InfoQuest

From **KMSystems, Inc.**

System User Guide

IN0501(09-00)

If you have any comments about the software or documentation, notify KMSystems, Inc., in writing at the following address:

KMSystems, Inc.
3225 Shallowford Road
Building 1000
Marietta, Georgia 30062
U.S.A.
Technical Support (770) 635-6363 - Main Number (770) 635-6350 - Fax (770) 635-6351

InfoQuest Release 5R6, November 1999

Contents

Contents

Chapter 1 Getting Started

InfoQuest is a new approach to data processing. InfoQuest allows an end user to produce sophisticated reports from a wide variety of data models, without having to worry about the procedural languages required to use the data models. As a Management Information System (MIS) Professional, your responsibilities are focused on data access, efficiency, and security, as opposed to generating endless procedural code to produce ad hoc reports.

This user guide is to assist in establishing a functional InfoQuest environment, which includes creating user views and database modules (DBMs). Also included are the steps required for user registration, application definition and security level validation. Maintenance functions are provided for report management and error analysis to KMSystems.

1.1 Using the InfoQuest System User Guide

The InfoQuest System User Guide is written for the person needing to maintain the InfoQuest operational environment, and requires an understanding of the data models (e.g., DMS 2200, PCIOS, RDMS 2200 and MAPPER DTM) to be accessed. This user guide is written primarily for the person needing to define the data and create the database access modules (DBMs) required to produce reports with the InfoQuest system. The DBMs are Q-LINK programs that contain record level access logic necessary to access selected file systems. For the most part, these DBMs can be automatically generated by the InfoQuest Maintenance Program.

Using the InfoQuest System User Guide

1.1.1 Summary of Chapters

This guide is divided into eight sections:

Chapter 1 contains an introduction to this guide and a brief overview of the maintenance process required to define applications and generate DBMs.

Chapter 2, called “A Brief Walk-through IQMNT”, contains a description of the InfoQuest Domain and a walk through the initial creation of a sample application from the time the application is defined up through the completed DBM. The section is meant as an introduction for those who are not already familiar with the InfoQuest maintenance programs.

Chapter 3 contains a description of each InfoQuest maintenance menu and a discussion of the fields on each screen. The section begins with a menu map which illustrates how the maintenance functions are organized.

Chapter 4, called “View Generation and Maintenance”, illustrates the steps necessary to produce the InfoQuest views that will be visible to the end user. These views contain a list of data items from which end users may produce reports.

Chapter 5, called “DBM Generation and Maintenance”, shows the steps required to automatically generate the DBMs which drive the report generation process. InfoQuest works in concert with the DBMs to produce the reports from a list of user selected items (presented in the view).

Chapter 6 contains examples of various InfoQuest paths used to access a variety of data structures.

Chapter 7 describes the procedure for installing the validation application. The validation application will allow you to verify that InfoQuest has been successfully installed. It is also required for the end user to perform the hands-on tutorial described in the InfoQuest User Handbook.

Chapter 8, called “Off-line Request Trace/Scheduler”, contains the operation requirements for the InfoQuest off-line request trace/scheduler and monitoring utilities.

The final part of the guide is the appendices. They contain the QINDEX reference; a description of the DBM components; examples of tailored DBMs; DBM error detection procedure (includes a description of the KMSystems supplied DBM debugging aide, IQDEBUG); security considerations for InfoQuest; Q-LINK run-time error descriptions; various listings (DMS schema, RDMS tables, etc.) used in the examples of this guide and the tutorial portion of the InfoQuest User Handbook; and a description of the messages and warnings that can occur during an InfoQuest session.

1.1.2 Conventions Used in this Guide

Important notes and warnings are encased in a gray box as shown around this paragraph.

Changes to this document since its last publication are marked with a change bar (an elongated vertical bar) as shown to the right of this paragraph.

The double greater-than sign (») character will be used in this guide to represent the Start-Of-Entry (SOE) character normally shown for OS2200 DEMAND session examples.

1.2 Additional InfoQuest Documentation

The InfoQuest User Handbook is included with the InfoQuest release. This handbook can be used to assist the user when generating InfoQuest requests. The examples shown in the handbook utilize the demonstration application included on the InfoQuest release tape. These examples can be performed by the user after the demonstration database and application have been installed as explained in Chapter 7 of this System User Guide.

Detailed instructions for installing InfoQuest may be found in the InfoQuest Installation Guide which is included with the release of InfoQuest.

1.3 Important Terms and Definitions

Application	The definition of an InfoQuest application differs from the traditional definition of an application. Traditionally, an application was used to define boundaries within an organization. An application could be defined as payroll, accounts receivable, etc. There may have been distinct data bases or files to support each application. While InfoQuest may be used in this environment, an InfoQuest application may encompass several traditional applications. To InfoQuest, an application comprises the universe of data that may be accessed as a group. With InfoQuest, reports may be generated from a variety of different data sources (assuming of course that a common key exists). An InfoQuest application may also be a subset of a traditional application. For example, an InfoQuest application could be a single area within a database. When creating an InfoQuest application, you must specify a two character code for the application and a short description for it. This process is covered in Chapter 3 of this manual.

Important Terms and Definitions

Data Item Index File	One or more data item index files are created for each application. The data item index file contains the names and definitions of all of the data items available to the application. For PCIOS or DMS data, the QINDEX processor is used to build the data item index files. For PCIOS data, input to the QINDEX processor will consist of standard COBOL record definitions. For DMS data files, QINDEX will build the data item index file from an invoked schema/subschema. For RDMS data and MAPPER data, an InfoQuest maintenance function (see Chapter 3) is used to build the data item index file. The use of the QINDEX processor is described in the “QINDEX Reference” (see Appendix A) included within this documentation.
Security	Security for InfoQuest is controlled by application and by view. A user must be registered to the application before being allowed access. Security levels may be established that provide different views based on a user’s security level. InfoQuest security is fully described in Chapter 3 and Appendix H.
View	The InfoQuest View is used to define subsets of the InfoQuest application. A view may be created that shows or allows access to any subset of the data described in an application’s data item index file. Many views may be created for an application. The views are usually modified to make the data names more understandable to end users. InfoQuest provides view editing and duplication capabilities to facilitate the creation of views. The generation and modification of views is fully discussed in Chapter 4.
Data Base Module (DBM)	The Data Base Module (DBM) specifies how InfoQuest will navigate the data models while generating end user reports. The DBMs are generally related to a view. It is possible for several views to use the same DBM. The DBMs represent the static portion of accessing the data bases. When the end user generates a request, the appropriate DBM and the required data items for the specific report are combined to create an InfoQuest program which will create the requested report. The DBMs may be modified to accommodate any site specific needs such as decoding data or translating keys. The generation and modification of DBMs is covered in Chapter 5.

Maintenance	InfoQuest Maintenance Functions provide the InfoQuest Administrator with tools to manage the InfoQuest environment. Some of the maintenance functions are:

User Registration	Controls the access of end users to the InfoQuest applications.
Application Import/Export	Allows an InfoQuest Application to be transported from one installation of InfoQuest to a second installation of InfoQuest, a second MAPPER or another computer system. This function is useful when developing new InfoQuest Applications or testing new software release levels.

The maintenance functions are described in greater detail in Chapter 3.

Important Terms and Definitions

Chapter 2 A Brief Walk-through of IQMNT

This chapter will take the InfoQuest Administrator through the steps required to establish an InfoQuest application, register users, create the data item index file, establish views and generate DBMs. The chapter begins with a brief description of the user interfaces, programs, runs and components within the InfoQuest Domain. The last part of the chapter provides an on-line walk-through or demonstration of the InfoQuest Maintenance Program, IQMNT. The walk-through requires no special preparation, as all the files required for the demonstration were placed on your system when InfoQuest was installed.

However, before you begin, spend a few minutes on a brief tour of the InfoQuest Domain.

2.1 The InfoQuest Domain

In the InfoQuest Domain, data is made available to the end user through the use of “InfoQuest Views”. The end user will use these views to create a variety of requests. The end user is not expected to be familiar with any particular data structure where the data resides or even familiar with the data processing techniques necessary to access these structures. Through a series of point-and-pick menus, the end user selects the data best suited for their needs and chooses the order the data is to be presented (line format, sort order, subtotal control breaks, totals, etc.).

The views are established from existing definitions of files, databases, relational tables and MAPPER RIDs. Someone in your organization (probably you, since you are reading this document) is appointed to administer InfoQuest, and as such, will be responsible for generating the views. View generation begins by identifying a single “InfoQuest Application”.

An InfoQuest Application comprises the universe of data that may be accessed as a group. In a traditional sense, it might be the database of a particular organization within your company (e.g., Payroll, Accounts Receivable, etc.). However, the definition of an application is not restricted to this interpretation. It could be broad enough to encompass several PCIOS files (any type) and areas of a DMS 2200 Database, include RDMS 2200 relational tables and MAPPER RIDs, or almost any combination of the supported OS 2200 file structures. An InfoQuest Application could be as simple as a subset of a traditional application such as a single order entry file or one area of a general ledger database.

2.1.1 MIS Setup/Maintenance Processes

Once the InfoQuest Application has been identified, the InfoQuest Administrator registers the application (described later in this chapter) and creates three required components: the Application's QINDEX File, a View and a Database Module (DBM).

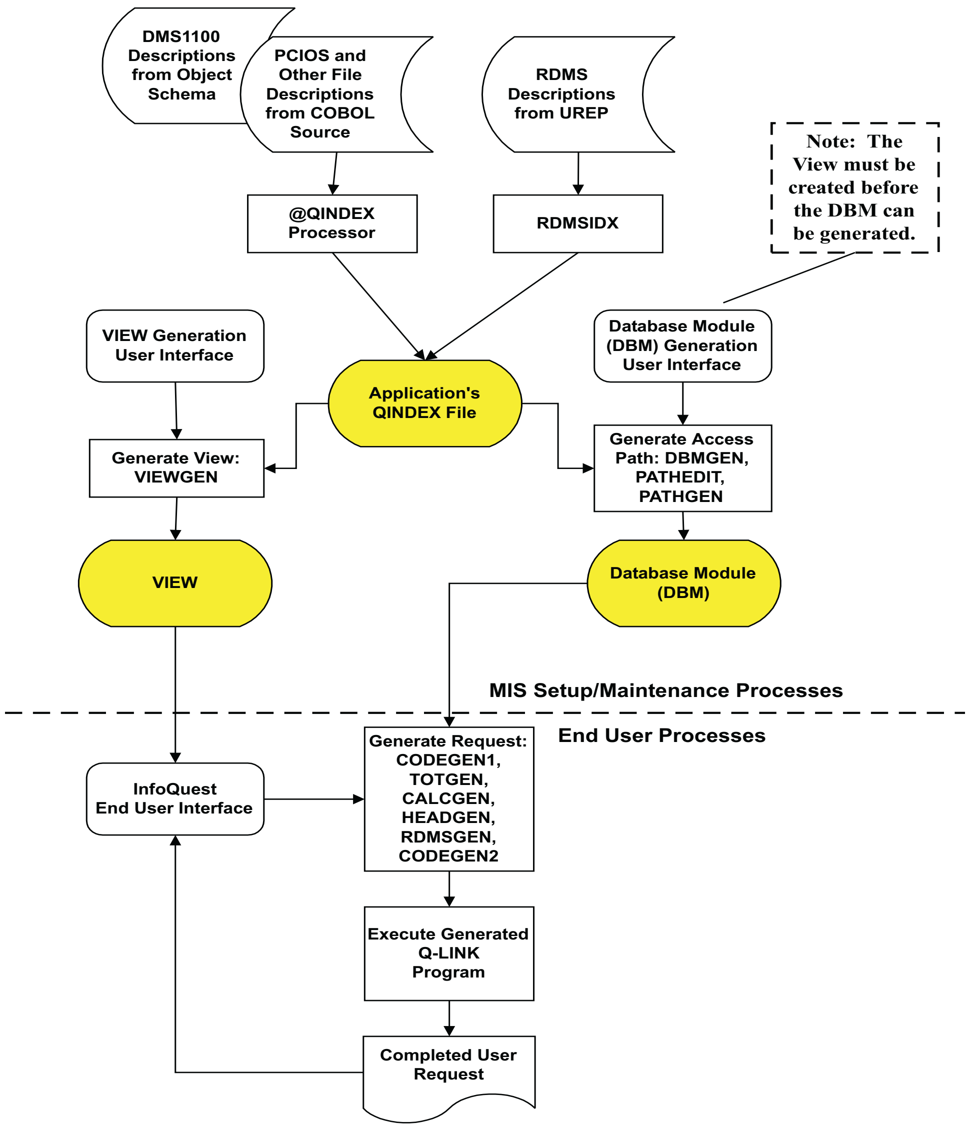

DMS1100 Descriptions from Object Schema
PCIOS and Other File Descriptions from COBOL Source
RDMS Descriptions from UREP
Note: The View must be created before the DBM can be generated.
@QINDEX Processor
RDMSIDX
VIEW Generation User Interface
Application's QINDEX File
Database Module (DBM) Generation User Interface
Generate View: VIEWGEN
Generate Access Path: DBMGEN, PATHEDIT, PATHGEN
VIEW
Database Module (DBM)
MIS Setup/Maintenance Processes
End User Processes
InfoQuest End User Interface
Generate Request: CODEGEN1, TOTGEN, CALCGEN, HEADGEN, RDMSGEN, CODEGEN2
Execute Generated Q-LINK Program
Completed User Request

Figure 1: The InfoQuest Domain

The first component, the InfoQuest Application's QINDEX File (see Figure 1), is created from existing data item definitions. This file is the point from which views/DBMs for the application are generated; however, an application is not limited to only one QINDEX file (a view is).

For views of DMS 2200 data and/or PCIOS data, the QINDEX processor (@QINDEX) is used to create the QINDEX file in an OS 2200 DEMAND session. Input definitions to the QINDEX processor can be from one of two sources. For DMS 2200 data, the definitions are taken from an existing object schema. PCIOS file definitions come from existing COBOL source such as COBOL COPY PROCs.

For all other file structures (RDMS 2200, MAPPER, etc.), the InfoQuest maintenance program, IQMNT, is used to create the QINDEX file. For RDMS 2200 data, the IQMNT utility, RDMSIDX, acquires the data item definitions from existing column descriptions found in the Unisys Repository (UREP). Item definitions for MAPPER data come from existing column headings in the RIDs.

Once the QINDEX file has been created, the InfoQuest Administrator uses the View Generation User Interface of IQMNT to specify which files/records/table data definitions are to be included in the view. The interface executes the InfoQuest Utility, VIEWGEN, to actually generate the second component, the InfoQuest View.[1] Once the view has been generated, the third component, the Database Module (DBM), may be generated.

The DBM is the code used to access the data when the end user submits an InfoQuest request. The InfoQuest Administrator uses the DBM Generation Interface to specify which entities (i.e., areas, records, files, etc.) are to be accessed by the DBM and the paths to those entities. The interface executes three IQMNT internal routines: DBMGEN, PATHEDIT and PATHGEN. PATHEDIT validates the DBM generation and path generation parameters entered by the InfoQuest Administrator. DBMGEN generates the base logic of the DBM, while PATHGEN generates the logic necessary to actually access the desired file structure(s). The final product of these three utilities is an incomplete Q-LINK program called the base DBM. It is incomplete because it must be further altered after the end user enters the selection and ordering parameters (see below).

Once the base DBM has been completed, it and an associated view are ready for use by the end user.

1 At this point, the view may be tailored to the specific needs of the end user (making user-friendly item names, establishing required keys, specifying special item editing, etc.) using other user interfaces of IQMNT.

2.1.2 End User Processes

The end user invokes InfoQuest by using one of the following InfoQuest End User Interfaces:

Interface:	**Software Component:**
The IQ run	InfoQuest (MAPPER-based)
The @IQEX[*x*] processor	InfoQuest/EX (DPS-based) where *x* is dependent upon the COMUS mode used to install InfoQuest — @IQEX for the default mode or @IQEXA through @IQEXK for modes INFOQA through INFOQK, respectively
The InfoQuest/WS Manager icon	InfoQuest/WS (Windows-based)

The end user Interface begins an InfoQuest Request by requiring the end user to select an InfoQuest Application and a view from that application. The user does not have to select a DBM as the view has a DBM associated with it.

At this point, the user begins to make the selections (items, order, options, etc.) that will become the InfoQuest Parameters. Once this selection process is completed and the user presses the key that represents the "RunNow" function, InfoQuest will store these InfoQuest Parameters within its database.

After the user enters a name for the request, InfoQuest will execute a number of internal routines (CODEGEN1, TOTGEN, CALCGEN, HEADGEN, RDMSGEN, CODEGEN2) that will generate, optionally store and execute a complete Q-LINK program. Once the execution has completed, the request is returned to the user or alternately routed as chosen by the user.

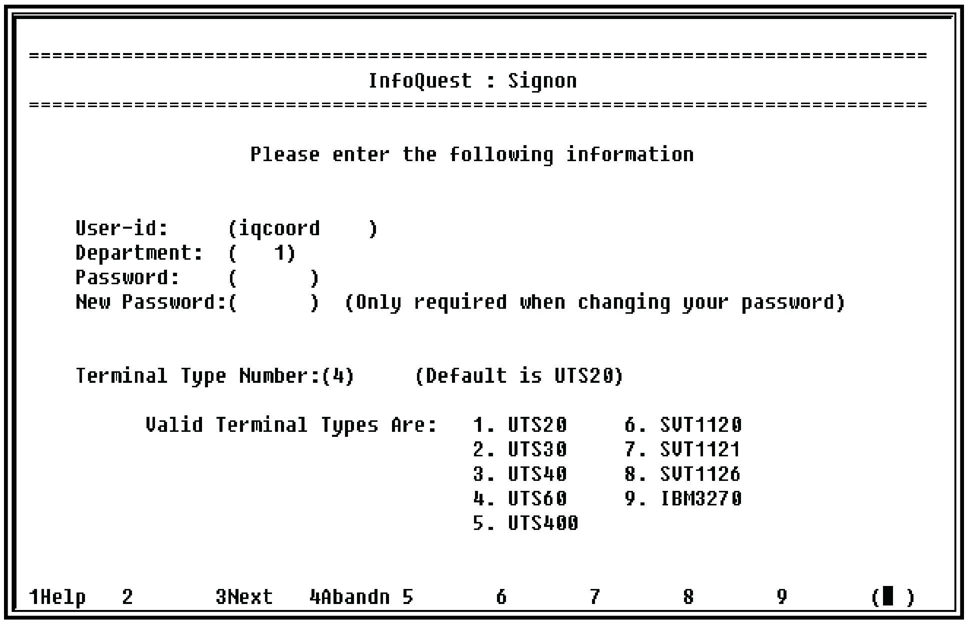
The remainder of this chapter is a "walk-through" that allows you to perform the basic maintenance functions of InfoQuest using the **InfoQuest Validation Demo** provided with InfoQuest.

2.2 The Walk-Through

When InfoQuest is installed, the data structures and files required for this walk-through are also installed. When you have completed this exercise, you will be able to generate ad hoc reports from a PCIOS file. The exercise is composed of several steps that are self contained. The walk-through can be interrupted at the completion of any section and resumed at a later time without losing what has already been processed. The entire walk-through can be completed in one-half hour.

After InfoQuest has been installed, initiate the InfoQuest Maintenance Subsystem by entering @IQMNT[*x*] (where *x* is dependent upon the COMUS mode used to install InfoQuest — @IQMNT for the default mode or @IQMNTA through @IQMNTK for modes INFOQA through INFOQK, respectively).

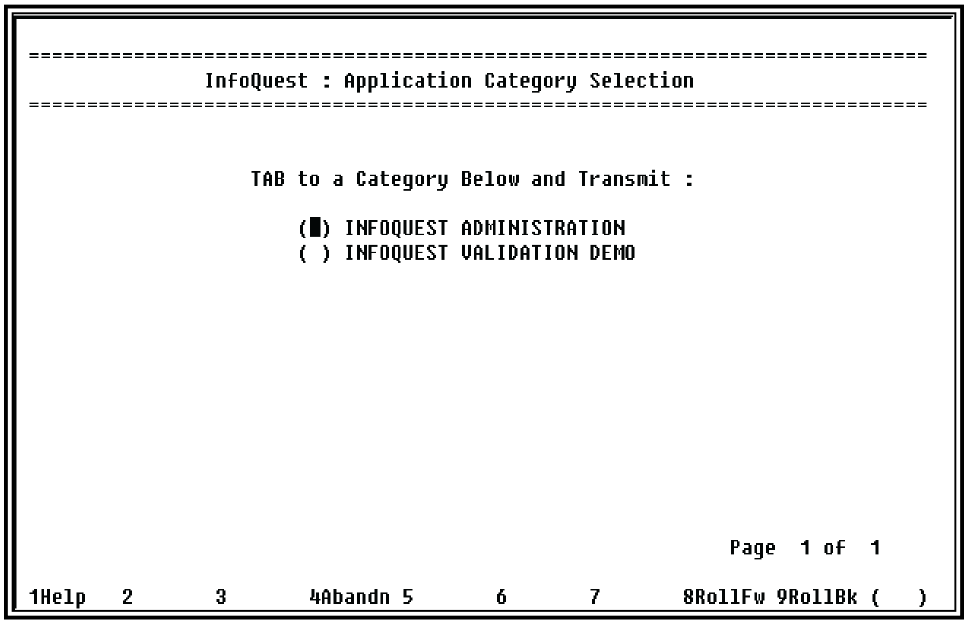
The Application Selection Menu will appear showing two applications that are supplied with InfoQuest at install time. The walk-through will use the InfoQuest Administration Application. The user-id, “IQCOORD”, in department 1 with a password of “IQEX” is installed when InfoQuest is installed. This user-id must be used until new user-ids are registered.



```
=============================================================================
                              InfoQuest : Signon
=============================================================================

                   Please enter the following information


    User-id:     (iqcoord    )
    Department:  (   1)
    Password:    (      )
    New Password:(      )  (Only required when changing your password)


    Terminal Type Number:(4)     (Default is UTS20)

          Valid Terminal Types Are:   1. UTS20     6. SVT1120
                                      2. UTS30     7. SVT1121
                                      3. UTS40     8. SVT1126
                                      4. UTS60     9. IBM3270
                                      5. UTS400


1Help   2       3Next   4Abandn 5       6       7       8       9       (█ )
```

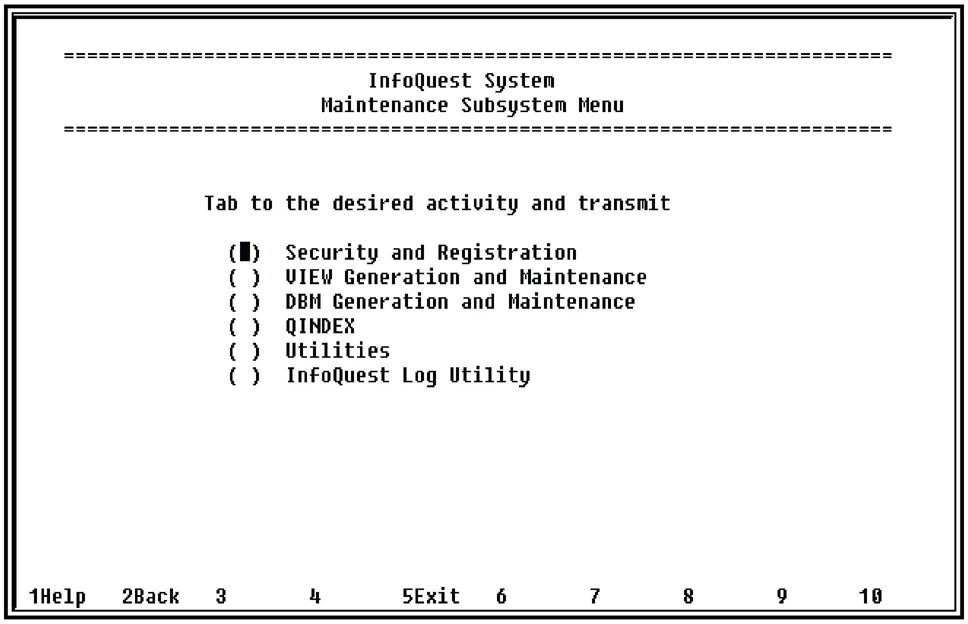
The Walk-Through

When the Application Selection screen menu appears, select “InfoQuest Administration”.



```
==============================================================================
               InfoQuest : Application Category Selection
==============================================================================


                   TAB to a Category Below and Transmit :

                       (█) INFOQUEST ADMINISTRATION
                       ( ) INFOQUEST VALIDATION DEMO










                                                           Page  1 of  1

1Help   2       3       4Abandn 5       6       7       8RollFw 9RollBk (    )
```

After selecting InfoQuest Administration, you will see the Maintenance Subsystem Menu. With the Security and Registration entry highlighted, press transmit.

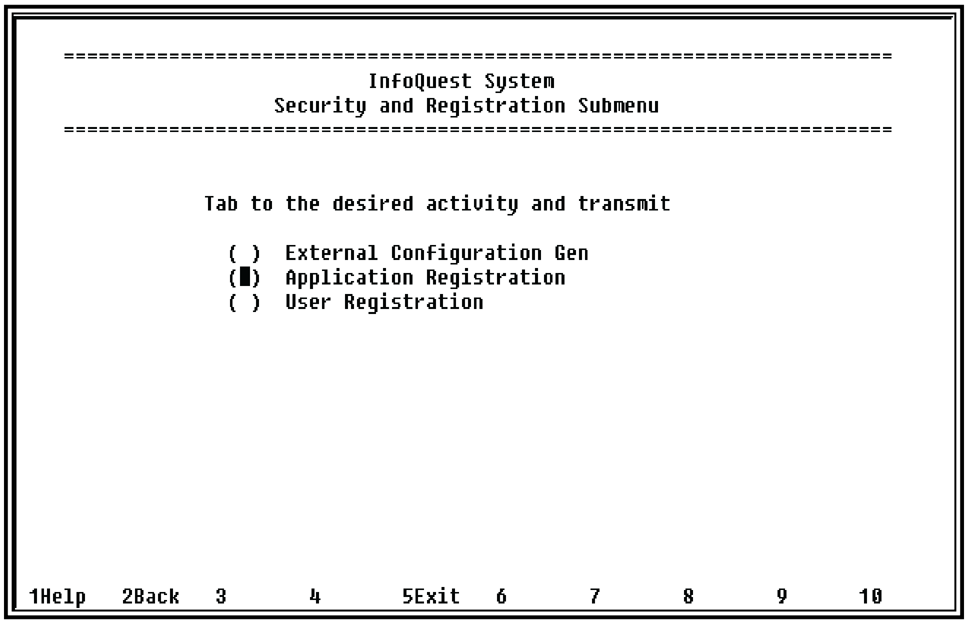


```
   =======================================================================
                              InfoQuest System
                          Maintenance Subsystem Menu
   =======================================================================


                Tab to the desired activity and transmit

                  (█)  Security and Registration
                  ( )  VIEW Generation and Maintenance
                  ( )  DBM Generation and Maintenance
                  ( )  QINDEX
                  ( )  Utilities
                  ( )  InfoQuest Log Utility







1Help   2Back   3       4       5Exit   6       7       8       9      10
```

2.3 Registering an Application

Note: The Security and Registration functions can only be performed by a user that is registered as an InfoQuest Administrator. The person who installed InfoQuest was automatically provided with these privileges when he ran the IQINSTALL run. For other users to be given administrator privileges, they must be registered (see "User Registration" page 3-12) with the pseudo application code of "00" (two zeros).

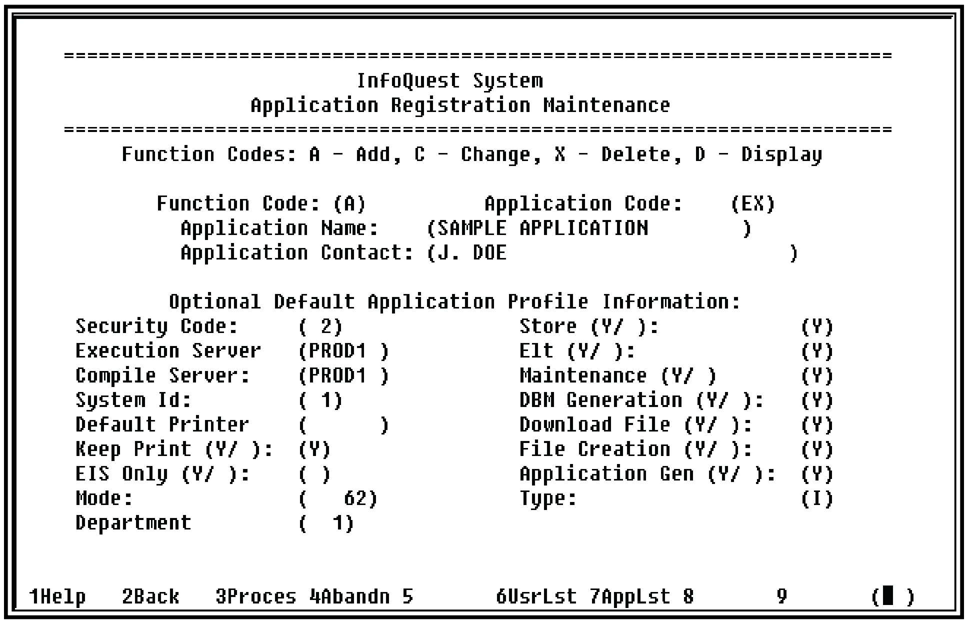
The next screen displayed will be the Security and Registration Submenu screen. On this screen, tab to the Application Registration option and press transmit.



```
======================================================================
                         InfoQuest System
                  Security and Registration Submenu
======================================================================


            Tab to the desired activity and transmit

              ( )  External Configuration Gen
              (█)  Application Registration
              ( )  User Registration









1Help   2Back   3       4       5Exit   6       7       8       9       10
```

Registering an Application

You will be asked for a function code and an application code. Enter “A” for the function code. For this sample application enter “EX” for the application code.



```
======================================================================
                        InfoQuest System
               Application Registration Maintenance
======================================================================
     Function Codes: A - Add, C - Change, X - Delete, D - Display

        Function Code: (A)          Application Code:    (EX)
          Application Name:    (SAMPLE APPLICATION       )
          Application Contact: (J. DOE                       )

         Optional Default Application Profile Information:
 Security Code:     ( 2)              Store (Y/ ):           (Y)
 Execution Server   (PROD1 )          Elt (Y/ ):             (Y)
 Compile Server:    (PROD1 )          Maintenance (Y/ )      (Y)
 System Id:         ( 1)              DBM Generation (Y/ ):  (Y)
 Default Printer    (      )          Download File (Y/ ):   (Y)
 Keep Print (Y/ ):  (Y)               File Creation (Y/ ):   (Y)
 EIS Only (Y/ ):    ( )               Application Gen (Y/ ): (Y)
 Mode:              (   62)           Type:                  (I)
 Department         (  1)

1Help   2Back   3Proces 4Abandn 5       6UsrLst 7AppLst 8       9       (█ )
```

You must now enter an Application Name for the EX application code. This name will be associated with the application code and is the name presented to the end user when they are asked to select an application. Enter “Example Application” in the Application Name field, an “01” in the System Id. field and press Function Key 3 to proceed (Note: The other parameters on this screen under “Optional Default Application Profile Information” will be discussed in Section 3.4.2, “Application Registration”, and Section 3.4.3, “User Registration”). IQMNT will respond with a confirmation message indicating that the application code EX has been added. At this point, you may add additional application codes if needed. When all entries have been completed, press the F2 key to return to the main IQMNT menu. A new application code, EX, has just been added to InfoQuest.

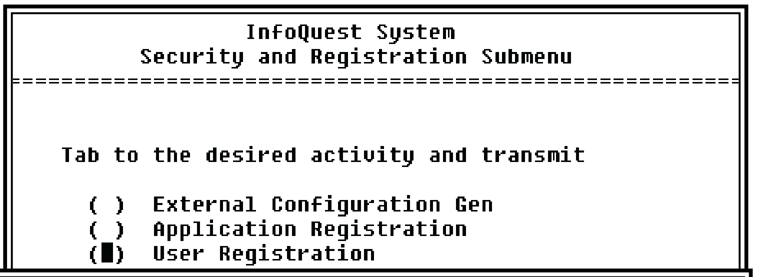
If you specify an application code that is already in use, IQMNT will inform you and ask you to select a different two character designator for your application.

2.4 Registering a User to an Application.

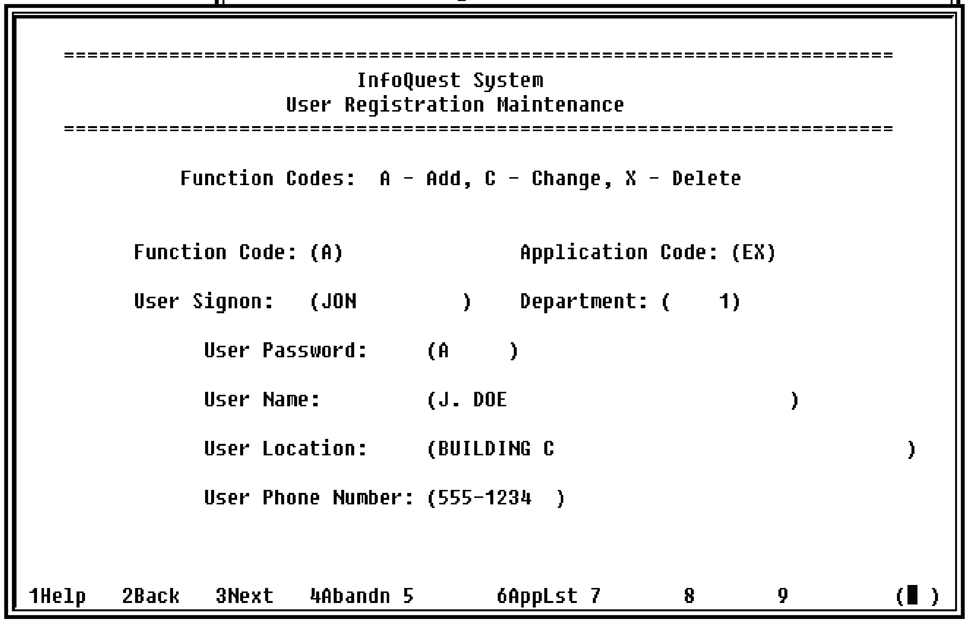
Now that the application has been registered, it is possible to register users to that application. IQMNT will not allow you to register users to an application that has not already been registered. To register the user, select the Security and Registration option from the IQMNT main menu and press transmit.

The Security and Registration Submenu screen will be displayed. Move the cursor to the User Registration option and press transmit. You will now see the User Registration Maintenance screen. Select "A" for the "Function Code" (to add a user registration). Enter the "Application Code" to which the user is to be allowed access (EX in this case). Enter the "User Signon" to be registered, which corresponds to the user's user-id. Finally, enter the user's "Department" number. When all of the fields are completed, press the F3 key to proceed to the next step.

The remaining fields on this screen are optional — even the User Password. The end user may choose their own password when they first sign on to InfoQuest.



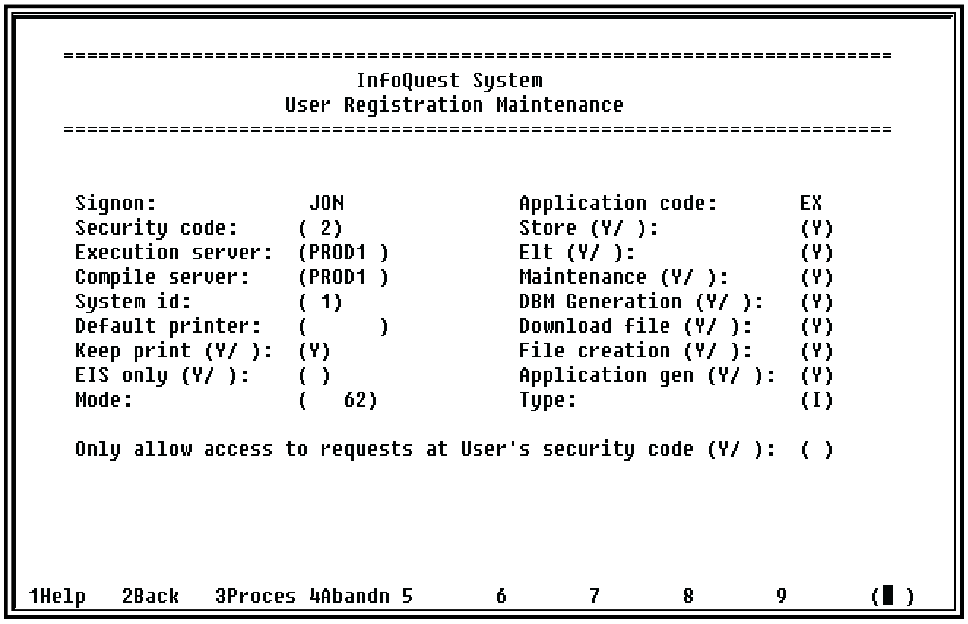
```
                  InfoQuest System
          Security and Registration Submenu
=======================================================


    Tab to the desired activity and transmit

      ( )  External Configuration Gen
      ( )  Application Registration
      (█)  User Registration
```



```
    ========================================================================
                             InfoQuest System
                       User Registration Maintenance
    ========================================================================

              Function Codes:  A - Add, C - Change, X - Delete


          Function Code: (A)              Application Code: (EX)

          User Signon:   (JON        )    Department: (    1)

                User Password:     (A     )

                User Name:         (J. DOE                       )

                User Location:     (BUILDING C                             )

                User Phone Number: (555-1234  )


 1Help   2Back   3Next   4Abandn 5       6AppLst 7       8       9        (█ )
```

Registering a User to an Application.

The next screen to be displayed confirms the sign-on and the application code. We must now specify the privileges that the user is allowed.



```
======================================================================
                        InfoQuest System
                  User Registration Maintenance
======================================================================


 Signon:            JON              Application code:       EX
 Security code:     ( 2)             Store (Y/ ):            (Y)
 Execution server:  (PROD1 )         Elt (Y/ ):              (Y)
 Compile server:    (PROD1 )         Maintenance (Y/ ):      (Y)
 System id:         ( 1)             DBM Generation (Y/ ):   (Y)
 Default printer:   (      )         Download file (Y/ ):    (Y)
 Keep print (Y/ ):  (Y)              File creation (Y/ ):    (Y)
 EIS only (Y/ ):    ( )              Application gen (Y/ ):  (Y)
 Mode:              (   62)          Type:                   (I)

 Only allow access to requests at User's security code (Y/ ):  ( )




1Help   2Back   3Proces 4Abandn 5       6       7       8       9       (█ )
```

We will proceed through the screen from left to right, top to bottom, using the tab key to move from field to field. The “Security code” is a value from 1 to 99 — the lower the number, the higher the privilege. In this case, security is set to 1, implying that Jon has no InfoQuest access restrictions. The “Store” field determines if the user is allowed to save reports in MAPPER (MAPPER-based InfoQuest only). The “Execution Server” field specifies which Q-LINK server class will be used to execute the InfoQuest request. The server can be specified to provide high or low priority to the user. For this example, enter your standard production server, usually PROD1. The “Elt” field controls the user’s ability to use the MAPPER ELT function to store reports in a program files outside of MAPPER (MAPPER-based InfoQuest only). The next field, “Compile Server”, specifies which server is to be used to compile the InfoQuest request. Normally this server is the same as the execution server specified previously. The “Maintenance” field determines if the user is allowed to perform maintenance functions via IQMNT for this application. In the “System id” field, enter the system id under which this user will be operating. This field will control the batch port to be used on batch InfoQuest runs and other configuration options (see 3.4.1, “External Configuration Generation”). For now, enter a “1”. The next field ,“DBM generation”, determines if the user is allowed to create DBMs for this application (this field is not currently in use and is reserved for future implementation within InfoQuest). The “Default Printer” is the OS 2200 printer id as configured in the EXEC where the user’s report files should be sent when the user selects the site printing option yet supplies no printer id. “PR” is the default OS 2200 system printer. The “Download file” field allows the user to download files to a personal computer. “Keep print” determines whether the user is allowed to maintain report print files on the system. The “File creation” field allows the user to create EXEC files from InfoQuest reports. The “EIS only” field can be used to restrict the user to only the Executive Information Systems feature of InfoQuest. For now, leave this field blank. The next field, “Application gen”, controls whether or not the user is allowed to create subordinate applications from an existing application. This procedure is known as application cloning. The “Mode” field contains the mode in which the user is allowed to operate (MAPPER-based InfoQuest only). In the “Type” field, enter the type where the InfoQuest reports will be stored (MAPPER-based InfoQuest only). Normally, you enter an “I” in this field.

When all of the parameters have been entered, press F3 to proceed. If no errors are detected, IQMNT will add the user information. Any errors will be displayed with a message indicating the problem. Correct the fields that are in error and re-submit the screen. When the user has been added, the user will have access to the application. The user registration process is complete.

2.5 Creating the Data Item Index File

If you will be accessing RDMS 2200 or MAPPER data structures, you will create the data item index file from IQMNT using the QINDEX option. For accessing PCIOS and DMS 2200 data structures, you will have to run the QINDEX processor from demand mode on the 2200. The QINDEX processor is fully described in Appendix A, “QINDEX Reference” of this guide.

The program file, IQ$DEMO*IQ$DEMO (installed along with InfoQuest), will contain everything needed to create the sample application illustrated in this chapter. In the IQ$DEMO*IQ$DEMO file you will find the COBOL definitions for the demonstration data files. While it is possible to access many files concurrently, this example will detail the steps required to provide InfoQuest access to the customer demonstration file, CUSTFILE.

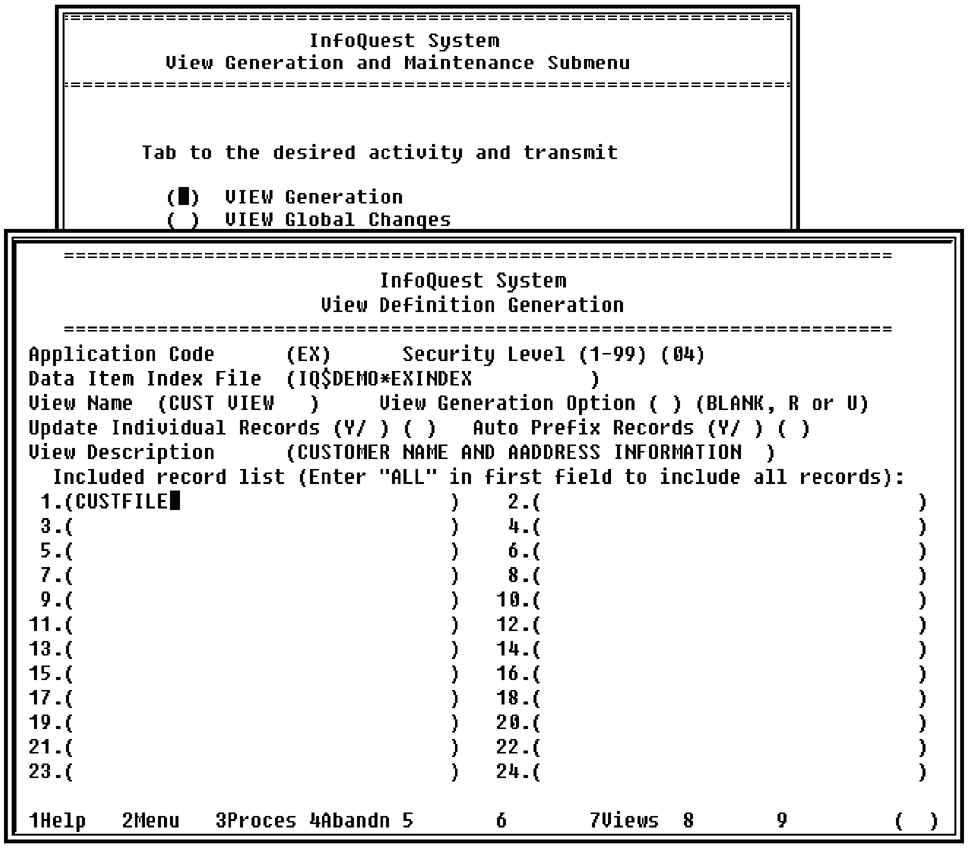
In a demand session, create the data item index file (QINDEX file). This file will contain the internal representation of the record layout of the CUSTFILE. To create the QINDEX file, IQ$DEMO*EXINDEX, enter the following commands:

```
@ASG,UPV    IQ$DEMO*EXINDEX.,F
@QINDEX,I   IQ$DEMO*EXINDEX.
FILE CUSTFILE
@ADD,E      IQ$DEMO*IQ$DEMO.CUSTFILE/COBOL-DEF
@FREE       IQ$DEMO*EXINDEX.
```

At this point, the IQ$DEMO*EXINDEX file contains the information needed to access data items in the file CUSTFILE. Now, we must return to IQMNT to establish the view.

2.6 Establishing the View

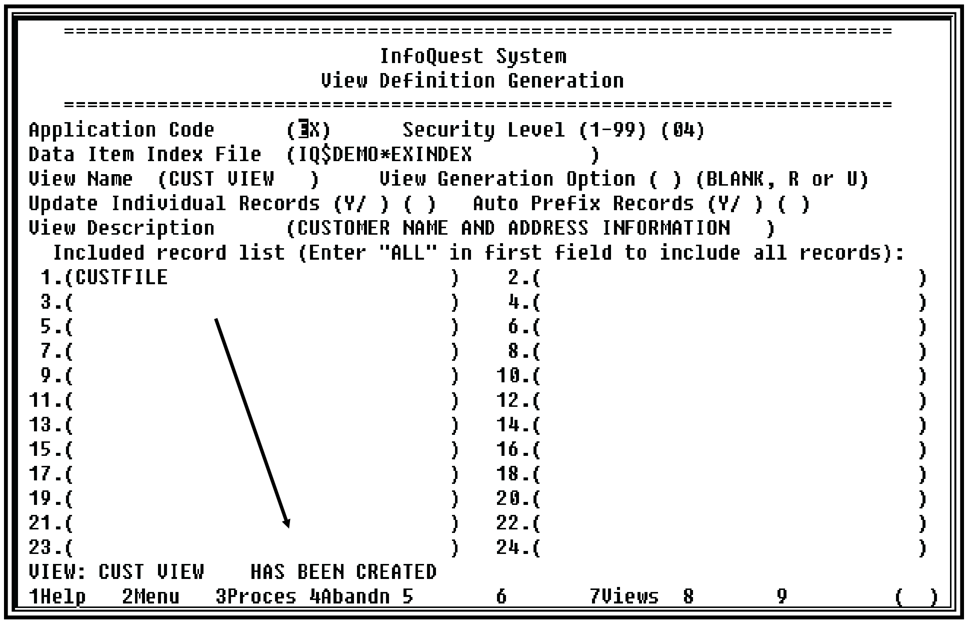
After entering IQMNT, you will see the Maintenance Subsystem menu. From this menu, move the cursor to the View Generation and Maintenance option and press transmit. You will now see the View Generation and Maintenance Submenu. The cursor should be positioned on the View Generation option. Press transmit to proceed with view generation.



```
==============================================================
                  InfoQuest System
        View Generation and Maintenance Submenu
==============================================================


      Tab to the desired activity and transmit

        (█)  VIEW Generation
        ( )  VIEW Global Changes
```

```
   ======================================================================
                           InfoQuest System
                      View Definition Generation
   ======================================================================
 Application Code      (EX)      Security Level (1-99) (04)
 Data Item Index File  (IQ$DEMO*EXINDEX          )
 View Name  (CUST VIEW    )     View Generation Option ( ) (BLANK, R or U)
 Update Individual Records (Y/ ) ( )   Auto Prefix Records (Y/ ) ( )
 View Description      (CUSTOMER NAME AND AADDRESS INFORMATION  )
   Included record list (Enter "ALL" in first field to include all records):
  1.(CUSTFILE█                     )    2.(                              )
  3.(                              )    4.(                              )
  5.(                              )    6.(                              )
  7.(                              )    8.(                              )
  9.(                              )   10.(                              )
 11.(                              )   12.(                              )
 13.(                              )   14.(                              )
 15.(                              )   16.(                              )
 17.(                              )   18.(                              )
 19.(                              )   20.(                              )
 21.(                              )   22.(                              )
 23.(                              )   24.(                              )

 1Help   2Menu   3Proces 4Abandn 5       6       7Views  8       9        (  )
```

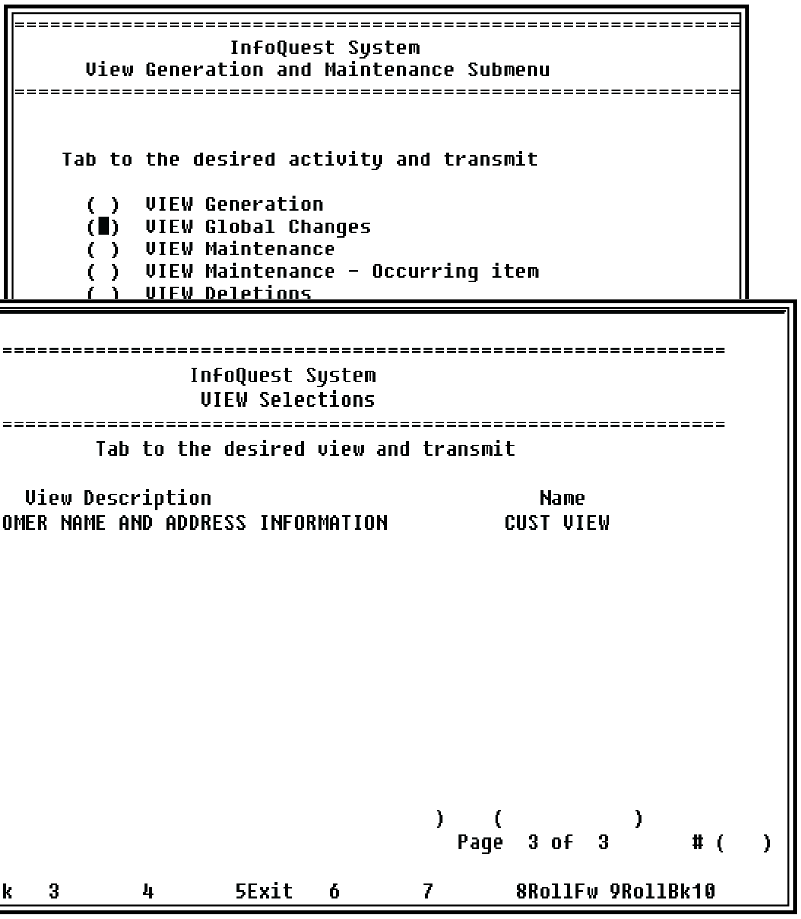
From the View Definition Generation screen, we will fill in the parameters required to create the view for our “EX” application. Enter “EX” in the application code field. The security level may be any number from 1 to 99; however, if it is set to a number lower than the security code given to the user when the user was registered, that user will not have access to this view. For this example, set the security level to 4. Tab to the next field (Data Item Index File), and enter the filename that was used by QINDEX (see previous page). In this case, enter IQ$DEMO*EXINDEX. In the “View Name” field, enter “CUST VIEW”, since this example allows us to view data from the customer file. The next field to be entered is the “View Description”. The View Descriptions are presented to the end user for selection and should be meaningful. Enter “Customer Name and Address Information", and TAB to the next field. While the data file index (IQ$DEMO*EXINDEX) only contains a single record description (CUSTFILE), it could easily contain descriptions for several records. In this field, you may specify which record descriptions you want to incorporate into this view. In our example, we will specify "CUSTFILE" and press transmit. A confirmation message will be displayed on the bottom of the screen. Now press the F2 key. We will be returned to the IQMNT main menu. The view is now ready for use.



```
   ======================================================================
                            InfoQuest System
                        View Definition Generation
   ======================================================================
Application Code      (3X)      Security Level (1-99) (04)
Data Item Index File  (IQ$DEMO*EXINDEX         )
View Name  (CUST VIEW   )     View Generation Option ( ) (BLANK, R or U)
Update Individual Records (Y/ ) ( )   Auto Prefix Records (Y/ ) ( )
View Description      (CUSTOMER NAME AND ADDRESS INFORMATION   )
  Included record list (Enter "ALL" in first field to include all records):
 1.(CUSTFILE                       )    2.(                              )
 3.(                               )    4.(                              )
 5.(                               )    6.(                              )
 7.(                               )    8.(                              )
 9.(                               )   10.(                              )
11.(                               )   12.(                              )
13.(                               )   14.(                              )
15.(                               )   16.(                              )
17.(                               )   18.(                              )
19.(                               )   20.(                              )
21.(                               )   22.(                              )
23.(                               )   24.(                              )
VIEW: CUST VIEW    HAS BEEN CREATED
1Help   2Menu   3Proces 4Abandn 5       6       7Views  8       9        (  )
```

Establishing the View

One feature of InfoQuest is to provide user-friendly names to the data items. This process is accomplished through the IQMNT View Generation and Maintenance facility. We will apply some global changes to the existing view items to make them more presentable. Once again, select the "View Generation and Maintenance" option and press transmit. At this point, move the cursor to the "VIEW Global Changes" option and press transmit.

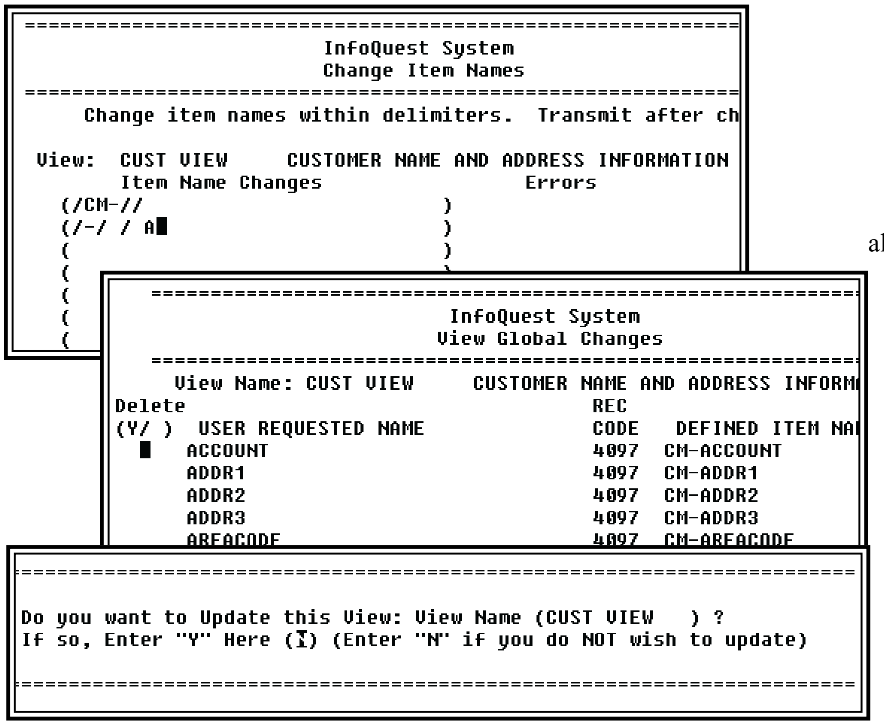


```
==============================================================
                   InfoQuest System
      View Generation and Maintenance Submenu
==============================================================


    Tab to the desired activity and transmit

      ( )  VIEW Generation
      (█)  VIEW Global Changes
      ( )  VIEW Maintenance
      ( )  VIEW Maintenance - Occurring item
      ( )  VIEW Deletions
```

```
  ======================================================================
                            InfoQuest System
                             VIEW Selections
  ======================================================================
                    Tab to the desired view and transmit

              View Description                            Name
  (█)   CUSTOMER NAME AND ADDRESS INFORMATION          CUST VIEW










TARGET (                                         )    (          )
                                                   Page  3 of  3       # (    )

 1Help   2Back   3       4       5Exit   6       7       8RollFw 9RollBk10
```

The screen that will be presented shows all the views to which we have access. If the view does not appear on the current screen, use the "TARGET" feature or you may use the F8 key (RollFw) or the F9 key (RollBk) to page through the views. Once you see the desired view, move the cursor to the view and press transmit. The "Change Items Names" screen will be seen next.

In this screen, we can make global changes to the view. For this example, we will eliminate all “CM–” which are prefixes to the data item as defined to COBOL. We will also



```
===========================================================
                          InfoQuest System
                          Change Item Names
===========================================================
     Change item names within delimiters.  Transmit after ch

 View:  CUST VIEW     CUSTOMER NAME AND ADDRESS INFORMATION
        Item Name Changes                 Errors
   (/CM-//                      )
   (/-/ / A█                    )
   (                            )
   (
   (
   (
   (
```

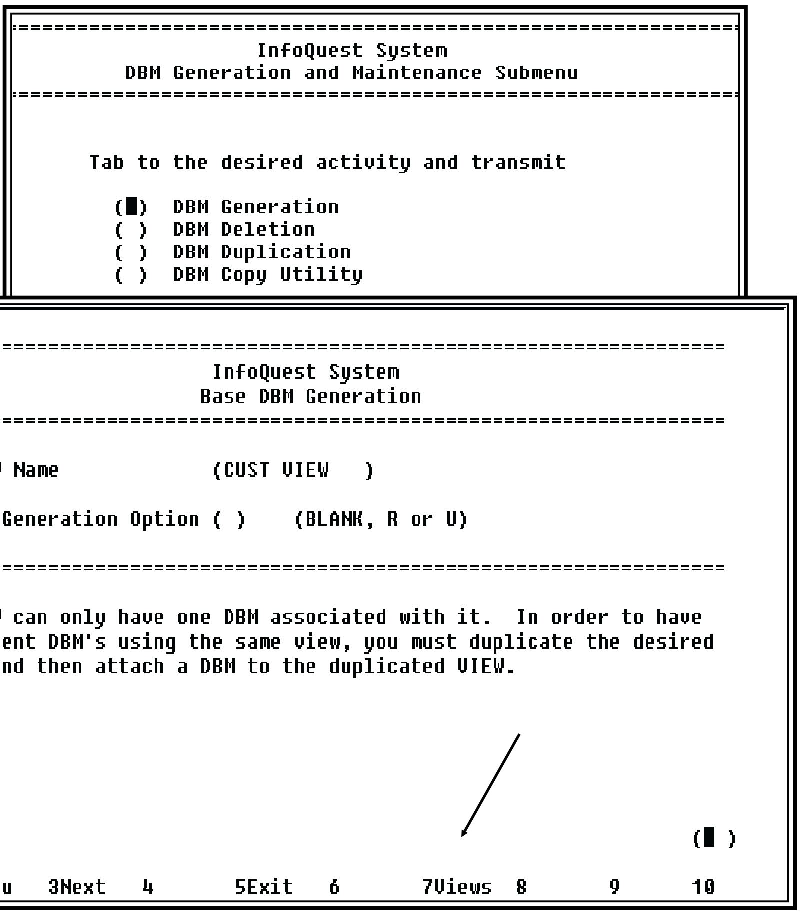
```
   ==========================================================
                            InfoQuest System
                           View Global Changes
   ==========================================================
      View Name: CUST VIEW     CUSTOMER NAME AND ADDRESS INFORM
 Delete                                 REC
 (Y/ )  USER REQUESTED NAME             CODE   DEFINED ITEM NA
   █   ACCOUNT                          4097  CM-ACCOUNT
       ADDR1                            4097  CM-ADDR1
       ADDR2                            4097  CM-ADDR2
       ADDR3                            4097  CM-ADDR3
       AREACODE                         4097  CM-AREACODE
```

```
======================================================================

Do you want to Update this View: View Name (CUST VIEW    ) ?
If so, Enter "Y" Here (Y) (Enter "N" if you do NOT wish to update)

======================================================================
```

replace all “–” symbols which appear in the record definition as word separators with a blank (“ ”). These changes are done by entering “/CM–//” on the first line and “/–/ /A” on the next line. The slashes are string delimiters. The first string contains the search value and the second string the result value. The “A” indicates that we want this change on all occurrences on each line. The absence of the “A” implies that only the first occurrence on each line is to be changed. After entering the above changes, press transmit. The new item names to be shown to the user will be displayed (see the “USER REQUESTED NAME” column on the second screen above). Any items that are not desired for this view may be eliminated by entering a “Y” and pressing transmit. You may also page through the data items by using F8 (RollFw) and F9 (RollBk). When you are done with any changes, press the F3 key. You will be asked if you want to update the view; enter “Y” and press transmit. There are other options in view maintenance to allow you to modify the display length, edit mask or data item name. These options are covered later in the manual. At this point, the view has been generated and modified.

2.7 Generating the DBM

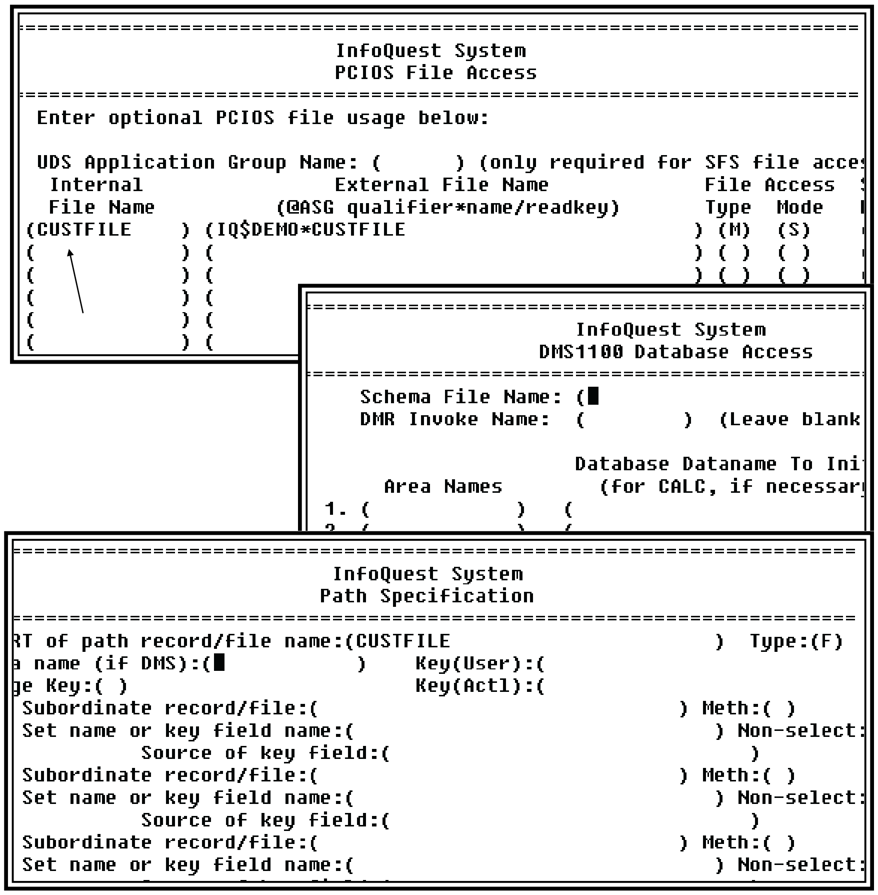
Once a view is available, the DBM may be generated. From the IQMNT "Maintenance Subsystem Menu", move the cursor to the "DBM Generation and Maintenance" option and press transmit. IQMNT will then display the "DBM Generation and Maintenance Submenu". The cursor should be positioned at the "DBM Generation" option. Press transmit to begin the DBM generation process. At the "VIEW Name" field, enter the view that was just created and press F3. If you do not rember the view name, you may use the F7 key to select from a list of names.



```
==============================================================
                      InfoQuest System
          DBM Generation and Maintenance Submenu
==============================================================


      Tab to the desired activity and transmit

        (█)  DBM Generation
        ( )  DBM Deletion
        ( )  DBM Duplication
        ( )  DBM Copy Utility
```

```
======================================================================
                            InfoQuest System
                           Base DBM Generation
======================================================================

     VIEW Name              (CUST VIEW   )

     DBM Generation Option ( )    (BLANK, R or U)

======================================================================

   A VIEW can only have one DBM associated with it.  In order to have
   different DBM's using the same view, you must duplicate the desired
   VIEW and then attach a DBM to the duplicated VIEW.




                                                                (█ )
 1Help   2Menu   3Next   4       5Exit   6       7Views  8       9      10
```

The next screen is used to describe the PCIOS files that are to be accessed by this DBM. Enter “CUSTFILE” for the internal file name, and “IQ$DEMO*CUSTFILE” for the external file name. The CUSTFILE is an MSAM type and will be accessed sequentially. Therefore, type an “M” in the “File Type” field and “S” in the “Access Mode” field. Multiple files may be specified. Press transmit to continue to the next screen. The next screen is used to define DMS 2200 access. Since we will not use DMS data in this example, press transmit. The final step of DBM generation shown is the “Path



```
=====================================================================
                          InfoQuest System
                          PCIOS File Access
=====================================================================
 Enter optional PCIOS file usage below:

 UDS Application Group Name: (      ) (only required for SFS file acce
  Internal                External File Name            File Access
  File Name          (@ASG qualifier*name/readkey)      Type  Mode
(CUSTFILE    ) (IQ$DEMO*CUSTFILE                       ) (M)  (S)
(            ) (                                       ) ( )  ( )
(            ) (                                       ) ( )  ( )
(            ) (
(            ) (
(            ) (
```

```
=============================================
                InfoQuest System
             DMS1100 Database Access
=============================================
    Schema File Name: (█
    DMR Invoke Name:  (        )  (Leave blank

                      Database Dataname To Ini
      Area Names        (for CALC, if necessar
  1. (            )   (
```

```
=====================================================================
                          InfoQuest System
                         Path Specification
=====================================================================
RT of path record/file name:(CUSTFILE                   )  Type:(F)
a name (if DMS):(█           )    Key(User):(
ge Key:( )                        Key(Actl):(
 Subordinate record/file:(                           ) Meth:( )
 Set name or key field name:(                           ) Non-select:
           Source of key field:(                        )
 Subordinate record/file:(                           ) Meth:( )
 Set name or key field name:(                           ) Non-select:
           Source of key field:(                        )
 Subordinate record/file:(                           ) Meth:( )
 Set name or key field name:(                           ) Non-select:
```

Specification” screen. IQMNT uses path specification to determine how to access all of the records in the path. In our example, we have one record. We only have to specify the internal filename, CUSTFILE, and the type, “F”. When we press transmit, IQMNT will build the DBM.

Generating the DBM

Chapter 3 InfoQuest Maintenance

This chapter is the starting point for all functions covered in this guide. It illustrates the use of an InfoQuest utility called, “IQMNT”.

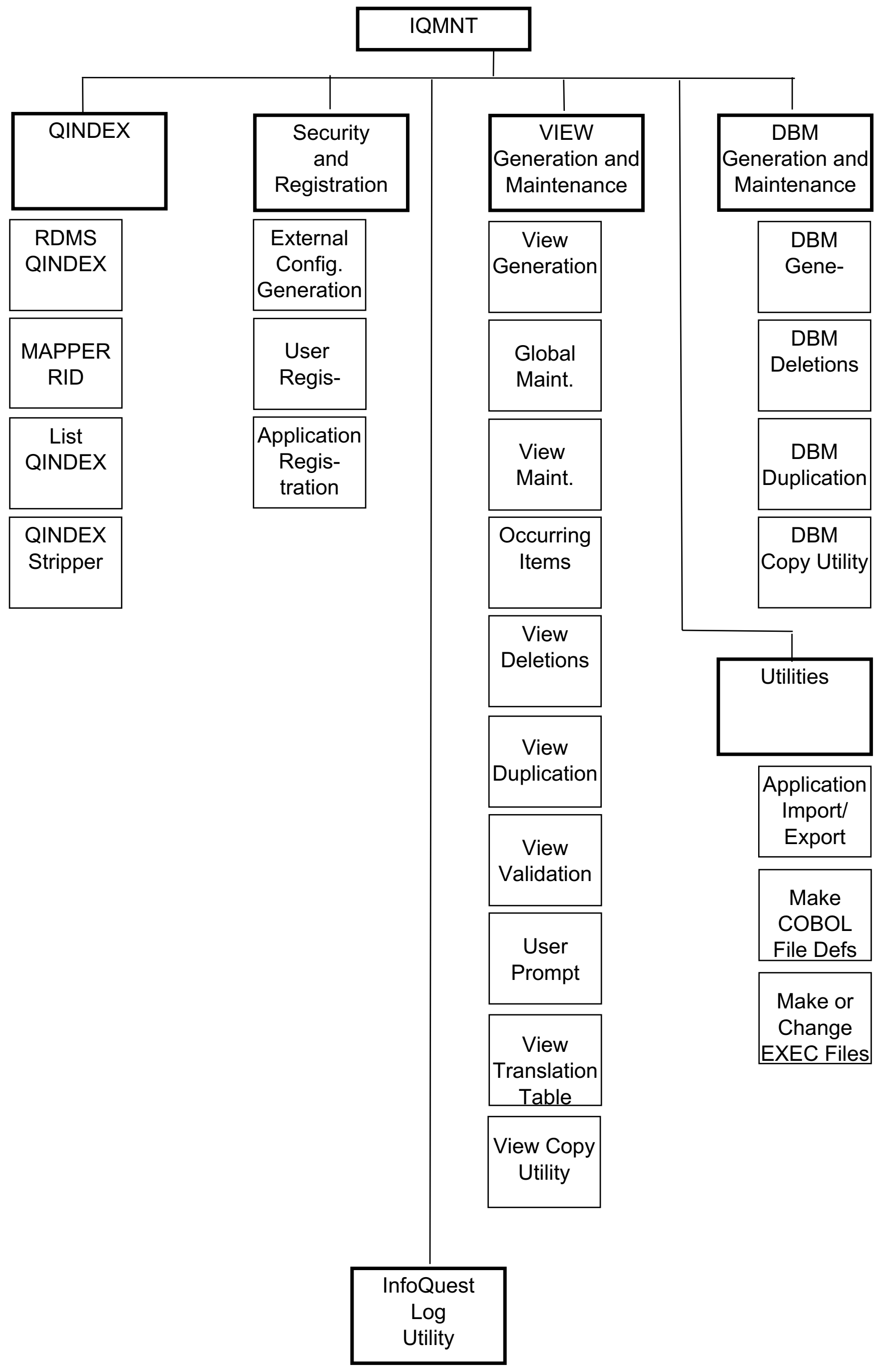
The IQMNT processor produces a menu from which all the maintenance functions can be accessed. The maintenance functions include:

- InfoQuest user registration, the maintenance of InfoQuest security codes and applications and InfoQuest external configuration maintenance;
- View generation, maintenance and validation;
- DBM generation and maintenance;
- Data item index generation (QINDEX) for RDMS 2200 tables and MAPPER RIDs;
- Utilities controlling the importing and exporting of InfoQuest applications and file creating;
- The InfoQuest log utility.

3.1 InfoQuest Maintenance Menu Map

The diagram or menu map on the following page shows the selection categories that are available from the IQMNT main menu:

InfoQuest Maintenance Menu Map

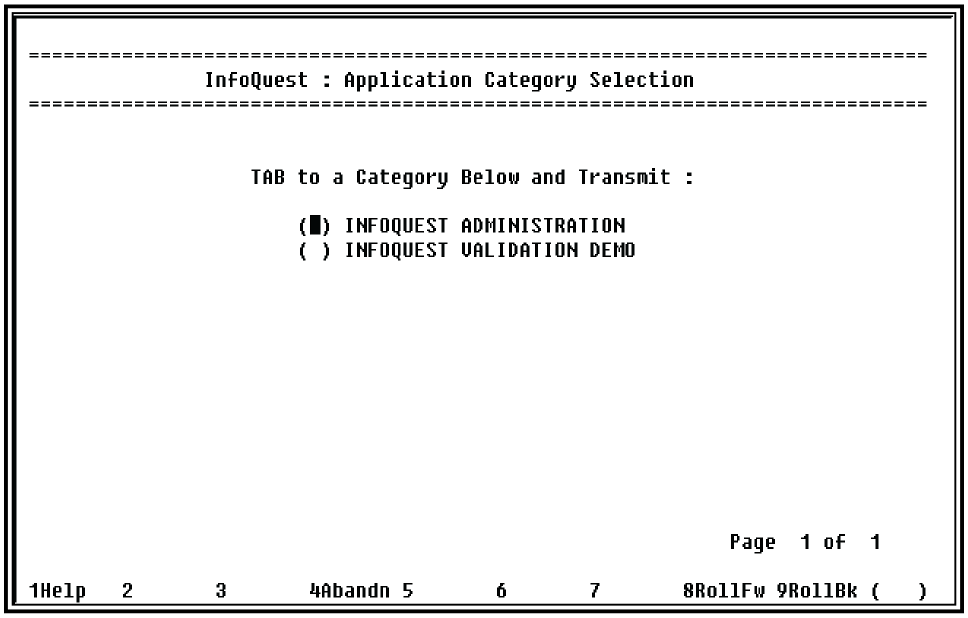


3.2 IQMNT: The Main Menu

To gain entry to the InfoQuest maintenance functions, invoke the IQMNT processor.

Sign on to a DEMAND session and initiate the InfoQuest Maintenance Subsystem by entering, @IQMNT[*x*], where *x* is dependent upon the COMUS mode used to install InfoQuest — @IQMNT for the default mode or @IQMNTA through @IQMNTK for modes INFOQA through INFOQK, respectively.

The first screen will be the Application Category Selection screen. The application is an integral part of the maintenance security system. It limits the IQMNT user to performing maintenance functions in their own application, that is, the user will only be able to generate, modify or delete views, DBMs and requests for the application entered. This restriction by application also applies to subset database generation and auto-scheduling.



```
==============================================================================
               InfoQuest : Application Category Selection
==============================================================================


                   TAB to a Category Below and Transmit :

                       (█) INFOQUEST ADMINISTRATION
                       ( ) INFOQUEST VALIDATION DEMO










                                                             Page  1 of  1

1Help  2      3      4Abandn 5      6      7      8RollFw 9RollBk (   )
```

Each application has an associated application code (see Section 3.4.2, "Application Registration"). Users who are registered in InfoQuest with an application code of all zeros have access to the "InfoQuest Administration" application that allows access to all InfoQuest components (views, DBMs, etc.) — regardless of which application the component is associated. Such users are provided full administrative privileges.

If you are uncertain as to which application should be selected, consult the person responsible for InfoQuest administration on your system.

IQMNT: The Main Menu

The majority of the InfoQuest System User Guide deals with the topics of view generation, view maintenance, DBM generation and automatic path generation. These topics involve two of the selections on the IQMNT menu and are covered in the following chapters of this guide:

Chapter 4, "View Generation and Maintenance"

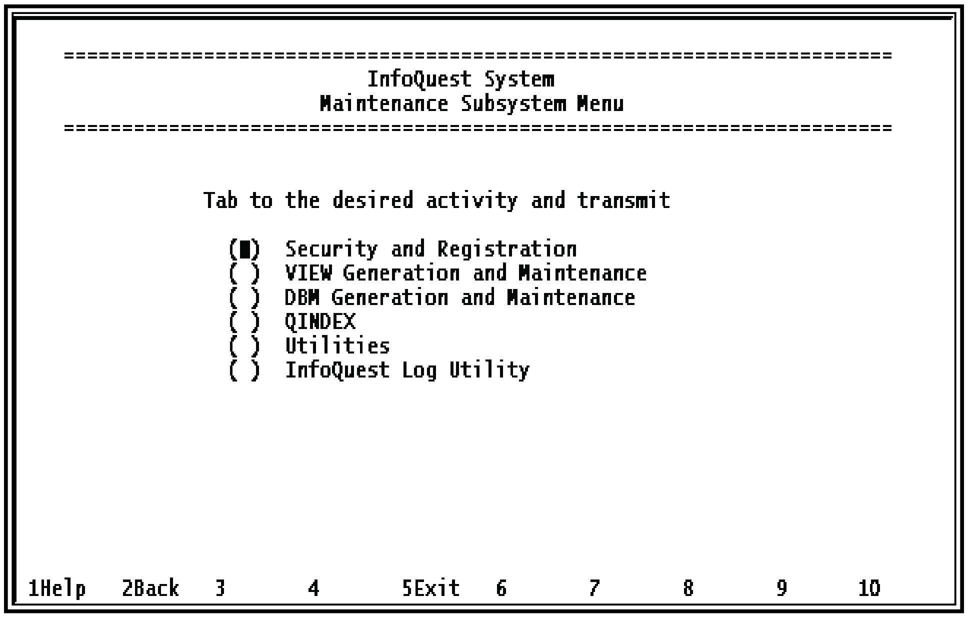
- View Generation
- View Maintenance
- View Global Changes
- Maintaining Occurring Items
- Deleting Views
- Duplicating Views
- View Validation
- View User Prompt Definition
- View Translation Table Creating
- View Copy Utility

Chapter 5, "DBM Generation and Maintenance"

- DBM Generation
- Deleting DBMs
- Duplicating DBMs
- DBM Copy Utility

The remaining InfoQuest maintenance functions are covered in this chapter.

The various maintenance activities are divided into a number of categories as shown on the Maintenance Subsystem Menu (see below). A separate section in this chapter is dedicated to each category (except view and DBM maintenance).



```
======================================================================
                          InfoQuest System
                      Maintenance Subsystem Menu
======================================================================


             Tab to the desired activity and transmit

               (■)  Security and Registration
               ( )  VIEW Generation and Maintenance
               ( )  DBM Generation and Maintenance
               ( )  QINDEX
               ( )  Utilities
               ( )  InfoQuest Log Utility







 1Help   2Back   3       4       5Exit   6       7       8       9       10
```

3.3 IQMNT Function Keys and Screen Paging

InfoQuest has a modified function key line for IQMNT (see the next to the last line on the screen image, above).

Function key usage is generally standardized within IQMNT; e.g., F1 provides on-line help, F2 goes back to the previous step, F3 proceeds to the next step (where applicable) or processes entered information, etc. Many screens provide a menu (see above) where you make a selection (by mouse or keystroke) and press the transmit key to proceed to the desired function.

All maintenance screens that require more than one screen to accommodate their selections provide a paging capability, that is, pages may be reached directly by entering a specific page number instead of scrolling from screen to screen. However, forward and backward scrolling is possible by entering a + or -, respectively, or by using Function Keys 8 and 9. F8 will scroll forward and F9 will scroll backward.

Note: In the interest of making a larger, more readable graphic, function keys are not always shown on the images displayed in this guide. In most instances the function keys are standard and self explanatory.

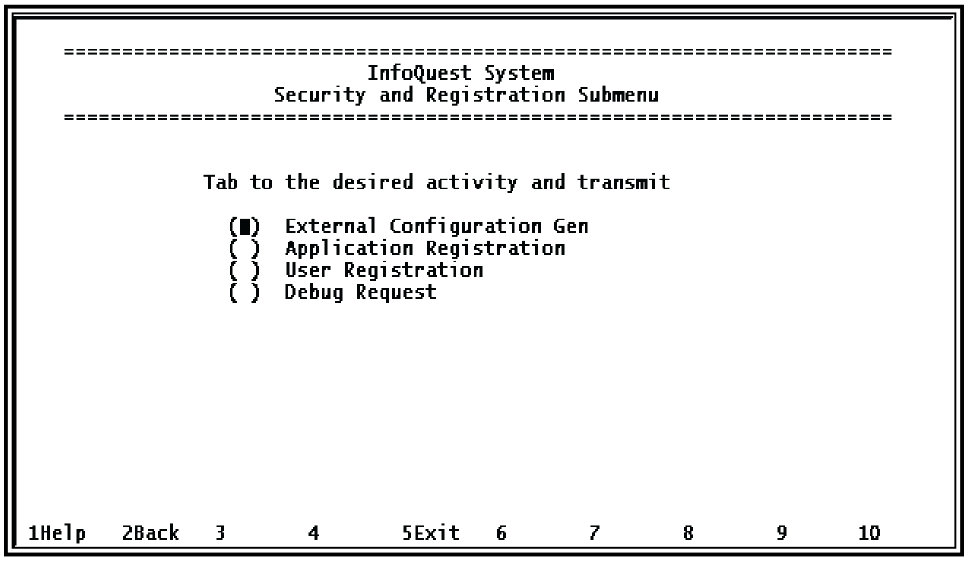
3.4 Security and Registration

When InfoQuest is first installed, there are two areas provided which contain information used for InfoQuest security. One of the areas contains the application code list required for user validation. These codes are used to determine the applications to which a user has access and which maintenance functions the user may perform. The application code list may be changed prior to view/DBM generation, or the defaults may be used as initially installed. The second area contains the user-id list used by InfoQuest to determine which applications and views can be accessed by the end user. There should be one user-id entry placed into this area for each application that a user may access. Each list should be maintained through the use of the IQMNT menu described in this section.

For additional considerations regarding security configurations at your site, refer to Appendix H, “InfoQuest Security”.

There are three activities that can be reached from the “Security and Registration Submenu”: “External Configuration Generation”, “User Registration” and “Application Registration”.

Each of the five selections are shown below and discussed in detail on the following pages:



```
  ======================================================================
                           InfoQuest System
                   Security and Registration Submenu
  ======================================================================


              Tab to the desired activity and transmit

                (■)  External Configuration Gen
                ( )  Application Registration
                ( )  User Registration
                ( )  Debug Request










1Help   2Back   3       4       5Exit   6       7       8       9      10
```

Note: The Security and Registration functions can only be performed by a user that is registered as an InfoQuest Administrator. For other users to be given administrator privileges, they must be registered (see “User Registration” page 3-12) with the pseudo application code of “00” (two zeros).

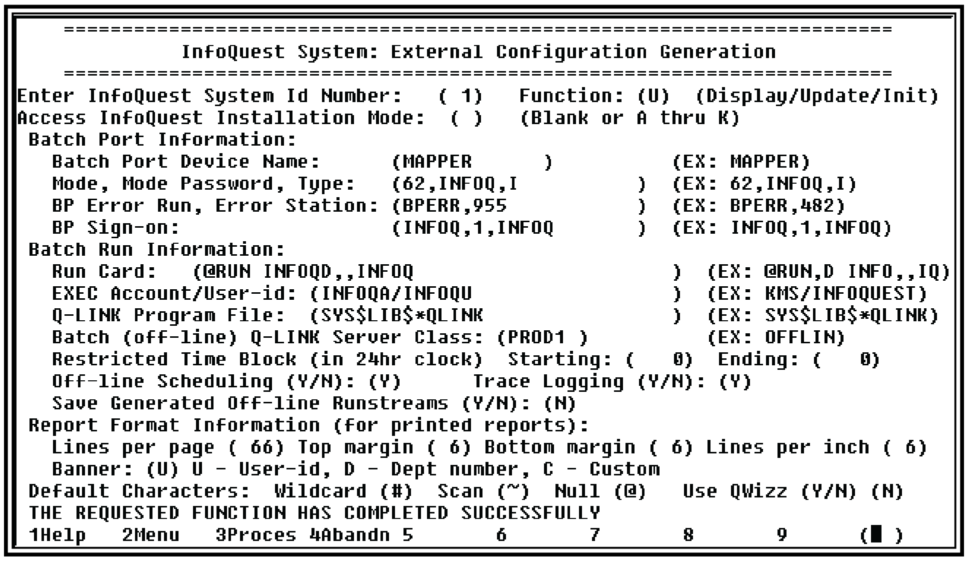
3.4.1 External Configuration Generation

By using the required “External Configuration Generation” function of IQMNT, the InfoQuest Administrator can gain access to the process that sets up the external configuration for InfoQuest. The configurations are primarily used for off-line request processing. A configuration may be established to control which InfoQuest installation will process the parameters, which Q-LINK server class will execute the off-line request, which MAPPER batch port (MAPPER-based InfoQuest only) will handle the request result, etc.

The InfoQuest “System Id” is a unique number associated with a particular InfoQuest system. The minimum requirement is one InfoQuest system id for each InfoQuest system; however, since there can be multiple EXEC account/user-id pairs configured for InfoQuest, there may be multiple InfoQuest system ids for a particular InfoQuest.

The configuration information entered is stored in the COMUS catalogued MSAM file, SYS$LIB$*INFOQ$CONF.

To enter or change a configuration, select “External Configuration Generation” from the “Security and Registration Submenu”. The “External Configuration Generation” screen is shown below:



```
    ======================================================================
              InfoQuest System: External Configuration Generation
    ======================================================================
Enter InfoQuest System Id Number:   ( 1)   Function: (U)  (Display/Update/Init)
Access InfoQuest Installation Mode:  ( )   (Blank or A thru K)
 Batch Port Information:
   Batch Port Device Name:      (MAPPER      )         (EX: MAPPER)
   Mode, Mode Password, Type:   (62,INFOQ,I          )  (EX: 62,INFOQ,I)
   BP Error Run, Error Station: (BPERR,955           )  (EX: BPERR,482)
   BP Sign-on:                  (INFOQ,1,INFOQ       )  (EX: INFOQ,1,INFOQ)
 Batch Run Information:
   Run Card:   (@RUN INFOQD,,INFOQ                      )  (EX: @RUN,D INFO,,IQ)
   EXEC Account/User-id: (INFOQA/INFOQU                 )  (EX: KMS/INFOQUEST)
   Q-LINK Program File:  (SYS$LIB$*QLINK                )  (EX: SYS$LIB$*QLINK)
   Batch (off-line) Q-LINK Server Class: (PROD1 )          (EX: OFFLIN)
   Restricted Time Block (in 24hr clock)  Starting: (   0)  Ending: (   0)
   Off-line Scheduling (Y/N): (Y)      Trace Logging (Y/N): (Y)
   Save Generated Off-line Runstreams (Y/N): (N)
 Report Format Information (for printed reports):
   Lines per page ( 66) Top margin ( 6) Bottom margin ( 6) Lines per inch ( 6)
   Banner: (U) U - User-id, D - Dept number, C - Custom
 Default Characters:  Wildcard (#)  Scan (~)  Null (@)   Use QWizz (Y/N) (N)
 THE REQUESTED FUNCTION HAS COMPLETED SUCCESSFULLY
 1Help   2Menu   3Proces 4Abandn 5       6       7       8       9      (█ )
```

Function “I” should be executed only the very first time InfoQuest is installed on your system. If it is executed a second time, any external configurations in SYS$LIB$*INFOQ$CONF previously configured will be lost.

The “System Id” identifies which InfoQuest configuration to use for each application/user. There is only one configuration file common to all InfoQuest installations on one computer system. An InfoQuest System can have multiple system ids from 1 to 99. You can establish your system ids for multiple Q-LINK, MAPPER and InfoQuest installations; various applications; and even department sign-ons.

For your initial InfoQuest installation, start off simple! First, initialize the configuration file with the "I" function. The first system id installed must be system id "01".

The system id is associated with the files of the InfoQuest installation mode. If the default installation mode was used to install InfoQuest, leave the Access InfoQuest Installation Mode filed blank. If another mode was used to install InfoQuest, enter the appropriate letter (A-K) that corresponds to the COMUS mode selected (INFOQA through INFOQK).

The external configuration program has three main components: parameters required for batch port operations, parameters required for the batch run and parameters required for report formatting on the system printer:

Batch Port Information

MAPPER Batch Port Device Name	Defined in the EXEC generation.
Mode, Mode Password, Type	This is the mode where the MAPPER coordinator placed the InfoQuest installation. You obtain the password for this mode from the coordinator. The Type must be "I"!
BP Error Run, Error Station	"BPERR" is the standard error run that comes with MAPPER. Error station is the station number to which batch port error messages are sent (normally the station number of the technical contact for InfoQuest).
BP Sign-On	InfoQuest comes with a BPORT sign-on (INFOQ) for the F-type in mode 218; check your registration in the F-type RID for this.

Batch Port Information only applies to the MAPPER-based InfoQuest.

Batch Run Information

Run Card	A valid run card image for the batch job.
EXEC Account/User-id	The account/user-id for the batch job.
Q-LINK Program File	The product file where Q-LINK was installed.
Batch Q-LINK Server Class	The server established through Q-LINK dynamic configuration to handle off-line request processing.

Batch Run Information

Restricted Time Block	The hours, between which, off-line requests may not be run. Starting and ending times are specified using the 24 hour clock; e.g., 1:30 p.m. would be entered as “1330”.
Off-line Scheduling	When set to “Y”, generated request runs are saved in the InfoQuest log file and subsequently started by the InfoQuest off-line request monitor. If set to “N”, off-line requests are @STARTed immediately. When set to “Y”, TRACE (see below) must be set to “Y”.
Trace Log	When set to “Y”, log entries are maintained in the InfoQuest log file that records the events of each request run; e.g., start time, finish time, error information, etc.
Save Generated Off-line Runstreams	When an off-line request is generated, the request and its runstream are saved in a file called InfoQuest*infoq#######. Normally, when the off-line request is finished, InfoQuest deletes this file. If this field is set to a “Y”, then the file is not deleted. When this facility is used, trace logging should be run so that the log file can be examined to find the file name associated with the request (see “Trace Log” above and Chapter 8).

Report Format Information

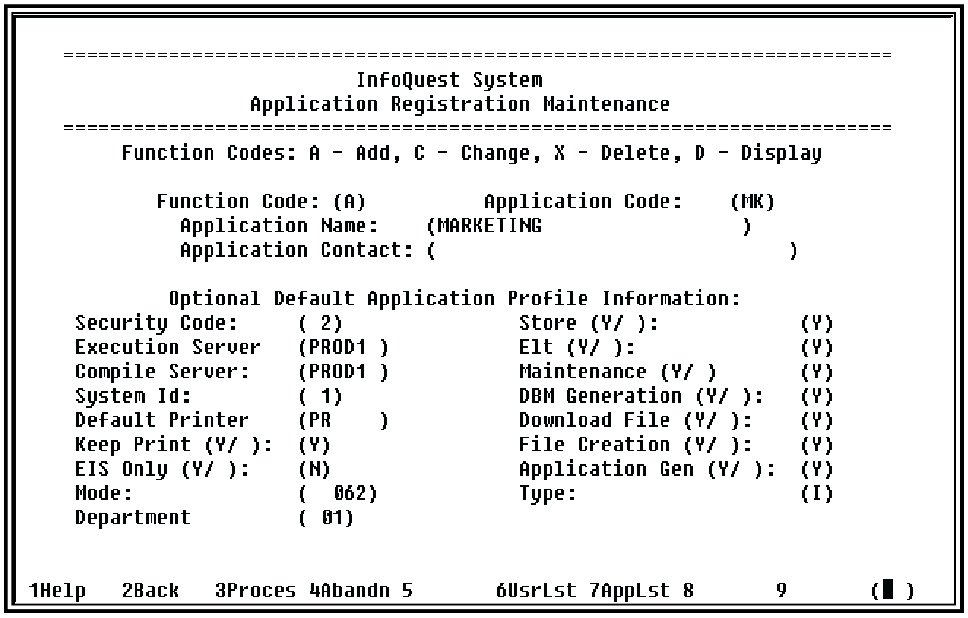
Lines per Page	The actual number of lines per page. For an 11 inch form printing at 6 lines per inch, lines per page would be set to 66 (6 x 11).
Top Margin	The number of blank lines at the top of the page before any printing occurs.
Bottom Margin	The number of blank lines to be reserved at the bottom of the page after the last printed line on the page.

Report Format Information	
Lines per Inch	Used in conjunction with the number of lines per page and the margins to determine the actual number of printed lines per page. For example: An 11 inch form at 6 lines per inch with top and bottom margins of 3 each will yield 60 print lines; i.e., (11 x 6) - 3 - 3 = 60.
Banner U/D	The banner page that will precede the printed report can show the user-id (“U”) or the department number (“D”).
Default Characters	
Wildcard	Sets the character to be used as a wildcard character when the user specifies the search values for a request.
Scan	Sets the character to be used as a scan character when the user specifies the search values for a request.
Null	Sets the character to be used as a null character when the user specifies the search values for a request. Null value searches are only valid for view items generated from RDMS 2200 table columns.
Use QWizz Interface	
Use QWizz	When set to “Y”, InfoQuest will attempt to use the QWizz statistical summary processor when producing summary requests. QWizz is a separately priced product from KMSystems, Inc., and must be installed on your system to use this feature. The default setting is “N”.

3.4.2 Application Registration

Application codes can be entered and modified by selecting “Application Registration” from the “Security and Registration Submenu”.

Any two unique alphanumeric characters (A-Z and 0-9) are valid application codes. The application code is used (along with the security code) when determining the views to which a user may have access; e.g., if the view was generated under application code “MK”, then there must be a user-id registered (see page 3-12) with the same application code for the user to have access to the “MK” application views. The following figures show the addition of a new application code:



```
 ======================================================================
                          InfoQuest System
                  Application Registration Maintenance
 ======================================================================
      Function Codes: A - Add, C - Change, X - Delete, D - Display

         Function Code: (A)          Application Code:    (MK)
           Application Name:    (MARKETING                )
           Application Contact: (                             )

          Optional Default Application Profile Information:
  Security Code:     ( 2)              Store (Y/ ):            (Y)
  Execution Server   (PROD1 )          Elt (Y/ ):              (Y)
  Compile Server:    (PROD1 )          Maintenance (Y/ )       (Y)
  System Id:         ( 1)              DBM Generation (Y/ ):   (Y)
  Default Printer    (PR    )          Download File (Y/ ):    (Y)
  Keep Print (Y/ ):  (Y)               File Creation (Y/ ):    (Y)
  EIS Only (Y/ ):    (N)               Application Gen (Y/ ):  (Y)
  Mode:              (  062)           Type:                   (I)
  Department         ( 01)


1Help   2Back   3Proces 4Abandn 5       6UsrLst 7AppLst 8       9       (█ )
```

If users have been previously registered for an application, you may enter the application code in the first screen and press F6 to get a list of users currently registered. To see a list of existing applications, press F7.

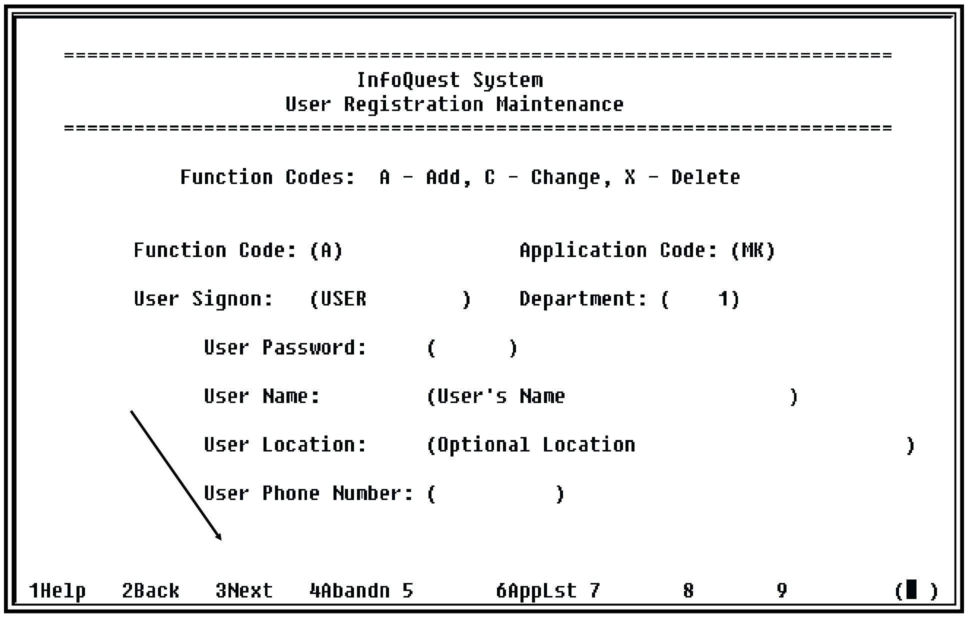
Application codes may be associated with a department, section or other groups within an organizational structure; a project, contract or other divisions of work; a group of individuals such as teams and committees; etc. You may change and/or assign these codes to satisfy the internal processing requirements of your particular organization.

The previous example shows the addition of application code “MK” for a group called “MARKETING”. Application codes can be changed with the “C” function, deleted with the “X” function and displayed with the “D” function.

The parameters in the Original Default Application Profile Information section of the screen can be used to establish default values to be used when registering users for the application. These parameters are optional. If you add them at a later time, they will only be applied to any new users registered, not those already registered. For an explanation of each parameter, see “User Registration”, below.

3.4.3 User Registration

Before an end user can use InfoQuest, a user-id must be registered with InfoQuest. From the IQMNT main menu, select “Security and Registration” followed by “User Registration”. The following figures illustrate the addition of a new user:



```
 ======================================================================
                          InfoQuest System
                    User Registration Maintenance
 ======================================================================

           Function Codes:  A - Add, C - Change, X - Delete


       Function Code: (A)              Application Code: (MK)

       User Signon:   (USER        )    Department: (    1)

             User Password:     (      )

             User Name:         (User's Name                 )

             User Location:     (Optional Location                      )

             User Phone Number: (          )


1Help   2Back   3Next   4Abandn 5       6AppLst 7       8       9        (█ )
```

There should be one user registration entry for each application to which an end user can have access.

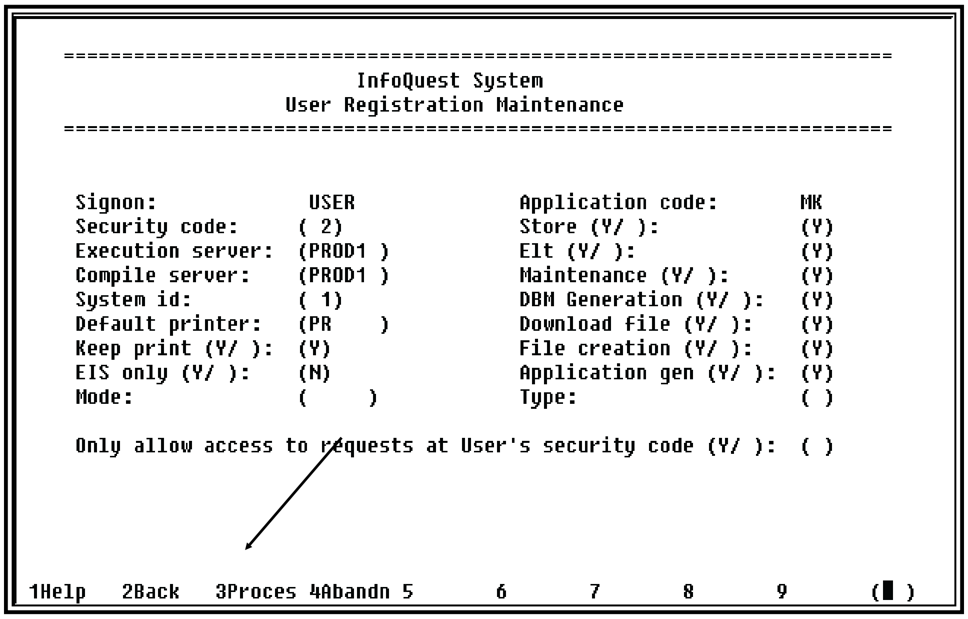
To change information about an existing user or to delete a user from the user registration area, choose the “C” or the “X” function code, respectively. The “D” action can be used to display the settings for an existing user.

The following fields are required:

User Signon	The user-id used by the user being registered.
Application Code	From the list of application codes developed for your site. Note: There should be one user registration entry for each application to which the user may have access.
Department	The Department Number of the user-id.

The remaining fields on this screen are optional — even the User Password. The end user may choose their own password when they first sign on to InfoQuest.

On the second screen, enter or change any of the default profile parameters where appropriate:



```
======================================================================
                        InfoQuest System
                  User Registration Maintenance
======================================================================


 Signon:            USER            Application code:       MK
 Security code:     ( 2)            Store (Y/ ):            (Y)
 Execution server:  (PROD1 )        Elt (Y/ ):              (Y)
 Compile server:    (PROD1 )        Maintenance (Y/ ):      (Y)
 System id:         ( 1)            DBM Generation (Y/ ):   (Y)
 Default printer:   (PR    )        Download file (Y/ ):    (Y)
 Keep print (Y/ ):  (Y)             File creation (Y/ ):    (Y)
 EIS only (Y/ ):    (N)             Application gen (Y/ ):  (Y)
 Mode:              (     )         Type:                   ( )

 Only allow access to requests at User's security code (Y/ ):  ( )




1Help   2Back   3Proces 4Abandn 5       6       7       8       9       (█ )
```

The following fields are required:

Security Code	The security code is used when determining the views to which a user may have access. Valid security codes are "00" (two zeroes) through "99" with "00" being the highest level, but least secure. The user may not have access to views generated with a higher security level. For example, if an entry for the user shows a security code level of "03", then the user may not have access to any view with a security level of "00" through "02". Only a security code of "02" or higher may have access to all InfoQuest maintenance functions. For more information on security, see Appendix H, "InfoQuest Security".
Execution Server	Server class through which the Q-LINK programs (DBMs) will execute.
Compile Server	Server class through which the Q-LINK programs will compile.
System Id	The InfoQuest System Id associated with a particular InfoQuest system. The InfoQuest System Id is assigned during the external configuration selection from the IQMNT menu (see "External System Configuration" page 3-7).

Security and Registration

Mode, Type, Store and Elt only apply to the MAPPER-based InfoQuest, and thus, need not be entered if the user is using InfoQuest/EX or InfoQuest Client.

The following fields are optional. In each case, the default is no (space):

Store	Yes or No (Y/). If yes, allows user to store requests in MAPPER RIDs.
Elt	Yes or No (Y/). If yes, allows user to store requests in DEMAND program files.
Maintenance	Yes or No (Y/). If yes, allows user to perform maintenance functions.
DBM Generation	Yes or No (Y/). If yes, allows user to perform DBM functions of maintenance.
Default Printer	Forces the user's print output to a specific printer-id.
Download Files	Yes or No (Y/). If yes, allows user to download requests to a personal computer (PC). If a user intends to use the "K" option on the "Special Print and Run Options" menu of InfoQuest (the IQ run), this option must be set to "Y".
Keep Print	Yes or No (Y/). If yes, allows user to keep print files on the system.
File Creation	Yes or No (Y/). If yes, allows user to catalog, change or delete EXEC files.
EIS Only	Yes or No (Y/). If yes, used to restrict user to only the Executive Information Systems features of InfoQuest.
Application Generation	Yes or No (Y/). If yes, allows user to create sub-applications from existing applications.
Mode	The mode under which InfoQuest reporting will occur.
Type	The type under which InfoQuest requests will be stored.
Only allow access to requests at User's security code.	Allows you to restrict a user to accessing request (overlay, create from existing or rerun) at **ONLY** their security level. This feature will allow you to compartmentalize users within a specific application. This setting is set on an application basis and is not a global user setting.

Example:

Current sample user registration:

User	App code	Security level
Bob	AA	10
Mary	AA	20
Joe	AA	30

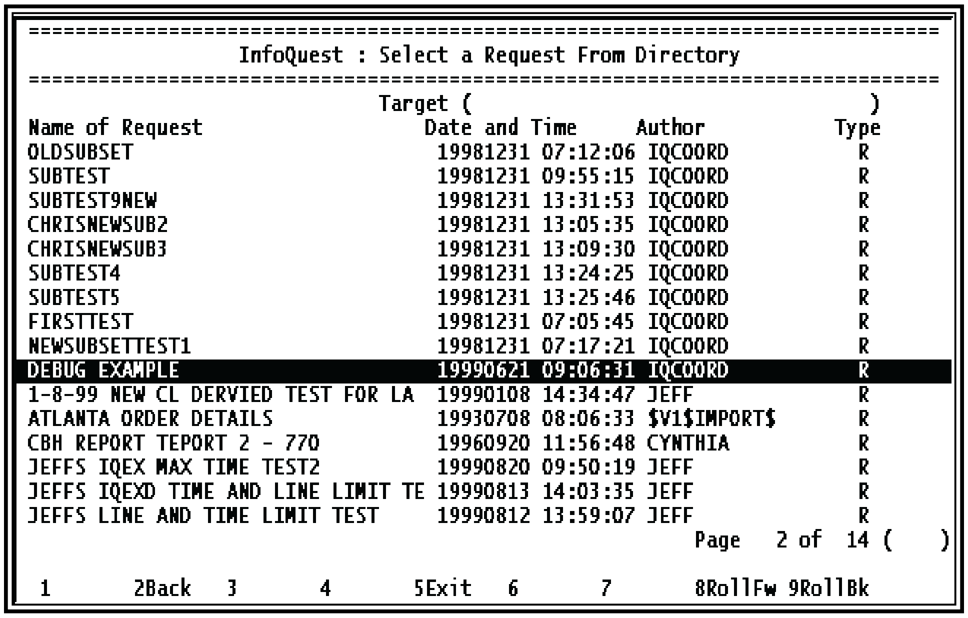
With the current InfoQuest settings, user Bob would be able to see any of Mary's or Joe’s requests, or any requests at a lower level. Mary would be able to see any of Joe’s requests but not any of Bob’s. Joe would only be able to see requests at his level or lower (40, 50, 60, etc.).

With the new user setting, you can restrict Bob to only viewing requests at security level 10, Mary to security level 20 and Joe to security level 30.

3.4.4 Debug Request

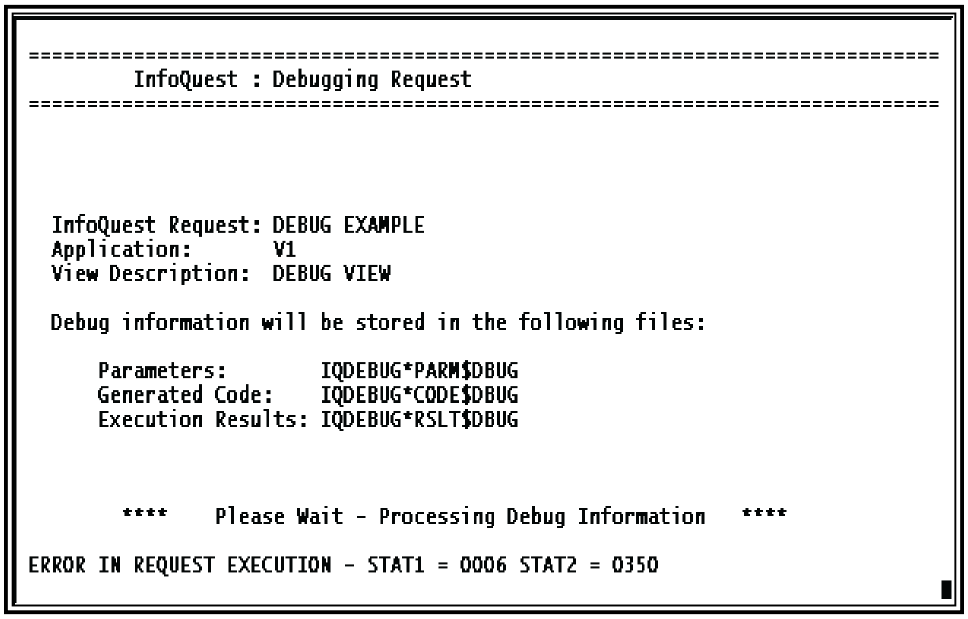
This function alls anyone with InfoQuest Administrator privileges to run any user's request in debug mode. This function gives access to any request created in InfoQuest regardless of whether the request was created in InfoQuest Client32, InfoQuest/MAPER or InfoQuest/EX. This function also ignores any security level or privacy indicator checking. It is for this reason that this function is only allowed for someone with Administrator privileges.

The only selection required is from a list of all requests current being maintained in InfoQuest. Tab to the request to debug and press transmit:



```
==============================================================================
                  InfoQuest : Select a Request From Directory
==============================================================================
                              Target (                                   )
Name of Request                   Date and Time     Author          Type
OLDSUBSET                          19981231 07:12:06 IQCOORD           R
SUBTEST                            19981231 09:55:15 IQCOORD           R
SUBTEST9NEW                        19981231 13:31:53 IQCOORD           R
CHRISNEWSUB2                       19981231 13:05:35 IQCOORD           R
CHRISNEWSUB3                       19981231 13:09:30 IQCOORD           R
SUBTEST4                           19981231 13:24:25 IQCOORD           R
SUBTEST5                           19981231 13:25:46 IQCOORD           R
FIRSTTEST                          19981231 07:05:45 IQCOORD           R
NEWSUBSETTEST1                     19981231 07:17:21 IQCOORD           R
DEBUG EXAMPLE                      19990621 09:06:31 IQCOORD           R
1-8-99 NEW CL DERVIED TEST FOR LA  19990108 14:34:47 JEFF              R
ATLANTA ORDER DETAILS              19930708 08:06:33 $V1$IMPORT$       R
CBH REPORT TEPORT 2 - 770          19960920 11:56:48 CYNTHIA           R
JEFFS IQEX MAX TIME TEST2          19990820 09:50:19 JEFF              R
JEFFS IQEXD TIME AND LINE LIMIT TE 19990813 14:03:35 JEFF              R
JEFFS LINE AND TIME LIMIT TEST     19990812 13:59:07 JEFF              R
                                                     Page   2 of  14 (    )

 1      2Back   3       4       5Exit   6       7       8RollFw 9RollBk
```

The Debugging Request screen shows three cycled files that are created as a result of the code generation and execution process:

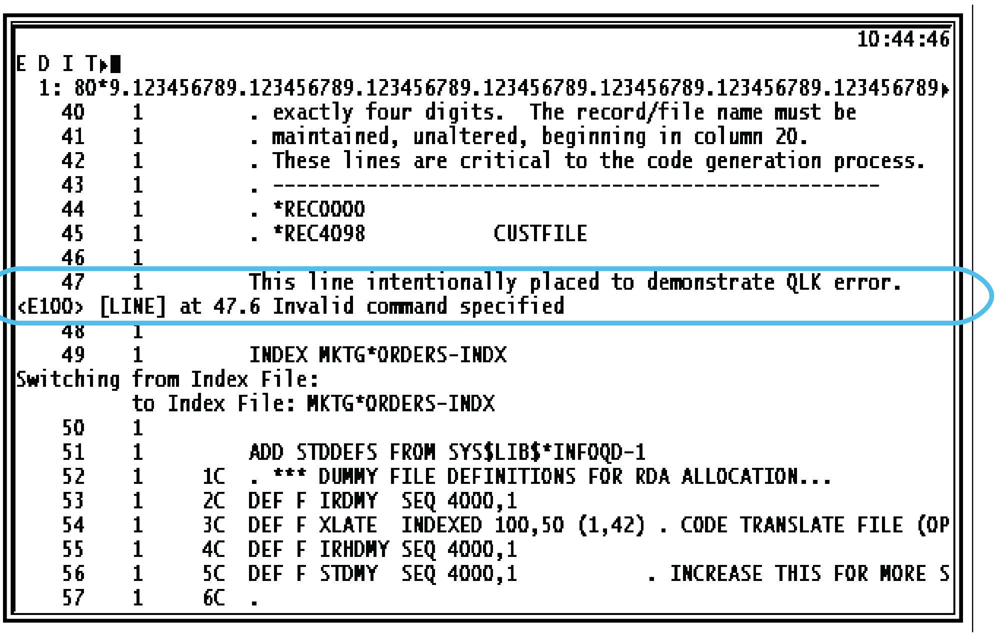


```
==============================================================================
         InfoQuest : Debugging Request
==============================================================================




  InfoQuest Request: DEBUG EXAMPLE
  Application:       V1
  View Description:  DEBUG VIEW

  Debug information will be stored in the following files:

      Parameters:        IQDEBUG*PARM$DBUG
      Generated Code:    IQDEBUG*CODE$DBUG
      Execution Results: IQDEBUG*RSLT$DBUG



        ****    Please Wait - Processing Debug Information   ****

ERROR IN REQUEST EXECUTION - STAT1 = 0006 STAT2 = 0350
```

The three files are described below and may be examined with any editor:

Physical File Name	@USE File Name	Description
IQDEBUG*PARM$DEBUG.	IQPARM.	The InfoQuest parameters and DBM that are input to the code generation process.
IQDEBUG*CODE$DEBUG.	IQCODE.	The generated request.
IQDEBUG*RSLT$DEBUG.	IQRSLT.	The result of the execution of the generated request.

When the debugging process completes, exit IQMAINT and edit the third file, IQRSLT. Notice the error circled below:

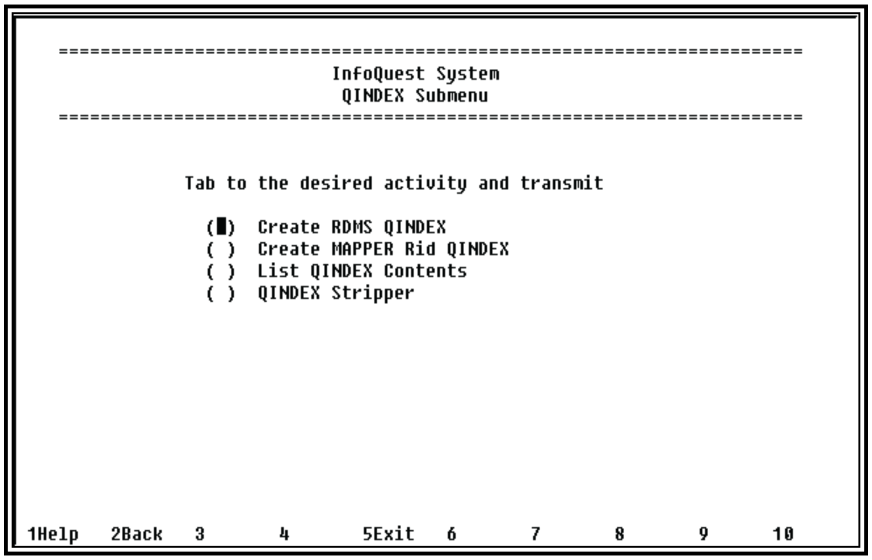


```
                                                                          10:44:46
E D I T▸█
  1: 80*9.123456789.123456789.123456789.123456789.123456789.123456789.123456789▸
    40    1         . exactly four digits.  The record/file name must be
    41    1         . maintained, unaltered, beginning in column 20.
    42    1         . These lines are critical to the code generation process.
    43    1         . ----------------------------------------------------
    44    1         . *REC0000
    45    1         . *REC4098           CUSTFILE
    46    1
    47    1         This line intentionally placed to demonstrate QLK error.
<E100> [LINE] at 47.6 Invalid command specified
    48    1
    49    1         INDEX MKTG*ORDERS-INDX
Switching from Index File:
          to Index File: MKTG*ORDERS-INDX
    50    1
    51    1         ADD STDDEFS FROM SYS$LIB$*INFOQD-1
    52    1     1C  . *** DUMMY FILE DEFINITIONS FOR RDA ALLOCATION...
    53    1     2C  DEF F IRDMY  SEQ 4000,1
    54    1     3C  DEF F XLATE  INDEXED 100,50 (1,42) . CODE TRANSLATE FILE (OP
    55    1     4C  DEF F IRHDMY SEQ 4000,1
    56    1     5C  DEF F STDMY  SEQ 4000,1          . INCREASE THIS FOR MORE S
    57    1     6C  .
```

For more information on debugging, see "Correcting the DBM" in Appendix F on page F-7.

3.5 QINDEX

The QINDEX function, selected from the main IQMNT menu, allows the user to create data item index files for RDMS 2200 tables and MAPPER RIDs. The data item index file must be created prior to creating the view for the user, as this file is input to the view generation process (see Chapter 4, “View Generation and Maintenance”). In addition to creating these new files, existing data item index files can be listed or duplicated including or excluding as many records/files/etc. as desired. The corresponding selections are shown in the figure below:

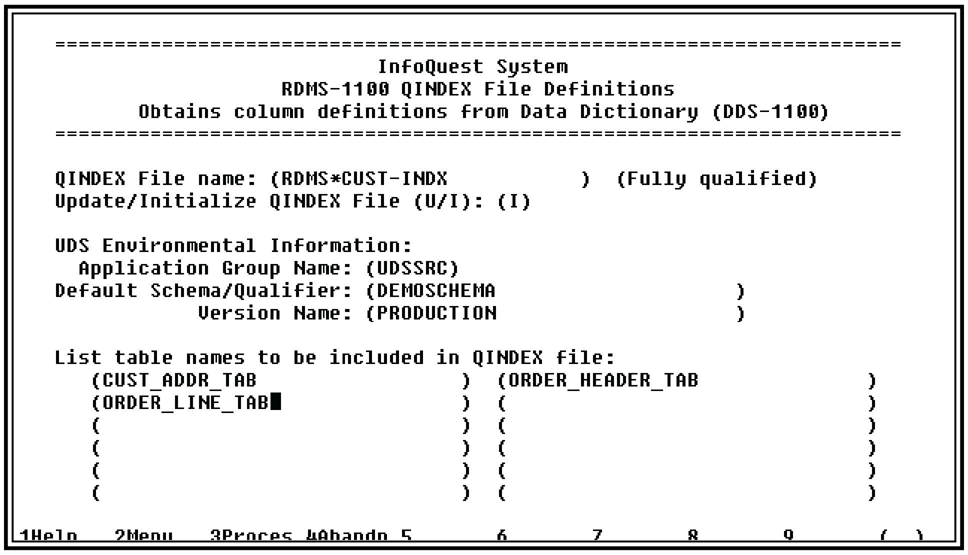


```
======================================================================
                          InfoQuest System
                           QINDEX Submenu
======================================================================


           Tab to the desired activity and transmit

             (█)  Create RDMS QINDEX
             ( )  Create MAPPER Rid QINDEX
             ( )  List QINDEX Contents
             ( )  QINDEX Stripper











1Help   2Back   3       4       5Exit   6       7       8       9       10
```

To create data item index files consisting of DMS 2200 record descriptions or PCIOS file definitions, use the QINDEX processor in OS 2200 DEMAND/BATCH mode (see Appendix A, “QINDEX Reference”, in this guide).

3.5.1 Create RDMS QINDEX

With this function, RDMS 2200 column definitions are obtained from the UREP 2200 data dictionary and used to build data item entries in the specified QINDEX file (see figure below). This file will be used to build the user views as shown in Chapter 4, "View Generation and Maintenance".



```
======================================================================
                          InfoQuest System
                  RDMS-1100 QINDEX File Definitions
       Obtains column definitions from Data Dictionary (DDS-1100)
======================================================================

QINDEX File name: (RDMS*CUST-INDX           )  (Fully qualified)
Update/Initialize QINDEX File (U/I): (I)

UDS Environmental Information:
  Application Group Name: (UDSSRC)
Default Schema/Qualifier: (DEMOSCHEMA                   )
            Version Name: (PRODUCTION                   )

List table names to be included in QINDEX file:
   (CUST_ADDR_TAB                )  (ORDER_HEADER_TAB              )
   (ORDER_LINE_TAB█              )  (                              )
   (                             )  (                              )
   (                             )  (                              )
   (                             )  (                              )
   (                             )  (                              )

1Help   2Menu   3Proces 4Abandn 5       6       7       8       9       (  )
```

The Application Group Name , "UDSSRC", shown in the above example is the default application group for UDS 2200; however, the name must be entered exactly as configured in the EXEC at your site.

The Default Schema/Qualifier and Version Name are needed to uniquely qualify each table entered.

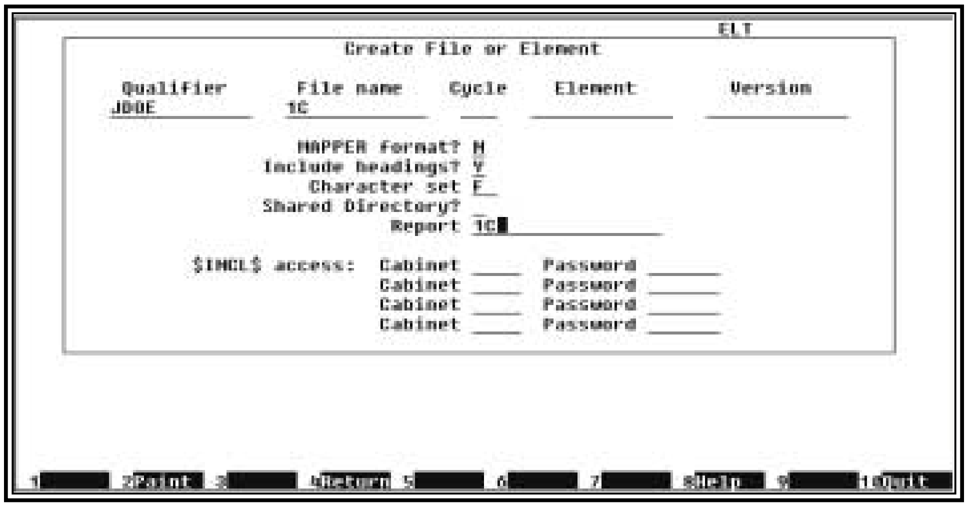
The Version Name, "PRODUCTION", in the above example is the default version for UREP 2200.

A data item reference will be created for every column defined in the named table(s).

All table names must be entered exactly as defined in the data dictionary (UREP 2200).

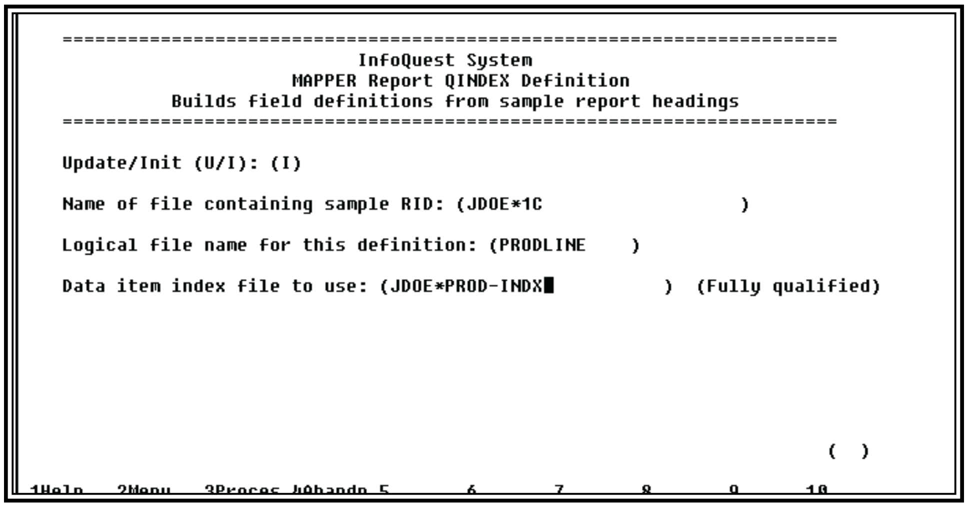
3.5.2 Create MAPPER RID QINDEX

This function creates a data item index file from the column headings defined for a RID. This file will be used to build the user views as shown in Chapter 4, “View Generation and Maintenance”. A data item definition will be created for each column heading in the RID. The following example shows a data item index file being created for RID 1C in mode 0, the JDOE mode. The RID must first be placed in a data file with the ELT function in MAPPER. Be sure to include the headings.



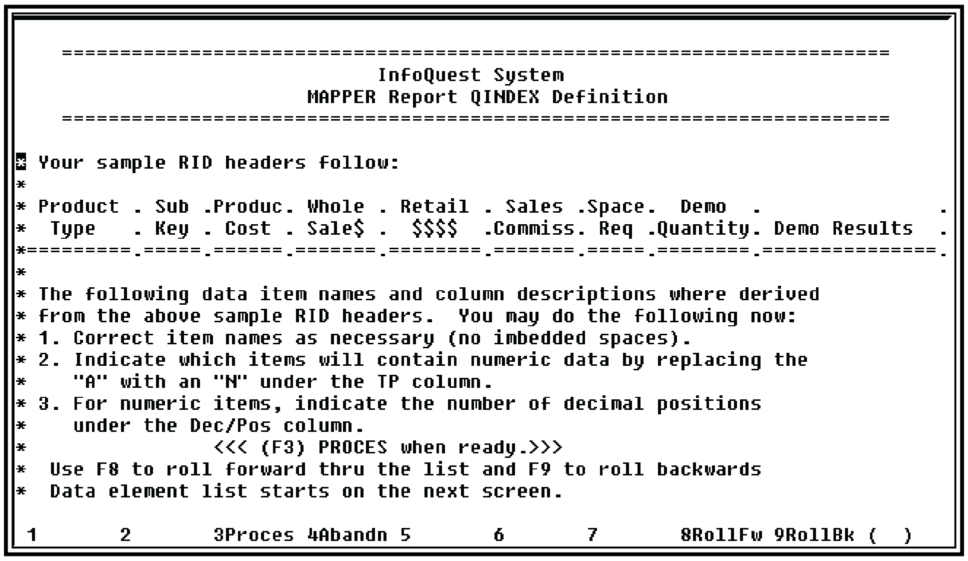
```
                                                              ELT
                          Create File or Element

   Qualifier       File name    Cycle     Element        Version
  JDOE            1C

                  MAPPER format? M
                Include headings? Y
                   Character set E
               Shared Directory?
                          Report 1C

          $INCL$ access:  Cabinet       Password
                          Cabinet       Password
                          Cabinet       Password
                          Cabinet       Password

1       2Point  3       4Return 5       6       7       8Help   9       10Quit
```

The data file is then referenced by the “Name of file containing sample RID” parameter in IQMNT.



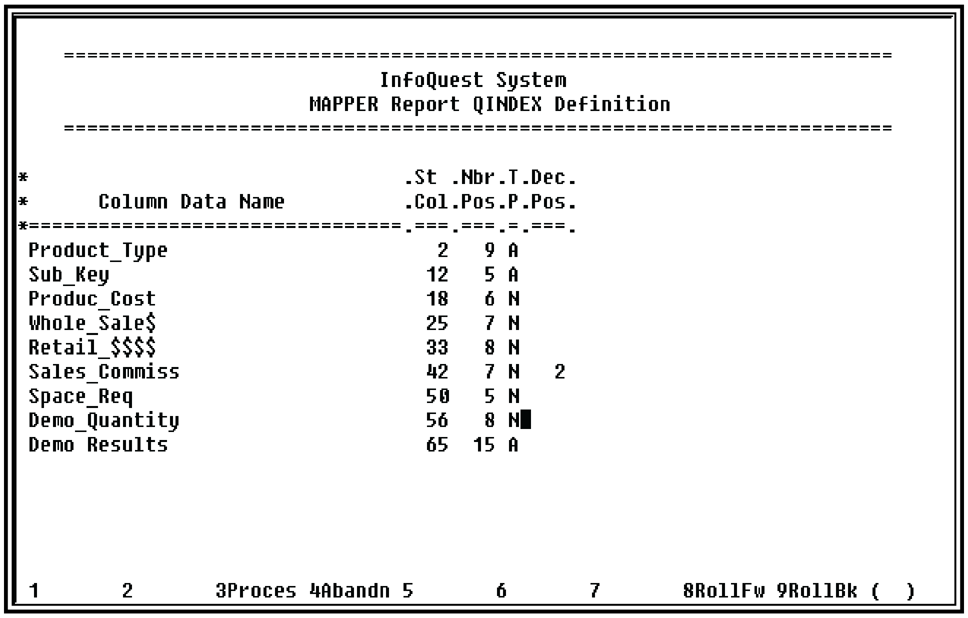
```
=======================================================================
                            InfoQuest System
                      MAPPER Report QINDEX Definition
             Builds field definitions from sample report headings
=======================================================================

Update/Init (U/I): (I)

Name of file containing sample RID: (JDOE*1C                  )

Logical file name for this definition: (PRODLINE    )

Data item index file to use: (JDOE*PROD-INDX          )  (Fully qualified)



                                                                   (  )
1Help   2Menu   3Proces 4Abandn 5       6       7       8       9       10
```

After pressing F3, the result screen is returned where several of the items are altered to indicate that they contain numeric data.



```
   ======================================================================
                              InfoQuest System
                        MAPPER Report QINDEX Definition
   ======================================================================

* Your sample RID headers follow:
*
* Product . Sub .Produc. Whole . Retail . Sales .Space.  Demo  .               .
*  Type   . Key . Cost . Sale$ .  $$$$  .Commiss. Req .Quantity. Demo Results  .
*=========.=====.======.=======.========.=======.=====.========.===============.
*
* The following data item names and column descriptions where derived
* from the above sample RID headers.  You may do the following now:
* 1. Correct item names as necessary (no imbedded spaces).
* 2. Indicate which items will contain numeric data by replacing the
*    "A" with an "N" under the TP column.
* 3. For numeric items, indicate the number of decimal positions
*    under the Dec/Pos column.
*               <<< (F3) PROCES when ready.>>>
*  Use F8 to roll forward thru the list and F9 to roll backwards
*  Data element list starts on the next screen.

 1       2       3Proces 4Abandn 5       6       7       8RollFw 9RollBk (  )
```

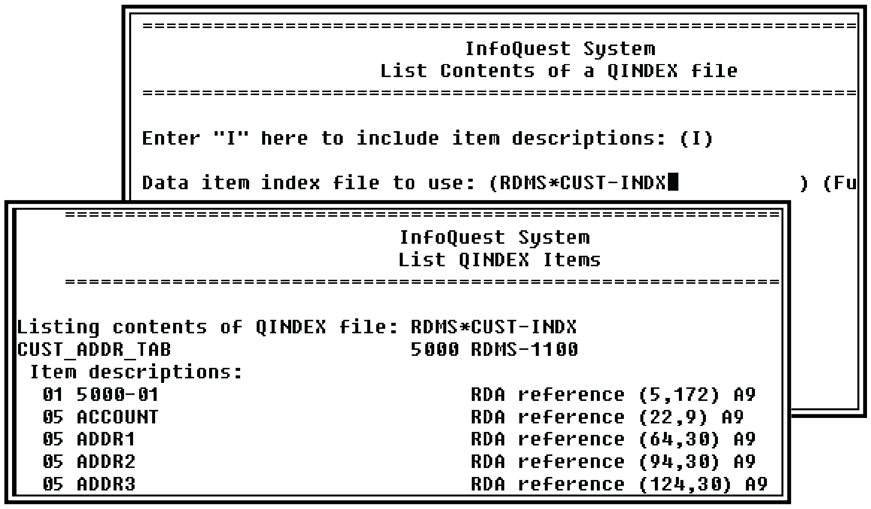
An “N” is placed in the “TP” column to indicate numeric, and a “2” in the “Dec Pos” column to allow for two decimal positions where appropriate.



```
   ======================================================================
                              InfoQuest System
                        MAPPER Report QINDEX Definition
   ======================================================================

*                              .St .Nbr.T.Dec.
*      Column Data Name        .Col.Pos.P.Pos.
*==============================.===.===.=.===.
 Product_Type                     2   9 A
 Sub_Key                         12   5 A
 Produc_Cost                     18   6 N
 Whole_Sale$                     25   7 N
 Retail_$$$$                     33   8 N
 Sales_Commiss                   42   7 N   2
 Space_Req                       50   5 N
 Demo_Quantity                   56   8 N
 Demo Results                    65  15 A




 1       2       3Proces 4Abandn 5       6       7       8RollFw 9RollBk (  )
```

3.5.3 List QINDEX Contents

This function can be used to display the contents of any data item index file. Only the record/file/table names can be listed, or, with the optional “I” option, each of the field definitions may be listed.

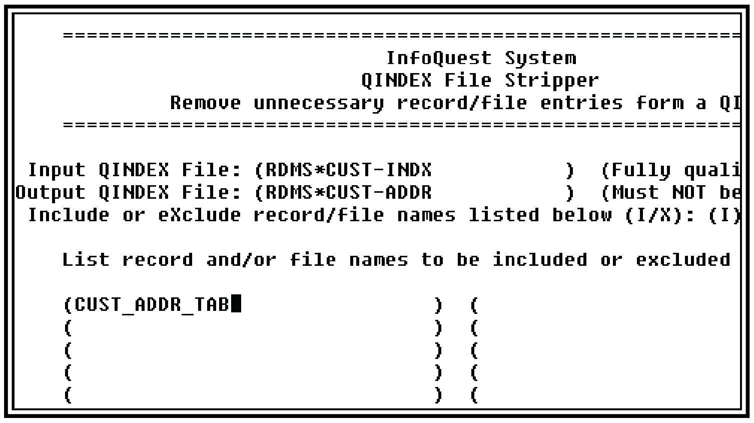


```
=============================================================
                         InfoQuest System
                  List Contents of a QINDEX file
=============================================================

Enter "I" here to include item descriptions: (I)

Data item index file to use: (RDMS*CUST-INDX█          ) (Fu
```

```
    ============================================================
                              InfoQuest System
                              List QINDEX Items
    ============================================================

Listing contents of QINDEX file: RDMS*CUST-INDX
CUST_ADDR_TAB                    5000 RDMS-1100
 Item descriptions:
  01 5000-01                          RDA reference (5,172) A9
  05 ACCOUNT                          RDA reference (22,9) A9
  05 ADDR1                            RDA reference (64,30) A9
  05 ADDR2                            RDA reference (94,30) A9
  05 ADDR3                            RDA reference (124,30) A9
```

3.5.4 QINDEX Stripper

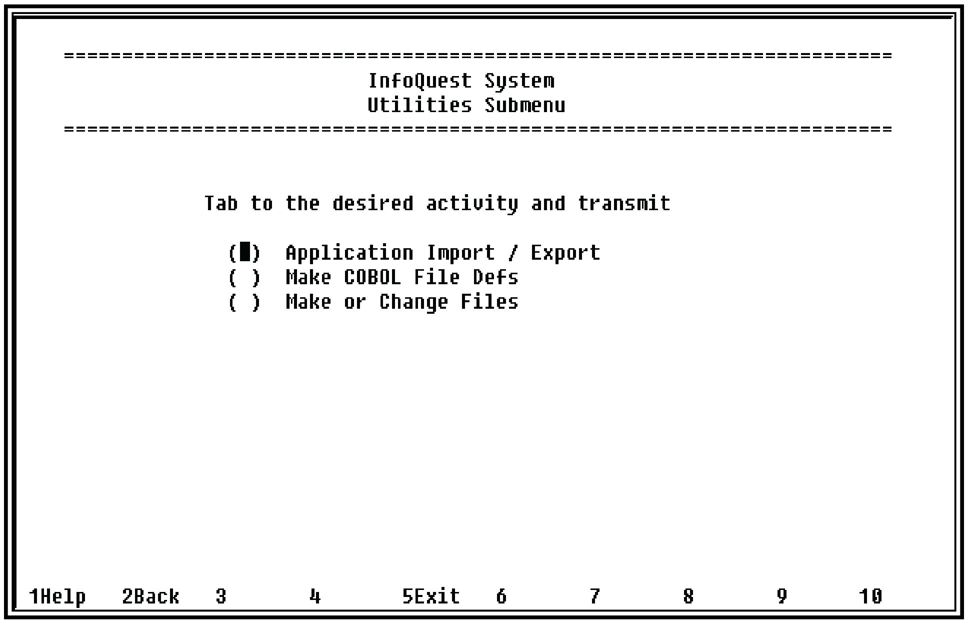
The QINDEX Stripper function can be used to duplicate the data item index file. Record/file definitions can be included (I) or excluded (X) as desired. Both QINDEX file specifications must be fully qualified. If the output QINDEX file does not exist, it will be cataloged.



```
   ==========================================================
                              InfoQuest System
                            QINDEX File Stripper
              Remove unnecessary record/file entries form a QI
   ==========================================================

 Input QINDEX File: (RDMS*CUST-INDX           )  (Fully quali
Output QINDEX File: (RDMS*CUST-ADDR           )  (Must NOT be
 Include or eXclude record/file names listed below (I/X): (I)

   List record and/or file names to be included or excluded

   (CUST_ADDR_TAB█               )  (
   (                             )  (
   (                             )  (
   (                             )  (
   (                             )  (
```

3.6 Utilities

The Utilities menu provides a number of useful routines for the individual needing to maintain the InfoQuest environment. These routines include the ability to import/export entire applications, make COBOL file definitions and maintain EXEC files.



```
======================================================================
                          InfoQuest System
                          Utilities Submenu
======================================================================


           Tab to the desired activity and transmit

             (█)  Application Import / Export
             ( )  Make COBOL File Defs
             ( )  Make or Change Files










1Help   2Back   3       4       5Exit   6       7       8       9       10
```

3.6.1 Application Import / Export

The Application Import/Export utility allows entire InfoQuest applications to be exported from one InfoQuest installation mode and subsequently imported to a second installation mode. This utility can be used to transport an application developed in one InfoQuest (e.g., a test version) to another InfoQuest (e.g., a production version), or even another computer system.

The export function moves all components of a specific application to a specified EXEC program file. The components include all views, DBMs and, optionally, any pre-written requests associated with the specified 2-character application code.

Each application to be exported must have its own unique file; i.e., only one application is allowed per file. The export function uses the names of the views and requests to build the element names in the program file.

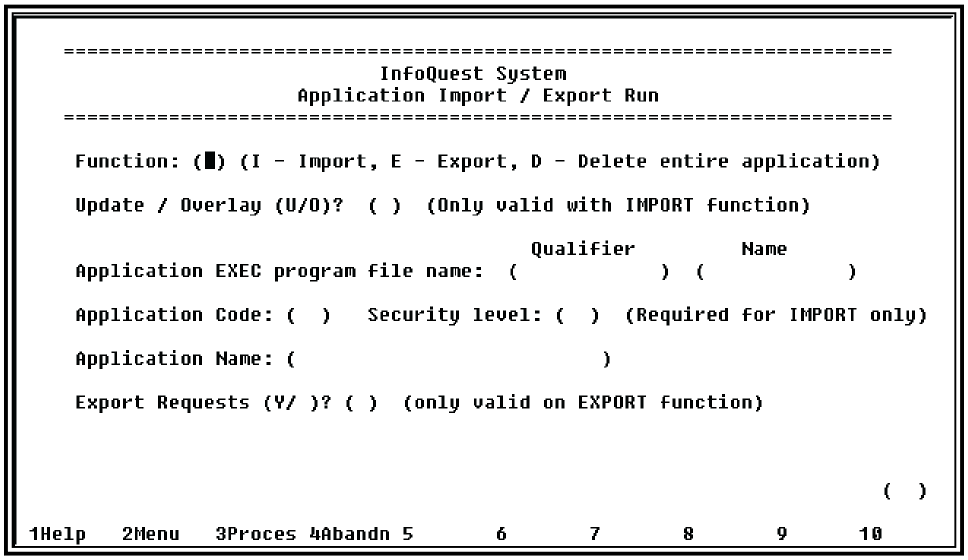
If errors occur during export, you will be notified of the permanent location where the error list is stored.

The import function is used to retrieve all components of an application from an EXEC program file created by the export function. The import function will register the application code, views and DBMs and pre-written requests in all appropriate InfoQuest indexes. User access must be manually established. The application must not exist prior to running the import except when using the update or overlay function. The import will also check to insure view names are not currently registered under a different application.

The delete function will remove all components, including registration and index entries of a specified application. The delete function may be used prior to importing an application which already exists in the InfoQuest installation (replacing an application). This function may also be used to delete partially imported applications where the import failed for some reason.

The import, export and delete functions do not affect data or QINDEX files in any way. These files must be transported manually with the EXEC program file containing the InfoQuest components.

The following screen is used for both the export and import functions:



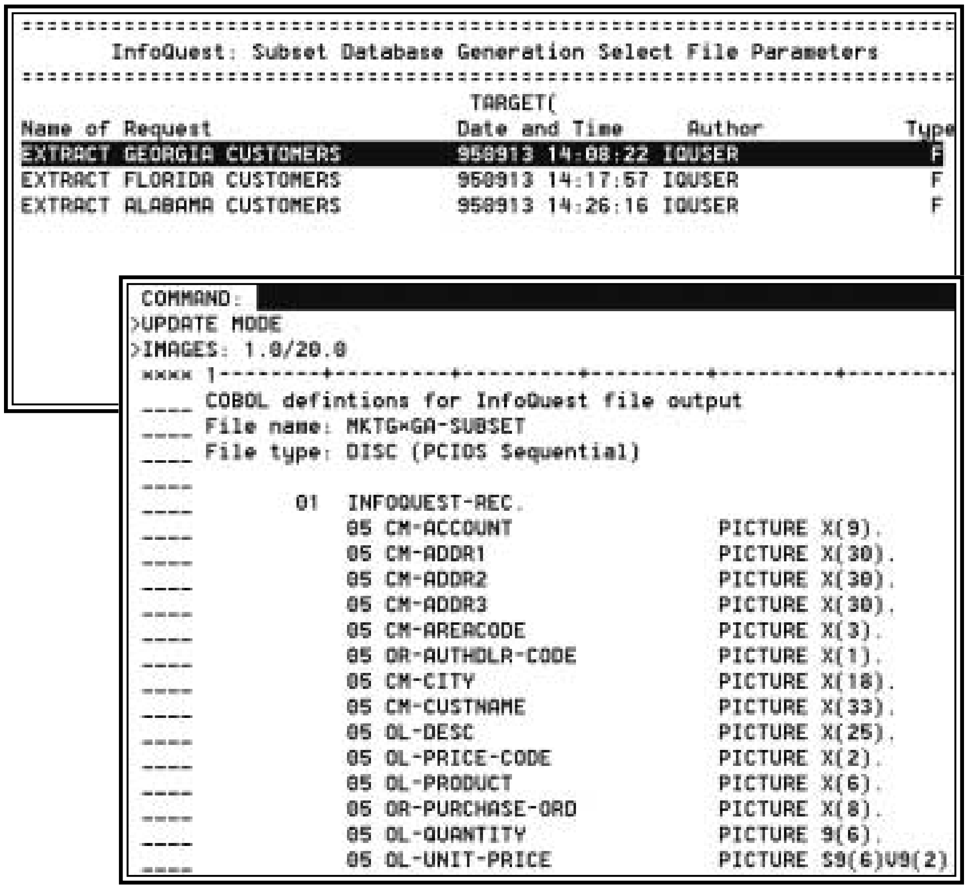
```
  ===================================================================
                          InfoQuest System
                   Application Import / Export Run
  ===================================================================

   Function: (█) (I - Import, E - Export, D - Delete entire application)

   Update / Overlay (U/O)?  ( )  (Only valid with IMPORT function)

                                       Qualifier        Name
   Application EXEC program file name:  (            )  (            )

   Application Code: (  )   Security level: (  )  (Required for IMPORT only)

   Application Name: (                         )

   Export Requests (Y/ )? ( )  (only valid on EXPORT function)



                                                                    (  )
 1Help   2Menu   3Proces 4Abandn 5       6       7       8       9      10
```

When using the Import function, a security level must be chosen. Also, to use this function, your user-id must be registered as an InfoQuest Administrator (see Section 3.4.2, "Application Registration").

When importing, the Update and Overlay options allow you to update existing applications with Views, DBMs and requests that were created in other InfoQuest systems. The Update option will only add Views, DBMs and requests that are not currently in the selected application. The Overlay option works in the same way as the Update option; however, the Overlay option will also ask if you want to overlay existing Views, DBMs and requests with ones that are being imported.

3.6.2 Make COBOL File Defs

This utility will create a COBOL file definition for a file that was created by InfoQuest. The utility run reads the parameters used to create the file and creates the file layout. This layout is then displayed to you on the screen. This layout can then be placed in a program file element and used in a COBOL program to read the file. To create a file layout, tab to the desired set of parameters and transmit.



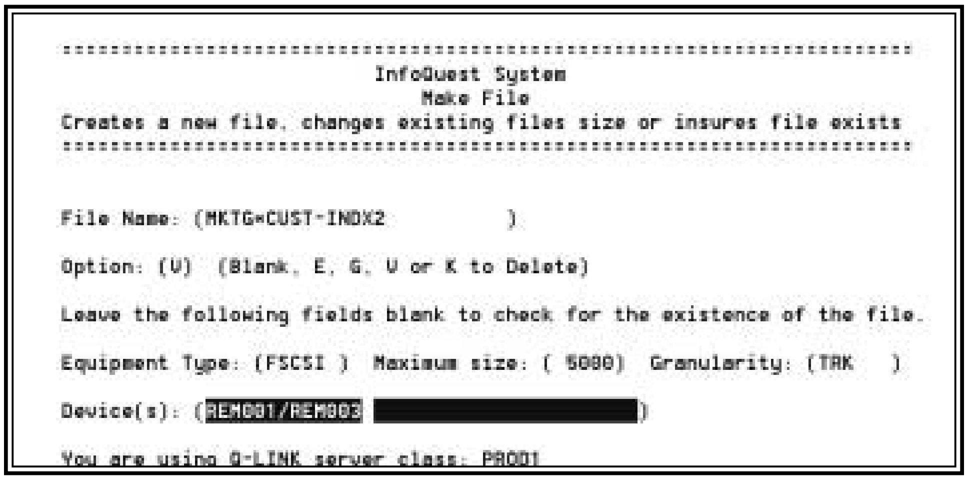
```
=====================================================================
       InfoQuest: Subset Database Generation Select File Parameters
=====================================================================
                                   TARGET(
Name of Request                   Date and Time     Author          Type
EXTRACT GEORGIA CUSTOMERS         950913 14:08:22 IQUSER               F
EXTRACT FLORIDA CUSTOMERS         950913 14:17:57 IQUSER               F
EXTRACT ALABAMA CUSTOMERS         950913 14:26:16 IQUSER               F
```

```
 COMMAND:
>UPDATE MODE
>IMAGES: 1.0/20.0
 **** 1--------+---------+---------+---------+---------+---------+--------
 ____ COBOL defintions for InfoQuest file output
 ____ File name: MKTG*GA-SUBSET
 ____ File type: DISC (PCIOS Sequential)
 ____
 ____        01  INFOQUEST-REC.
 ____            05 CM-ACCOUNT               PICTURE X(9).
 ____            05 CM-ADDR1                 PICTURE X(30).
 ____            05 CM-ADDR2                 PICTURE X(30).
 ____            05 CM-ADDR3                 PICTURE X(30).
 ____            05 CM-AREACODE              PICTURE X(3).
 ____            05 OR-AUTHDLR-CODE          PICTURE X(1).
 ____            05 CM-CITY                  PICTURE X(18).
 ____            05 CM-CUSTNAME              PICTURE X(33).
 ____            05 OL-DESC                  PICTURE X(25).
 ____            05 OL-PRICE-CODE            PICTURE X(2).
 ____            05 OL-PRODUCT               PICTURE X(6).
 ____            05 OR-PURCHASE-ORD          PICTURE X(8).
 ____            05 OL-QUANTITY              PICTURE 9(6).
 ____            05 OL-UNIT-PRICE            PICTURE S9(6)V9(2)
 ____
```

The COBOL definitions are placed in "*user-id**COBDEF(+1)." where *user-id* is the user-id of the user as registered in InfoQuest. Also, notice that the plus one cycle of the file is always used.

3.6.3 Make or Change File

This function provides a means to create, change or delete EXEC files from within the InfoQuest environment. There is no need to open to a separate demand session in order to accomplish EXEC file maintenance. This function is only allowed to those users whose registration has the “File Creation” switch set to “Y” .



```
=================================================================
                        InfoQuest System
                           Make File
Creates a new file, changes existing files size or insures file exists
=================================================================


File Name: (MKTG*CUST-INDX2          )

Option: (V)  (Blank, E, G, V or K to Delete)

Leave the following fields blank to check for the existence of the file.

Equipment Type: (FSCSI )  Maximum size: ( 5000)  Granularity: (TRK    )

Device(s): (REM001/REM003                      )

You are using Q-LINK server class: PROD1
```

The “File name” must be a fully qualified file name as in the above example.

The “Option” field can be one of the following values:

blank	Cataloged as a public file.
E	Check for the existence of the file.
G	Do not backup the file.
V	Ensure the file is never unloaded to tape.
K	Delete the file.

The “Equipment type” is dependent upon the mass-storage devices installed at your site. Please consult the person responsible for your site’s administration and the Unisys OS 2200 ECL Programming Reference Manual for a list of valid equipment types.

”Maximum size” is the maximum number of tracks or positions the file may occupy. A track is 64 sectors. A position is 64 tracks.

“Granularity” may be blank or “TRK” for tracks or “POS” for positions. The default is tracks.

“Device(s)” is a device or list of devices upon which the file may be stored. Please consult the person responsible for your site’s administration for a list of valid device mnemonics at your site. Multiple devices should be separated by slashes (/).

3.7 Log Utility

The Log Utility gives the InfoQuest administrator access to the lock and trace log files which are maintained by the Off-line Request Trace/Scheduling feature of InfoQuest. This feature is an integral part of the InfoQuest environment when managing off-line requests or implementing an Executive Information System with InfoQuest (see Chapter 8, “Off-line Request Trace/Scheduler” in this guide). The utility provides the following functions:

1. List status of log trace file and retrieve current status counts (S function).
2. List status of log trace file and view trace log entries (L function).
3. Combine both of the two above functions (B function).

   Note: All entries and counts may be viewed or only those from a selected date and time.
4. Delete, or Kill, catalogued print files no longer needed (K function);
5. Purge by date/time (P function) to eliminate trace log entries and delete all associated files (off-line runstreams and print files) up to the select date entered.

All Log Utility functions, except retrieve current status counts, provide multiple selections by status code(s), user-id, department and print file name. For the purge function, the additional selection criteria, date and time, are required.

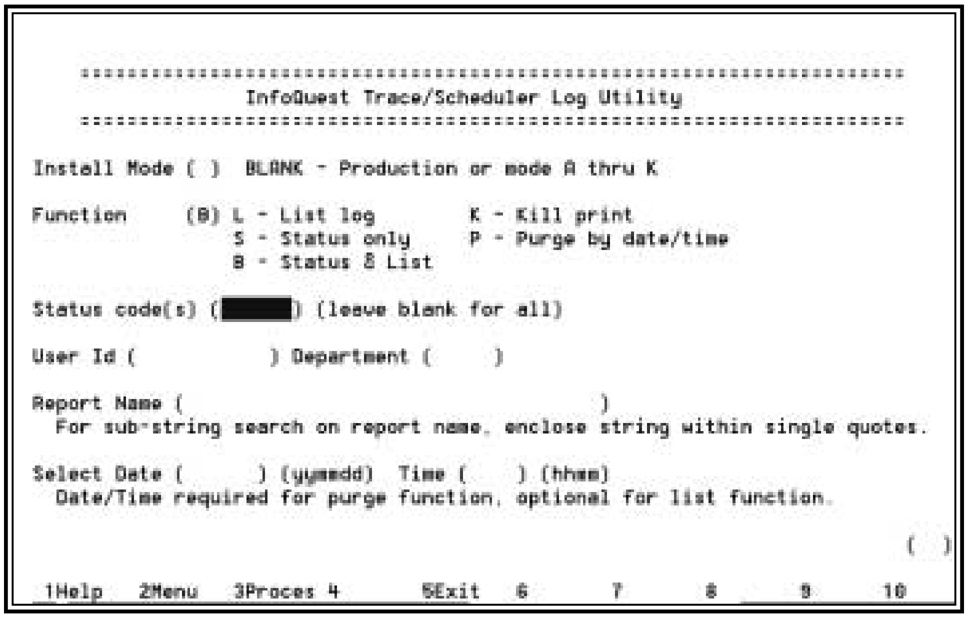
The valid status codes are:

B	Run built and ready to be started by the IQMON monitor program;
E	Run is in execution;
F	Run finished normally;
H	Run held due to the maximum limit;
L	Run lost but restartable (was in execution, but no longer running);
Q	Run @STARTed;
R	Run was errored off;
T	Run @START failed;
X	Run lost and not restartable (run file is deleted).

When using request name for selection, a search for a particular string match within the request name may be invoked by enclosing the search value within single quotes (‘). If the request name is not quoted, an exact match will be assumed.

All print files created by off-line requests will be printed, then deleted (unless the user chose to save the print file on the host) upon completion. If for any reason a completed print file does not print, it can be found by searching the trace log and then manually retrieved or printed.

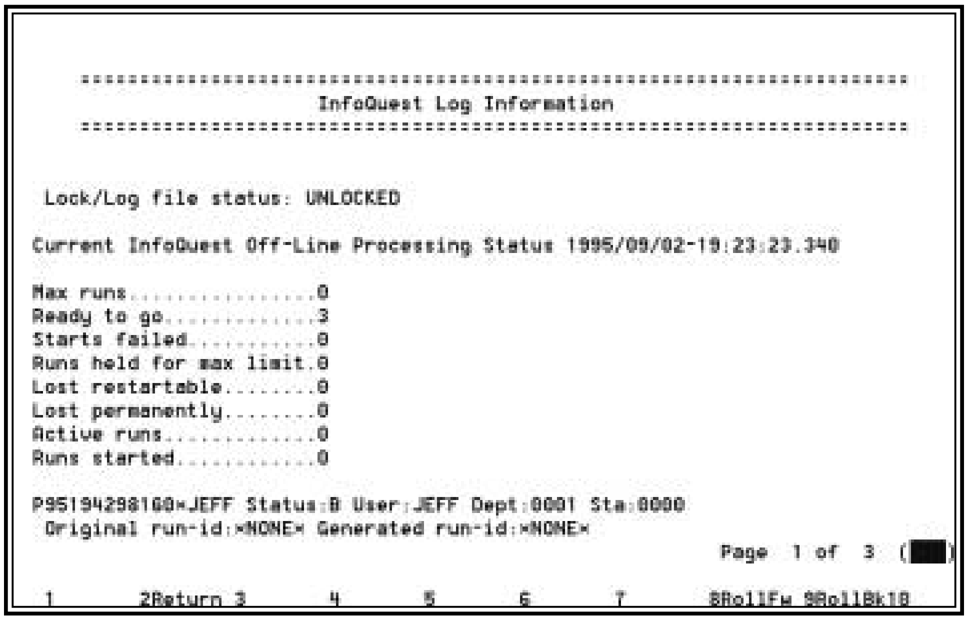
The following example shows the B function being used to display the current status and trace log entries:



```
   ======================================================================
                 InfoQuest Trace/Scheduler Log Utility
   ======================================================================

 Install Mode ( )  BLANK - Production or mode A thru K

 Function     (B) L - List log        K - Kill print
                  S - Status only     P - Purge by date/time
                  B - Status & List

 Status code(s) (      ) (leave blank for all)

 User Id (           ) Department (     )

 Report Name (                                  )
   For sub-string search on report name, enclose string within single quotes.

 Select Date (      ) (yymmdd)  Time (    ) (hhmm)
   Date/Time required for purge function, optional for list function.

                                                                         ( )
  1Help   2Menu   3Proces 4       5Exit   6       7       8       9       10
```

Log Utility

The example shows no active off-line request currently in execution; however, three (3) are “Ready to go”. The last entry in the trace log shows that the request for JEFF has a status of F (run finished normally). Notice that the last line of the log entry shows the date/time stamps when the run was built (B status) and @STARTed (Q status) by the IQMON monitor program. For more information on off-line request trace/scheduling, see Chapter 8, “Off-line Request Trace/Scheduler” in this guide.



```
    ======================================================================
                          InfoQuest Log Information
    ======================================================================


 Lock/Log file status: UNLOCKED

Current InfoQuest Off-Line Processing Status 1995/09/02-19:23:23.340

Max runs.................0
Ready to go..............3
Starts failed............0
Runs held for max limit.0
Lost restartable........0
Lost permanently........0
Active runs.............0
Runs started............0

P95194298100*JEFF Status:B User:JEFF Dept:0001 Sta:0000
 Original run-id:*NONE* Generated run-id:*NONE*
                                                     Page  1 of  3  (   )

 1       2Return 3       4       5       6       7       8RollFw 9RollBk10
```

Chapter 4 View Generation and Maintenance

4.1 View: Definition

An InfoQuest View contains all the data items that the end user will have available when generating InfoQuest reports. A view defines a portion of an application that will be visible to the end user. There can be many views associated with one application. A view can contain all the data items that are included in a DMS1100 subschema plus data items from PCIOS files that are defined in COBOL copy procedures. On the other hand, a view can be a subset of a DMS1100 subschema by including only those records that are required in order to produce the reports needed by the end user. Views can contain data items whose names are derived from the column names in RDMS 1100 tables or MAPPER RIDs. Furthermore, data items that reside in DMS1100 files, PCIOS records or any other file system may be excluded from the view by tailoring the view once it has been produced.

A part of the tailoring process allows for the renaming of data items in order to provide data names that are more meaningful to the end user. This feature is particularly useful when a site uses structured programming standards that require each data item to be prefixed with a structured code that would be meaningless to the end user.

When a view is generated, access to the view can be restricted by supplying a security code level higher than the security code given to the user as registered in the user registration table (see Section 3.4.3, “User Registration”, and Appendix H,“InfoQuest Security”).

4.2 Selecting View Generation and Maintenance

An IQMNT function that allows for the generation of the base view, rebuilding (replacing all data items) an existing view, and updating (integrate additional data items) an existing view is provided with the release of InfoQuest. This function is the “View Generation” function which is selected from the “View Generation and Maintenance” menu of IQMNT. Once the view has been generated it can be tailored to the particular needs of the end user with the “View Global Change” and “View Maintenance” functions. Other functions are also available for: "View Maintenance - Occurring Items", "View Deletions", "View Duplication", "View Validation", "View User Prompt Definition", "View Translation Table Creation" and "View Copy".

4.3 View: Preparation

At some point prior to selecting the VIEW Generation function it will be necessary to sign on to demand mode and create a data item index file if the view is to contain data items from DMS 1100 records or PCIOS files. In this case, the data item index file is created through the use of the supplied utility, QINDEX.

For views which will contain data items from RDMS 1100 tables or MAPPER RIDs, there are separate IQMNT functions for this purpose (see Section 3.5, "QINDEX").

The data item index file can contain all definitions for an entire application. Several views can be generated from a single data item index file.

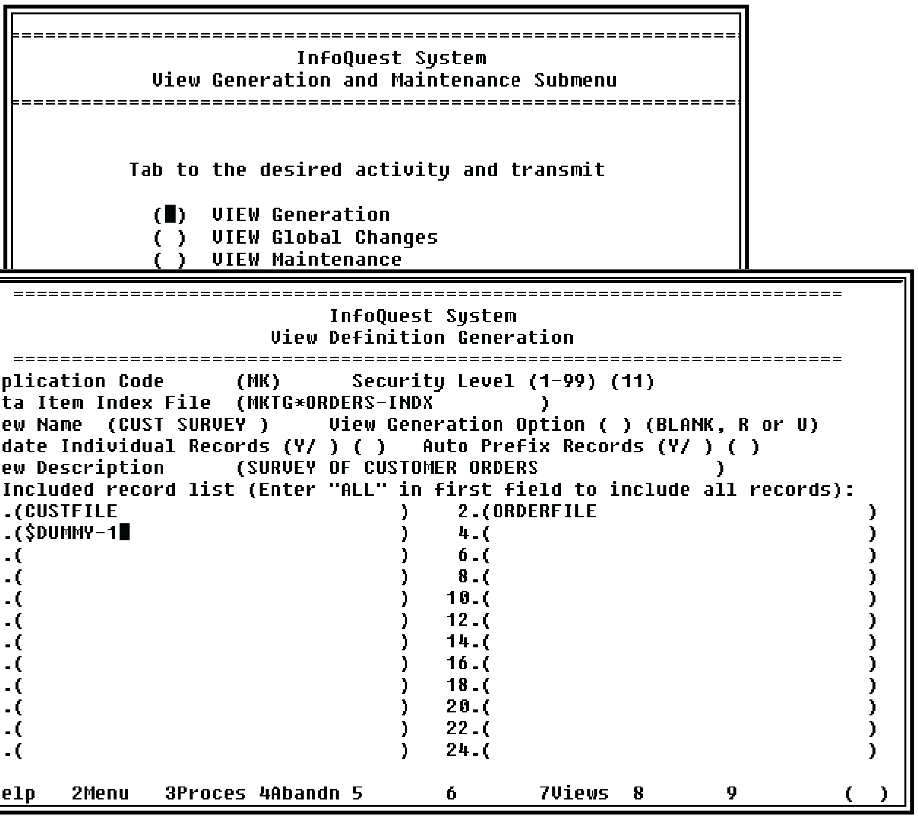
The following example of an execution of QINDEX will build the data item index file needed to generate the view. The example invokes only a subschema and does not reference any PCIOS file definitions. For other examples of QINDEX file creation, see Chapter 2, "A Brief Walk-through of IQMNT", and Appendix A, "QINDEX Reference".

The QINDEX processor call name will depend upon how it was installed on your system. The possible values are: QINDEX, QINDXA, QINDXB ... QINDXK.

```
>@asg,up dms*co-index.
>I:002333 ASG complete.
>@qindex,i dms*co-index.
>QINDEX 6-20 (Release 6-20) (990610 0634:06) 1999 Jun 14 Mon 0853:40
>(C) Copyright 1985-1997 by KMSystems, Inc.  All Rights reserved.
>This program licensed for use by KMSYSTEMS, INC.
>File:DMS*CO-INDEX. already assigned
>***Index will be INITIALIZED.
>invoke demosub in demosch file uds$$src*schabs
>@eof
>.....End of QINDEX processing.
```

4.4 View Generation

The following figures illustrate the basic steps to a view generation. The first screen that will appear after selecting "VIEW Generation" from the View Generation and Maintenance Submenu is the View Definition Generation menu.



```
===========================================================
                      InfoQuest System
            View Generation and Maintenance Submenu
===========================================================


         Tab to the desired activity and transmit

           (█)  VIEW Generation
           ( )  VIEW Global Changes
           ( )  VIEW Maintenance
```

```
   ===================================================================
                             InfoQuest System
                        View Definition Generation
   ===================================================================
Application Code      (MK)      Security Level (1-99) (11)
Data Item Index File  (MKTG*ORDERS-INDX         )
View Name  (CUST SURVEY )     View Generation Option ( ) (BLANK, R or U)
Update Individual Records (Y/ ) ( )   Auto Prefix Records (Y/ ) ( )
View Description      (SURVEY OF CUSTOMER ORDERS             )
  Included record list (Enter "ALL" in first field to include all records):
 1.(CUSTFILE                    )    2.(ORDERFILE                   )
 3.($DUMMY-1█                   )    4.(                            )
 5.(                            )    6.(                            )
 7.(                            )    8.(                            )
 9.(                            )   10.(                            )
11.(                            )   12.(                            )
13.(                            )   14.(                            )
15.(                            )   16.(                            )
17.(                            )   18.(                            )
19.(                            )   20.(                            )
21.(                            )   22.(                            )
23.(                            )   24.(                            )

1Help   2Menu   3Proces 4Abandn 5       6       7Views  8       9        (  )
```

Application Code and Security Level:

The Application Code and Security Level are required fields and must have been previously entered with the IQMNT Security and Registration Sections 3.4.2, "Application Registration", and 3.4.3, "User Registration", discussed in the previous chapter (also see, Appendix H, "InfoQuest Security"). For users to have access to the view, the Security Level must be equal to or greater than that entered for registered users. Also, each user must be registered for the application entered.

Data Item Index File:

The Data Item Index File field is also required and is the name of a QINDEX built file as discussed earlier.

View Generation

The Record List must be selected from the record/file/table/etc. included in the data item index file when it was originally built. To include all records/files in the data item index file, enter "ALL" as the first (only) record name.

View Name:

The View Name must be unique as InfoQuest maintenance uses this name to maintain the InfoQuest tables.

If you plan to use the export function of IQMNT, the slash (/) should not be used when naming views as it will cause an error during export.

View Generation Option:

The View Generation Option should remain blank when generating new views as in the above example.

When the View Generation Option is set to a "U", the View Generation function will update or integrate additional data items into an existing view without altering data items previously included/changed in the view. For example, the update option can be used to restore data items that have been previously deleted from the view or when new data items are being added to a record in a schema or PCIOS file.

When the View Generation Option is set to an "R", the view generation will regenerate any data items previously defined/changed in the view. This option is used when changes have been made to the file or subschema which affect codes in the view tables. Such changes might be adding or deleting records from a subschema or file, changing the data item lengths listed in the view or changing the data type of items listed in the view.

The only fields changed when using the regenerate option are: internal character lengths, decimal positions and data type. Customized user names, output lengths and edit masks will remain unaffected.

In all cases, only data items from the named records will be included. Therefore, if a view is updated or rebuilt (U or R), the records and/or file names specified for the original view must be re-entered unless they are to be omitted from the view.

Update Individual Records:

If it is desirable to update or regenerate a view for selected records without effecting data items from other records in the view, the "Update Individual Records" option may be used. By entering a "Y" in the Individual Records field, view generation will only update the items for those records that are entered on the screen. The "ALL" option for record inclusion must not be used.

Auto Prefix Records:

If set to “Y”, this option will automatically prefix each itemname in the record with the InfoQuest generated record code. This option is useful when there are duplicate item names between records and you do not want to manually change the duplicate occurences.

View Description:

The View Description is the name that will appear on the view list when the user selects the InfoQuest application.

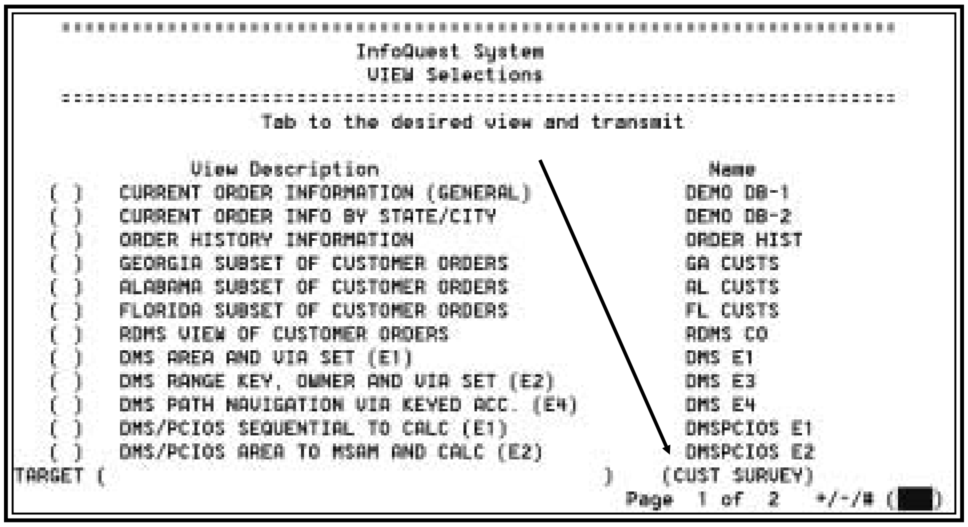
4.5 View Global Changes

Another function that can be selected from the View Generation and Maintenance menu is the “View Global Change” selection. A naming convention commonly used for data items in file and record definitions calls for each data item to use a record/file prefix; e.g., “AM-” for the ACCOUNT-MASTER record. Such a convention can be beneficial to the programmer or data processing organization but meaningless when viewed by the end user. The View Global Change function can be used to strip these prefixes and also replace all dashes (-) with spaces.

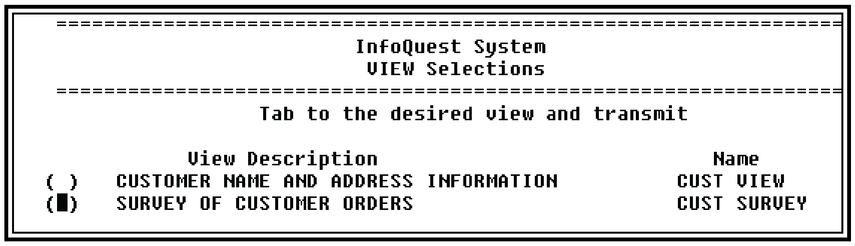
First select the view you wish to alter by tabbing to the appropriate entry and transmitting. If the desired view is not visible on the screen, you may scroll forward or backward by using function keys 8 and 9, respectively, or select a specific page by entering a specific page number over the plus sign (+) at the bottom of the screen and transmitting. Another option is to use one of the target fields to search for the view.

View Global Changes

When using a target field, it is not necessary to enter the entire target value. A partial value may be entered for view description or view name. The following example illustrates a target search on view name:



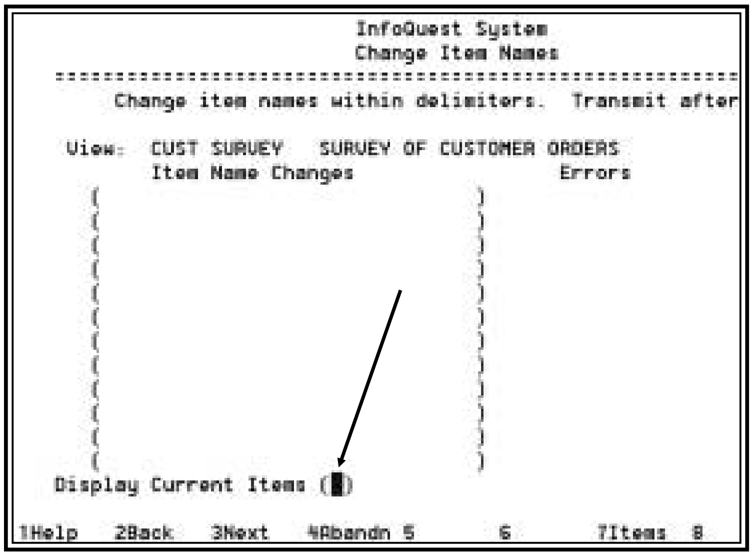
```
 *********************************************************************
                          InfoQuest System
                           VIEW Selections
 =====================================================================
                   Tab to the desired view and transmit

            View Description                                 Name
  ( )   CURRENT ORDER INFORMATION (GENERAL)               DEMO DB-1
  ( )   CURRENT ORDER INFO BY STATE/CITY                  DEMO DB-2
  ( )   ORDER HISTORY INFORMATION                         ORDER HIST
  ( )   GEORGIA SUBSET OF CUSTOMER ORDERS                 GA CUSTS
  ( )   ALABAMA SUBSET OF CUSTOMER ORDERS                 AL CUSTS
  ( )   FLORIDA SUBSET OF CUSTOMER ORDERS                 FL CUSTS
  ( )   RDMS VIEW OF CUSTOMER ORDERS                      RDMS CO
  ( )   DMS AREA AND VIA SET (E1)                         DMS E1
  ( )   DMS RANGE KEY, OWNER AND VIA SET (E2)             DMS E3
  ( )   DMS PATH NAVIGATION VIA KEYED ACC. (E4)           DMS E4
  ( )   DMS/PCIOS SEQUENTIAL TO CALC (E1)                 DMSPCIOS E1
  ( )   DMS/PCIOS AREA TO MSAM AND CALC (E2)              DMSPCIOS E2
TARGET (                                         )   (CUST SURVEY)
                                             Page  1 of  2   +/-/# (   )
```



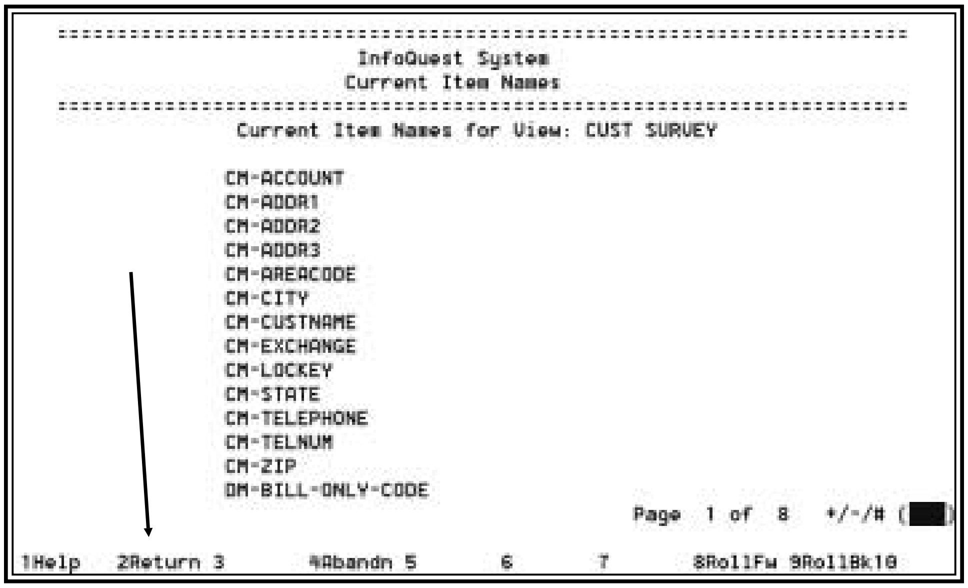
```
 ==================================================================
                    InfoQuest System
                     VIEW Selections
 ==================================================================
             Tab to the desired view and transmit

          View Description                           Name
 ( )   CUSTOMER NAME AND ADDRESS INFORMATION        CUST VIEW
 (█)   SURVEY OF CUSTOMER ORDERS                    CUST SURVEY
```

In the figures above, the “CUST SURVEY” view was selected and the Global Change screen displayed. If you are uncertain which global changes might be required, tab to

"Display Current Items" and transmit as was done in the following example, or use function key F7. F2 will return you to the global change screen once you have finished scrolling through the list of items.



```
                          InfoQuest System
                          Change Item Names
  ======================================================
      Change item names within delimiters.  Transmit after

   View:  CUST SURVEY   SURVEY OF CUSTOMER ORDERS
          Item Name Changes                 Errors
     (                              )
     (                              )
     (                              )
     (                              )
     (                              )
     (                              )
     (                              )
     (                              )
     (                              )
     (                              )
     (                              )
     (                              )
  Display Current Items ( )

1Help   2Back   3Next   4Abandn 5       6       7Items  8
```



```
  =======================================================================
                          InfoQuest System
                         Current Item Names
  =======================================================================
                 Current Item Names for View: CUST SURVEY

                 CM-ACCOUNT
                 CM-ADDR1
                 CM-ADDR2
                 CM-ADDR3
                 CM-AREACODE
                 CM-CITY
                 CM-CUSTNAME
                 CM-EXCHANGE
                 CM-LOCKEY
                 CM-STATE
                 CM-TELEPHONE
                 CM-TELNUM
                 CM-ZIP
                 DM-BILL-ONLY-CODE
                                               Page  1 of  8   +/-/# (   )

1Help   2Return 3       4Abandn 5       6       7       8RollFw 9RollBk10
```

View Global Changes

The “Item Names” will be used to construct the column headers when the user generates reports. InfoQuest allows for three lines when constructing these column headers. A space between two words in an item name causes InfoQuest to place the second word on the next line. For example, the item name, “NEW LOAN PAYMENT CODE”, would appear on the report as:

```
.NEW        .
.LOAN       .
.PAYMENT    .
```

Notice that the word, “CODE”, does not appear since InfoQuest only allows for a three-line header.

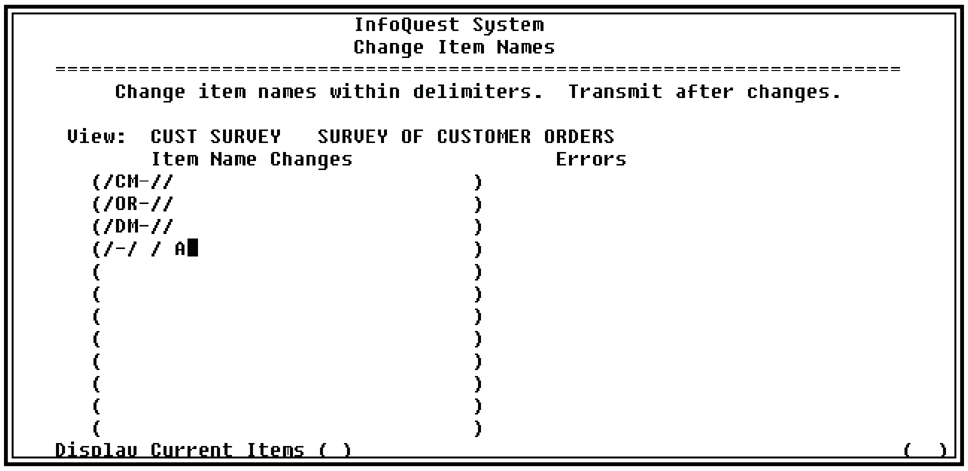
When constructing column headers, InfoQuest interprets two special characters as follows:

1) An underscore (_) keeps words together on the same line of the header but displays as a space.
2) A slash (/) in the heading causes all remaining characters to be ignored.

For example, the item names, “NEW_LOAN PAYMENT CODE” and “LOAN AMOUNT/J Doe” would appear on the report as:

```
.NEW LOAN   .LOAN       .
.PAYMENT    .AMOUNT     .
.CODE       .           .
```

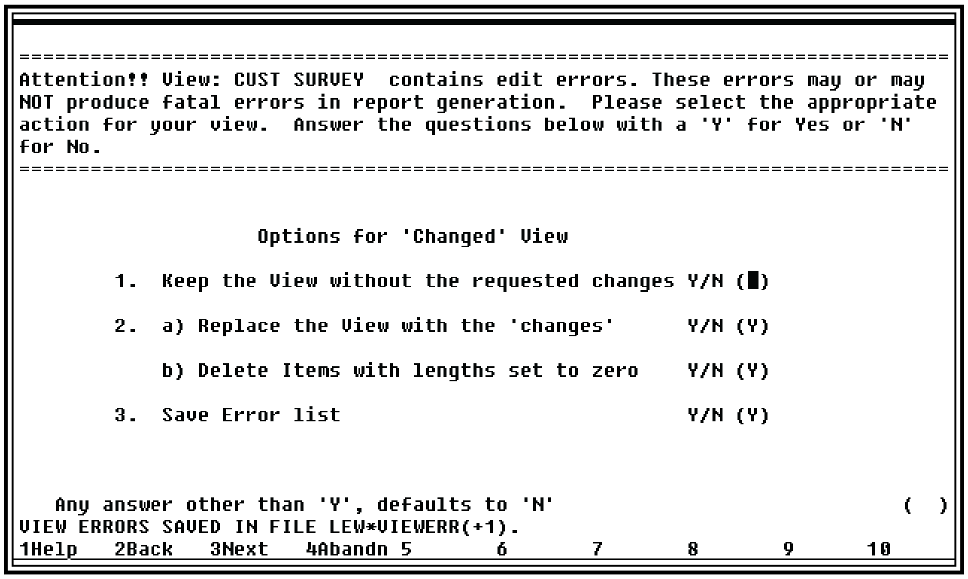
When changing item names, any character may be used as a string delimiter. The following example uses a slash (/) and will remove all the prefixes from the data item names to be viewed by the end user. Furthermore, all dashes (-) will be replaced by a single space.



```
                              InfoQuest System
                              Change Item Names
  ======================================================================
       Change item names within delimiters.  Transmit after changes.

   View:  CUST SURVEY   SURVEY OF CUSTOMER ORDERS
          Item Name Changes                 Errors
     (/CM-//                        )
     (/OR-//                        )
     (/DM-//                        )
     (/-/ / A█                      )
     (                              )
     (                              )
     (                              )
     (                              )
     (                              )
     (                              )
     (                              )
     (                              )
  Display Current Items ( )                                          (   )
```

The “A” option (3rd field on an “Item Name Changes” line) is only required when all occurrences of the search string are to be changed in each item name; therefore, it is not required to change prefixes.

If there are more changes than will fit on one screen, you may simply transmit beyond the last change entered, and the changes on the current screen will be stored and another blank screen will allow you to enter more changes. Once you have finished with all screens, F3 will cause the changes to take effect.



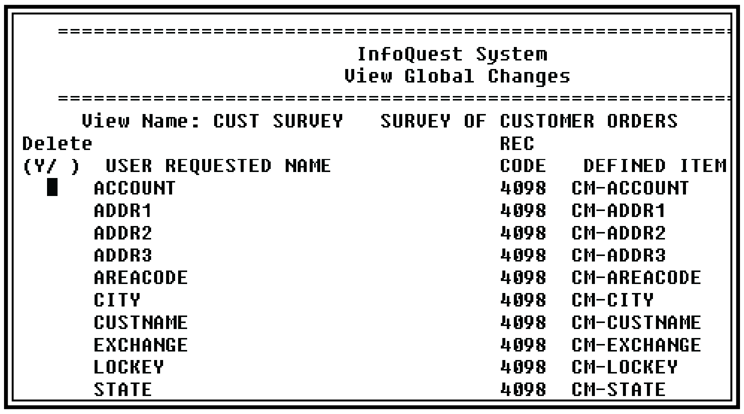
```
================================================================================
Attention!! View: CUST SURVEY  contains edit errors. These errors may or may
NOT produce fatal errors in report generation.  Please select the appropriate
action for your view.  Answer the questions below with a 'Y' for Yes or 'N'
for No.
================================================================================


                    Options for 'Changed' View

        1.  Keep the View without the requested changes Y/N (█)

        2.  a) Replace the View with the 'changes'      Y/N (Y)

            b) Delete Items with lengths set to zero    Y/N (Y)

        3.  Save Error list                             Y/N (Y)


   Any answer other than 'Y', defaults to 'N'                          (  )
VIEW ERRORS SAVED IN FILE LEW*VIEWERR(+1).
1Help   2Back   3Next   4Abandn 5       6       7       8       9       10
```

If errors occur (i.e., duplicate user item names, output fields with a length of zero, etc.), a screen will give the opportunity to:

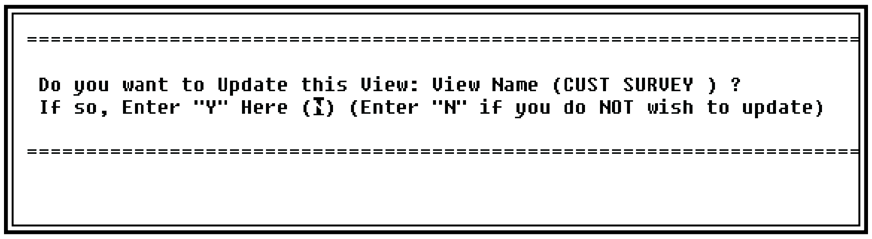
1. Keep the view without the requested changes;
2 Replace the view with the changes (including the errors) and/or delete items with a length of zero;
3. Save a list of errors. The errors will be stored in “*user-id**VIEWERR(+1).” where *user-id* is the EXEC user-id. Also, notice that the plus one cycle of the file is always used.

View Global Changes

The next screen shows the altered “USER REQUESTED NAME” item names. This screen will allow you to delete any items that are not to be selected by the end user. The deletes will not occur until you depress F3.



```
   ======================================================
                         InfoQuest System
                        View Global Changes
   ======================================================
     View Name: CUST SURVEY   SURVEY OF CUSTOMER ORDERS
Delete                                 REC
(Y/ )  USER REQUESTED NAME             CODE   DEFINED ITEM
  █    ACCOUNT                         4098  CM-ACCOUNT
       ADDR1                           4098  CM-ADDR1
       ADDR2                           4098  CM-ADDR2
       ADDR3                           4098  CM-ADDR3
       AREACODE                        4098  CM-AREACODE
       CITY                            4098  CM-CITY
       CUSTNAME                        4098  CM-CUSTNAME
       EXCHANGE                        4098  CM-EXCHANGE
       LOCKEY                          4098  CM-LOCKEY
       STATE                           4098  CM-STATE
```



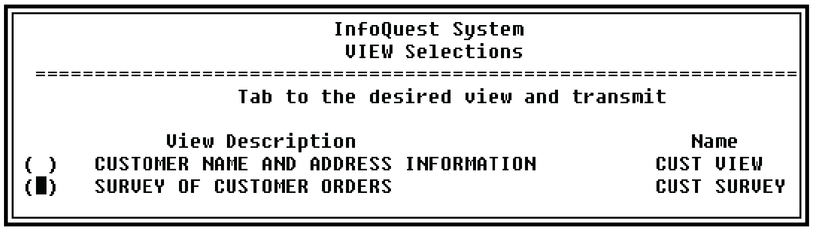
```
======================================================================

 Do you want to Update this View: View Name (CUST SURVEY ) ?
 If so, Enter "Y" Here (Y) (Enter "N" if you do NOT wish to update)

======================================================================
```

Once the items to be deleted have been marked (if any) and F3 is depressed, a verification screen will appear allowing all changes to be backed out at the last minute.

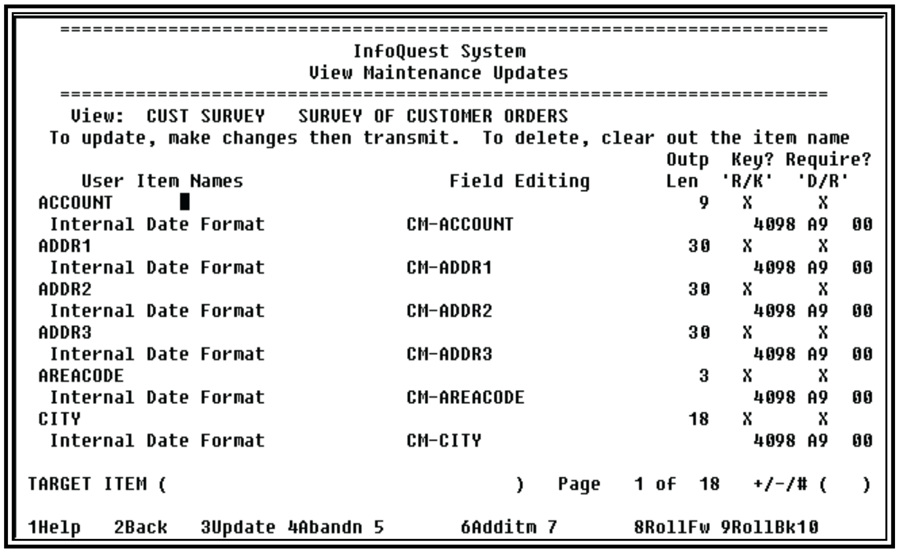
4.6 View Maintenance

Once the view has been generated, it may be tailored to the users' specific needs. The next series of figures shows how the data names are changed in order to be more meaningful when displayed to the end user during an InfoQuest session.

First, select the "View Maintenance" option from IQMNT's View Generation and Maintenance menu. The following screen will list all of the views that have been generated. Simply tab to the desired view and transmit to see the items in the view. If the desired view is not shown on the screen, use the paging feature, the scroll function keys, or the target feature to select the view. The target feature is identical as that described for global view maintenance (see page 4-5).



```
                         InfoQuest System
                         VIEW Selections
 ================================================================
                  Tab to the desired view and transmit

          View Description                            Name
( )   CUSTOMER NAME AND ADDRESS INFORMATION        CUST VIEW
(█)   SURVEY OF CUSTOMER ORDERS                    CUST SURVEY
```



```
  =================================================================
                          InfoQuest System
                      View Maintenance Updates
  =================================================================
   View:  CUST SURVEY   SURVEY OF CUSTOMER ORDERS
 To update, make changes then transmit.  To delete, clear out the item name
                                                     Outp  Key? Require?
    User Item Names                  Field Editing    Len  'R/K'  'D/R'
ACCOUNT      █                                          9   X      X
 Internal Date Format         CM-ACCOUNT                     4098 A9  00
ADDR1                                                  30   X      X
 Internal Date Format         CM-ADDR1                       4098 A9  00
ADDR2                                                  30   X      X
 Internal Date Format         CM-ADDR2                       4098 A9  00
ADDR3                                                  30   X      X
 Internal Date Format         CM-ADDR3                       4098 A9  00
AREACODE                                                3   X      X
 Internal Date Format         CM-AREACODE                    4098 A9  00
CITY                                                   18   X      X
 Internal Date Format         CM-CITY                        4098 A9  00

TARGET ITEM (                             )   Page   1 of  18   +/-/# (   )

1Help   2Back   3Update 4Abandn 5      6Additm 7      8RollFw 9RollBk10
```

User Item Names:

Upon selecting the view, you will see the list of items in the view in alphabetical order. The name of the item (as displayed to the end user) may be changed to an item name that is more expressive or meaningful to the end user (for more on user item names, see page 4-8). A "Field Editing" mask may be set to center or justify alphabetic items or edit numeric items (see the following pages). Also, the output field length may be increased to accommodate edited numeric items or totals containing large values.

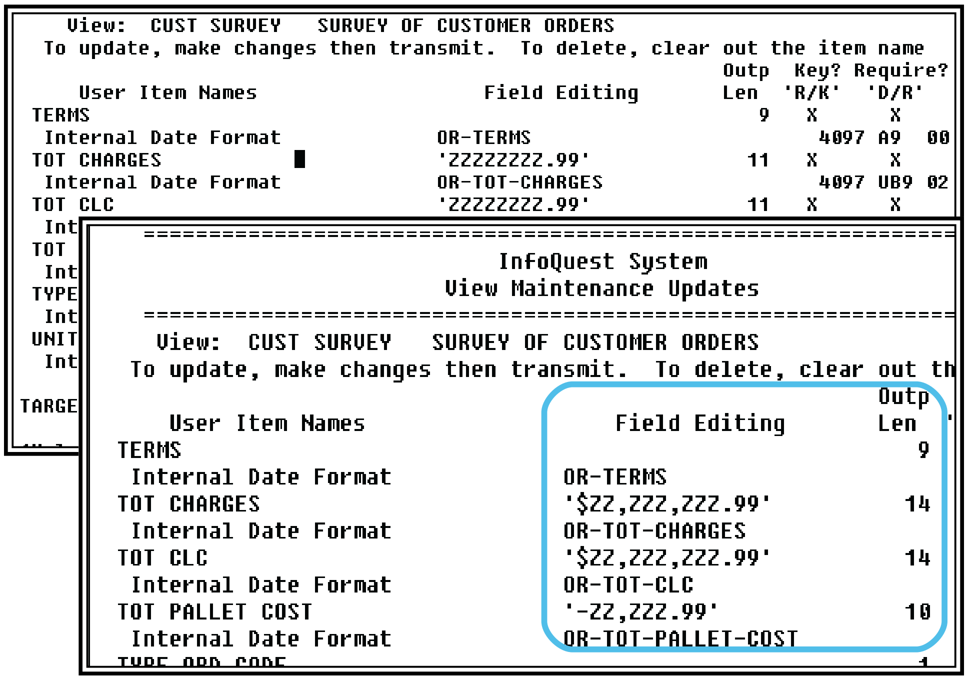
To see other data items that are not shown on the screen, use the paging feature, the scroll function keys, or the target feature to find the item. The target feature is similar to that described for global view maintenance (see page 4-5).

Only use the F3 key once you have made all of the changes desired.

To eliminate items from the user's view, blank out the item name in the "User Item Names" column.

Field Editing:

The edit masks for each numeric data item are derived from the data type and maximum output length. They may be changed at this point; however, if the change results in a change to the length of the mask, the "Outp Len" field must be altered accordingly. In the example below, the edit mask for the dollar amount fields (e.g., "TOT CHARGES") were altered to add commas (,) as part of the editing mask. Also, a floating dollar sign precedes the zero suppression symbol (Z) on all amounts except "TOT PALLET COST" which uses a floating minus sign (–) Consequently, the output length fields were increased to allow for the commas and floating symbols.



```
    View:  CUST SURVEY   SURVEY OF CUSTOMER ORDERS
  To update, make changes then transmit.  To delete, clear out the item name
                                                        Outp  Key? Require?
     User Item Names                  Field Editing     Len  'R/K'  'D/R'
 TERMS                                                     9    X      X
  Internal Date Format          OR-TERMS                         4097 A9   00
 TOT CHARGES       █            'ZZZZZZZZ.99'             11    X      X
  Internal Date Format          OR-TOT-CHARGES                   4097 UB9  02
 TOT CLC                        'ZZZZZZZZ.99'             11    X      X
  Int
 TOT
  Int
 TYPE
  Int
 UNIT
  Int

TARGE
```

```
 ====================================================================
                       InfoQuest System
                   View Maintenance Updates
 ====================================================================
   View:  CUST SURVEY   SURVEY OF CUSTOMER ORDERS
 To update, make changes then transmit.  To delete, clear out th
                                                          Outp
    User Item Names                    Field Editing      Len
TERMS                                                        9
 Internal Date Format            OR-TERMS
TOT CHARGES                      '$ZZ,ZZZ,ZZZ.99'           14
 Internal Date Format            OR-TOT-CHARGES
TOT CLC                          '$ZZ,ZZZ,ZZZ.99'           14
 Internal Date Format            OR-TOT-CLC
TOT PALLET COST                  '-ZZ,ZZZ.99'               10
 Internal Date Format            OR-TOT-PALLET-COST
```

The following edit mask characters may be used for editing decimal numbers:

1. Insertion characters:

   Any character may be used as an insertion character; however, there are two special insertion characters which are:

"B"	blank
"H"	hyphen (-)

2. Zero suppression control characters are:

"Z"	suppress leading zeros
"9"	display leading zeros

3. Sign control characters are:

"+"	print plus or minus sign depending on the value of the number
"–"	print minus sign only if the number is negative

4. Floating characters are:

"$"	print dollar sign to the left of the most significant digit when the dollar sign is the first character of the mask and followed by the character, "Z"
"–"	for negative numbers, print minus sign to the left of the most significant digit when the minus sign is the first character of the mask and is followed by the character, "Z"

The following edit mask characters may be used for aligning **alpha** items:

"LEFT"	Item is left justified in the column
"CENTER"	Item is centered in the column
"RIGHT"	Item is right justified in the column

Field Editing Date Data Items:

Data items can be identified in the view as "date data" items. To indicate that an item is a date, the InfoQuest Administrator marks the item by showing the specific **internal** date format that was used when the item was stored on the 2200 host; e.g., 'Y4M2D2', 'M2D2Y2', etc. Also, the administrator specifies the **output** format of the date as it should appear on reports and in PC files; e.g., 'M2/D2/Y2', 'M9BD2,Y4', etc.

If a data item is defined in an RDMS 2200 table as a "DATE" data type, the InfoQuest view generation process will automatically detect that the data item's **internal** format is Y4-M2-D2. For a DMS 2200 data item defined with a "USAGE IS DATE" clause in the schema, the view generation will detect that the **internal** format is Y4M2D2. In both cases, the vew generation process will automatically set the **output** format to the same as the **internal** format. The InfoQuest Administator may change the **output** format; however, the **internal** format may not be altered.

When using a date item for comparisons or calculations, InfoQuest first converts the date from its internal format to an **intermediate integer** format. This intermediate integer format is used by InfoQuest when making the actual comparisons and calculations.

Internal Date Format and Field Editing of Date Items:

The InfoQuest Administrator can identify date data items during View Maintenance as follows:

1. Enter the Internal Date Format field to correspond to the format of the date as stored on the host (automatically set by View Generation if the the item is defined with a data type of "DATE" in a DMS 2200 schema or RDMS 2200 table — the Internal Date Format for these items may ***not*** be altered in the view);
2. Enter the Output Date Format in the Field Editing field as the date is to appear on reports and in PC files (automatically set by View Generation if the the item is defined with a data type of "DATE" in a DMS 2200 schema or RDMS 2200 table — the Output Date Format for these items ***may be altered*** in the view);
3. Change the Output Length field to include enough characters to cover the date format specified in Field Editing.

Date data items may be any currently supported data type (unsigned ASCII numeric display, ASCII alphanumeric display, unsigned ASCII aligned binary, etc.) except floating point. Date formats may be specified as a predefined $DATE*n* format (without quotes) or as an administrator defined literal bound in quotes.

The following $DATE*n* formats are predefined:

$DATE*n*	**Format**	**Example**
$DATE0	YMMDD (Y1M2D2)	20327
$DATE1	YYMMDD (Y2M2D2)	920327
$DATE2	DD MMM YY (D2BM3BY2)	27 MAR 92
$DATE3	YDDD (Y1D3)	2087
$DATE4	YYDDD (Y2D3)	92087
$DATE5	DDMMYY (D2M2Y2)	270392
$DATE6	MM/DD/YY (M2/D2/Y2)	03/27/92
$DATE7	MMMMMMMMM DD,YYYY (M9BD2,Y4)	MARCH 27,1992
$DATE8	MMDDYY (M2D2Y2)	032792
$DATE20	YYYYMMDD (Y4M2D2)	19920327
$DATE21	YYYYDDD (Y4D3)	1992087
$DATE22	DDMMYYYY (D2M2Y4)	27031992
$DATE23	MMDDYYYY (M2D2Y4)	03271992

Date Descriptors	Explanation

Key?:

There are two flags that can be set on a data item in a view. These flags are used internally to specify what action is to be taken on a specific data item. The flags are set in a copy of the view as the end user proceeds through an InfoQuest session.

The first flag, “Key”, may be changed to a “K” designating that the associated database module (DBM) will access the record/file by a specific key value.

If range processing is desired, the “Key” flag should be set to an “R”. Range processing is not possible for DMS1100 CALC records.

This flag should be set only when the data item is defined as a key for random record selection in the DMS1100 schema, PCIOS MSAM file or PCIOS ISAM file. For MSAM files, the key can be either the primary key or an alternate key (if defined).

If this flag is set, the second flag, “Require?”, should be set to an “R”.

Require?:

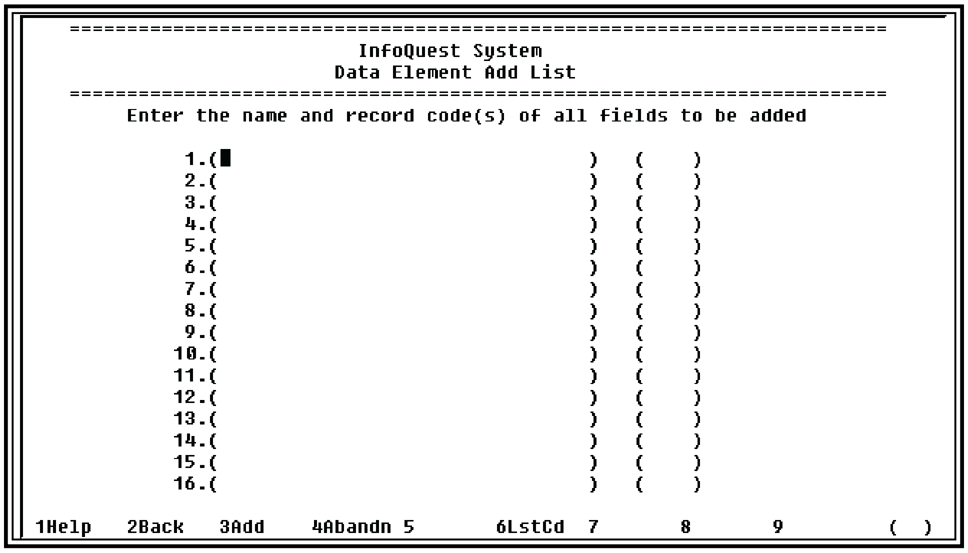
The second flag, “Require?”, may be changed to an “R”, designating that the data item is required for selection. In this case, the end user will be required to supply a value for selection regardless of whether the end user has specifically chosen to select on the data item.

If the flag is set to a “D” (defined), the data item will be marked as required for access but not for selection; i.e., the end user will not be prompted to enter any values unless they choose the item as a selection item.

If the data item is defined as a part of an RDMS 1100 table and its value is to be used to gain random entry to another file system (DMS, PCIOS, etc.), the “Required” flag must be set to an “R” or a “D”. Either of these settings provides the mechanism by which the DBM can initialize the key used for random access to the other file system.

AddItm(F6):

To add items to the view use the F6 key and a screen will appear which allows you to enter item names and record codes. These items must have been previously deleted from the view; i.e., still present in the QINDEX data item index file. The item names and record codes entered must be exactly as specified in the data item index file.



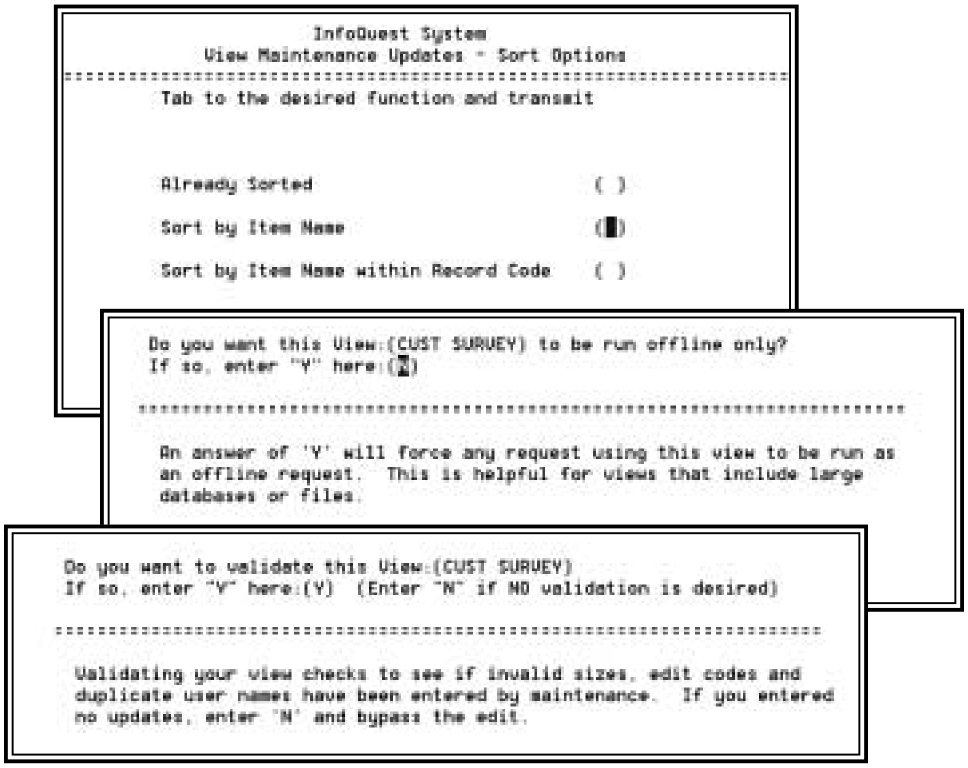
```
=====================================================================
                         InfoQuest System
                        Data Element Add List
=====================================================================
     Enter the name and record code(s) of all fields to be added

          1.(█                             )   (    )
          2.(                              )   (    )
          3.(                              )   (    )
          4.(                              )   (    )
          5.(                              )   (    )
          6.(                              )   (    )
          7.(                              )   (    )
          8.(                              )   (    )
          9.(                              )   (    )
         10.(                              )   (    )
         11.(                              )   (    )
         12.(                              )   (    )
         13.(                              )   (    )
         14.(                              )   (    )
         15.(                              )   (    )
         16.(                              )   (    )

1Help   2Back   3Add    4Abandn 5       6LstCd  7       8       9        (  )
```

If you do not remember a name or record code, use the F6(LstCd) key again to get a list of the record/file codes available in the QINDEX data item index file.

Only use the F3 key when you have entered all the item names desired.

View Maintenance

Once the desired changes have been completed, the View Maintenance function allows the view to be sorted. Simply tab to the desired selection and transmit. Following the sort option screen, you will be asked if the view is to be restricted to off-line processing.



```
                    InfoQuest System
          View Maintenance Updates - Sort Options
====================================================================
        Tab to the desired function and transmit


        Already Sorted                          ( )

        Sort by Item Name                       (█)

        Sort by Item Name within Record Code    ( )
```

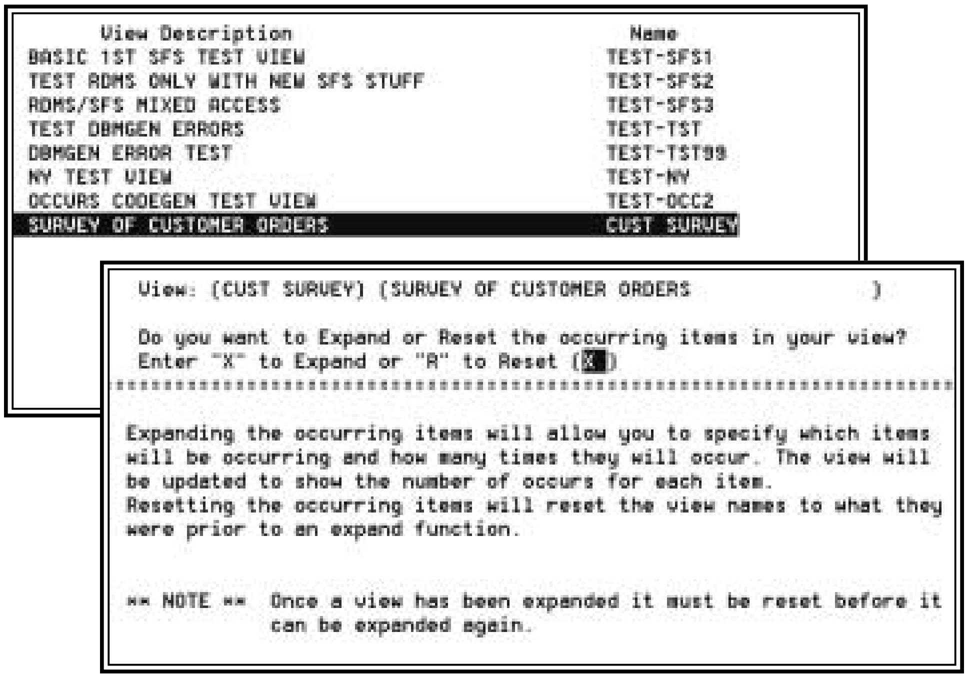
```
Do you want this View:(CUST SURVEY) to be run offline only?
If so, enter "Y" here:(█)

=====================================================================

 An answer of 'Y' will force any request using this view to be run as
 an offline request.  This is helpful for views that include large
 databases or files.
```

```
Do you want to validate this View:(CUST SURVEY)
If so, enter "Y" here:(Y)  (Enter "N" if NO validation is desired)

=====================================================================

 Validating your view checks to see if invalid sizes, edit codes and
 duplicate user names have been entered by maintenance.  If you entered
 no updates, enter 'N' and bypass the edit.
```

The final screen allows the altered view to be validated for possible errors. This validation is the same as described in Section 4.10, “View Validation”.

4.7 Occurring Item Maintenance

InfoQuest has the capability of handling data items contained within a COBOL OCCURS clause. This function will rename each occurrence of an item in order for the user to see the item as a unique occurrence. This function is the “View Maintenance - Occurring Items” selection on the View Generation and Maintenance screen. For a complete example illustrating view generation and path generation with occurring items, see Section 6.13, “OCCURS Groups”.



```
     View Description                                Name
BASIC 1ST SFS TEST VIEW                            TEST-SFS1
TEST RDMS ONLY WITH NEW SFS STUFF                  TEST-SFS2
RDMS/SFS MIXED ACCESS                              TEST-SFS3
TEST DBMGEN ERRORS                                 TEST-TST
DBMGEN ERROR TEST                                  TEST-TST99
MY TEST VIEW                                       TEST-MY
OCCURS CODEGEN TEST VIEW                           TEST-OCC2
SURVEY OF CUSTOMER ORDERS                          CUST SURVEY
```

```
View: (CUST SURVEY) (SURVEY OF CUSTOMER ORDERS              )

Do you want to Expand or Reset the occurring items in your view?
Enter "X" to Expand or "R" to Reset (X)
*****************************************************************

Expanding the occurring items will allow you to specify which items
will be occurring and how many times they will occur. The view will
be updated to show the number of occurs for each item.
Resetting the occurring items will reset the view names to what they
were prior to an expand function.


** NOTE **  Once a view has been expanded it must be reset before it
            can be expanded again.
```

Once the
desired view has been selected, you may choose to expand (“X” option) or reset (“R” option) the view.

Occurring Items Expansion:

The expansion option displays on the screen only those items that are a part of an occurring group. To expand an occurring item, change the value of “Number of Occurs” to the number of occurrences to which you need to expand. The numbers 0000 and 9999 are ignored; any value between 0000 and 9999 is valid. However, be careful not to expand the item past the actual number of occurrences, as it will result in an invalid data reference (error 6) in the Q-LINK program (DBM).

Each data item expanded will have multiple generated user item names that consist of the original user item name suffixed with “–number”, where “number” = 1, 2, 3, etc., up to “Number of Occurs” (see example).

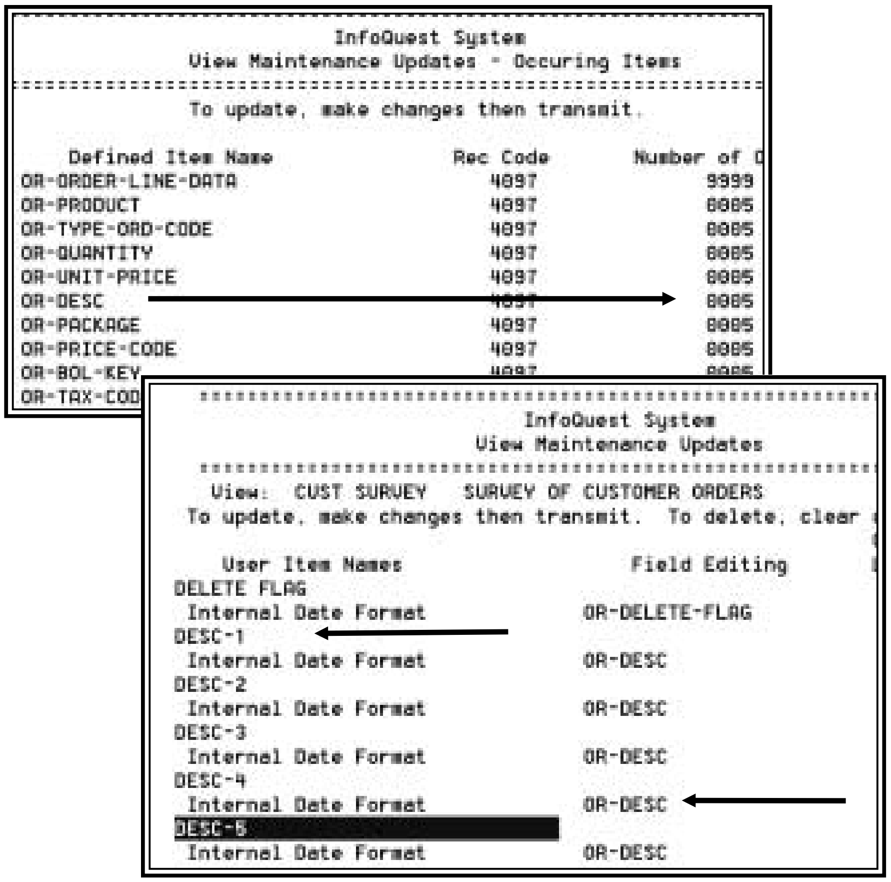
Occurring Item Maintenance

Occurring Items Reset:

The reset option returns the occurring items to their original state prior to expansion; i.e., only one occurrence of each user item. It does not reset "Number of Occurs" entered when the view was originally expanded. Therefore, if an item was last expanded to 9 occurrences, the "Number of Occurs" field will still contain the value "9".

Once the view has been expanded, it must be reset (R) before it can be expanded again.

The following example shows OR-DESC being expanded to 5 occurrences. The result of the expansion can be seen by displaying the view with the View Maintenance function. Notice the generated user item names DESC-1 through DESC-5 (the "OR-" prefix was dropped).



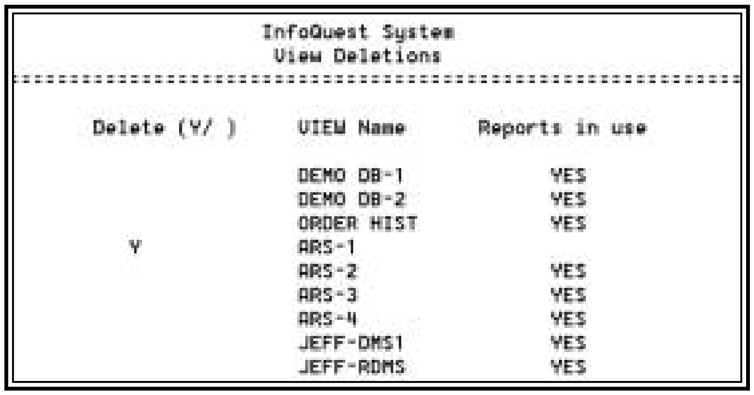
```
                         InfoQuest System
              View Maintenance Updates - Occuring Items
=====================================================================
              To update, make changes then transmit.

     Defined Item Name                 Rec Code       Number of O
OR-ORDER-LINE-DATA                       4097              9999
OR-PRODUCT                               4097              0005
OR-TYPE-ORD-CODE                         4097              0005
OR-QUANTITY                              4097              0005
OR-UNIT-PRICE                            4097              0005
OR-DESC                                  4097              0005
OR-PACKAGE                               4097              0005
OR-PRICE-CODE                            4097              0005
OR-BOL-KEY                               4097              0005
OR-TAX-COD
```

```
=====================================================================
                         InfoQuest System
                      View Maintenance Updates
=====================================================================
  View:  CUST SURVEY   SURVEY OF CUSTOMER ORDERS
 To update, make changes then transmit.  To delete, clear

    User Item Names                      Field Editing
DELETE FLAG
 Internal Date Format               OR-DELETE-FLAG
DESC-1
 Internal Date Format               OR-DESC
DESC-2
 Internal Date Format               OR-DESC
DESC-3
 Internal Date Format               OR-DESC
DESC-4
 Internal Date Format               OR-DESC
DESC-5
 Internal Date Format               OR-DESC
```

Not every occurring item has to be expanded. Also, occurring items need not be expanded to the maximum number of occurrences.

4.8 View Deletions

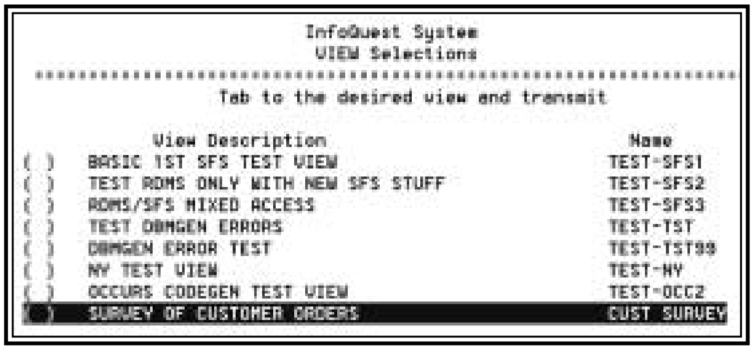
Another function available from the IQMNT menu is the “View Deletions” selection. Simply mark each view to be deleted with an “Y”. If the view has reports in use, a warning message will be displayed asking if you want to delete the view and reports. Once all views have been marked for deletion, depress F3, and the marked entries will be deleted.



```
                       InfoQuest System
                        View Deletions
==========================================================

     Delete (Y/ )       VIEW Name       Reports in use

                        DEMO DB-1             YES
                        DEMO DB-2             YES
                        ORDER HIST            YES
        Y               ARS-1
                        ARS-2                 YES
                        ARS-3                 YES
                        ARS-4                 YES
                        JEFF-DMS1             YES
                        JEFF-RDMS             YES
```

4.9 View Duplication

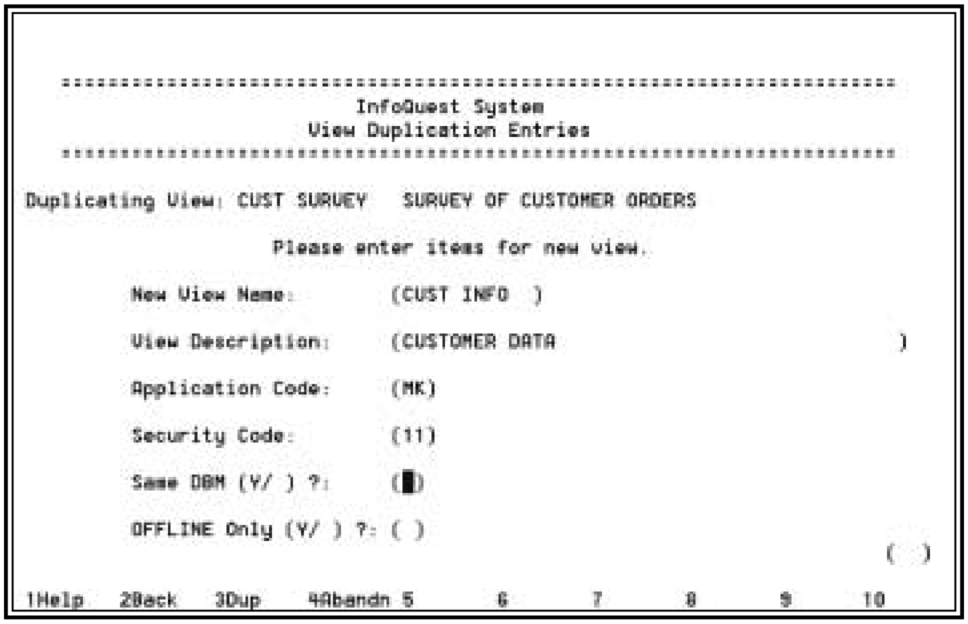
Another function that can be selected from the View Generation and Maintenance menu is the “View Duplication” selection. This option allows an existing view to be copied, and might be advantageous when different access paths (DBMs) are required to the same data. Also, copying a view that has previously been tailored can save time and provide consistent data item names between views. First select the view you wish to duplicate by tabbing to the appropriate entry and transmitting. If the desired view is not shown, use the scroll or target features described earlier (see page 4-5).



```
                          InfoQuest System
                           VIEW Selections
  ***************************************************************
                 Tab to the desired view and transmit

          View Description                              Name
( )   BASIC 1ST SFS TEST VIEW                         TEST-SFS1
( )   TEST RDMS ONLY WITH NEW SFS STUFF               TEST-SFS2
( )   RDMS/SFS MIXED ACCESS                           TEST-SFS3
( )   TEST DBMGEN ERRORS                              TEST-TST
( )   DBMGEN ERROR TEST                               TEST-TST99
( )   MY TEST VIEW                                    TEST-MY
( )   OCCURS CODEGEN TEST VIEW                        TEST-OCC2
( )   SURVEY OF CUSTOMER ORDERS                       CUST SURVEY
```

View Duplication

Once the view to be copied has been selected, the following screen will allow the view name and view description to be entered. The application code and security code may be altered as required.

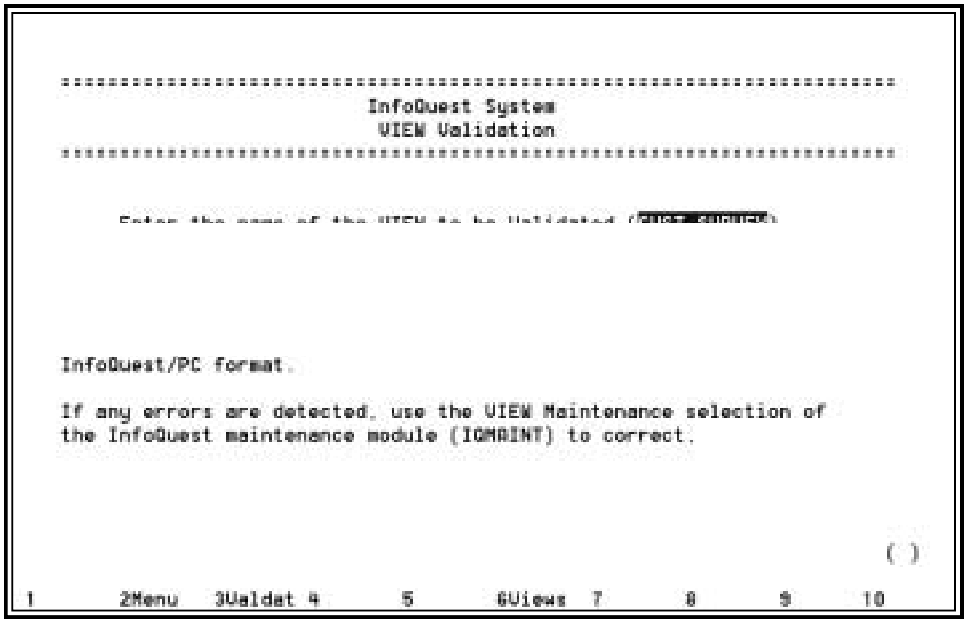


```
   ====================================================================
                            InfoQuest System
                        View Duplication Entries
   ====================================================================

Duplicating View: CUST SURVEY   SURVEY OF CUSTOMER ORDERS

                   Please enter items for new view.

        New View Name:       (CUST INFO  )

        View Description:    (CUSTOMER DATA                          )

        Application Code:    (MK)

        Security Code:       (11)

        Same DBM (Y/ ) ?:    (█)

        OFFLINE Only (Y/ ) ?: ( )
                                                                     (  )
1Help   2Back   3Dup    4Abandn 5       6       7       8       9       10
```

The "Same DBM" options allows the view to use a copy of the same DBM as the original view.

4.10 View Validation

The View Validation selection from the View Generation and Maintenance menu allows the user to validate a view for errors (duplicate user names, invalid lengths, etc.). If errors occur, the View Maintenance selection can be used to correct the errors.

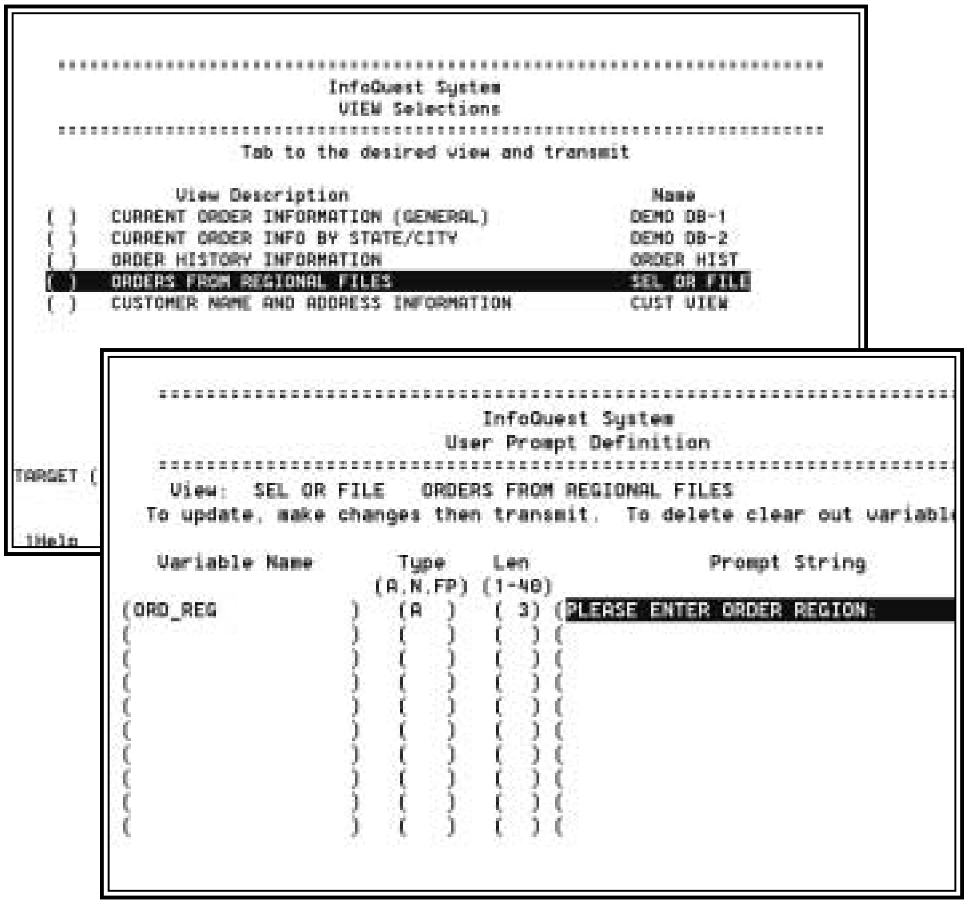


```
======================================================================
                        InfoQuest System
                         VIEW Validation
======================================================================


     Enter the name of the VIEW to be Validated ([illegible])




InfoQuest/PC format.

If any errors are detected, use the VIEW Maintenance selection of
the InfoQuest maintenance module (IQMAINT) to correct.



                                                                    ( )
1       2Menu   3Valdat 4       5       6Views  7       8       9       10
```

View Validation will assure that no duplicate user names exist, that output lengths are long enough to cover the edit masks supplied, etc. Furthermore, the DBM associated with the view, will be checked to assure that items in the view have corresponding record access logic included in the DBM.

4.11 View User Prompt Definitions

The View User Prompt Definition selection from the View Generation and Maintenance menu allows the application administrator the ability to predefine prompt variables to hold user entered data and to be used within a tailored InfoQuest Database Module (DBM).



```
******************************************************************
                         InfoQuest System
                          VIEW Selections
==================================================================
                 Tab to the desired view and transmit

          View Description                           Name
( )   CURRENT ORDER INFORMATION (GENERAL)          DEMO DB-1
( )   CURRENT ORDER INFO BY STATE/CITY             DEMO DB-2
( )   ORDER HISTORY INFORMATION                    ORDER HIST
( )   ORDERS FROM REGIONAL FILES                   SEL OR FILE
( )   CUSTOMER NAME AND ADDRESS INFORMATION        CUST VIEW

TARGET (
 1Help
```

```
==================================================================
                         InfoQuest System
                      User Prompt Definition
==================================================================
  View:  SEL OR FILE   ORDERS FROM REGIONAL FILES
 To update, make changes then transmit.  To delete clear out variabl

  Variable Name      Type    Len           Prompt String
                   (A,N,FP) (1-40)
 (ORD_REG         )   (A  )   ( 3) (PLEASE ENTER ORDER REGION:
 (                )   (   )   (  ) (
 (                )   (   )   (  ) (
 (                )   (   )   (  ) (
 (                )   (   )   (  ) (
 (                )   (   )   (  ) (
 (                )   (   )   (  ) (
 (                )   (   )   (  ) (
 (                )   (   )   (  ) (
 (                )   (   )   (  ) (
```

After selecting the desired view, up to 10 user prompt variables may be defined. There are four entries for each field definition: the first is the name of the variable to hold the data, the second is the field type (Alpha, Numeric or Floating point), next is the length of the field (can be from 1 to 40 characters) and the last is the actual prompt string that the user will see.

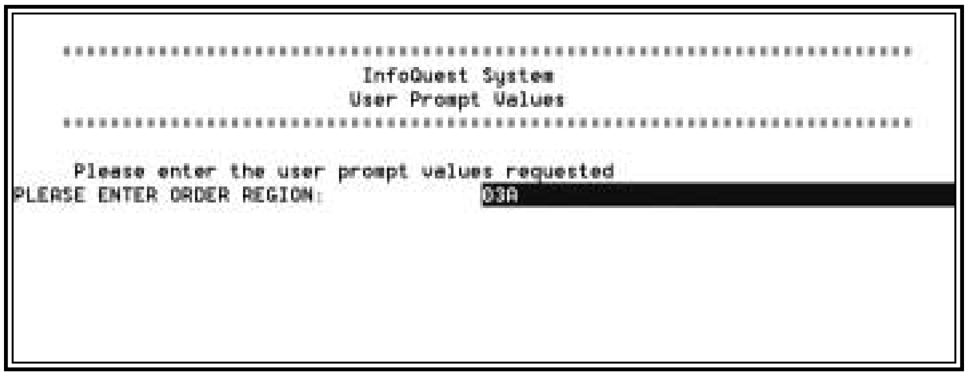
Numeric variables (Type=N) must be limited to a length of 18 positions.

In the above example when the user selects the ORDERS FROM REGIONAL FILES view, the user will be asked, “Please enter order REGION:”. The user’s response will be placed in a three character alphanumeric variable called, “ORD_REG”. The region code will be used to select only the data for that region. Region code is not an item in the view but is used in this example as a part of a file name; i.e., each region has its own source of data (file). The DBM had to be modified slightly in order to include this region code as a part of the file name. For instance, the file assignment commands might look like this:

```
display ‘@asg,a iq$demo*orderfile’ +
display ORD_REG +
a50 = $pbuff
csf x a50
IF X  0
  D ‘Can’’t assign file ORDERFILE’
  DO CLOSE-RPT
  STOP 9999
ENDIF

display ‘@use orderfile.,iq$demo*orderfile’ +
display ORD_REG +
a50 = $pbuff
csf x a50
OPEN ORDERFILE INPUT DYNAMIC
```

The variable “a50” is a user defined variable to contain ECL commands. Its definition is not shown, but it is 50 alphanumeric characters.

The value for the prompt variable will be set during the request generation immediately prior to the user entering the search values. The following screen will be presented at that time.



```
   ***********************************************************************
                            InfoQuest System
                           User Prompt Values
   ***********************************************************************

    Please enter the user prompt values requested
PLEASE ENTER ORDER REGION:            00A
```

The variable name may be used in a variety of ways when customizing DBMs (see Appendix E, “Customized DBMs”, for other examples).

4.12 View Translation Table Generation

The translation file function allows you to create or list a translation file to be used in association with a particular view. A translation file is a file that contains field translations for a particular view. Translations are fields in the database that contain codes that have other meanings. One example might be a field called "SEX". In the database, a "1" might mean "FEMALE"; and a "2", "MALE". By creating a translation file and attaching it to the view, InfoQuest will substitute "MALE" and "FEMALE" for "1" and "2", respectively, whenever the "SEX" field is printed. The same could be done for state abbreviations. Instead of printing "GA" or "AL", you could have a translation file that translates the abbreviations of the states so that "GEORGIA" or "ALABAMA" would be printed.

The following steps are required to generate the translation table and create the translation file:

1) Create the translation definitions for the translation run.
2) Run the View Translation Table Generation function from the View Generation and Maintenance Submenu to create the translation file.

The translation values will stored in a file that will be referenced by the DBM; therefore, the DBM must be generated prior to this function being used.

In the following example, two different fields will be translated: CM–STATE and OH–PRODLIN–CODE:

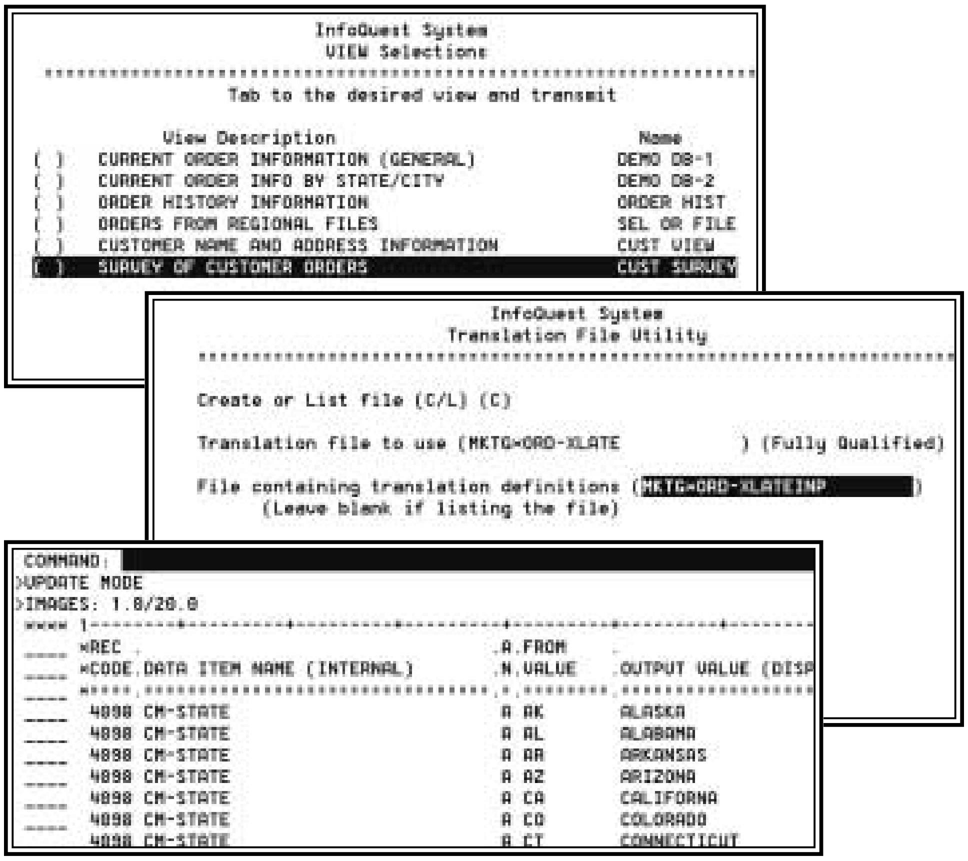
```
         1    1    2    2    3    3    4
1...5....0....5....0....5....0....5....0
N4099 CM-STATE
VGA       GEORGIA
VAL       ALABAMA
VCA       CALIFORNIA
VFL       FLORDIA
N0006 OH-PRODLINE-CODE
V1        PRINTER
V2        MONITOR
V3        KEYBOARD
```

The translation definitions tell InfoQuest what fields will be translated and what data values will be substituted.

The "N" lines specify the record code containing the field to be translated and the name of the field being translated. Note: Use the List QINDEX Contents function of IQMNT to view the record codes, field names and field lengths available (see page 3-22).

The “V” lines contain the translation values. The first 8 positions are the actual data values, then 1 space and then the next 50 characters are the value to be printed. The translated values are only used for printing purposes; they are not used for record selection.

The Translation File Utility screen lists the views that currently exist in your system and are accessible through InfoQuest. To select the view to process, tab to the view name and transmit. If the view to process is not shown on the current screen, press the F8 key to page forward, or press the F9 key to page backwards. You can also enter a specific page number or use the target item search to locate a certain view. When using the target search, you can enter either a complete or partial name or description.



```
                         InfoQuest System
                          VIEW Selections
 *******************************************************************
                  Tab to the desired view and transmit

           View Description                          Name
( )   CURRENT ORDER INFORMATION (GENERAL)          DEMO DB-1
( )   CURRENT ORDER INFO BY STATE/CITY             DEMO DB-2
( )   ORDER HISTORY INFORMATION                    ORDER HIST
( )   ORDERS FROM REGIONAL FILES                   SEL OR FILE
( )   CUSTOMER NAME AND ADDRESS INFORMATION        CUST VIEW
( )   SURVEY OF CUSTOMER ORDERS                    CUST SURVEY
```

```
                         InfoQuest System
                     Translation File Utility
 *******************************************************************

 Create or List file (C/L) (C)

 Translation file to use (MKTG*ORD-XLATE          ) (Fully Qualified)

 File containing translation definitions (MKTG*ORD-XLATEINP          )
       (Leave blank if listing the file)
```

```
COMMAND:
>UPDATE MODE
>IMAGES: 1.0/20.0
**** 1---------+---------+---------+---------+---------+---------+--------
____ *REC                               A.FROM
____ *CODE.DATA ITEM NAME (INTERNAL)     N.VALUE    OUTPUT VALUE (DISP
____ *****.******************************.*.********.******************
____  4098 CM-STATE                       A AK       ALASKA
____  4098 CM-STATE                       A AL       ALABAMA
____  4098 CM-STATE                       A AR       ARKANSAS
____  4098 CM-STATE                       A AZ       ARIZONA
____  4098 CM-STATE                       A CA       CALIFORNA
____  4098 CM-STATE                       A CO       COLORADO
      4098 CM-STATE                       A CT       CONNECTICUT
```

On the second screen, enter “C” to create a file or “L” to list the contents of an existing file. Also enter the name of the file to contain the translation table and the location where the translation definitions are stored.

View Translation Table Generation

The translation file utility places a listing of the translation values and any input errors that might occur in "*user-id**XLATEOFILE(+1)." where *user-id* is the EXEC user-id. Also, notice that the plus one cycle of the file is always used. This file may be viewed with any editor.

When using the View Translation Table feature, be sure to increase the size of the output length (see "View Maintenance" on page 4-11) if the"OUTPUT VALUE" is larger than the original field. In the above example, the output length for CM-STATE will be set to the length of the largest name to be printed: DISTRICT OF COLUMBIA (20 characters).

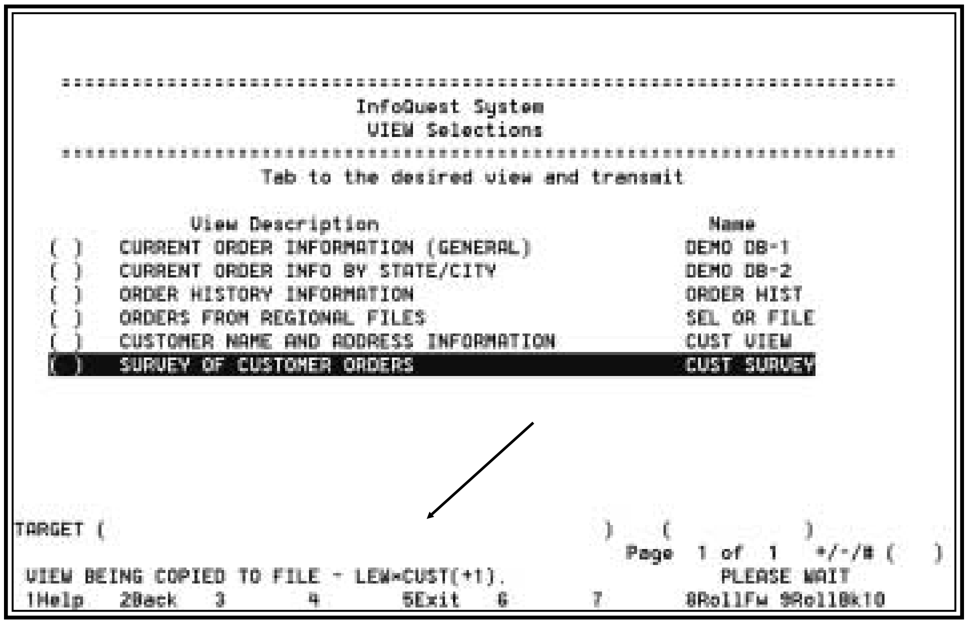
Once the translation table result is displayed, press F1 (Resume) to return to the View Translation Table Generation menu.

The following special rules apply to translation table generation:

Items containing decimals cannot be translated.
Actual data values cannot be longer than eight (8) characters.
Translated values cannot be longer than 50 characters.
Actual data values are used for all selecting, sorting, and calculating derived items.
Sorts may appear to be incorrect because the actual data values are being sorted, not the translated values.

4.13 View Copy Utility

The View Copy Utility allows you to copy a view to a data file for subsequent viewing with any 2200 editor. The data file to which the view is written will be automatically cataloged if it does not exist. The name of the data file is constructed as follows: "*user-id***view-name*(+1).", where *user-id* is the EXEC user-id and *view-name* is the leftmost significant characters of the view name, up to but not including the first space (e. g., the "CUST SURVEY" view, if copied by the user "JOSE", would be placed in the "JOSE*CUST(+1)." file). Also, notice that the plus one cycle of the file is always used.



```
  ======================================================================
                            InfoQuest System
                             VIEW Selections
  ======================================================================
                     Tab to the desired view and transmit

                View Description                             Name
 ( )   CURRENT ORDER INFORMATION (GENERAL)                 DEMO DB-1
 ( )   CURRENT ORDER INFO BY STATE/CITY                    DEMO DB-2
 ( )   ORDER HISTORY INFORMATION                           ORDER HIST
 ( )   ORDERS FROM REGIONAL FILES                          SEL OR FILE
 ( )   CUSTOMER NAME AND ADDRESS INFORMATION               CUST VIEW
 ( )   SURVEY OF CUSTOMER ORDERS                           CUST SURVEY



TARGET (                                          )   (          )
                                                    Page  1 of  1   +/-/# (   )
 VIEW BEING COPIED TO FILE - LEW*CUST(+1).              PLEASE WAIT
 1Help   2Back   3     4      5Exit   6      7       8RollFw 9RollBk10
```

View Copy Utility

Chapter 5 DBM Generation and Maintenance

5.1 DBM: Definition

An InfoQuest Database Module (DBM) is a Q-LINK program that will be used to access the data when the end user is generating a report through an InfoQuest session. The DBM contains all the path navigation logic necessary to retrieve the data (at the record level) required for the end user's report(s).

5.2 Selecting DBM Generation and Maintenance

An IQMNT function that allows for the generation of the new DBMs, regenerating (replacing all existing code) an existing DBM, and updating (integrating additional record/file access code or deleting existing code) an existing DBM is provided with the release of InfoQuest. This function is the "DBM Generation" function which is selected from the "DBM Generation and Maintenance" menu of IQMNT. Other functions are also available for "DBM Duplication" and "DBM Deletion".

5.3 DBM: Prerequisites

Before generating the DBM, you will need to have the following information about your data structure ready:

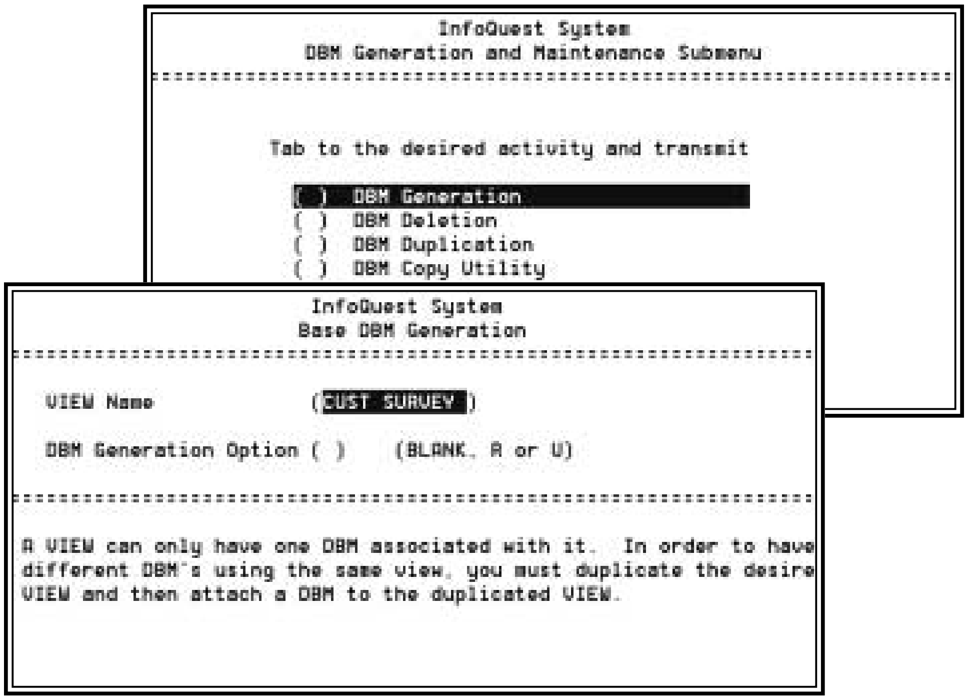
- The VIEW name to be used by the DBM.
- If PCIOS files will be accessed, the external name of the file for the @ASG statement; the internal name for the file by which it was defined during view generation; the file's type and access mode. Note: record size, block size and MSAM key specifications (location, length and data type) are not required as they will be read from the internal PCIOS file label.
- For DMS1100 database access, you will need the names of all areas that must be opened by the DBM plus all necessary database data names, record names and set names.

DBM: Prerequisites

- If a DMS1100 CALC record will be accessed by its key, you will need the name of the database-dataname that must be initialized with an appropriate area name. Also, a database-dataname may be required when accessing an Index Sequential record by its key. In this case, the database-dataname is required only if the record spans more than one area, and the database-dataname is not initialized by an Index Sequential Database Procedure.
- You will also need the names of all records and sets that the DBM must navigate (the path) for report generation, which may require usage of a schema listing or representative block diagram.
- For RDMS 1100 access, the name of the UDS 1100 application (UDSSRC is the UDS default), the table qualifier (schema name), the table version (PRODUCTION is the UREP 1100 default), and the name of each table to be accessed by the DBM. In addition, if tables are to be linked or joined, the defined primary key of each table or defined secondary indexes should be used for efficiency.
- For MAPPER DTM access, the name of the logical files as defined in the view generation process (previous chapter).
- If keyed access is required, you may fill in the key information (item name or names) which generates the code to find the user-entered key values at run time. Item names can also be used to interlink DMS 1100 records, RDMS 1100 tables, PCIOS files, etc.

5.4 DBM Generation

The DBM build can be invoked from the IQMNT menu by selecting “DBM Generation and Maintenance” followed by the “DBM Generation” selection as shown in the following figure. The DBM generation function will add the code to actually access the data based on the information entered on a series of generation screens. One DBM is required for each view generated. Also, only one DBM can be associated with an InfoQuest view.



```
                    InfoQuest System
           DBM Generation and Maintenance Submenu
=================================================================


            Tab to the desired activity and transmit

              ( )  DBM Generation
              ( )  DBM Deletion
              ( )  DBM Duplication
              ( )  DBM Copy Utility
```

```
                      InfoQuest System
                    Base DBM Generation
=================================================================

  VIEW Name                   (CUST SURVEY )

  DBM Generation Option ( )    (BLANK, A or U)

=================================================================
 A VIEW can only have one DBM associated with it.  In order to have
 different DBM's using the same view, you must duplicate the desire
 VIEW and then attach a DBM to the duplicated VIEW.
```

Note: On occassion, it might be necessary to add additional code to a generated DBM to handle special circumstances such as customized navigational path logic. The “DBM Copy Utility” (see page 5-20) may be used in IQMNT to copy a generated DBM to a file for possible editing with any 2200 editor (also see, Appendix E, “Customized DBMs”).

VIEW Name:

The VIEW Name must be a valid view name as entered when the view was generated. When selecting a view, you may also use the F7 key to display the list of views previously generated for the application, tab to the desired view and press transmit.

DBM Generation Option:

The DBM Generation Option should remain blank when generating new DBMs as in the example on the previous page.

When the DBM Generation Option is set to an “R”, the DBM generation process will regenerate the current DBM associated with the view number entered. The DBM will be regenerated using the original path specification parameters. Alteration of the original parameters is not possible. Any customized code manually inserted in the DBM since the last generation will be lost.

If the DBM Generation Option is set to “U”, the existing path parameters (from a previous generation or update) can be altered. Unlike the “R” option, each parameter will be displayed as originally entered where it may be changed or deleted as necessary. Additional path specification parameters may be added during this “replay” of the original parameters. Any customized code in the original DBM will be lost.

Once the above information is entered, the path generation screens will appear as described below.

5.5 Automatic Path Generation

The InfoQuest Path Navigation Logic is a part of the DBM generation process. It actually generates the Q-LINK command language program to access various data structures under InfoQuest. The path generation process will be automatically called once the previous screen has been entered. While this process can be used to generate paths to access many varying data structures, it is important to remember that you can further enhance the generated logic to improve performance or add access logic.

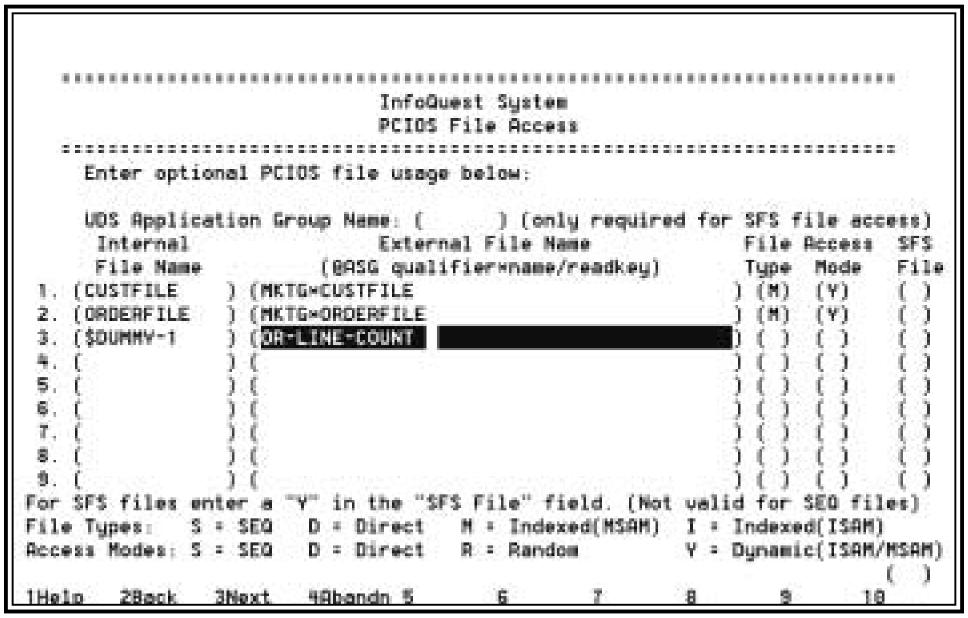
Path generation, like most other InfoQuest functions, is performed under the control of an InfoQuest run interface. It will prompt you for the required parameters. The parameters will be edited by a Q-LINK program, and if errors are found, you will be given the opportunity to correct and re-submit. Once there are no errors, the parameters will be passed to a second Q-LINK program which will actually generate the Q-LINK command navigational logic, thus, creating the finished DBM.

5.6 Path Generation Screens

The following subsections describe each of the path generation screens and the type of information to be entered on each screen. Some information identifies the entities (file names, record names, set names, access keys, etc.) included in the path while other information describes the access method(s) to be used. No data item values are entered since they are supplied by the end user during report generation.

5.6.1 PCIOS Files

This screen is only required if PCIOS files will be accessed in the generation path. To bypass this screen, depress F3; otherwise, supply the names, type and mode for each file as described below:



```
  ************************************************************************
                             InfoQuest System
                             PCIOS File Access
  ========================================================================
    Enter optional PCIOS file usage below:

    UDS Application Group Name: (       ) (only required for SFS file access)
     Internal                External File Name                File Access  SFS
     File Name          (@ASG qualifier*name/readkey)          Type  Mode   File
 1. (CUSTFILE    ) (MKTG*CUSTFILE                            ) (M)   (Y)    ( )
 2. (ORDERFILE   ) (MKTG*ORDERFILE                           ) (M)   (Y)    ( )
 3. ($DUMMY-1    ) (OR-LINE-COUNT                            ) ( )   ( )    ( )
 4. (            ) (                                         ) ( )   ( )    ( )
 5. (            ) (                                         ) ( )   ( )    ( )
 6. (            ) (                                         ) ( )   ( )    ( )
 7. (            ) (                                         ) ( )   ( )    ( )
 8. (            ) (                                         ) ( )   ( )    ( )
 9. (            ) (                                         ) ( )   ( )    ( )
For SFS files enter a "Y" in the "SFS File" field. (Not valid for SEQ files)
File Types:   S = SEQ   D = Direct   M = Indexed(MSAM)  I = Indexed(ISAM)
Access Modes: S = SEQ   D = Direct   R = Random         Y = Dynamic(ISAM/MSAM)
                                                                       (   )
1Help   2Back   3Next   4Abandn 5       6       7       8       9       10
```

Internal File Name:

The first field, Internal File Name, must be entered as included in the view except when a $DUMMY-n entry is being entered to handle a record with an OCCURS DEPENDING ON clause (see below).

External File Name:

The second entry, External File Name, is the name required to @ASG the file. It must include a qualifier, file name and, if necessary, a read key.

In one case, the External File Name is not a cataloged file. If the file name entry is being used to handle a record with an OCCURS DEPENDING ON clause, the name of the elementary item that contains the occurs count must be placed in the External File Name entry.

For more information on handling occuring items, see Section 6.13, “OCCURS Groups”, on page 6-31).

File Type:

The third entry specifies the file type required by Q-LINK. There are four acceptable file types:

> Sequential;
> Direct (relative);
> MSAM (most commonly used indexed file access);
> ISAM (older version single-key indexed sequential).

Most programming shops use MSAM as opposed to the old ISAM format. The programming techniques required to handle ISAM are basically the same as that required for MSAM. However, alternate key access is not supported for ISAM. If there is any question as to whether a file created in a COBOL program is MSAM or ISAM, look for the ASSIGN clause on the SELECT statement in the COBOL ENVIRONMENT DIVISION. If it is assigned to DISC, it is MSAM; if assigned to MASS-STORAGE, ISAM.

The path generation process cannot automatically generate the record and block size parameters in the DBM for ISAM files; therefore, the DBM must be altered manually to supply the record size and block size on the DEFINE File directive before attempting to use the DBM. Also, since the ISAM key cannot be specified as a data item in the record, the user defined key variable must be specified. The following generated line of code in the DBM must be altered:

```
DEFINE  F  FILE-NAME  ISAM  [recsize,blocksize  key-var]
```

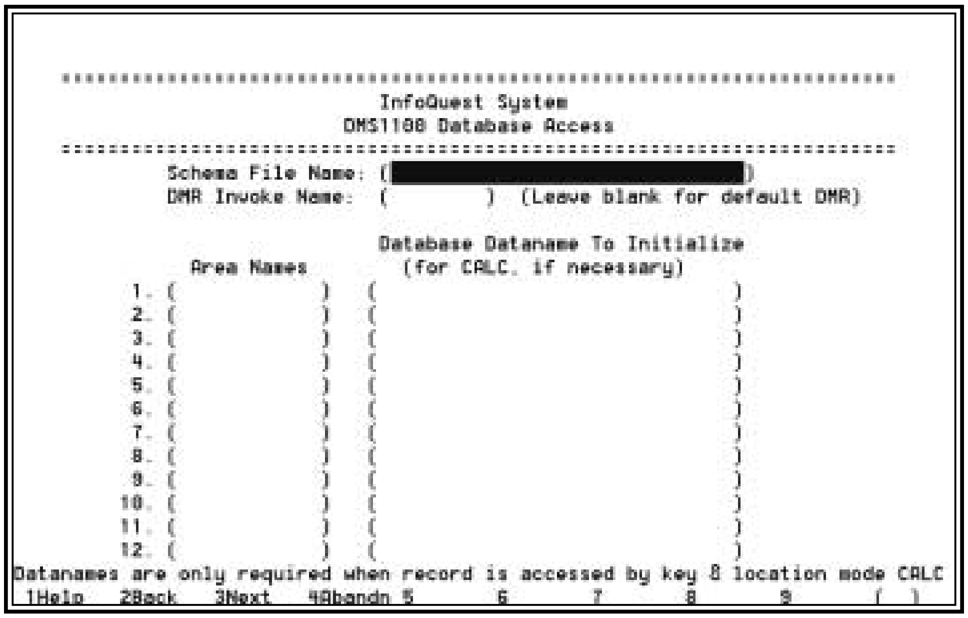
In addition, the “key-var” must be defined prior to the above DEFINE F directive.

Access Mode:

The fourth entry specifies how the file will be accessed. File access will be dictated by the file’s relationship within a path. If a file is the start or root of a path, its access must be Sequential, Random or Dynamic (as required for exact key or range key access of MSAM or ISAM files). If a file is a subordinate to the start or root of the path, it may only be accessed randomly. By these rules then, a sequential file may never appear as a subordinate within the path. To access a direct file sequentially, you must define it as a sequential file. Random and dynamic access is only valid for MSAM and ISAM file types. Direct access is only valid for direct file types.

5.6.2 DMS1100 Access Information

Information on this screen is only required when DMS1100 database access is required as part of the path. The example being shown in this section does not involve DMS access; therefore, this screen is bypassed. To bypass this screen, depress F3; otherwise, enter the fields as described below:



```
*****************************************************************
                        InfoQuest System
                     DMS1100 Database Access
=================================================================
          Schema File Name: (                          )
          DMR Invoke Name:  (        )  (Leave blank for default DMR)

                        Database Dataname To Initialize
            Area Names        (for CALC, if necessary)
        1. (            )  (                            )
        2. (            )  (                            )
        3. (            )  (                            )
        4. (            )  (                            )
        5. (            )  (                            )
        6. (            )  (                            )
        7. (            )  (                            )
        8. (            )  (                            )
        9. (            )  (                            )
       10. (            )  (                            )
       11. (            )  (                            )
       12. (            )  (                            )
Datanames are only required when record is accessed by key & location mode CALC
 1Help   2Back   3Next   4Abandn 5      6      7      8      9      (  )
```

Schema File Name:

The first entry is the schema file name to be used in the Q-LINK INVOKE directive. The schema file may be specified as a full file name (qualifier*filename.), or as a TIP file number.

DMR invoke Name:

The "DMR Invoke Name" may be set so that the DBM will invoke a DMR other than the default DMR. This name corresponds to the DMR name configured in Q-LINK. It can be used to provide a means to access DMS databases under different EXEC application groups. For sites executing DMS under UDS where the schema in an alternate application group is registered as an ALIAS in the default application group, this parameter is not generally required.

Area Names:

The second portion of the screen allows you to list all the required area names. This list must include all areas that may be accessed within the path.

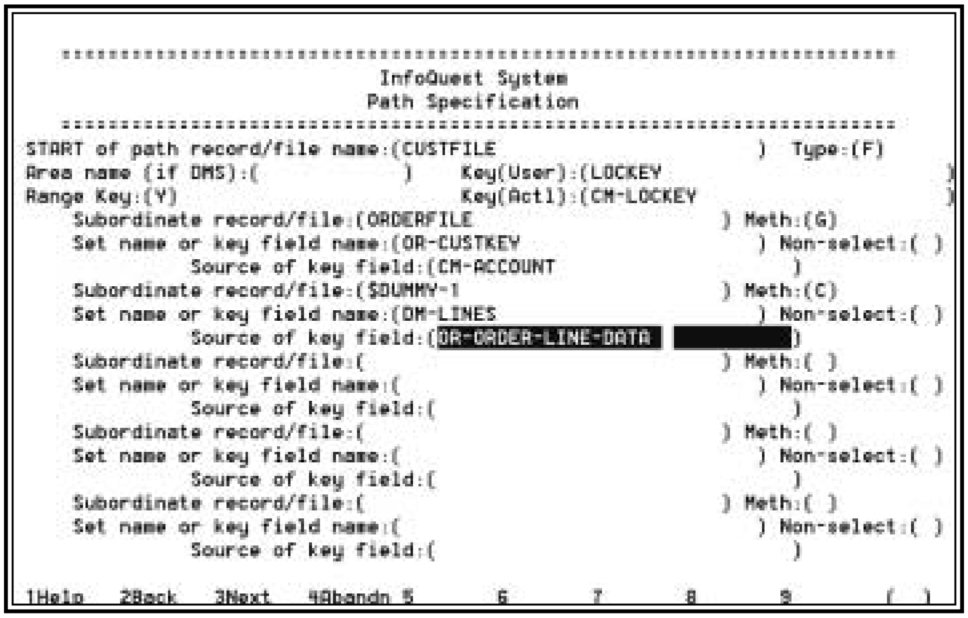
Database Dataname to Initialize:

The second column, “Database Dataname To Initialize”, is used only when a DMS1100 CALC record will be accessed by key, or when an Index Sequential record is to be accessed by key, and the Index Sequential record spans more than one area. The database-dataname specified here will be set to the value of the corresponding area name.

The path generation process does not allow for the access of DMS1100 Direct records by their area key and area name database-datanames. However, if a database-dataname to contain the area name is entered for a Direct record, the completed base DBM must be manually modified (after the path generation process) to initialize the database-dataname for the area key with the appropriate page and record numbers.

5.6.3 Path Specification

The actual path navigation requirements will be specified for all file systems except RDMS 1100 on this screen. For RDMS 1100, enter an “R” in the “Type” field. A fourth screen will be displayed as described in the next subsection.



```
    ======================================================================
                              InfoQuest System
                             Path Specification
    ======================================================================
START of path record/file name:(CUSTFILE                       )  Type:(F)
Area name (if DMS):(            )    Key(User):(LOCKEY                     )
Range Key:(Y)                        Key(Actl):(CM-LOCKEY                  )
    Subordinate record/file:(ORDERFILE                        ) Meth:(G)
    Set name or key field name:(OR-CUSTKEY                       ) Non-select:( )
              Source of key field:(CM-ACCOUNT                      )
    Subordinate record/file:($DUMMY-1                         ) Meth:(C)
    Set name or key field name:(DM-LINES                         ) Non-select:( )
              Source of key field:(OR-ORDER-LINE-DATA              )
    Subordinate record/file:(                                 ) Meth:( )
    Set name or key field name:(                                 ) Non-select:( )
              Source of key field:(                                )
    Subordinate record/file:(                                 ) Meth:( )
    Set name or key field name:(                                 ) Non-select:( )
              Source of key field:(                                )
    Subordinate record/file:(                                 ) Meth:( )
    Set name or key field name:(                                 ) Non-select:( )
              Source of key field:(                                )

1Help   2Back   3Next   4Abandn 5       6       7       8       9       (  )
```

START of path record/file name:

The path always begins with a starting (root) record type or file name. The starting record type or file name may be accessed sequentially or randomly by key. Sequential access of a DMS1100 record implies next-of-area type access.

The first field of the starting record/file entry must be a DMS1100 record type, a PCIOS file name or a MAPPER DTM logical file name as defined in the view.

Type:

The next field, “Type”, is used to tell the path generation process which type the record/file entry is:

D	Specifies that the start of path entry is a DMS-1100 record.
R	Specifies that an RDMS-1100 table will be the root of the path (the start of path entry must be blank). Use of this type, causes the RDMS Environment & Table Relationship Specifications screen to appear.
F	Specifies that the start of path entry is a PCIOS file.
M	Specifies that the start of path entry is a MAPPER logical file (RID) via DTM.

When accessing RDMS data, the RDMS database access may only appear as the starting, or root access. The path generation process does not generate database access SQL, but does generate appropriate RDMS commands in preparation for database access. The SQL commands to actually retrieve data (DECLARE CURSOR and FETCH) are generated at run time from user-entered parameters. DMS-1100 and PCIOS files may be included in the path as subordinate records.

Area Name:

The “Area name” field is used only for DMS1100 records. It names the area in which the record resides.

Key (User) and Key (Actl):

The “Key” fields of the starting record/file entry are only required when the root record/file is to be accessed randomly. When random entry is desired, both the user item name “(User)” and the actual item name as specified in the QINDEX data item index file “(Actl)” must be supplied in the “key” fields. For MSAM files, the key may be the primary key or an alternate key (if defined).

The key item must be defined in the view as a key field that is required for selection. To define the key item, set the K/R-flag to a value of “R” for range access or “K” for keyed access, whichever is desired. Also, set the R/D-flag to a value of “R” for required. See Section 4.6, “View Maintenance”, in the previous chapter.

Range Key:

The “Range Key” option (set it to “Y” if YES) is used when index sequential records are to be accessed within a range of values, thus taking advantage of their physical order by key. This option allows random entry to the first value in the range. Each subsequent record will be processed sequentially until a record with a value greater than the highest value entered is encountered. Range key processing is provided only for DMS1100 INDEX SEQUENTIAL records, MSAM (most commonly used indexed file access), and ISAM (older version single-key indexed sequential file access). For MSAM files, the range key can be the primary key or an alternate key (if defined). If range key processing is desired for DMS1100 INDEX SEQUENTIAL records, the RANGE parameter must be specified on the LOCATION MODE clause of the record in the schema.

Subordinate record/file:

Following the starting record/file entry are five subordinate record/file access entries. Each one is made up of five fields to define how each subordinated record in the hierarchy is to be accessed (see next page).

Meth:

The first field, like the starting record/file, can be a DMS1100 record type, a PCIOS file name or a MAPPER DTM logical file name as defined in the view. The next field, "Meth", describes how the record type or file is to be accessed. The valid methods are:

Code	Method
C	Specifies that access will be to an occurring item group (see 6.13 "Occurs Groups" on page 6-31 for an example).
F	Specifies that a PCIOS MSAM file is to be accessed by its primary key.
G	Specifies that a PCIOS MSAM file is to be accessed by an alternate key
H	Specifies that a DMS1100 CALC or INDEX SEQUENTIAL record can have duplicate keys. This method will generate a loop to retrieve all records having the same key value.
K	Specifies that the DMS1100 record is to be accessed by key.
L	Fetch only the last member of a DMS set.
O	Specifies that the record is to be accessed as an owner of a set. The next field must specify the set name. The previous record on the path must be a member of the set.
S	Specifies default access which is via set. The next entry must specify the valid set name by which the record will be accessed.
T	Fetch only the first record of a DMS set.
U	User defined access can be for any file type where user-written access code is to be inserted into the DBM; e.g., MAPPER access, accessing a DMS1100 record using a database-key retrieved from another record type, access a record that requires a special code to determine key values, etc.

Set name or key field name:

The key field name in a subordinate entry is specified in the next entry field, "Set name or key field name", and the field containing the value to initialize that key is specified in the last entry field (see "Source key field" below).

If a subordinate entry is a PCIOS file, it may not be a SEQuential file. If the subordinate entry is an MSAM file, the key field, “Set name or key field name”, may be the primary key or any defined alternate key.

If a subordinate entry is a DMS1100 record, it may only have a location mode of CALC or INDEXED SEQUENTIAL, or VIA SET. DMS1100 location mode of DIRECT is not supported; however, the DBM may be generated as if the direct record were CALC, then manually modified later to include the necessary code for area key initialization.

If a PCIOS DIRECT file is to be accessed as a subordinate, its key field must be left blank, as it will be accessed by relative key value (not a record key) which will always be stored in the predefined Q-LINK variable X. The source key field for a DIRECT field must be numeric.

Source of key field:

The source key field must be a field in a record/file that appears in a previously specified subordinate entry or the starting (root) record/file entry.

Non-select:

The “Non-select” entry is used to specify that a subordinate record or file is included in the path only because it is required to access another record or file. In this case no selection logic will be generated, but the record or file will be accessed in order to access a subsequent subordinate. A non-select is indicated by entering a “Y”. Any records marked for non-selection should have its fields removed from the view. Also, the last subordinate record in the path should never be marked as a non-select record.

The “Non-select” may also be set to an “F” indicating that when access to a record on a one-to-one relationship results in a NO-FIND, a NULL record occurrence is to be generated. All alphanumeric fields in a NULL record will be filled with spaces, all numeric fields will be filled with zeros, and the DBM logic will not stop traversing the path. Normally, when a NO-FIND occurs, access down the path is stopped and considered a non-select relationship. This option should only be used where the record is related on a one-to-one basis, and the record is not required for access to another subordinate record within the path.

You may enter a “G” in the non-select entry of subordinate records on the path. It works similar to the “F” option. The difference is that ALL records following and including the current record in the path are null-filled if the current record is not found. With the “F” option only the current record is null-filled if its not found.

5.6.4 RDMS 1100 Access

On the RDMS Environment and Table Relationship Specifications screen, space is provided to enter table linkages. Since RDMS does not provide a mechanism to define how one table relates to another, the person setting up the InfoQuest path must provide the relationships. Inter-table relationships, or linkages, are described in the form of table names along with the column name that links that table. These linkages are used to build an RDMS SQL WHERE clause. The WHERE clauses become permanent in the DBM path logic and end user selections are added to these permanent WHEREs.

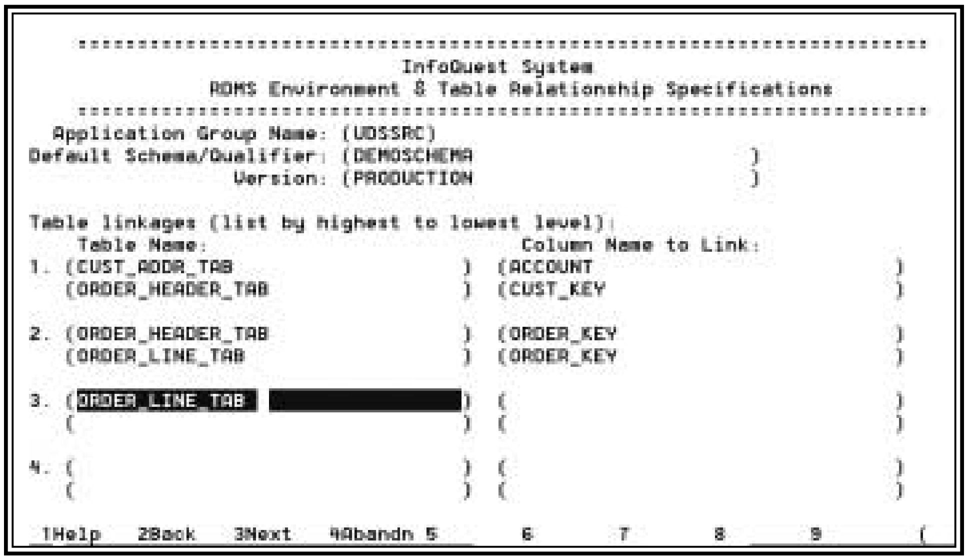
An alternative to linking or joining two or more tables in this manner would be to define an RDMS view and use it like a single table (see the RDMS CREATE VIEW command, 7830 8160, RDMS 1100 SQL Programming Reference).

When more than two tables are to be linked, a hierarchy must be established so that tables are properly related. This hierarchy also prevents unnecessary access to tables when the end user does not require data from all tables covered in the view. How and why the hierarchy is used is best described by the following example.

Assume a database for customer order entry consists of three tables: CUST_ADDR_TAB containing customer account number, name, address, etc.; ORDER_HEADER_TAB containing the customer account, order number, shipping information and other items pertaining to the entire order; and ORDER_LINE_TAB containing the order number, product code, unit price, quantity ordered, etc. A hierarchy can be established among these three tables: ORDER_LINE_TAB is subordinate to ORDER_HEADER_TAB and ORDER_HEADER_TAB is subordinate to CUST_ADDR_TAB. The three tables also have very clear logical relationships: the CUST_ADDR_TAB and ORDER_HEADER_TAB are related (linked) by the customer account number, and ORDER_HEADER_TAB and ORDER_LINE_TAB are related by the order number. Note that no relationship exists between CUST_ADDR_TAB and ORDER_LINE_TAB without ORDER_HEADER_TAB. The hierarchy of access to data in the order entry database can then be assumed to be as follows:

$$\begin{tikzcd}
\textbf{CUST\_ADDR\_TAB} \arrow[dr] & & \\
& \textbf{ORDER\_HEADER\_TAB} \arrow[dr] & \\
& & \textbf{ORDER\_LINE\_TAB}
\end{tikzcd}$$

Since the end user can not be responsible for always specifying the inter-table relationships, InfoQuest's path generation function provides a screen on which to specify the logical relationships between the tables. These relationship specifications are permanently embedded in the path logic. When entering the table linkages, the hierarchy is specified from top to bottom. In our order entry database, the screen entries for table linkages would look as follows:



```
    ======================================================================
                                InfoQuest System
               RDMS Environment & Table Relationship Specifications
    ======================================================================
  Application Group Name: (UDSSRC)
Default Schema/Qualifier: (DEMOSCHEMA                        )
                 Version: (PRODUCTION                        )

Table linkages (list by highest to lowest level):
    Table Name:                          Column Name to Link:
1. (CUST_ADDR_TAB                    ) (ACCOUNT                    )
   (ORDER_HEADER_TAB                 ) (CUST_KEY                   )

2. (ORDER_HEADER_TAB                 ) (ORDER_KEY                  )
   (ORDER_LINE_TAB                   ) (ORDER_KEY                  )

3. (ORDER_LINE_TAB                   ) (                           )
   (                                 ) (                           )

4. (                                 ) (                           )
   (                                 ) (                           )

 1Help   2Back   3Next   4Abandn 5       6       7       8       9       (
```

Application Group Name:

The Application Group Name is the name that was used in the UDS installation. UDSSRC is the standard default.

Default Schema/Qualifier:

Default Schema/Qualifier is the name that is registered in the UREP 1100 data dictionary.

Version:

The Version is the user-defined name that is defined in the data dictionary. The standard default is PRODUCTION.

Table linkages:

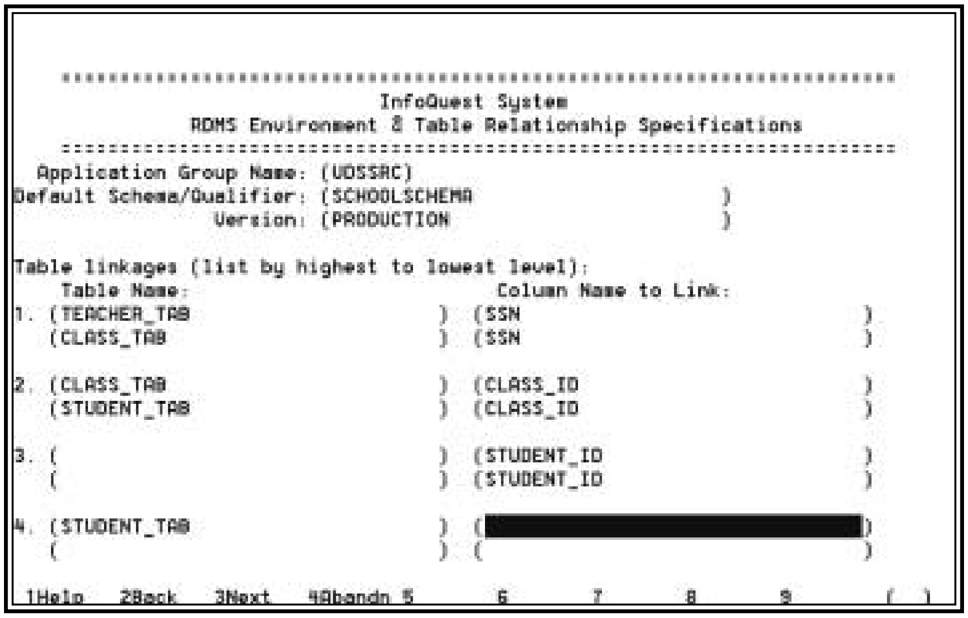
The entries on the remaining part of the screen are for linking tables. The linkages should be specified in highest to lowest order level. For each set, enter a table and column name on the first line. On the next line, enter the table and column name for which they are to be linked. For the table at the bottom of the hierarchy (or for single table access), enter just the table name on the first line. Do not enter a column name or fill in the second line for this table. You can link as many tables as you need. If you fill a screen, you will be asked if you need more entries. If you do, you will be given another blank screen in which to make additional entries.

Note: At least one table or view name must be entered.

In the example on the previous page, if the end user needs access to data at the lowest level of the hierarchy, ORDER_LINE_TAB, then CUST_ADDR_TAB, ORDER_HEADER_TAB and ORDER_LINE_TAB must be linked. If data from ORDER_HEADER_TAB is required, then only CUST_ADDR_TAB and ORDER_HEADER_TAB need to be linked. If only data from CUST_ADDR_TAB is needed, no inter-table links are needed. When InfoQuest generates the SQL DECLARE CURSOR statement, the appropriate WHERE syntax required to join tables will be inserted in front of any WHERE syntax generated by the end user's selection entries.

Notice that the third entry, ORDER_LINE_TAB, has no "Column Name to Link". It is at the bottom of the hierarchy; i.e., it does not link to any table below it. However, it must be named here in order to generate the appropriate WHERE clause.

If two tables are linked by multiple columns (i.e., a FOREIGN KEY clause has more than one column name specified), the additional column names should be entered on subsequent table linkage lines, but without repeating the table name (see below).



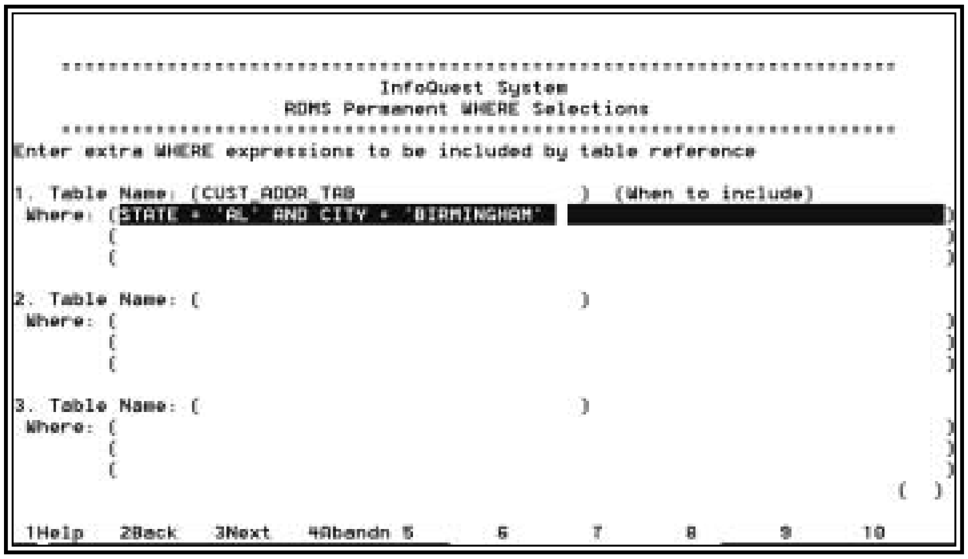
```
   *****************************************************************
                         InfoQuest System
              RDMS Environment & Table Relationship Specifications
   =================================================================
  Application Group Name: (UDSSRC)
Default Schema/Qualifier: (SCHOOLSCHEMA                    )
                 Version: (PRODUCTION                      )

Table linkages (list by highest to lowest level):
    Table Name:                         Column Name to Link:
1. (TEACHER_TAB                     )  (SSN                        )
   (CLASS_TAB                       )  (SSN                        )

2. (CLASS_TAB                       )  (CLASS_ID                   )
   (STUDENT_TAB                     )  (CLASS_ID                   )

3. (                                )  (STUDENT_ID                 )
   (                                )  (STUDENT_ID                 )

4. (STUDENT_TAB                     )  (                           )
   (                                )  (                           )

 1Help   2Back   3Next   4Abandn 5      6       7       8       9       (  )
```

In the example above, CLASS_TAB and STUDENT_TAB are linked based on the values in two columns: CLASS_ID and STUDENT_ID. Both make up the primary key of CLASS_TAB and are named as the foreign key of STUDENT_TAB.

When the RDMS DBM is built, InfoQuest inserts *RECnnnn definitions for all tables that are defined in the QINDEX file. This is done because it is not possible to determine, ahead of time, which tables are going to be used during the actual generation of each request. When a request is generated, InfoQuest generates a DECLARE CURSOR SQL command with the table names found on the *RECnnnn definitions. You should delete the *RECnnnn line from the DBM for any table not required. By deleting that line, the table will be omitted from the DECLARE CURSOR.

Also note that the order in which the tables are listed (the order of the *RECnnnn entries) can be changed, if it will make RDMS run faster.

Once the RDMS Environment and Table Relationship Specifications have been completed, press F3 and the optional "RDMS Permanent WHERE Selections" screen will appear as shown below:


```
   **************************************************************************
                             InfoQuest System
                    RDMS Permanent WHERE Selections
   **************************************************************************
Enter extra WHERE expressions to be included by table reference

1. Table Name: (CUST_ADDR_TAB                      )  (When to include)
 Where: (STATE = 'AL' AND CITY = 'BIRMINGHAM'                               )
        (                                                                   )
        (                                                                   )

2. Table Name: (                                   )
 Where: (                                                                   )
        (                                                                   )
        (                                                                   )

3. Table Name: (                                   )
 Where: (                                                                   )
        (                                                                   )
        (                                                                   )
                                                                        (  )
 1Help   2Back   3Next   4Abandn 5     6       7       8       9      10
```

The Permanent WHERE Selections are optional, but may be used to improve overall performance by specifying index values for selection into the various RDMS tables. In the above example, STATE and CITY are defined as secondary indexes into the CUST_ADDR_TAB table (at the top of the hierarchy).

Table Name:

Enter the table name to which the WHERE clauses will be attached.

Where:

The WHERE clauses are linked by table. This means that they will not be used unless the table is selected in the report. You can enter as many sets of WHERE clauses as needed.

In the above example, the CUST_ADDR_TAB is at the top of the hierarchy and has two secondary indexes defined (STATE and CITY) when the table was created. The above example provides values to be used to randomly enter that table.

When specific items are to be used for selection on the RDMS Permanent WHERE Selection screen, the items can be marked in the view as required for access or required for selection. To mark the item as required for access, set the R/D-flag to a value of "D" for defined (required for access but not selection). We can use this setting in the above example since values (literals) are being placed in a permanent WHERE. If values are not being entered here, you could mark the item as required for selection by setting the R/D flag to "R". In this instance, the end user would be responsible for entering the values when they generate the request. See Section 4.6, "View Maintenance", in the previous chapter.

When entering literals, as in the above example, the literals may be bound by either single ('*literal* ') or double ("*literal* ") quotes.

Notice that the parentheses mark the beginning and end of each field. If parentheses are required to force the conditions to be evaluated at a higher precedence than end user selections, they (the parentheses) must be entered within the delimited field. For example:

```
( ( CUST_ADDR_TAB.STATE = 'GA' AND                           )
(   ORDER_LINE_TAB.QUANTITY > 50                           ) )
```

Only these two parentheses are evaluated as part of the expression.

WHERE syntax is used, as is, by InfoQuest; that is, there is no syntax checking. For this reason, it should be checked and tested carefully.

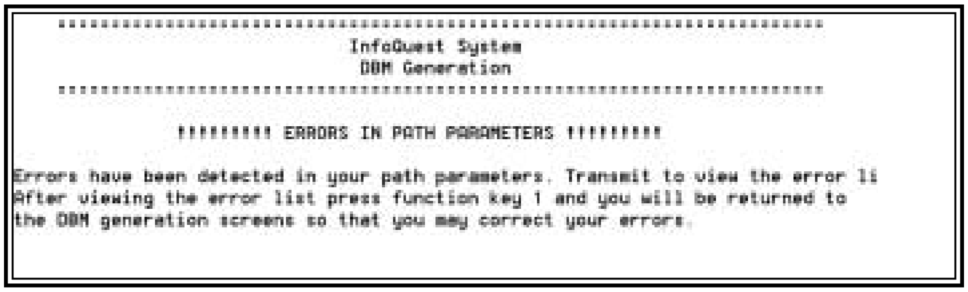
Each WHERE clause is used only if the associated table is needed in the user's request. WHEREs for tables not required are ignored.

Program variables may be used in the RDMS path specification; however, some modification to the DBM will be required. For example, if you wanted the RDMS "Version" name as a variable, you could enter the variable name on the path specification screen and then modify the DBM to use that variable (see Section E.3, "Using Variables on the RDMS Permanent WHERE" and Section E.4, "Using Variables on RDMS USE Commands").

Unlike other database access, InfoQuest generates SQL statements for RDMS to preselect only that data required to produce the end user's desired result. Preselection is possible due to the RDMS implementation of SQL. With other database models, the InfoQuest path logic must retrieve each record, then test whether or not it meets end user selection criteria.

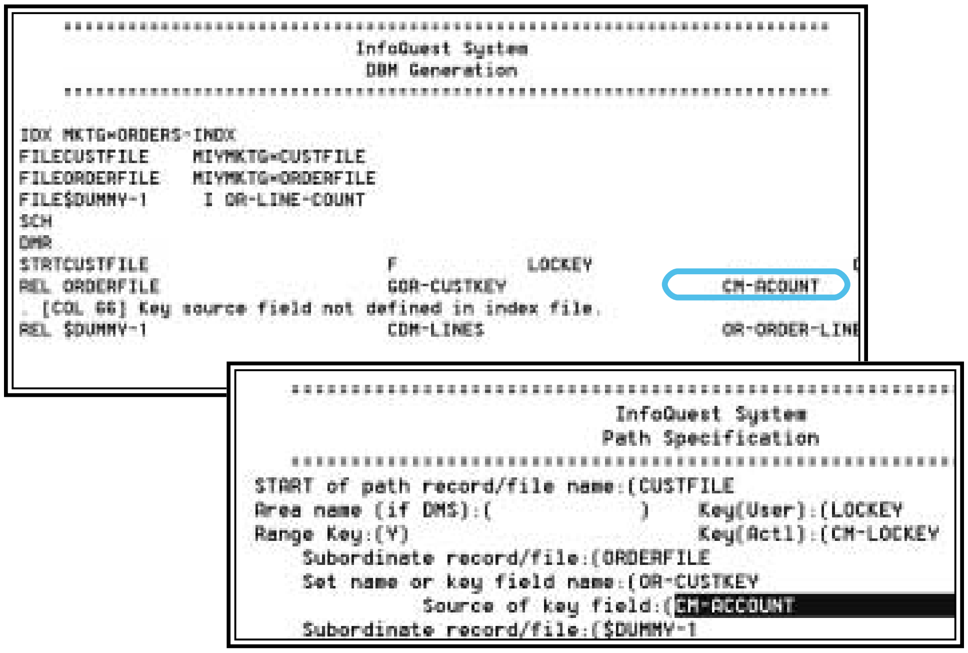
5.6.5 Correcting Path Generation Errors

When errors are detected in the path generation process, they very often can be corrected without the necessity of repeating the entire process. The following screen occurred because an error was made on a previous screen:



```
    ************************************************************************
                              InfoQuest System
                               DBM Generation
    ************************************************************************

               !!!!!!!!! ERRORS IN PATH PARAMETERS !!!!!!!!!

Errors have been detected in your path parameters. Transmit to view the error li
After viewing the error list press function key 1 and you will be returned to
the DBM generation screens so that you may correct your errors.
```

Path generation, while editing the parameters, determined that an incorrect field name (CM-ACCOUNT) was entered for the key. When an error is detected, press F1 and you will be returned to the first screen of DBM generation parameters. The error in the example below occurred due to a misspelled data name on the path specification screen.



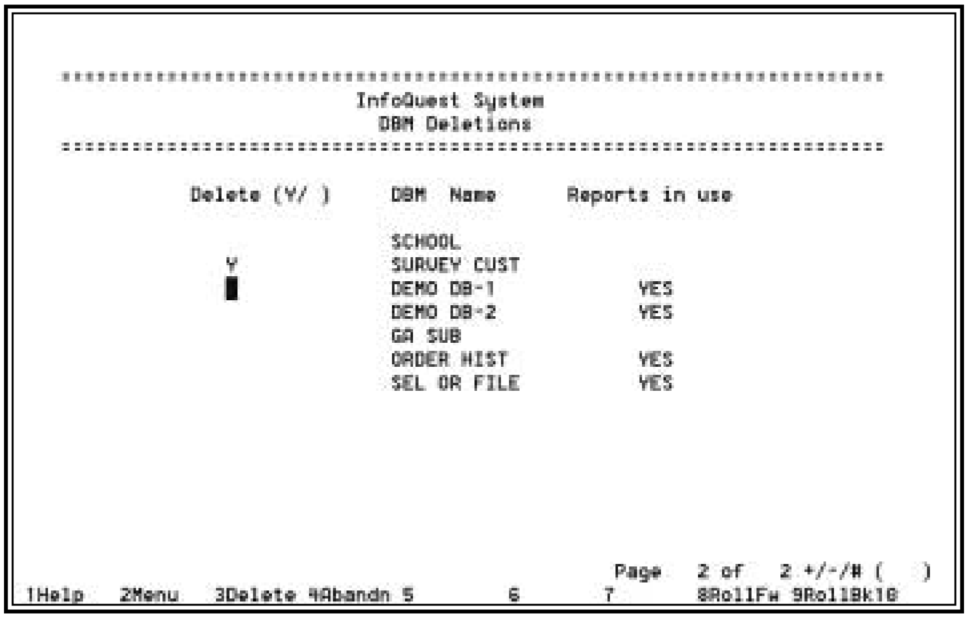
```
    ************************************************************************
                              InfoQuest System
                               DBM Generation
    ************************************************************************

IDX MKTG*ORDERS-INDX
FILECUSTFILE    MIYMKTG*CUSTFILE
FILEORDERFILE   MIYMKTG*ORDERFILE
FILE$DUMMY-1     I OR-LINE-COUNT
SCH
DMR
STRTCUSTFILE                     F            LOCKEY                          C
REL ORDERFILE                    GOR-CUSTKEY                  CM-ACOUNT
. [COL 66] Key source field not defined in index file.
REL $DUMMY-1                     CDM-LINES                    OR-ORDER-LINE
```

```
    *************************************************************
                         InfoQuest System
                        Path Specification
    *************************************************************
START of path record/file name:(CUSTFILE
Area name (if DMS):(             )    Key(User):(LOCKEY
Range Key:(Y)                         Key(Actl):(CM-LOCKEY
    Subordinate record/file:(ORDERFILE
    Set name or key field name:(OR-CUSTKEY
              Source of key field:(CM-ACCOUNT
    Subordinate record/file:($DUMMY-1
```

When a parameter is in error, the column on which the parameter begins will be displayed along with the error message immediately below the line in error. Notice that the parameter was corrected to read, “CM-ACCOUNT”. Then by keying Function 3 (F3) at the keyboard, all parameters will be re-evaluated.

5.7 DBM Deletion

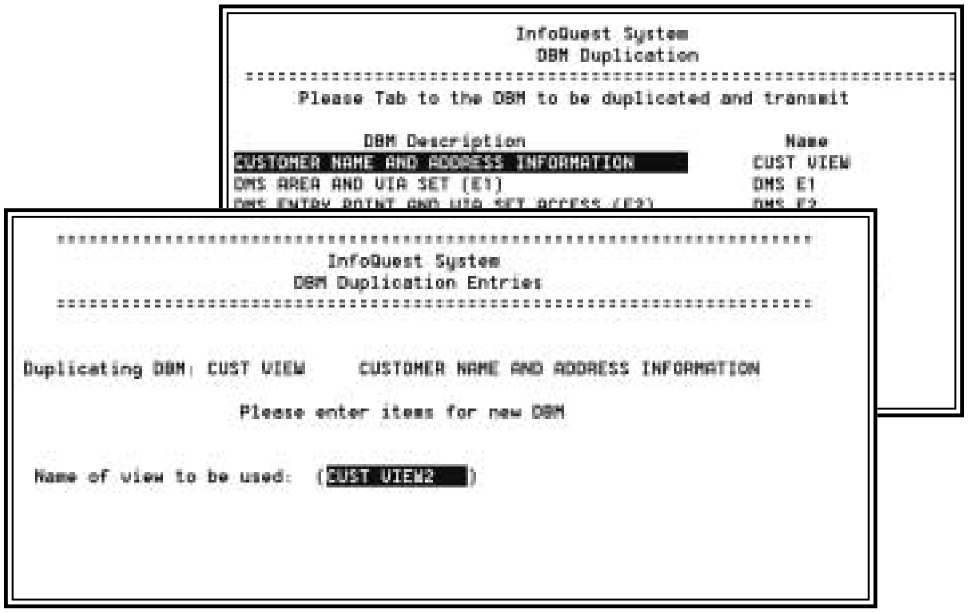
Existing DBMs may be deleted by selecting “DBM Deletion” from the “DBM Generation and Maintenance” screen. The following screen allows you to mark the DBM(s) to be deleted with a “Y”. If there are more DBMs than will fit on one screen, use a plus (+) or minus (-) sign in the bottom right-hand position of the screen to scroll forward or backward through the DBM list. Use the transmit key for scrolling and the F3 key to complete the deletion process.



```
=====================================================================
                         InfoQuest System
                           DBM Deletions
=====================================================================

           Delete (Y/ )     DBM  Name      Reports in use

                            SCHOOL
              Y             SURVEY CUST
              █             DEMO DB-1            YES
                            DEMO DB-2            YES
                            GA SUB
                            ORDER HIST           YES
                            SEL OR FILE          YES




                                        Page   2 of   2 +/-/# (   )
1Help   2Menu   3Delete 4Abandn 5       6       7       8RollFw 9RollBk10
```

5.8 DBM Duplication

DBMs may be duplicated from existing DBMs by selecting “DBM Duplication” from the “DBM Generation and Maintenance” screen. Tab to the DBM to be duplicated and transmit as shown below.



```
                    InfoQuest System
                     DBM Duplication
==================================================================
     Please Tab to the DBM to be duplicated and transmit

            DBM Description                          Name
CUSTOMER NAME AND ADDRESS INFORMATION               CUST VIEW
DMS AREA AND VIA SET (E1)                           DMS E1
DMS ENTRY POINT AND VIA SET ACCESS (E2)             DMS E2
```

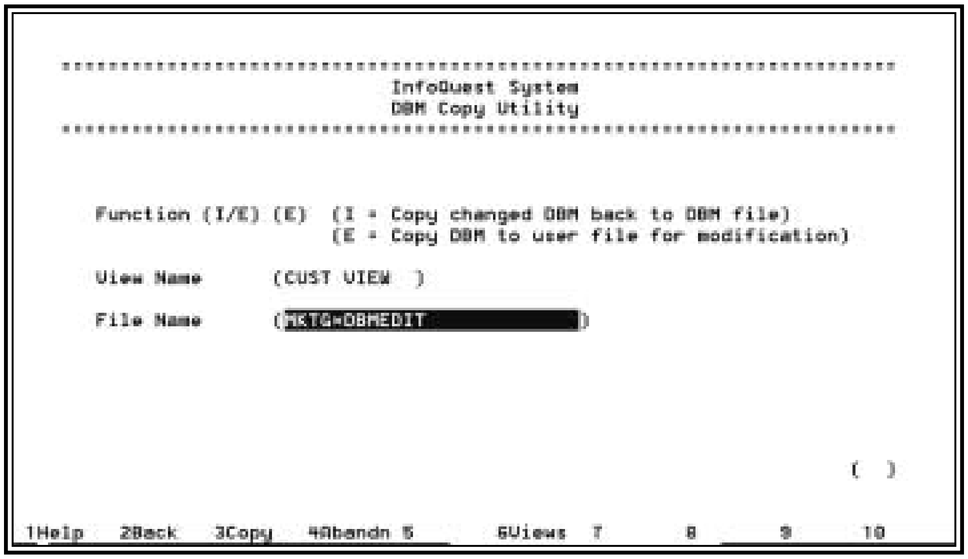
```
=====================================================================
                      InfoQuest System
                   DBM Duplication Entries
=====================================================================

Duplicating DBM: CUST VIEW     CUSTOMER NAME AND ADDRESS INFORMATION

                  Please enter items for new DBM

 Name of view to be used:  (CUST VIEW2     )
```

Duplicating DBMs in this manner can save time and effort when the original DBM contains extensive custom program code and you need to duplicate the path for a different view.

5.9 DBM Copy Utility

The DBM Copy Utility provides the means to copy an existing DBM to and from an SDF file. This utility is used when it is necessary to tailor a DBM for customized path navigational logic that involves manually altering the generated DBM (see Appendix E, “Customized DBMs”).

The DBM Copy Utility can only be executed from the application in which the DBM resides. If you have been working in the InfoQuest Administration application, you will need to make sure you are registered as a user of the DBM’s application and select that application in IQMNT prior to performing the DBM Copy Utility.



```
*************************************************************************
                            InfoQuest System
                            DBM Copy Utility
*************************************************************************


  Function (I/E) (E)  (I = Copy changed DBM back to DBM file)
                      (E = Copy DBM to user file for modification)

  View Name       (CUST VIEW  )

  File Name       (MKTG*DBMEDIT             )



                                                              (  )

1Help   2Back   3Copy   4Abandn 5      6Views  7      8      9      10
```

Function:

There are two functions of the copy utility. The E-function allows the DBM to be exported from IQMNT to an SDF file where it may be modified or viewed with an editor. After the DBM has been altered, the I-function imports the DBM back into IQMNT.

View Name:

The VIEW Name must be a valid view name as entered when the view was generated. When selecting a view, you may also use the F6 key to display the list of views previously generated for the application, tab to the desired view and press transmit.

File Name:

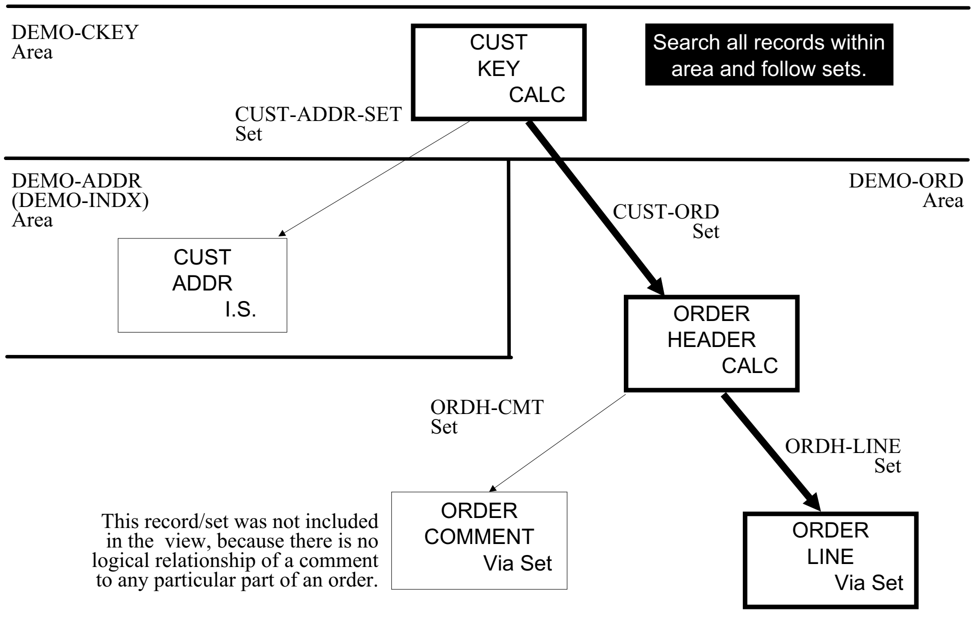
The file name is any standard OS 2200 SDF file name. The file is a data file that must have been previously cataloged.

Chapter 6 Path Generation Examples

In this chapter, we will discuss the generation of path navigational logic for several different types of data structures, including DMS1100, PCIOS, RDMS 1100 and others.

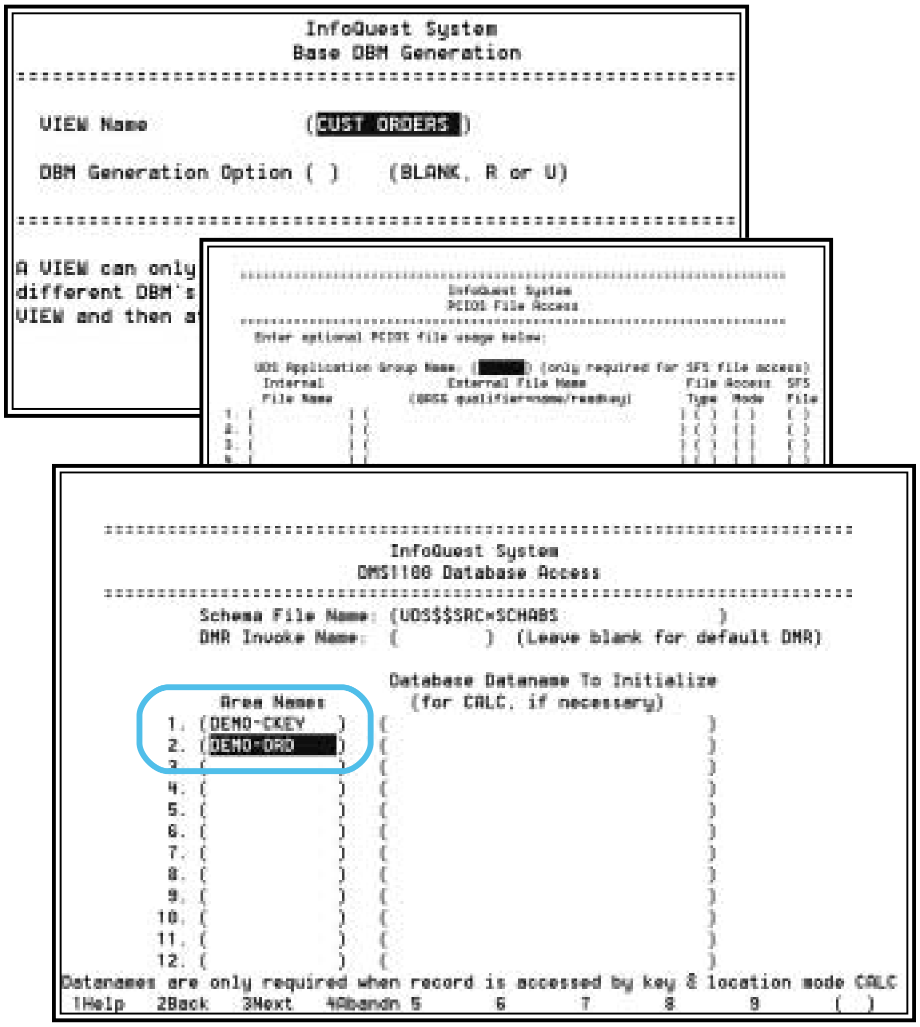
6.1 DMS: Area and Via Set Access

The access path for this first example will include a search of the area for all CUST-KEY-RECs (the schema listing for this and other DMS examples may be found in Appendix B, "DEMO Schema"). For each CUST-KEY-REC owner, we will traverse the CUST-ORD set accessing the ORDER-HEADER-REC members. In addition, each ORDER-HEADER-REC is an owner of the ORDH-LINE set; therefore, we will access each ORDER-LINE-REC member of the ORDH-LINE set.


DEMO-CKEY Area
CUST KEY CALC
Search all records within area and follow sets.
CUST-ADDR-SET Set
DEMO-ADDR (DEMO-INDX) Area
CUST ADDR I.S.
CUST-ORD Set
DEMO-ORD Area
ORDER HEADER CALC
ORDH-CMT Set
ORDH-LINE Set
This record/set was not included in the view, because there is no logical relationship of a comment to any particular part of an order.
ORDER COMMENT Via Set
ORDER LINE Via Set

DMS: Area and Via Set Access

On this and the following page, we see four of the path generation screens used to establish the navigational logic for DBMs. After selecting a previously generated view and providing a brief identifying description for the DBM, we will skip the second screen (by depressing F3) since there are no PCIOS files to be accessed in this example.



```
                    InfoQuest System
                  Base DBM Generation
=============================================================

  VIEW Name               (CUST ORDERS )

  DBM Generation Option ( )    (BLANK, R or U)

=============================================================

A VIEW can only
different DBM's
VIEW and then a
```

```
==================================================================
                        InfoQuest System
                        PCIOS File Access
==================================================================
  Enter optional PCIOS file usage below:

  UDS Application Group Name: (      ) (only required for SFS file access)
   Internal                External File Name            File Access  SFS
   File Name           (UDSS qualifier*name/readkey)     Type  Mode   File
 1. (              ) (                                  ) ( ) ( )    ( )
 2. (              ) (                                  ) ( ) ( )    ( )
 3. (              ) (                                  ) ( ) ( )    ( )
 4. (              ) (                                  ) ( ) ( )    ( )
```

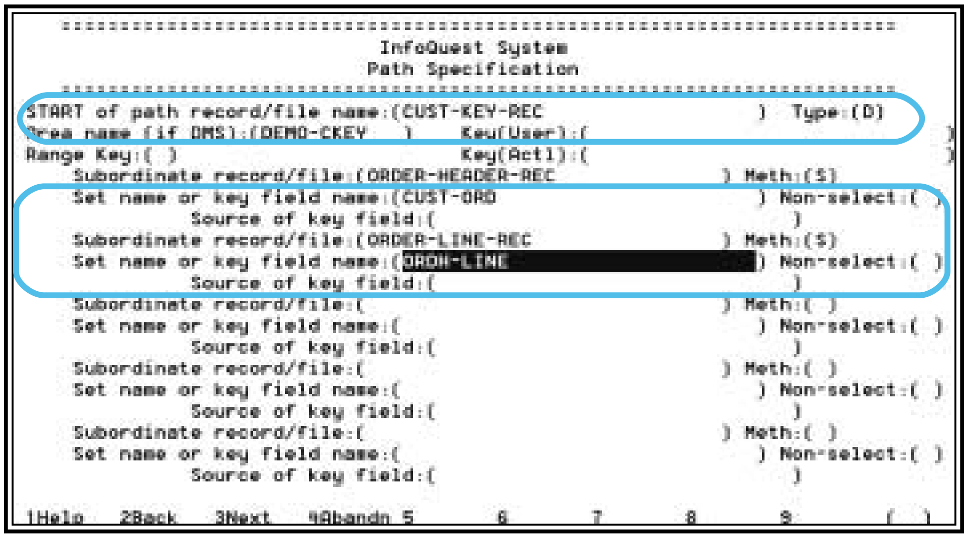
```
    =======================================================================
                               InfoQuest System
                            DMS1100 Database Access
    =======================================================================
              Schema File Name: (UDS$$SRC*SCHABS                    )
              DMR Invoke Name:  (        )  (Leave blank for default DMR)

                                 Database Datename To Initialize
                   Area Names      (for CALC, if necessary)
               1. (DEMO-CKEY   ) (                              )
               2. (DEMO-ORD    ) (                              )
               3. (            ) (                              )
               4. (            ) (                              )
               5. (            ) (                              )
               6. (            ) (                              )
               7. (            ) (                              )
               8. (            ) (                              )
               9. (            ) (                              )
              10. (            ) (                              )
              11. (            ) (                              )
              12. (            ) (                              )
Datenames are only required when record is accessed by key & location mode CALC
 1Help   2Back   3Next   4Abandn 5       6       7       8       9       ( )
```

On the DMS1100 Database Access screen we will enter the schema file name (or TIP schema file number) and the areas to be opened in the DBM. The "DMR Invoke Name" is used to link the DBM to a DMR in an alternate EXEC application group (see Section 5.6.2, "DMS1100 Access Information"). In this example, the default DMR is being used.

For this example, we will take advantage of the hierarchical or logical database structure to navigate from an owner record type to the member record types subordinate in the structure utilizing the logical set linkages. However, since the DBM is to support the end users' requirements to scan through a large amount of data, we will search for owner data in physical page number sequence. This area scan will give us the most efficient means of accessing the owner data; i.e., in a single pass of the owner area. Notice that no database datanames were entered for the DEMO-CKEY area since the CUST-KEY-REC is not being accessed by its CALC key. The same is true for the DEMO-ORD area. In this area, the ORDER-HEADER-RECs are being accessed via the CUST-ORD set.

The efficiency of member record access will depend on the physical organization of the member areas. In the example, the first member record type to be accessed (ORDER-HEADER-REC) is a CALC record. When CALC records are stored, their physical location is determined by the values of their CALC key(s) and not by their logical location in a set occurrence. For this reason, it is very unlikely that any two members within a set occurrence will appear on the same physical page. Here we can expect some re-accessing of pages when chasing the member records in the CUST-ORD set. On the other hand, the members of the ORDH-LINE set occurrence should be clustered close together since they are stored via a primary set. Hence, all the members of a set occurrence can be accessed with a minimum number of physical I/Os.

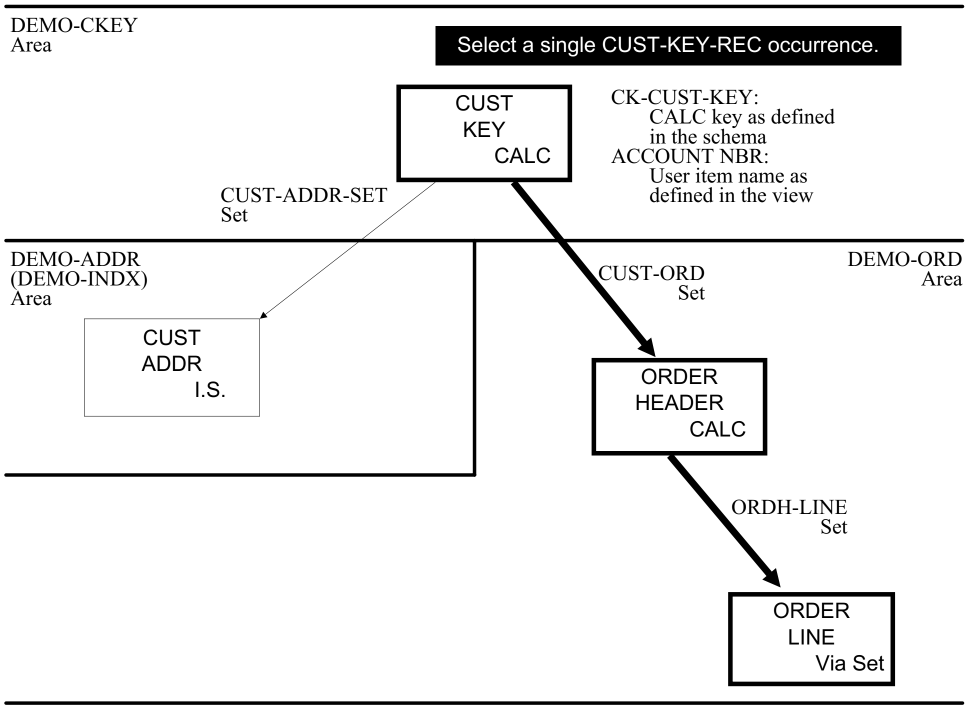
The following screen illustrates the path previously discussed. The CUST-KEY-REC is the start, or root, of the path which implies that the area will be read sequentially from beginning to end. The first subordinate entry specifies that the ORDER-HEADER-REC is to be accessed using the set linkages (Meth: S) in the CUST-ORD set. Likewise, the ORDER-LINE-REC record will be accessed through the ORDH-LINE set.



```
    ==========================================================================
                               InfoQuest System
                             Path Specification
    ==========================================================================
START of path record/file name:(CUST-KEY-REC                         )  Type:(D)
Area name (if DMS):(DEMO-CKEY    )     Key(User):(                              )
Range Key:( )                          Key(Actl):(                              )
    Subordinate record/file:(ORDER-HEADER-REC                   ) Meth:($)
    Set name or key field name:(CUST-ORD                           ) Non-select:( )
              Source of key field:(                                  )
    Subordinate record/file:(ORDER-LINE-REC                     ) Meth:($)
    Set name or key field name:(ORDH-LINE                          ) Non-select:( )
              Source of key field:(                                  )
    Subordinate record/file:(                                   ) Meth:( )
    Set name or key field name:(                                   ) Non-select:( )
              Source of key field:(                                  )
    Subordinate record/file:(                                   ) Meth:( )
    Set name or key field name:(                                   ) Non-select:( )
              Source of key field:(                                  )
    Subordinate record/file:(                                   ) Meth:( )
    Set name or key field name:(                                   ) Non-select:( )
              Source of key field:(                                  )

1Help   2Back   3Next   4Abandn 5       6       7       8       9          ( )
```

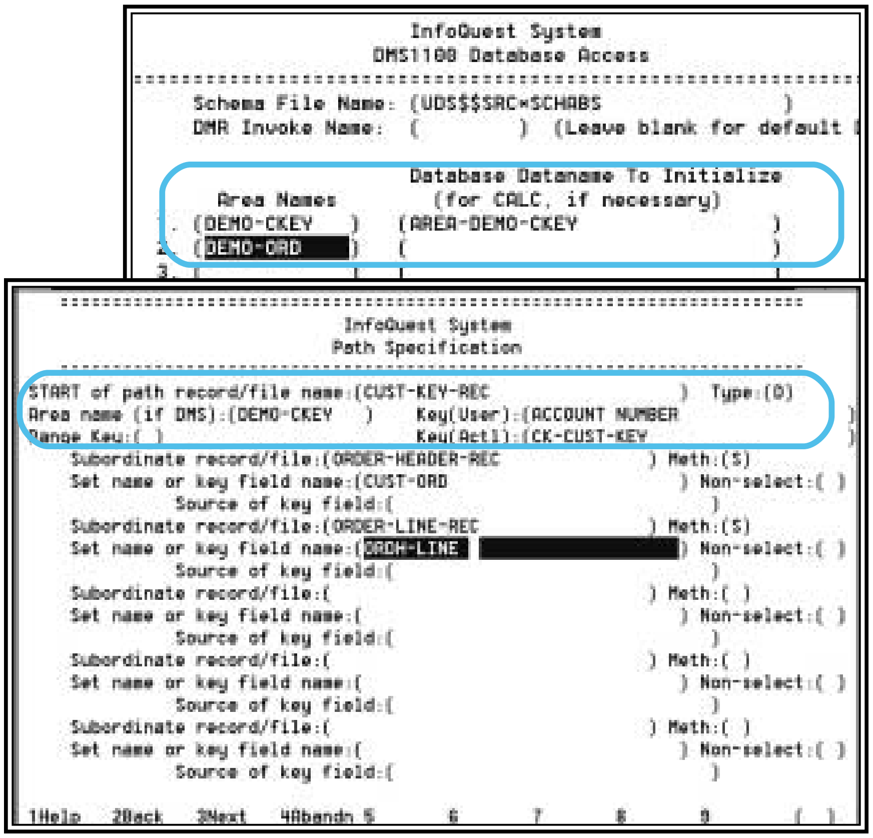
6.2 DMS: Entry Point and Via Set Access

The access path for this example will start with a single occurrence of an entry point record (root record). This is only possible when the root record is CALC or INDEX SEQUENTIAL. The root record, CUST-KEY-REC, will be accessed by its CALC key. This method is useful when it is possible to limit the number of records scanned to find the single entry point record. Comparing this method to the first example, we will access only one CUST-KEY-REC as opposed to all of the CUST-KEY-RECs in the area. For the single CUST-KEY-REC owner accessed, we will traverse the CUST-ORD set accessing the ORDER-HEADER-REC members. In addition, each ORDER-HEADER-REC is an owner of the ORDH-LINE set; therefore, we will access each ORDER-LINE-REC member of the ORDH-LINE set.


DEMO-CKEY Area
Select a single CUST-KEY-REC occurrence.
CUST KEY CALC
CK-CUST-KEY: CALC key as defined in the schema
ACCOUNT NBR: User item name as defined in the view
CUST-ADDR-SET Set
DEMO-ADDR (DEMO-INDX) Area
CUST ADDR I.S.
CUST-ORD Set
DEMO-ORD Area
ORDER HEADER CALC
ORDH-LINE Set
ORDER LINE Via Set

For this example, we will take advantage of the hierarchical, or logical database structure to navigate from an owner record type to the member record types subordinate in the structure utilizing the logical set linkages. However, since there is a large amount of data and we wish to limit the amount of data searched, we will require that the user search for a specific owner by access key. The access key will limit the search to a single owner and only the member records, lower in the hierarchy, that are logically linked to that owner.

Since the CUST-KEY-REC is to be accessed by its CALC key, the database dataname must be provided in order for the DBM to initialize the area name. In this example, AREA-DEMO-CKEY will be set to “DEMO-CKEY”.



```
                           InfoQuest System
                        DMS1100 Database Access
=====================================================================
    Schema File Name: (UDS$$SRC*SCHABS                  )
    DMR Invoke Name:  (         )  (Leave blank for default

                         Database Dataname To Initialize
      Area Names           (for CALC, if necessary)
 1. (DEMO-CKEY   )    (AREA-DEMO-CKEY                     )
 2. (DEMO-ORD    )    (                                   )
 3. (            )    (                                   )
```

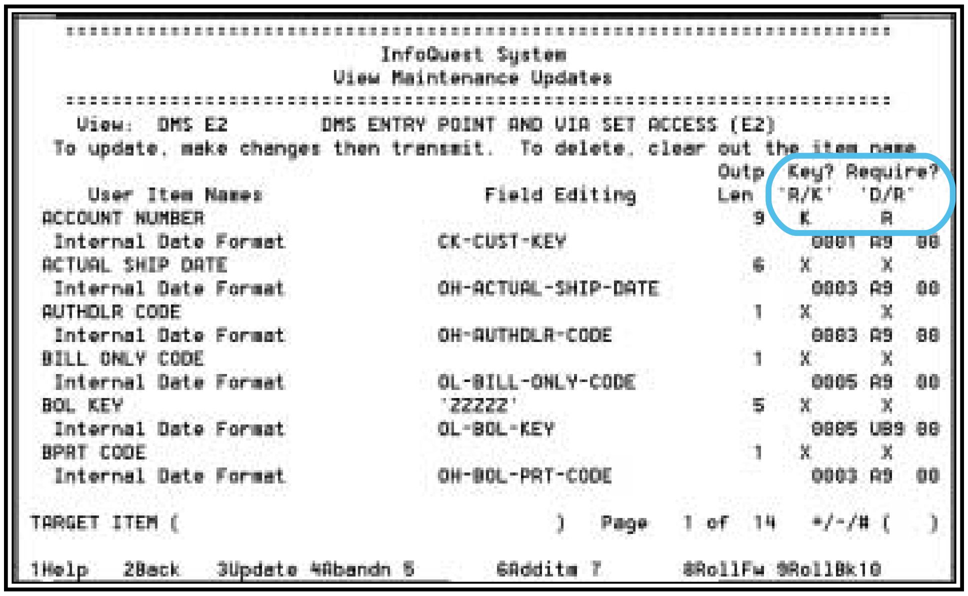
```
 ======================================================================
                          InfoQuest System
                         Path Specification
 ----------------------------------------------------------------------
START of path record/file name:(CUST-KEY-REC                  )  Type:(D)
Area name (if DMS):(DEMO-CKEY   )    Key(User):(ACCOUNT NUMBER          )
Range Key:( )                        Key(Actl):(CK-CUST-KEY             )
    Subordinate record/file:(ORDER-HEADER-REC              ) Meth:(S)
    Set name or key field name:(CUST-ORD                     ) Non-select:( )
             Source of key field:(                           )
    Subordinate record/file:(ORDER-LINE-REC                ) Meth:(S)
    Set name or key field name:(ORDH-LINE                    ) Non-select:( )
             Source of key field:(                           )
    Subordinate record/file:(                              ) Meth:( )
    Set name or key field name:(                             ) Non-select:( )
             Source of key field:(                           )
    Subordinate record/file:(                              ) Meth:( )
    Set name or key field name:(                             ) Non-select:( )
             Source of key field:(                           )
    Subordinate record/file:(                              ) Meth:( )
    Set name or key field name:(                             ) Non-select:( )
             Source of key field:(                           )

1Help   2Back   3Next   4Abandn 5      6      7      8      9      ( )
```

The previous screen illustrates that a single CUST-KEY-REC will be the start or root of the path. The two fields required for keyed access are “Key(user)” and “Key(Actl)”. “Key(user)” must contain the view item name as it appears to the user. “Key(Actl)” must contain the item name as specified in the schema.

As in the first example, the first subordinate entry specifies that the ORDER-HEADER-REC is to be accessed using the set linkages (Meth: S) in the CUST-ORD set. Likewise, the ORDER-LINE-REC record will be accessed through the ORDH-LINE set.

DMS: Entry Point and Via Set Access

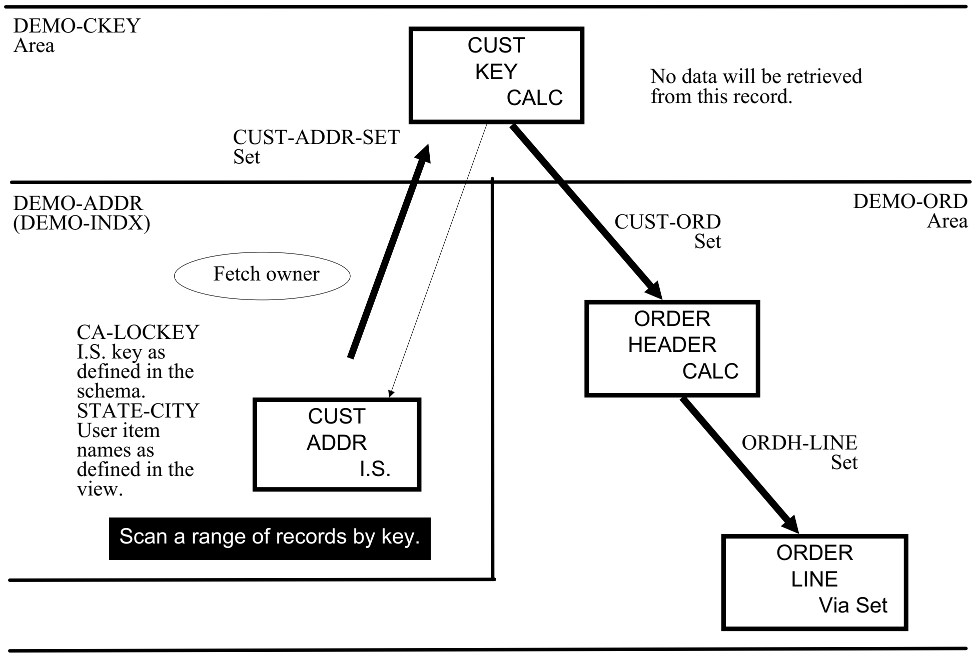
When keyed access is used, as in the above example, the item specified as the key must be flagged as an access key (“K”) and as an item required for selection (“R”) in the view. The key can be set on the “View Maintenance Updates” screen as shown below (see “View Maintenance”, page 4-11):



```
 ======================================================================
                            InfoQuest System
                        View Maintenance Updates
 ======================================================================
   View:  DMS E2        DMS ENTRY POINT AND VIA SET ACCESS (E2)
  To update, make changes then transmit.  To delete, clear out the item name
                                                     Outp  Key? Require?
     User Item Names                 Field Editing    Len  'R/K'  'D/R'
 ACCOUNT NUMBER                                          9   K      R
  Internal Date Format          CK-CUST-KEY                0001 A9   00
 ACTUAL SHIP DATE                                        6   X      X
  Internal Date Format          OH-ACTUAL-SHIP-DATE        0003 A9   00
 AUTHDLR CODE                                            1   X      X
  Internal Date Format          OH-AUTHDLR-CODE            0003 A9   00
 BILL ONLY CODE                                          1   X      X
  Internal Date Format          OL-BILL-ONLY-CODE          0005 A9   00
 BOL KEY                        'ZZZZZ'                  5   X      X
  Internal Date Format          OL-BOL-KEY                 0005 UB9 00
 BPRT CODE                                               1   X      X
  Internal Date Format          OH-BOL-PRT-CODE            0003 A9   00

TARGET ITEM (                                  )  Page   1 of  14   +/-/# (   )

1Help   2Back   3Update 4Abandn 5      6Additm 7       8RollFw 9RollBk10
```

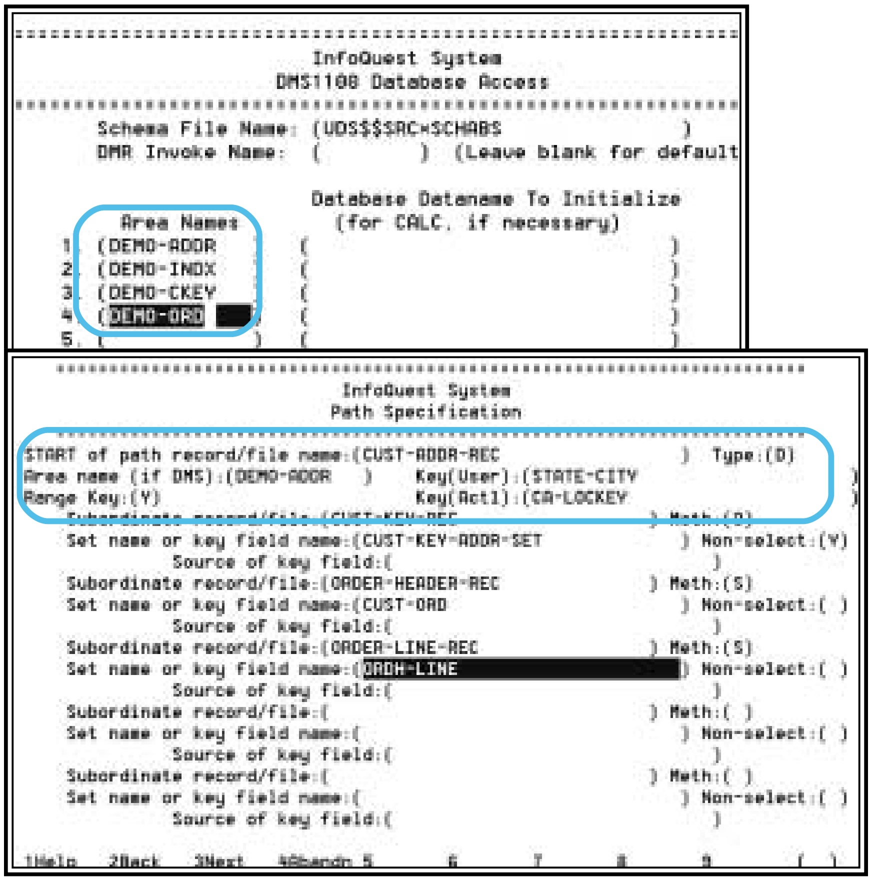
6.3 DMS: Range Key, Owner and Via Set Access

The root record for this DBM will be the CUST-ADDR-REC record. However, not all CUST-ADDR-RECs will be accessed. This example, like the second, limits the total number of records accessed for efficiency. It will search CUST-ADDR-RECs within a range of values and will take advantage of their physical order by key.


DEMO-CKEY Area
CUST KEY CALC
No data will be retrieved from this record.
CUST-ADDR-SET Set
DEMO-ADDR (DEMO-INDX)
CUST-ORD Set
DEMO-ORD Area
Fetch owner
CA-LOCKEY I.S. key as defined in the schema. STATE-CITY User item names as defined in the view.
ORDER HEADER CALC
CUST ADDR I.S.
ORDH-LINE Set
Scan a range of records by key.
ORDER LINE Via Set

DMS: Range Key, Owner and Via Set Access

The CUST-ADDR-REC's location mode is INDEX SEQUENTIAL which allows random entry to the first value in the range (Note: the associated index area name, DEMO-INDX, must be supplied on the Database Access screen). Each subsequent record will be processed sequentially until a record with a value greater than the highest value entered is encountered.

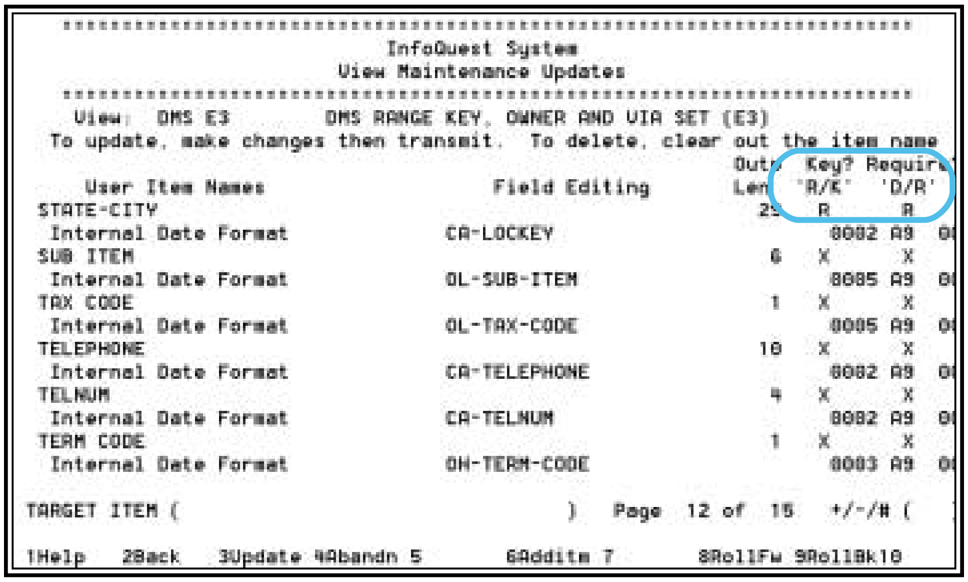


```
==================================================================
                      InfoQuest System
                   DMS1100 Database Access
******************************************************************
      Schema File Name: (UDS$$SRC*SCHABS                  )
      DMR Invoke Name:  (         )  (Leave blank for default

                        Database Dataname To Initialize
          Area Names      (for CALC, if necessary)
     1. (DEMO-ADDR    )  (                               )
     2. (DEMO-INDX    )  (                               )
     3. (DEMO-CKEY    )  (                               )
     4. (DEMO-ORD     )  (                               )
     5. (             )  (                               )
```

```
    *****************************************************************
                           InfoQuest System
                          Path Specification
    *****************************************************************
START of path record/file name:(CUST-ADDR-REC                ) Type:(D)
Area name (if DMS):(DEMO-ADDR   )    Key(User):(STATE-CITY             )
Range Key:(Y)                        Key(Act1):(CA-LOCKEY              )
    Subordinate record/file:(CUST-KEY-REC                ) Meth:(O)
    Set name or key field name:(CUST-KEY-ADDR-SET            ) Non-select:(Y)
                 Source of key field:(                         )
    Subordinate record/file:(ORDER-HEADER-REC            ) Meth:(S)
    Set name or key field name:(CUST-ORD                     ) Non-select:( )
                 Source of key field:(                         )
    Subordinate record/file:(ORDER-LINE-REC              ) Meth:(S)
    Set name or key field name:(ORDH-LINE                    ) Non-select:( )
                 Source of key field:(                         )
    Subordinate record/file:(                            ) Meth:( )
    Set name or key field name:(                             ) Non-select:( )
                 Source of key field:(                         )
    Subordinate record/file:(                            ) Meth:( )
    Set name or key field name:(                             ) Non-select:( )
                 Source of key field:(                         )

1Help  2Back  3Next  4Abandn 5      6      7      8      9        (  )
```

For each CUST-ADDR-REC accessed, the owner of the CUST-KEY-ADDR-SET will be accessed (FETCH3 OWNER). However, notice on the Path Specification screen that the CUST-KEY-REC record is set for non-selection (Non-Select: Y). This means that the DBM logic will make no attempt to do value selection on the CUST-KEY-REC record. The rest of the path is the same as the previous example.

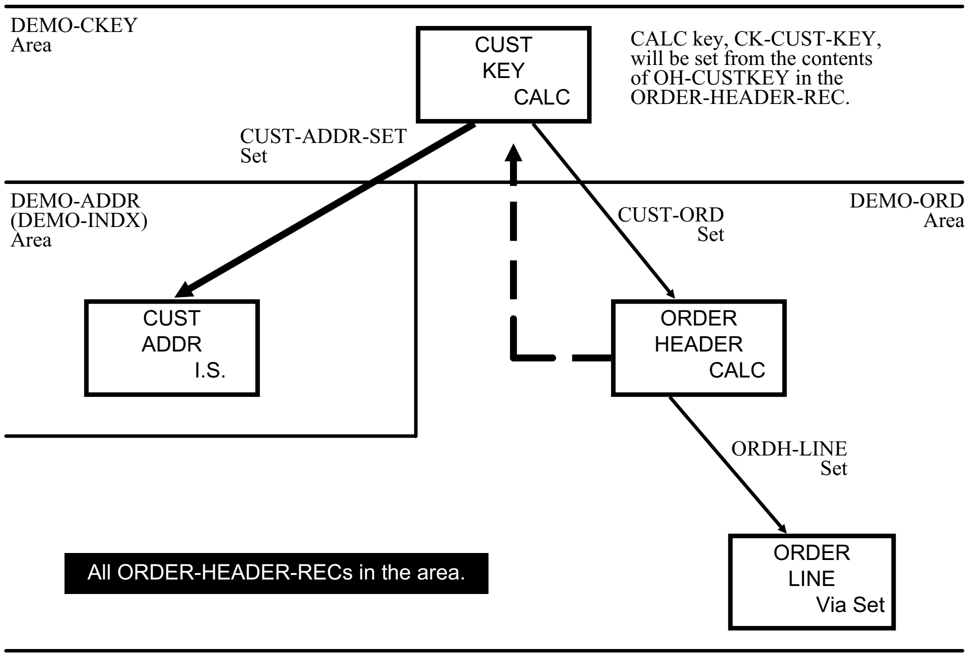
When range key processing is used, as in the above example, the item specified as the key must be flagged as a range key ("R") and as an item required for selection (also "R") in the view. The flagging is accomplished by choosing the VIEW MAINTENANCE function in the IQMNT menu. On the "View Maintenance Updates" screen, the item must be marked with an "R" in both fields as shown below:



```
   ************************************************************************
                              InfoQuest System
                          View Maintenance Updates
   ************************************************************************
    View:  DMS E3        DMS RANGE KEY, OWNER AND VIA SET (E3)
  To update, make changes then transmit.  To delete, clear out the item name
                                                           Outp  Key? Require
     User Item Names                     Field Editing      Len  'R/K'  'D/R'
 STATE-CITY                                                  29    R      R
  Internal Date Format           CA-LOCKEY                         0002 A9  0
 SUB ITEM                                                     6    X      X
  Internal Date Format           OL-SUB-ITEM                       0005 A9  0
 TAX CODE                                                     1    X      X
  Internal Date Format           OL-TAX-CODE                       0005 A9  0
 TELEPHONE                                                   10    X      X
  Internal Date Format           CA-TELEPHONE                      0002 A9  0
 TELNUM                                                       4    X      X
  Internal Date Format           CA-TELNUM                         0002 A9  0
 TERM CODE                                                    1    X      X
  Internal Date Format           OH-TERM-CODE                      0003 A9  0

TARGET ITEM (                                )   Page  12 of  15   +/-/# (

1Help   2Back   3Update 4Abandn 5       6Additm 7       8RollFw 9RollBk10
```

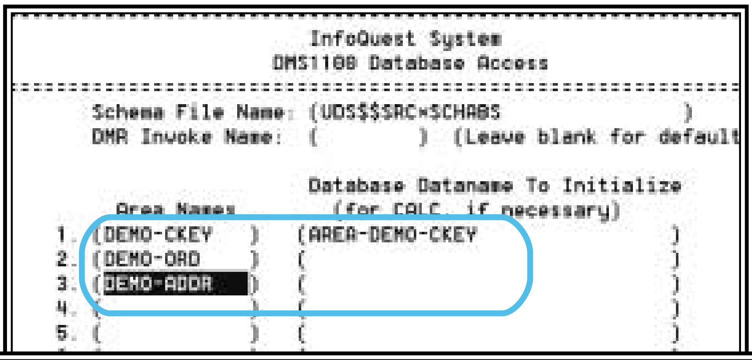
6.4 DMS: Path Navigation via Keyed Access

In this example we will extract data from a record in the path and use the value(s) of that data to access records subordinate to that record by key (randomly). In particular, the CUST-KEY-REC will be accessed by its CALC key, CK-CUST-KEY.

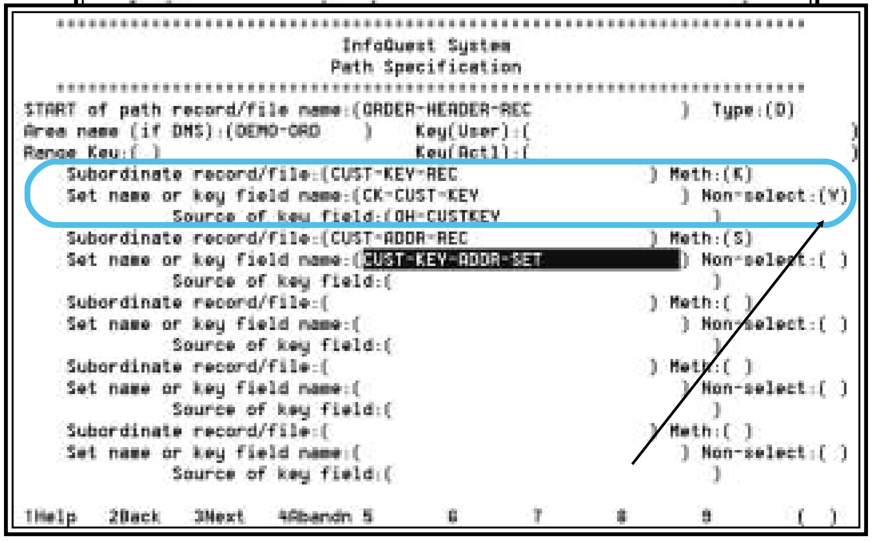


Here the root record is the ORDER-HEADER-REC record. The first subordinate in the path is the owner of the CUST-ORD set. However, the CUST-KEY-REC will not be accessed through the set (FETCH3 OWNER) but by its CALC key (FETCH5). Using the CALC key could represent quite a savings in physical I/Os since the ORDER-HEADER-REC record is an entry point record and has no owner pointers on the CUST-ORD set.

Notice that, on the DMS1100 Database Access screen, we must supply the database dataname (AREA-DEMO-CKEY) that will contain the CALC record's area name (a requirement of CALC entry point access).



```
                       InfoQuest System
                   DMS1100 Database Access
==============================================================
    Schema File Name: (UDS$$SRC*SCHABS                       )
    DMR Invoke Name:  (         )  (Leave blank for default

                      Database Dataname To Initialize
      Area Names         (for CALC, if necessary)
  1. (DEMO-CKEY   )   (AREA-DEMO-CKEY                   )
  2. (DEMO-ORD    )   (                                 )
  3. (DEMO-ADDR   )   (                                 )
  4. (            )   (                                 )
  5. (            )   (                                 )
```



```
   ****************************************************************
                          InfoQuest System
                         Path Specification
   ****************************************************************
START of path record/file name:(ORDER-HEADER-REC           )  Type:(D)
Area name (if DMS):(DEMO-ORD    )    Key(User):(                      )
Range Key:( )                        Key(Actl):(                      )
    Subordinate record/file:(CUST-KEY-REC                   ) Meth:(K)
    Set name or key field name:(CK-CUST-KEY                  ) Non-select:(Y)
             Source of key field:(OH-CUSTKEY                  )
    Subordinate record/file:(CUST-ADDR-REC                  ) Meth:(S)
    Set name or key field name:(CUST-KEY-ADDR-SET            ) Non-select:( )
             Source of key field:(                            )
    Subordinate record/file:(                               ) Meth:( )
    Set name or key field name:(                             ) Non-select:( )
             Source of key field:(                            )
    Subordinate record/file:(                               ) Meth:( )
    Set name or key field name:(                             ) Non-select:( )
             Source of key field:(                            )
    Subordinate record/file:(                               ) Meth:( )
    Set name or key field name:(                             ) Non-select:( )
             Source of key field:(                            )

1Help   2Back   3Next   4Abandn 5       6       7       8       9       (  )
```

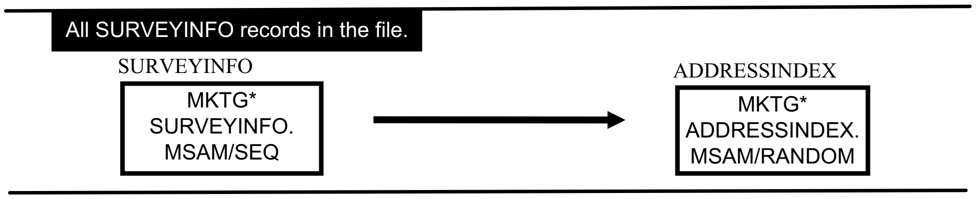
On the Path Specification screen, we must supply the name of the CALC key (CK-CUST-KEY) and the source of the key field (OH-CUSTKEY) whose value comes from a record higher in the path hierarchy.

Notice that the CUST-KEY-REC is marked so that no selection logic will be generated for the record (the Non-select parameter is set to "Y"); however, it is required to access the CUST-ADDR-REC lower in the hierarchy.

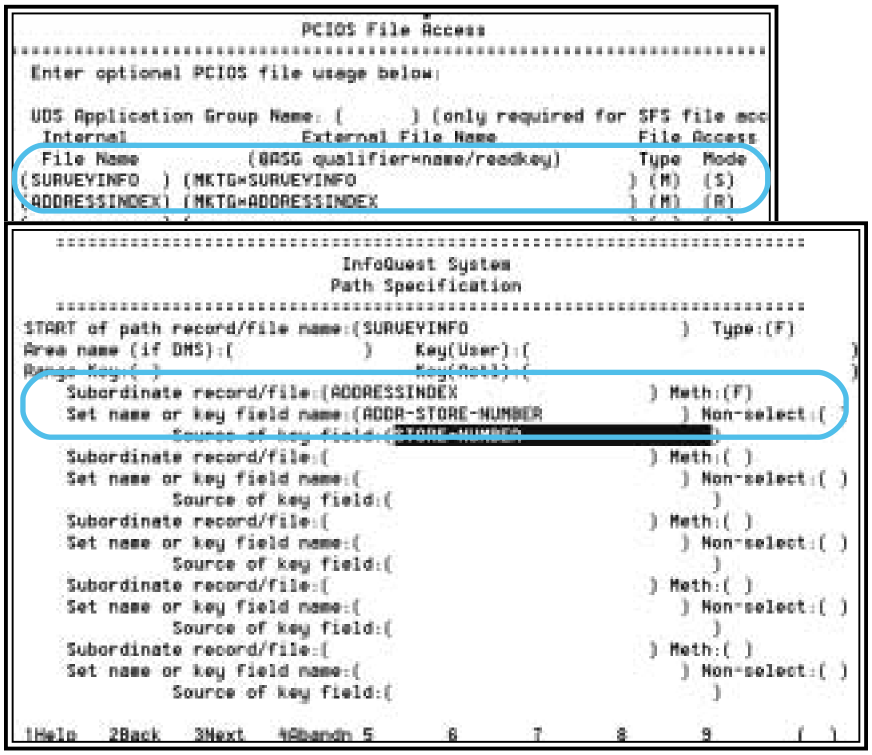
When a record is marked as "Non-select", the record must be included in the view, but its data items should be deleted from the view.

6.5 PCIOS: Sequential to MSAM Access

In the following PCIOS (only) example, the root file is the SURVEYINFO file. As with all PCIOS root records, the access must be sequential or dynamic. The STORE-NUMBER from SURVEYINFO will be used to initialize the primary MSAM key, ADDR-STORE-NUMBER, in the subordinate file, ADDRESSINDEX, in order to access it randomly.


All SURVEYINFO records in the file.
SURVEYINFO
MKTG* SURVEYINFO. MSAM/SEQ
ADDRESSINDEX
MKTG* ADDRESSINDEX. MSAM/RANDOM

With PCIOS files, the external filename must be supplied so that the DBM can assign the file (CSF X '@ASG,AZ ...') and equate it to the internal file name (CSF X '@USE ...').



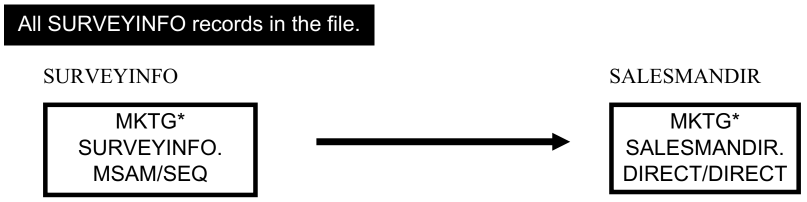
```
                              PCIOS File Access
*****************************************************************************
 Enter optional PCIOS file usage below:

 UDS Application Group Name: (      ) (only required for SFS file acc
  Internal                   External File Name                File Access
  File Name           (@ASG qualifier*name/readkey)            Type  Mode
(SURVEYINFO   ) (MKTG*SURVEYINFO                              ) (M)  (S)
(ADDRESSINDEX) (MKTG*ADDRESSINDEX                             ) (M)  (R)
```

```
   ======================================================================
                             InfoQuest System
                            Path Specification
   ======================================================================
START of path record/file name:(SURVEYINFO                     )  Type:(F)
Area name (if DMS):(            )    Key(User):(                          )
Range Key:( )                        Key(Retl):(                          )
    Subordinate record/file:(ADDRESSINDEX                      ) Meth:(F)
    Set name or key field name:(ADDR-STORE-NUMBER                ) Non-select:( )
              Source of key field:(STORE-NUMBER                 )
    Subordinate record/file:(                                  ) Meth:( )
    Set name or key field name:(                                 ) Non-select:( )
              Source of key field:(                             )
    Subordinate record/file:(                                  ) Meth:( )
    Set name or key field name:(                                 ) Non-select:( )
              Source of key field:(                             )
    Subordinate record/file:(                                  ) Meth:( )
    Set name or key field name:(                                 ) Non-select:( )
              Source of key field:(                             )
    Subordinate record/file:(                                  ) Meth:( )
    Set name or key field name:(                                 ) Non-select:( )
              Source of key field:(                             )

1Help   2Back   3Next   4Abandn 5       6       7       8       9       ( )
```

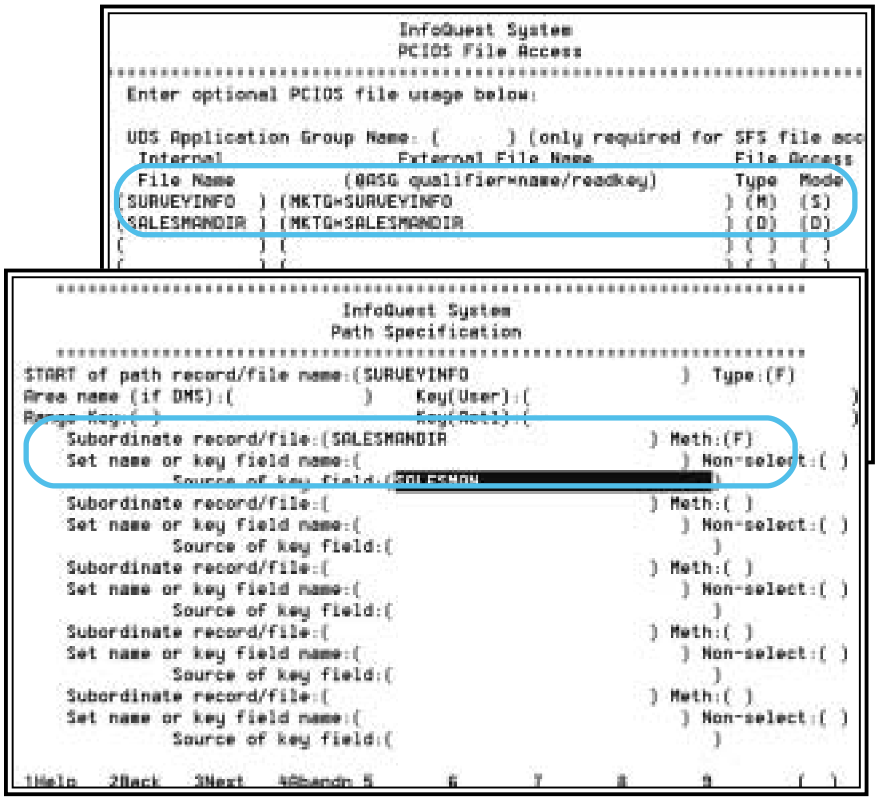
The path generation process will read the internal file label information in order to automatically generate the DEF F directive. The path generation will include the record and block size as well as the key specifications for INDEXED records.

6.6 PCIOS: Sequential to Direct Access

This example illustrates the use of a DIRECT file in the access path. Again, the root file is SURVEYINFO. The key used to randomly access the DIRECT file, SALESMANDIR, is a relative record key not contained in the DIRECT record. Instead,


All SURVEYINFO records in the file.
SURVEYINFO
MKTG* SURVEYINFO. MSAM/SEQ
SALESMANDIR
MKTG* SALESMANDIR. DIRECT/DIRECT

path generation requires the use of the reserved variable, X. For this reason, we must leave the key field name blank for the SALESMANDIR file (see the Parh Specification screen, below). The source field that will be used to initialize X is SALESMAN in the SURVEYINFO root file.



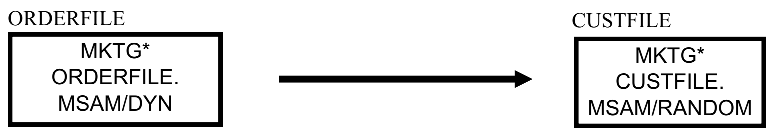
```
                          InfoQuest System
                          PCIOS File Access
**************************************************************************
 Enter optional PCIOS file usage below:

 UDS Application Group Name: (      ) (only required for SFS file acc
  Internal                  External File Name                File Access
  File Name          (@ASG qualifier*name/readkey)            Type  Mode
(SURVEYINFO  ) (MKTG*SURVEYINFO                             ) (M)  (S)
(SALESMANDIR ) (MKTG*SALESMANDIR                            ) (D)  (D)
(            ) (                                            ) ( )  ( )
```

```
    ***********************************************************************
                          InfoQuest System
                         Path Specification
    ***********************************************************************
START of path record/file name:(SURVEYINFO                  )  Type:(F)
Area name (if DMS):(             )    Key(User):(                         )
Range Key:( )                         Key(Rel1):(                         )
    Subordinate record/file:(SALESMANDIR                  ) Meth:(F)
    Set name or key field name:(                              ) Non-select:( )
             Source of key field:(SALESMAN                  )
    Subordinate record/file:(                             ) Meth:( )
    Set name or key field name:(                              ) Non-select:( )
             Source of key field:(                          )
    Subordinate record/file:(                             ) Meth:( )
    Set name or key field name:(                              ) Non-select:( )
             Source of key field:(                          )
    Subordinate record/file:(                             ) Meth:( )
    Set name or key field name:(                              ) Non-select:( )
             Source of key field:(                          )
    Subordinate record/file:(                             ) Meth:( )
    Set name or key field name:(                              ) Non-select:( )
             Source of key field:(                          )

1Help   2Back   3Next   4Abandn 5     6     7     8     9          ( )
```

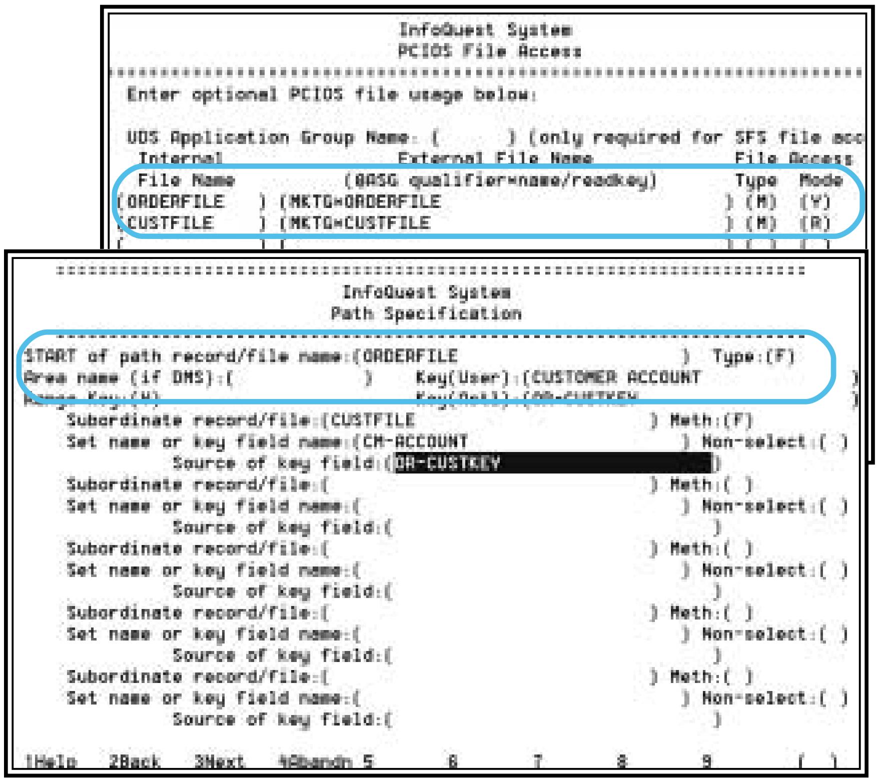
6.7 PCIOS: Range Key to Random Access

The root record for this DBM will also be an MSAM file, ORDERFILE. However, the total number of records accessed will be limited by a range of values (the end user will be prompted for a low and high search value). Also, the key used for the range processing will not be the primary record key but an alternate key (the record descriptions for these two files can be found at the end of Chapter 7, “Demonstration/Validation”).

All ORDERFILE records within a range of values.



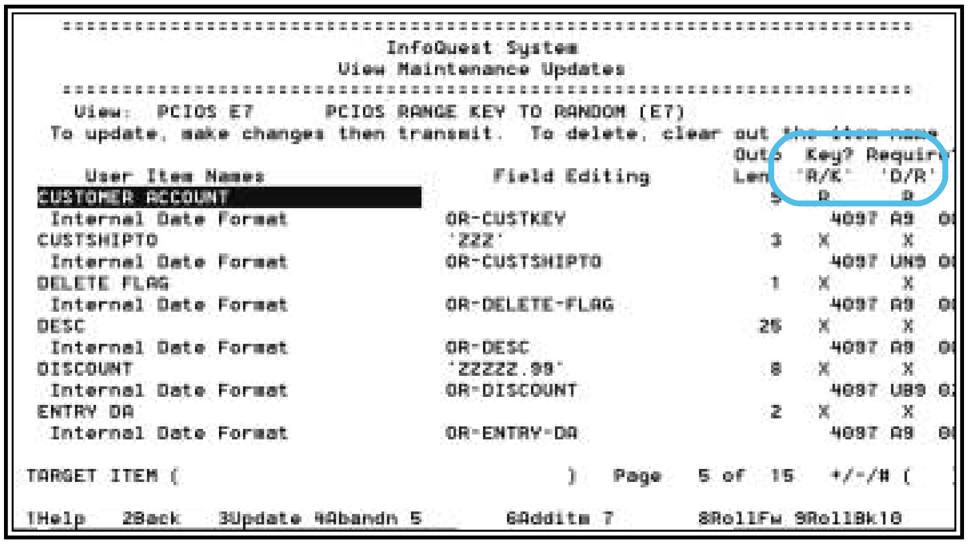
To allow range processing, the access mode of the ORDERFILE must be set to “Y” for dynamic access. Also, the range key field on the third screen must be set to “Y”. The alternate key is selected automatically by the DBM generation process based on the “Actl” key field name entered.



```
                          InfoQuest System
                          PCIOS File Access
**************************************************************************
 Enter optional PCIOS file usage below:

 UDS Application Group Name: (      ) (only required for SFS file acc
  Internal                External File Name                File Access
  File Name         (@ASG qualifier*name/readkey)           Type  Mode
(ORDERFILE    ) (MKTG*ORDERFILE                           ) (M)   (Y)
(CUSTFILE     ) (MKTG*CUSTFILE                            ) (M)   (R)
(             ) (                                         ) ( )   ( )
```

```
   ======================================================================
                          InfoQuest System
                         Path Specification
   ----------------------------------------------------------------------
START of path record/file name:(ORDERFILE                  )  Type:(F)
Area name (if DMS):(              )     Key(User):(CUSTOMER ACCOUNT        )
Range Key:(Y)                           Key(Actl):(OR-CUSTKEY              )
    Subordinate record/file:(CUSTFILE                     ) Meth:(F)
    Set name or key field name:(CM-ACCOUNT                   ) Non-select:( )
              Source of key field:(OR-CUSTKEY                  )
    Subordinate record/file:(                             ) Meth:( )
    Set name or key field name:(                             ) Non-select:( )
              Source of key field:(                            )
    Subordinate record/file:(                             ) Meth:( )
    Set name or key field name:(                             ) Non-select:( )
              Source of key field:(                            )
    Subordinate record/file:(                             ) Meth:( )
    Set name or key field name:(                             ) Non-select:( )
              Source of key field:(                            )
    Subordinate record/file:(                             ) Meth:( )
    Set name or key field name:(                             ) Non-select:( )
              Source of key field:(                            )

 1Help   2Back   3Next   4Abandn 5       6       7       8       9       (  )
```

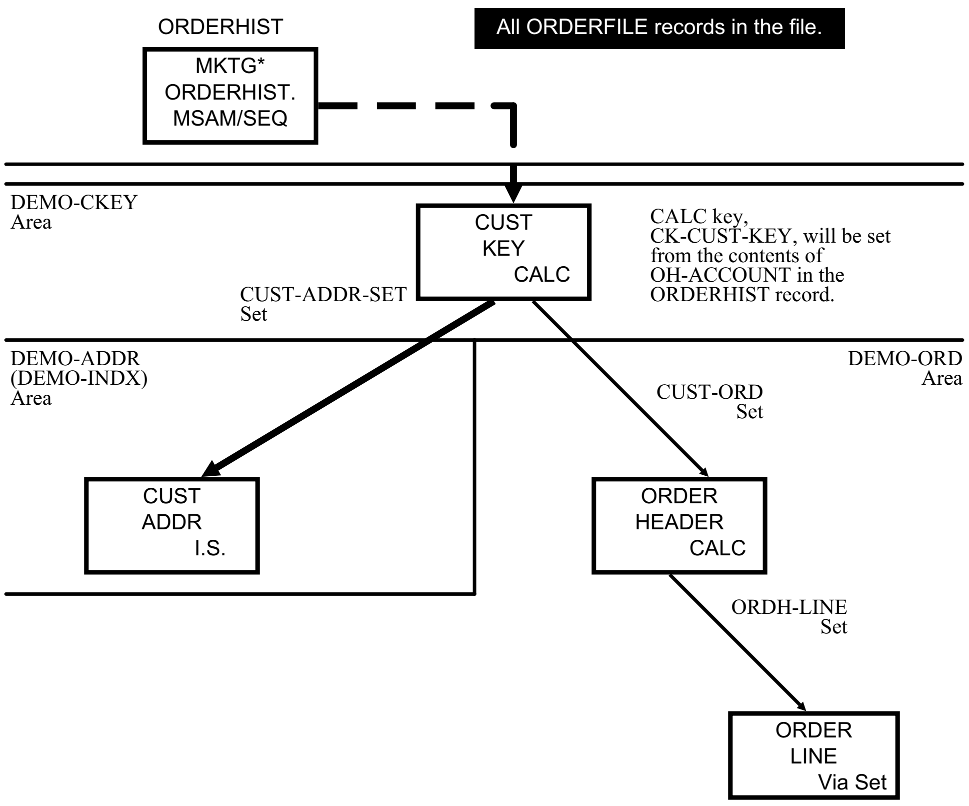
When range key processing is used, as in the above example, the item specified as the key must be flagged as a range key (“R”) and as an item required for selection (also “R”) in the view. The flagging is accomplished by choosing the VIEW MAINTENANCE function in the IQMNT menu. On the “View Maintenance Updates” screen, the item must be marked with an “R” in both fields as shown below:



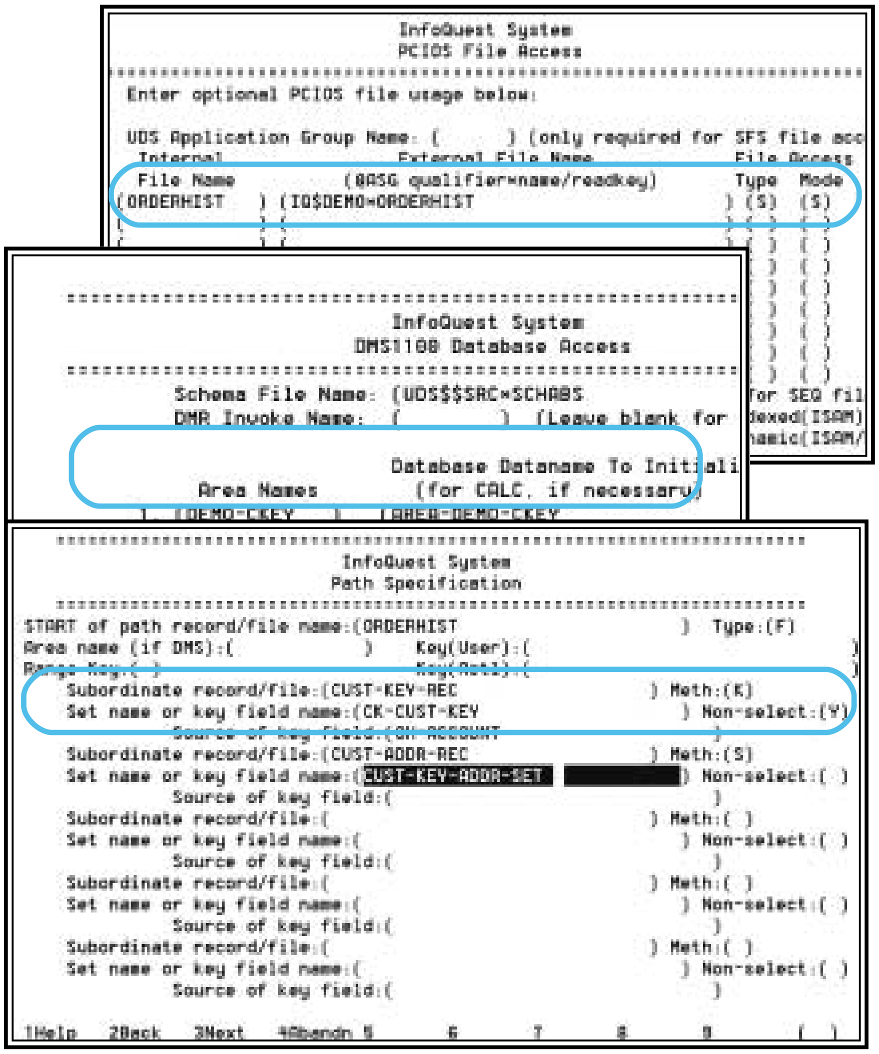
```
    ======================================================================
                              InfoQuest System
                          View Maintenance Updates
    ======================================================================
     View:   PCIOS E7      PCIOS RANGE KEY TO RANDOM (E7)
   To update, make changes then transmit.  To delete, clear out the item name
                                                          Outp  Key? Requir
      User Item Names                        Field Editing     Len  'R/K'  'D/R'
  CUSTOMER ACCOUNT                                          5    R      R
   Internal Date Format          OR-CUSTKEY                         4097 A9   0
  CUSTSHIPTO                     'ZZZ'                      3    X      X
   Internal Date Format          OR-CUSTSHIPTO                      4097 UN9  0
  DELETE FLAG                                               1    X      X
   Internal Date Format          OR-DELETE-FLAG                     4097 A9   0
  DESC                                                     25    X      X
   Internal Date Format          OR-DESC                            4097 A9   0
  DISCOUNT                       'ZZZZZ.99'                 8    X      X
   Internal Date Format          OR-DISCOUNT                        4097 UB9  0
  ENTRY DA                                                  2    X      X
   Internal Date Format          OR-ENTRY-DA                        4097 A9   0

 TARGET ITEM (                                )  Page   5 of  15   +/-/# (

 1Help   2Back   3Update 4Abandn 5        6Additm 7       8RollFw 9RollBk10
```

6.8 DMS/PCIOS: Sequential to CALC Access

This example combines the use of both PCIOS and DMS1100 access in one DBM. The PCIOS file, ORDERHIST, is the root file and a data item (OH-ACCOUNT) from this file will be used to gain entry to the database (the record description for this and other files used in the examples can be found at the end of Chapter 7, "Demonstration/Validation").


ORDERHIST
All ORDERFILE records in the file.
MKTG* ORDERHIST. MSAM/SEQ
DEMO-CKEY Area
CUST KEY CALC
CALC key, CK-CUST-KEY, will be set from the contents of OH-ACCOUNT in the ORDERHIST record.
CUST-ADDR-SET Set
DEMO-ADDR (DEMO-INDX) Area
DEMO-ORD Area
CUST-ORD Set
CUST ADDR I.S.
ORDER HEADER CALC
ORDH-LINE Set
ORDER LINE Via Set

The entry point record is the CUST-KEY-REC (CALC) whose CALC key, CK-CUST-KEY, will be initialized with the value of OH-ACCOUNT. Notice that the CUST-KEY-REC will not be used for data selection because the Non-select parameter is set to "Y". Here again it is necessary to supply the database data name (AREA-DEMO-CKEY) required for initialization with the name of the CALC area (DEMO-CKEY).



```
                        InfoQuest System
                        PCIOS File Access
****************************************************************************
 Enter optional PCIOS file usage below:

 UDS Application Group Name: (      ) (only required for SFS file acc
  Internal                  External File Name                File Access
  File Name           (@ASG qualifier*name/readkey)           Type   Mode
(ORDERHIST    ) (IQ$DEMO*ORDERHIST                          ) (S)    (S)
```

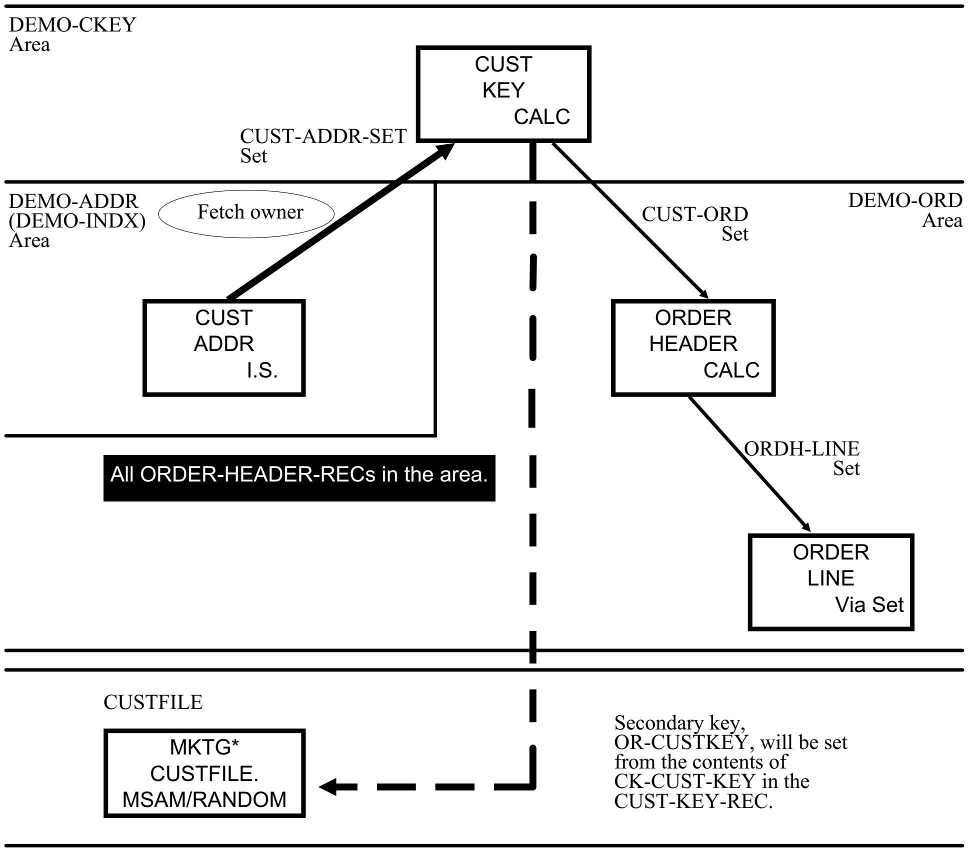
```
=====================================================================
                         InfoQuest System
                      DMS1100 Database Access
=====================================================================
          Schema File Name: (UDS$$SRC*SCHABS
          DMR Invoke Name:  (           )  (Leave blank for

                                 Database Dataname To Initiali
             Area Names            (for CALC, if necessary)
      1. (DEMO-CKEY     )     (AREA-DEMO-CKEY
```

```
*****************************************************************************
                              InfoQuest System
                             Path Specification
*****************************************************************************
START of path record/file name:(ORDERHIST                       )  Type:(F)
Area name (if DMS):(              )     Key(User):(                          )
Range Key:( )                           Key(Actl):(                          )
    Subordinate record/file:(CUST-KEY-REC                       ) Meth:(K)
    Set name or key field name:(CK-CUST-KEY                        ) Non-select:(Y)
             Source of key field:(OH-ACCOUNT                        )
    Subordinate record/file:(CUST-ADDR-REC                      ) Meth:(S)
    Set name or key field name:(CUST-KEY-ADDR-SET                  ) Non-select:( )
             Source of key field:(                                  )
    Subordinate record/file:(                                   ) Meth:( )
    Set name or key field name:(                                   ) Non-select:( )
             Source of key field:(                                  )
    Subordinate record/file:(                                   ) Meth:( )
    Set name or key field name:(                                   ) Non-select:( )
             Source of key field:(                                  )
    Subordinate record/file:(                                   ) Meth:( )
    Set name or key field name:(                                   ) Non-select:( )
             Source of key field:(                                  )

1Help   2Back   3Next   4Abandn 5       6       7       8       9       (  )
```

The final record in the path is the CUST-ADDR-REC that will be accessed via the CUST-KEY-ADDR-SET.

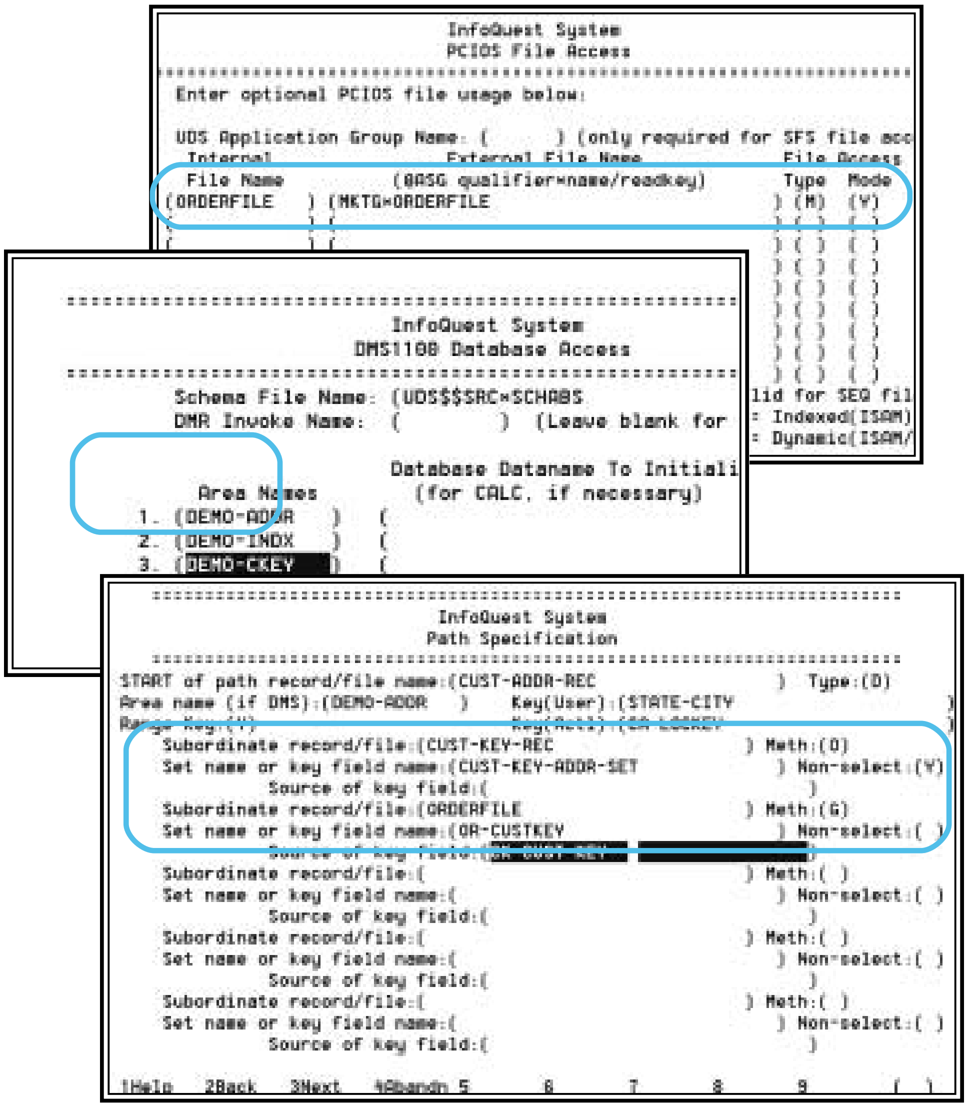
6.9 DMS/PCIOS: Range Key to Secondary Key

This example will do a range search on a DMS 1100 Index Sequential record, pick up a key from its owner and use that key to access an alternate PCIOS order file by a secondary key. The root record is the DMS1100 Index Sequential record, CUST-ADDR-REC. The key required to access the PCIOS file is in its owner record, CUST-KEY-REC, in the CUST-ADDR-SET. The FETCH OWNER access to this record will be efficient as there is always a one-to-one relationship between these two records. Also notice that the PCIOS file, ORDERFILE, is being accessed by a secondary key (Method G), not its primary key.


DEMO-CKEY Area
CUST KEY CALC
CUST-ADDR-SET Set
DEMO-ADDR (DEMO-INDX) Area
Fetch owner
CUST-ORD Set
DEMO-ORD Area
CUST ADDR I.S.
ORDER HEADER CALC
ORDH-LINE Set
All ORDER-HEADER-RECs in the area.
ORDER LINE Via Set
CUSTFILE
MKTG* CUSTFILE. MSAM/RANDOM
Secondary key, OR-CUSTKEY, will be set from the contents of CK-CUST-KEY in the CUST-KEY-REC.

DMS/PCIOS: Range Key to Secondary Key

The final record in the path is the CUST-ADDR-REC that will be accessed via the CUST-KEY-ADDR-SET. Notice the key fields and source fields required for random access. Also notice that the CUST-KEY-REC will not be used for data selection because the Non-select parameter is set to “Y”.



```
                         InfoQuest System
                         PCIOS File Access
**************************************************************************
 Enter optional PCIOS file usage below:

 UDS Application Group Name: (      ) (only required for SFS file acc
  Internal                  External File Name                File Access
  File Name            (@ASG qualifier*name/readkey)          Type  Mode
(ORDERFILE    ) (MKTG*ORDERFILE                             ) (M)  (Y)
(             ) (                                           ) ( )  ( )
(             ) (                                           ) ( )  ( )
                                                            ) ( )  ( )
                                                            ) ( )  ( )
                                                            ) ( )  ( )
                                                            ) ( )  ( )
                                                            ) ( )  ( )
                                                            ) ( )  ( )
                                                            ) ( )  ( )
                                                          lid for SEQ fil
                                                          : Indexed(ISAM)
                                                          : Dynamic(ISAM/
```

```
=================================================================
                    InfoQuest System
                 DMS1100 Database Access
=================================================================
        Schema File Name:  (UDS$$SRC*SCHABS
        DMR Invoke Name:   (          )  (Leave blank for

                           Database Dataname To Initiali
          Area Names         (for CALC, if necessary)
    1. (DEMO-ADDR    )    (
    2. (DEMO-INDX    )    (
    3. (DEMO-CKEY    )    (
```

```
    ======================================================================
                              InfoQuest System
                             Path Specification
    ======================================================================
START of path record/file name:(CUST-ADDR-REC                    )  Type:(D)
Area name (if DMS):(DEMO-ADDR   )    Key(User):(STATE-CITY                     )
Range Key:(Y)                        Key(Alt1):(OR-LOCKEY                      )
   Subordinate record/file:(CUST-KEY-REC                       ) Meth:(D)
   Set name or key field name:(CUST-KEY-ADDR-SET                  ) Non-select:(Y)
            Source of key field:(                                  )
   Subordinate record/file:(ORDERFILE                          ) Meth:(G)
   Set name or key field name:(OR-CUSTKEY                         ) Non-select:( )
            Source of key field:(OR-CUST-KEY                       )
   Subordinate record/file:(                                   ) Meth:( )
   Set name or key field name:(                                   ) Non-select:( )
            Source of key field:(                                  )
   Subordinate record/file:(                                   ) Meth:( )
   Set name or key field name:(                                   ) Non-select:( )
            Source of key field:(                                  )
   Subordinate record/file:(                                   ) Meth:( )
   Set name or key field name:(                                   ) Non-select:( )
            Source of key field:(                                  )

1Help   2Back   3Next   4Abandn 5       6       7       8       9        (  )
```

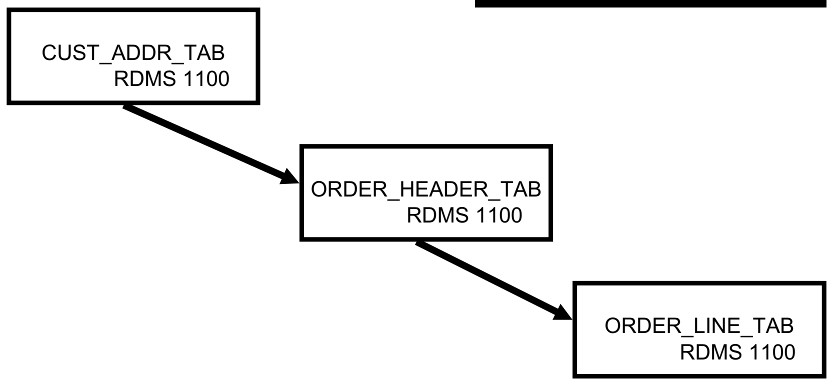
6.10 RDMS: Join Multiple Tables

The following example illustrates the generation of a DBM to join three RDMS tables through the use of defined primary keys and secondary indexes (the listing of the RDMS tables used in this and other examples may be found in Appendix G, “RDMS Tables”).

When more than two tables are to be linked, a hierarchy must be established so that tables are properly related. This hierarchy also prevents unnecessary access to tables when the end user does not require data from all tables covered in the view. How and why the hierarchy is used is best described by example.

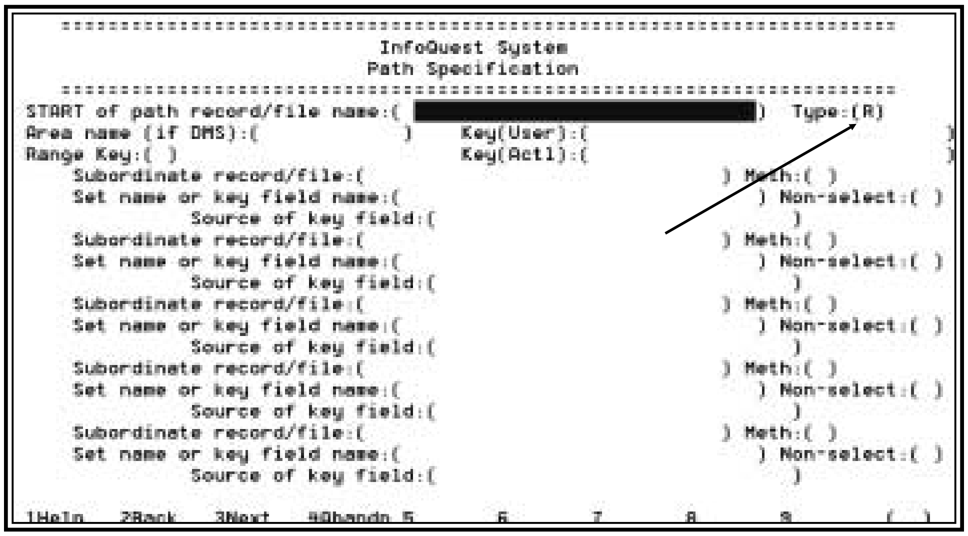
This example uses a database for customer order entry consisting of three tables: CUST_ADDR_TAB (table) containing customer account number, name, address, etc.; ORDER_HEADER_TAB containing the customer account, order number, shipping information and other items pertaining to the entire order; and ORDER_LINE_TAB containing the order number, product code, unit price, quantity ordered, etc. A hierarchy can be established among these three tables: ORDER_LINE_TAB is subordinate to ORDER_HEADER_TAB and ORDER_HEADER_TAB is subordinate to CUST_ADDR_TAB. The three tables also have very clear logical relationships: CUST_ADDR_TAB and ORDER_HEADER_TAB are related (linked) by the customer account number, and ORDER_HEADER_TAB and ORDER_LINE_TAB are related by the order number. Note that no relationship exists between CUST_ADDR_TAB and ORDER_LINE_TAB without ORDER_HEADER_TAB. The hierarchy of access to data in the order entry database can then be assumed to be as follows:

All rows in all tables through a join.



When accessing RDMS data, the RDMS database access may only appear as the starting, or root access. The path generation process does not generate database access SQL, but does generate appropriate RDMS commands in preparation for database access. The commands to actually retrieve data (DECLARE CURSOR and FETCH) are generated at run time from user-entered parameters.

When specifying the starting or root access of the path (see below), an 'R' in type is required. A subsequent screen will be displayed for RDMS environmental and table access information.

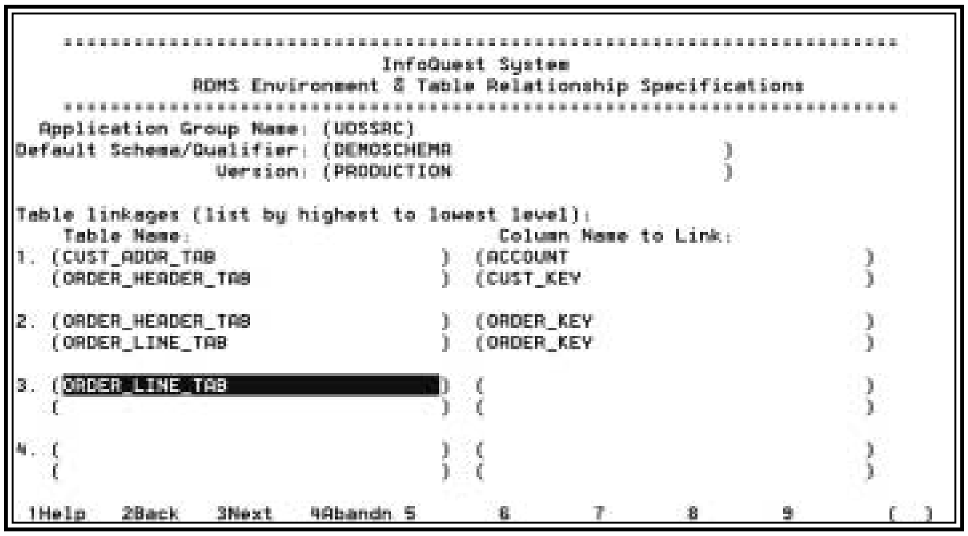


```
     ======================================================================
                               InfoQuest System
                              Path Specification
     ======================================================================
 START of path record/file name:(                             )  Type:(R)
 Area name (if DMS):(            )      Key(User):(                      )
 Range Key:( )                          Key(Actl):(                      )
     Subordinate record/file:(                                ) Meth:( )
     Set name or key field name:(                                ) Non-select:( )
               Source of key field:(                                )
     Subordinate record/file:(                                ) Meth:( )
     Set name or key field name:(                                ) Non-select:( )
               Source of key field:(                                )
     Subordinate record/file:(                                ) Meth:( )
     Set name or key field name:(                                ) Non-select:( )
               Source of key field:(                                )
     Subordinate record/file:(                                ) Meth:( )
     Set name or key field name:(                                ) Non-select:( )
               Source of key field:(                                )
     Subordinate record/file:(                                ) Meth:( )
     Set name or key field name:(                                ) Non-select:( )
               Source of key field:(                                )

 1Help   2Back   3Next   4Abandn 5       6       7       8       9       ( )
```

On the RDMS environmental screen (next page), space is provided to enter table linkages. Since RDMS does not provide a mechanism to define how one table relates to another, the person setting up the InfoQuest path must provide the relationships. Inter-table relationships, or linkages, are described in the form of table names along with the column name that links that table. These linkages are used to build an RDMS SQL WHERE clause. The WHERE clause becomes permanent in the DBM path logic and end user selections are added to this permanent WHERE.

RDMS: Join Multiple Tables

Since the end user can not be responsible for always specifying the inter-table relationships, InfoQuest's path generation function provides a screen on which to specify the logical relationships between the tables. These relationship specifications are permanently embedded in the path logic. When entering the table linkages, the hierarchy is specified from top to bottom. In our order entry database, the screen entries for table linkages would look as follows:



```
    ******************************************************************
                            InfoQuest System
              RDMS Environment & Table Relationship Specifications
    ******************************************************************
  Application Group Name: (UDSSRC)
Default Schema/Qualifier: (DEMOSCHEMA                    )
                 Version: (PRODUCTION                    )

Table linkages (list by highest to lowest level):
    Table Name:                         Column Name to Link:
1. (CUST_ADDR_TAB                 )  (ACCOUNT                      )
   (ORDER_HEADER_TAB              )  (CUST_KEY                     )

2. (ORDER_HEADER_TAB              )  (ORDER_KEY                    )
   (ORDER_LINE_TAB                )  (ORDER_KEY                    )

3. (ORDER_LINE_TAB                )  (                             )
   (                              )  (                             )

4. (                              )  (                             )
   (                              )  (                             )

 1Help   2Back   3Next   4Abandn 5      6      7      8      9      (  )
```

This specification is translated as follows: If the end user needs access to data at the lowest level of the hierarchy, ORDER_LINE_TAB, then CUST_ADDR_TAB, ORDER_HEADER_TAB and ORDER_LINE_TAB tables must be linked; If data from ORDER_HEADER_TAB is required, then only the CUST_ADDR_TAB and ORDER_HEADER_TAB need to be linked; If only data from CUST_ADDR_TAB is needed, no inter-table links are needed. When InfoQuest generates the SQL DECLARE CURSOR statement, the appropriate WHERE syntax specified above will be inserted in front of any WHERE syntax generated by the end user's selection entries.

Notice that the third entry, ORDER_LINE_TAB, has no "Column Name to Link". It is at the bottom of the hierarchy; i.e., it does not link to any table below it. However, it must be named here in order to generate the appropriate WHERE clause.

The table linkage WHERE syntax is used, as is, by InfoQuest, that is, there is no syntax checking. For this reason, it should be checked and tested carefully.

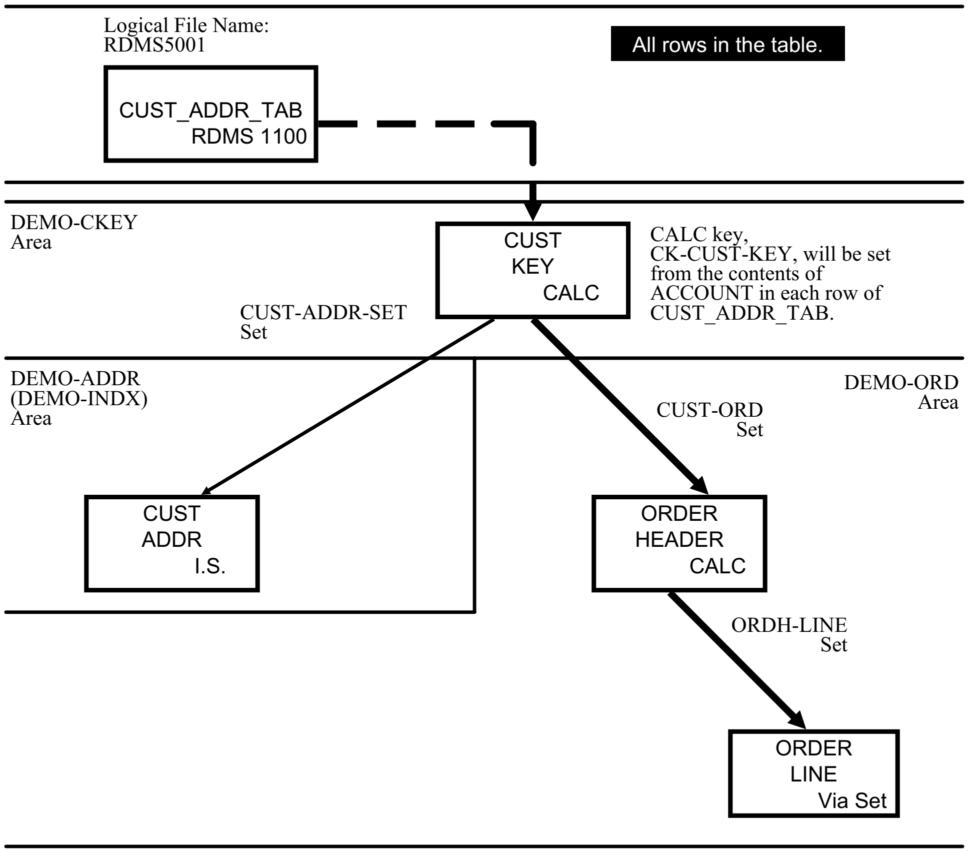
Unlike other database access, InfoQuest generates SQL statements for RDMS to preselect only that data required to produce the end user's desired result. Preselection is possible due to the RDMS implementation of SQL. With other database models, the InfoQuest path logic must retrieve each record, then test whether or not it meets end user selection criteria.

The next screen, the optional “RDMS Permanent WHERE Selections” screen, will be bypassed by pressing F3 as all of the rows in the CUST_ADDR_TAB are to be read.

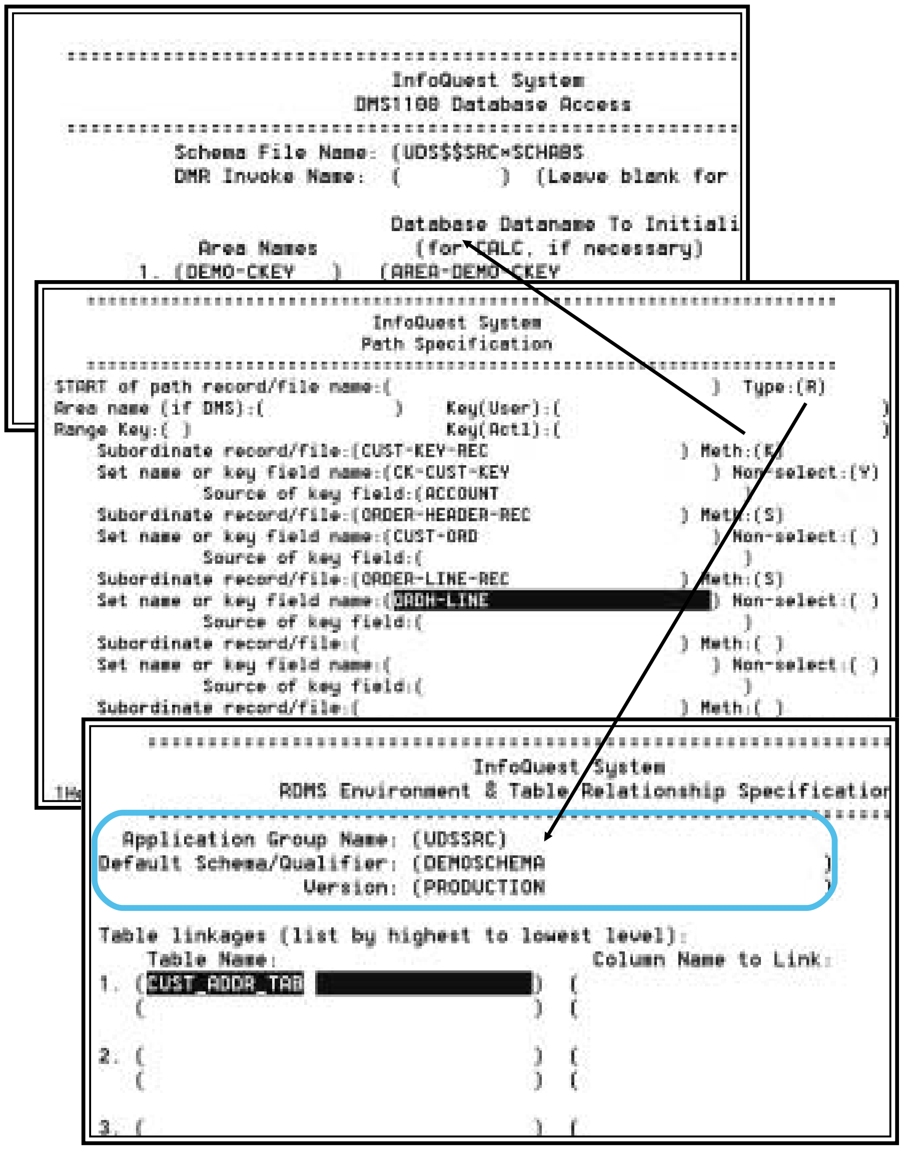
For more information on table linkages and the use of RDMS permanent WHERE selections, see Section 5.6.4, “RDMS 1100 Access”.

6.11 RDMS/DMS: RDMS Table to CALC Access

This example combines the use of both RDMS 1100 and DMS1100 access in one DBM. The RDMS table, CUST_ADDR_TAB, is the root of the path and a column (ACCOUNT) from this table will be used to gain entry to the DMS1100 database.



Notice that no record or file name is specified for the start of the path on the Path Specification screen. However, setting the Type field to an “R” will result in the RDMS Path Specification screen being displayed, where the pertinent RDMS 1100 information will be entered: Application Group, Qualifier, Version, etc.



```
==========================================================
                  InfoQuest System
               DMS1100 Database Access
==========================================================
        Schema File Name: (UDS$$SRC*SCHABS
        DMR Invoke Name:  (         )  (Leave blank for

                          Database Dataname To Initiali
          Area Names        (for CALC, if necessary)
     1. (DEMO-CKEY    )   (AREA-DEMO-CKEY
```

```
==================================================================
                      InfoQuest System
                     Path Specification
==================================================================
START of path record/file name:(                              )  Type:(R)
Area name (if DMS):(            )    Key(User):(                       )
Range Key:( )                        Key(Act1):(                       )
    Subordinate record/file:(CUST-KEY-REC                     ) Meth:(K)
    Set name or key field name:(CK-CUST-KEY                     ) Non-select:(Y)
             Source of key field:(ACCOUNT                        )
    Subordinate record/file:(ORDER-HEADER-REC                 ) Meth:(S)
    Set name or key field name:(CUST-ORD                        ) Non-select:( )
             Source of key field:(                               )
    Subordinate record/file:(ORDER-LINE-REC                   ) Meth:(S)
    Set name or key field name:(ORDH-LINE                       ) Non-select:( )
             Source of key field:(                               )
    Subordinate record/file:(                                 ) Meth:( )
    Set name or key field name:(                                ) Non-select:( )
             Source of key field:(                               )
    Subordinate record/file:(                                 ) Meth:( )
```

```
==============================================================
                        InfoQuest System
          RDMS Environment & Table Relationship Specification
==============================================================
  Application Group Name: (UDSSRC)
Default Schema/Qualifier: (DEMOSCHEMA                         )
                 Version: (PRODUCTION                         )

Table linkages (list by highest to lowest level):
    Table Name:                              Column Name to Link:
1. (CUST_ADDR_TAB                         )  (
   (                                      )  (

2. (                                      )  (
   (                                      )  (

3. (                                      )  (
```

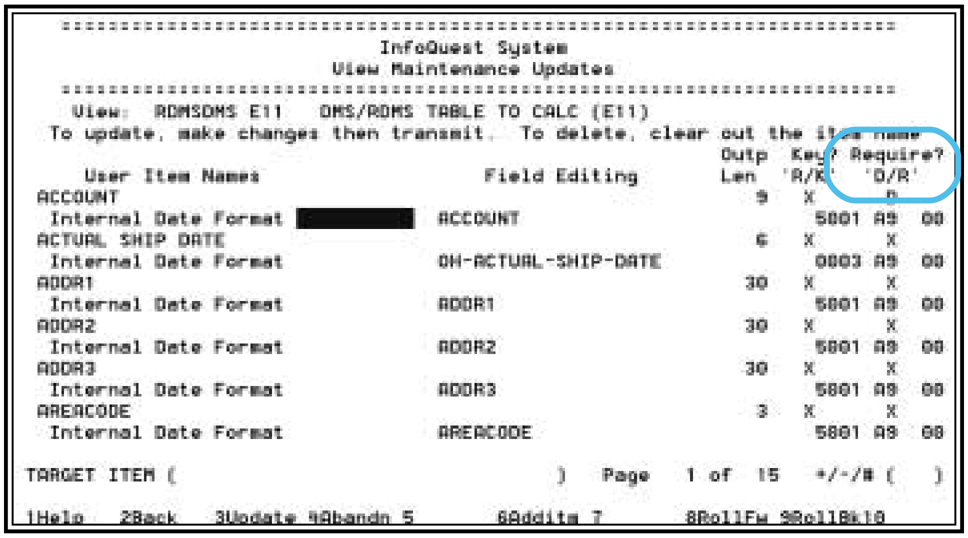
In this example, only one RDMS table is to be accessed; therefore, the “Table linkage” section of the screen has only the single table named with no column linkage as explained in Section 5.6.4, “RDMS 1100 Access”.

RDMS/DMS: RDMS Table to CALC Access

The DMS1100 database record, CUST-KEY-REC, is the first subordinate record in the path and will be accessed by key (Meth: K). Its CALC key, CK-CUST-KEY, will be initialized with the value of ACCOUNT for each row selected in the RDMS table. Here again it is necessary to supply the database data name (AREA-DEMO-CKEY) required for initialization with the name of the CALC area (DEMO-CKEY).

The second subordinate entry specifies that the ORDER-HEADER-REC is to be accessed using the set linkages (Meth: S) in the CUST-ORD set. Likewise, the ORDER-LINE-REC record will be accessed through the ORDH-LINE set.

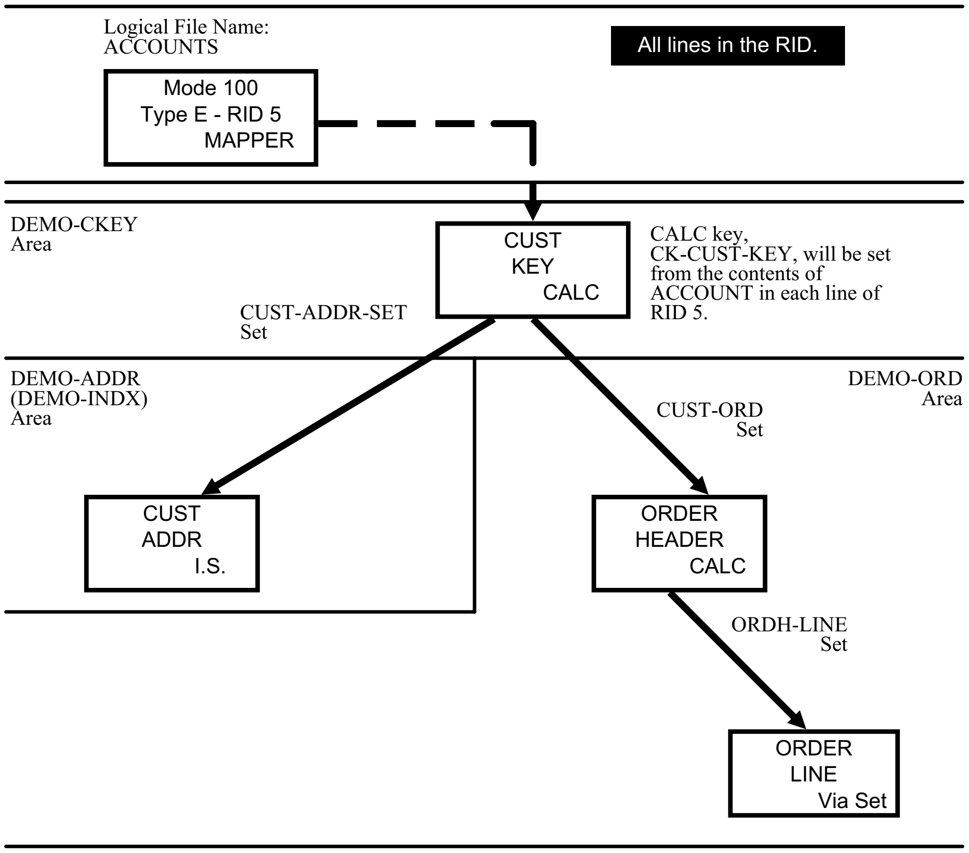
When RDMS column values are to be used to initialize a key for random entry in another file system (DMS, PCIOS, etc.), as in the above example, the item specified as the "Source key field" must be flagged as a defined item ("D") or an item required for selection ("R") in the view. The flagging is accomplished by choosing the VIEW MAINTENANCE function in the IQMNT menu. On the "View Maintenance Updates" screen, the item must be marked with a "D" or an "R" in the "Required" field as shown below:



```
    ======================================================================
                              InfoQuest System
                          View Maintenance Updates
    ======================================================================
     View:   RDMSDMS E11   DMS/RDMS TABLE TO CALC (E11)
   To update, make changes then transmit.  To delete, clear out the item name
                                                     Outp  Key? Require?
      User Item Names                 Field Editing   Len  'R/K'  'D/R'
  ACCOUNT                                              9    X      D
   Internal Date Format             ACCOUNT                  5001 A9   00
  ACTUAL SHIP DATE                                     6    X      X
   Internal Date Format             OH-ACTUAL-SHIP-DATE      0003 A9   00
  ADDR1                                               30    X      X
   Internal Date Format             ADDR1                    5001 A9   00
  ADDR2                                               30    X      X
   Internal Date Format             ADDR2                    5001 A9   00
  ADDR3                                               30    X      X
   Internal Date Format             ADDR3                    5001 A9   00
  AREACODE                                             3    X      X
   Internal Date Format             AREACODE                 5001 A9   00

 TARGET ITEM (                               )  Page   1 of  15   +/-/# (   )

 1Help   2Back   3Update 4Abandn 5      6Additm 7      8RollFw 9RollBk10
```

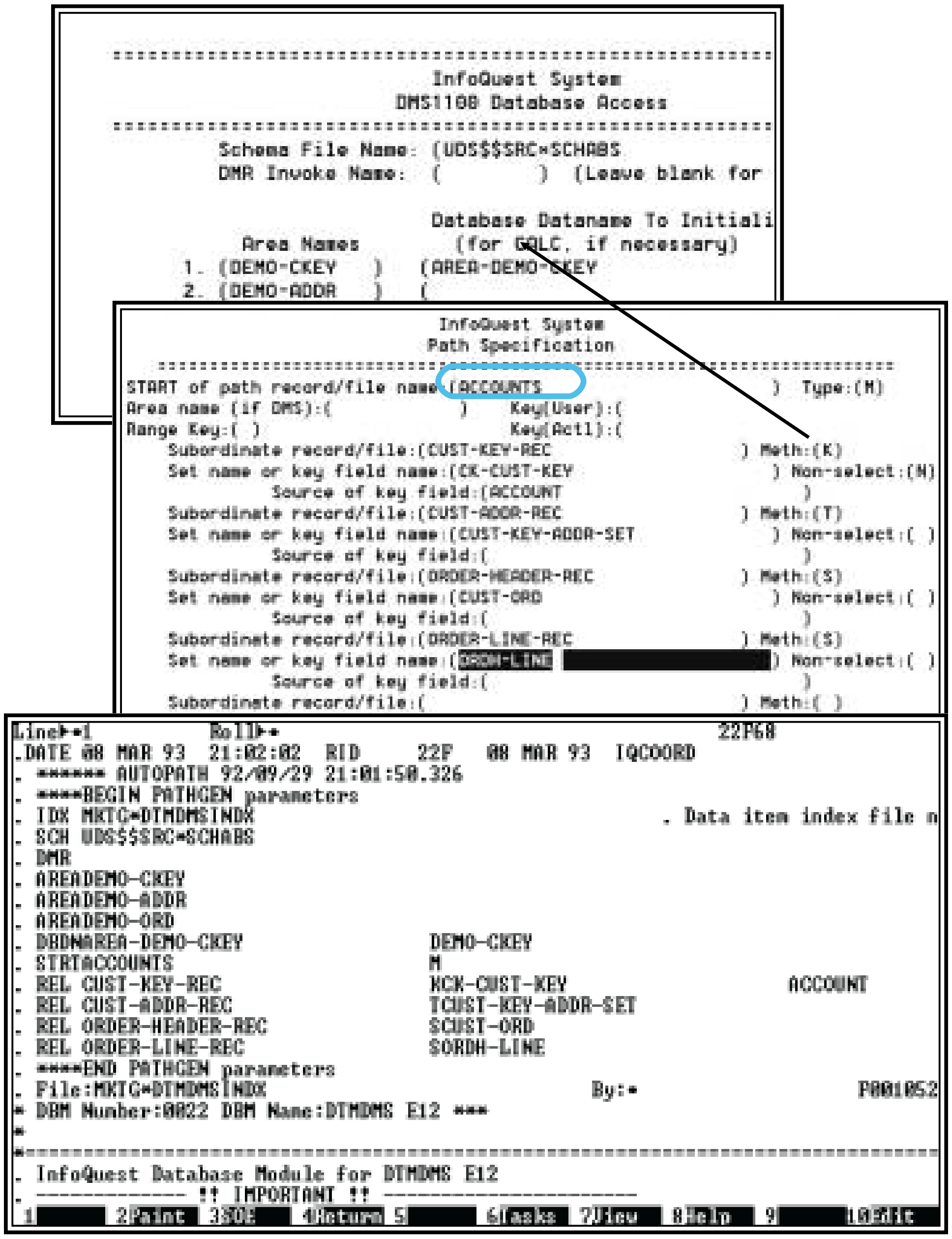
6.12 DTM/DMS: MAPPER RID to CALC Access

This example processes MAPPER data and DMS1100 data in one DBM. A MAPPER RID is the root of the path and a column (ACCOUNT) from this RID will be used to gain entry to the DMS1100 database.



DTM/DMS: MAPPER RID to CALC Access

When accessing a MAPPER RID, the logical file name from the generated view must be used on the Path Specification screen for a “record/file” reference. In the example below, “ACCOUNTS” was assigned as the logical file name when the view was generated by IQMNT’s “Create MAPPER RID QINDEX” function.



```
=======================================================
                         InfoQuest System
                      DMS1100 Database Access
=======================================================
        Schema File Name: (UDS$$SRC*SCHABS
        DMR Invoke Name:  (         )  (Leave blank for

                         Database Dataname To Initiali
          Area Names        (for CALC, if necessary)
      1. (DEMO-CKEY    )   (AREA-DEMO-CKEY
      2. (DEMO-ADDR    )   (
```

```
                        InfoQuest System
                       Path Specification
 =====================================================================
START of path record/file name (ACCOUNTS                     )  Type:(M)
Area name (if DMS):(              )    Key(User):(
Range Key:( )                          Key(Actl):(
   Subordinate record/file:(CUST-KEY-REC                  ) Meth:(K)
   Set name or key field name:(CK-CUST-KEY                  ) Non-select:(N)
           Source of key field:(ACCOUNT                      )
   Subordinate record/file:(CUST-ADDR-REC                 ) Meth:(T)
   Set name or key field name:(CUST-KEY-ADDR-SET            ) Non-select:( )
           Source of key field:(                             )
   Subordinate record/file:(ORDER-HEADER-REC              ) Meth:(S)
   Set name or key field name:(CUST-ORD                     ) Non-select:( )
           Source of key field:(                             )
   Subordinate record/file:(ORDER-LINE-REC                ) Meth:(S)
   Set name or key field name:(ORDH-LINE                    ) Non-select:( )
           Source of key field:(                             )
   Subordinate record/file:(                              ) Meth:( )
```

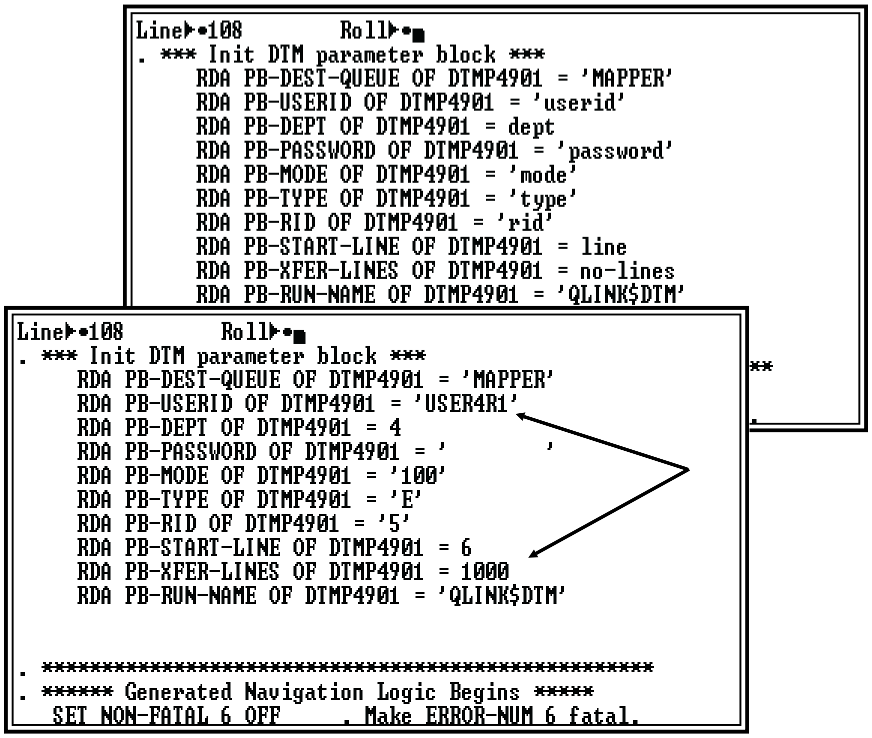
```
Line▸•1          Roll▸•                                   22P68
.DATE 08 MAR 93  21:02:02  RID     22F   08 MAR 93  IQCOORD
. ****** AUTOPATH 92/09/29 21:01:50.326
. ****BEGIN PATHGEN parameters
. IDX MKTG*DTMDMSINDX                                   . Data item index file n
. SCH UDS$$SRC*SCHABS
. DMR
. AREADEMO-CKEY
. AREADEMO-ADDR
. AREADEMO-ORD
. DBDNAREA-DEMO-CKEY              DEMO-CKEY
. STRTACCOUNTS                    M
. REL CUST-KEY-REC                KCK-CUST-KEY                  ACCOUNT
. REL CUST-ADDR-REC               TCUST-KEY-ADDR-SET
. REL ORDER-HEADER-REC            SCUST-ORD
. REL ORDER-LINE-REC              SORDH-LINE
. ****END PATHGEN parameters
. File:MKTG*DTMDMSINDX                         By:*                P001052
* DBM Number:0022 DBM Name:DTMDMS E12 ***
*
*=============================================================================
. InfoQuest Database Module for DTMDMS E12
. ------------- !! IMPORTANT !! -----------------------
 1        2Paint  3SOE    4Return 5        6Tasks  7View   8Help   9        10Edit
```

The DMS1100 database record, CUST-KEY-REC, is the first subordinate record in the path and will be accessed by key (Meth: K). Its CALC key, CK-CUST-KEY, will be initialized with the value of ACCOUNT for each line read from the RID. As in previous examples, it is necessary to supply the database data name (AREA-DEMO-CKEY) required for initialization with the name of the CALC area (DEMO-CKEY).

The second subordinate entry shows that only the first CUST-ADDR-REC (Meth: T) is to be accessed in the CUST-KEY-ADDR-SET and that the fields will be null-filled if the record is not there due to the “F” on the Non-select.

The third subordinate entry specifies that all ORDER-HEADER-REC s are to be accessed using the set linkages (Meth: S) in the CUST-ORD set. Likewise, all ORDER-LINE-RECs will be accessed through the ORDH-LINE set.

The generated DBM must be altered to provide the necessary MAPPER related information (shown below) to effect the data transfer.



```
Line▸•108        Roll▸•
. *** Init DTM parameter block ***
     RDA PB-DEST-QUEUE OF DTMP4901 = 'MAPPER'
     RDA PB-USERID OF DTMP4901 = 'userid'
     RDA PB-DEPT OF DTMP4901 = dept
     RDA PB-PASSWORD OF DTMP4901 = 'password'
     RDA PB-MODE OF DTMP4901 = 'mode'
     RDA PB-TYPE OF DTMP4901 = 'type'
     RDA PB-RID OF DTMP4901 = 'rid'
     RDA PB-START-LINE OF DTMP4901 = line
     RDA PB-XFER-LINES OF DTMP4901 = no-lines
     RDA PB-RUN-NAME OF DTMP4901 = 'QLINK$DTM'
```

```
Line▸•108        Roll▸•
. *** Init DTM parameter block ***
     RDA PB-DEST-QUEUE OF DTMP4901 = 'MAPPER'
     RDA PB-USERID OF DTMP4901 = 'USER4R1'
     RDA PB-DEPT OF DTMP4901 = 4
     RDA PB-PASSWORD OF DTMP4901 = '        '
     RDA PB-MODE OF DTMP4901 = '100'
     RDA PB-TYPE OF DTMP4901 = 'E'
     RDA PB-RID OF DTMP4901 = '5'
     RDA PB-START-LINE OF DTMP4901 = 6
     RDA PB-XFER-LINES OF DTMP4901 = 1000
     RDA PB-RUN-NAME OF DTMP4901 = 'QLINK$DTM'


. ***************************************************
. ****** Generated Navigation Logic Begins *****
   SET NON-FATAL 6 OFF      . Make ERROR-NUM 6 fatal.
```

The following requirements should be considered when altering the values of the above parameters:

PB-DEST-QUEUE must contain the MAPPER DTM queue name as generated in MAPPER (the default is “MAPPER”).

DTM/DMS: MAPPER RID to CALC Access

PB-USERID, PB-DEPT, and PB-PASSWORD must be for a valid MAPPER sign-on.

The PB-MODE, PB-TYPE and PB-RID must be the same format as the mode, type and RID used to generate the data item index file (see Section 3.5.2, “Create MAPPER RID QINDEX”).

PB-START-LINE should be set to exclude any MAPPER headers generated for the form type.

PB-XFER-LINES is dependent upon the total number of lines to be accessed in the RID plus the number of characters on each line. The DTM/MTQ interface in MAPPER limits the size of messages which may be passed to 113,000 words (approximately 5100 80-character or 3000 132-character ASCII lines). If the number of lines to be transferred exceeds this limit, the DBM procedural code must be modified to allow multiple OPENs of the logical file (see Example 2 in Section 9 of the Q-LINK Application Development User Guide).

PB-RUN-NAME contains the name of the MAPPER run (QLINK$DTM) which is required to effect the data transfer.

Notice that the values assigned to PB-MODE and PB-RID must be bound by quotation marks.

6.13 OCCURS Groups

While the current release of InfoQuest does not directly support the access of individual data elements within an OCCURS group, there are two very acceptable methods in which such data elements may be made available to the end user. One method simply involves redefinition of occurring items, while the other takes advantage of a feature of InfoQuest's path generation process.

The one factor that distinguishes the two methods of accessing occurring data elements is how the data is presented to the end user. In the first method, occurring elements are presented horizontally, while in the second, data elements are presented vertically. Which method is chosen may be dictated by how the end user uses or perceives the data elements involved. Let's look at some common uses of occurring data elements and how they are normally perceived by the user.

One application that will almost surely use occurring data elements is budgeting. In most financial applications, budgets for each account are spread out over the twelve months of a year. In this case, all twelve months are usually kept as elements within a table (OCCURS 12 TIMES). Another application that may involve occurring data elements is order entry. In this case, a typical layout for an order record may include some fixed header information (i.e., ship to, bill to, total due, entry date, etc.), with some number of line item entries with detailed order information (i.e., product, quantity ordered, unit of issue, unit price, etc.). In an order record, the number of line entries may vary (OCCURS 1 TO 30 TIMES DEPENDING ON LINE-COUNT). Applying these two common uses of occurring data elements, we can see how user perception of data elements can vary between applications. In the first use, budgets, the monthly budget items are perceived horizontally within each account record, much like it would be shown on a manual or electronic spread sheet. In the case of the order record, line item entries are perceived as being vertical as they normally appear on a printed order or invoice. Now let's look at how these two methods can be applied to InfoQuest.

6.13.1 Horizontal Presentation

We will look at horizontal data presentation first. Consider the following budget record definition:

```
01    GEN-LED-BUDGET.
            05 GL-ACCOUNT.
                  10 GL-MAJ-ACCOUNT        PIC 9(5).
                  10 GL-SUB-ACCOUNT        PIC 9(4).
            05 GL-BUDGET-YEAR              PIC 9(4).
            05 GL-CHANGED-DATE             PIC 9(6).
            05 GL-MONTHLY-AMOUNTS OCCURS 12.
                  10 GL-AMOUNT             PIC S9(10)V99 COMP.
```

If this record definition were processed in InfoQuest, unchanged, the resulting view would contain the following data elements:

GL-ACCOUNT
GL-MAJ-ACCOUNT
GL-SUB-ACCOUNT
GL-BUDGET-YEAR
GL-CHANGED-DATE
GL-MONTHLY-AMOUNTS
GL-AMOUNT

There are two problems with this view, if left unmodified. First, the group level item, GL-MONTHLY-AMOUNTS, would not be of much use as it consists of 12 subordinate computation fields. This item should therefore be deleted from the view. Secondly, the GL-AMOUNT would only refer to the first occurrence of GL-AMOUNT since InfoQuest does not automatically support subscripted reference. Since the user would normally perceive the monthly amounts as appearing horizontally, the best solution here is to redefine the twelve occurrences as individual monthly budget amounts. The record definition might be modified as follows:

```
01    GEN-LED-BUDGET.
            05 GL-ACCOUNT.
                  10 GL-MAJ-ACCOUNT        PIC 9(5).
                  10 GL-SUB-ACCOUNT        PIC 9(4).
            05 GL-BUDGET-YEAR              PIC 9(4).
            05 GL-CHANGED-DATE             PIC 9(6).
            05 GL-MONTHLY-AMOUNTS OCCURS 12.
                  10 GL-AMOUNT             PIC S9(10)V99 COMP.
            05 GL-MTHLY-AMTS REDEFINES GL-MONTHLY-AMOUNTS.
                  10 GL-AMT-JAN            PIC S9(10)V99 COMP.
                  10 GL-AMT-FEB            PIC S9(10)V99 COMP.
                  10 GL-AMT-MAR            PIC S9(10)V99 COMP.
                  10 GL-AMT-APR            PIC S9(10)V99 COMP.
                  10 GL-AMT-MAY            PIC S9(10)V99 COMP.
                  10 GL-AMT-JUN            PIC S9(10)V99 COMP.
                  10 GL-AMT-JUL            PIC S9(10)V99 COMP.
                  10 GL-AMT-AUG            PIC S9(10)V99 COMP.
                  10 GL-AMT-SEP            PIC S9(10)V99 COMP.
                  10 GL-AMT-OCT            PIC S9(10)V99 COMP.
                  10 GL-AMT-NOV            PIC S9(10)V99 COMP.
                  10 GL-AMT-DEC            PIC S9(10)V99 COMP.
```

The above modification produces a view containing the following elements (the elements marked by an asterisk are deleted from the view):

GL-ACCOUNT
GL-MAJ-ACCOUNT
GL-SUB-ACCOUNT
GL-BUDGET-YEAR
GL-CHANGED-DATE
GL-MONTHLY-AMOUNTS *
GL-AMOUNT *
GL-MTHLY-AMTS *
GL-AMT-JAN
GL-AMT-FEB
GL-AMT-MAR
GL-AMT-APR
GL-AMT-MAY
GL-AMT-JUN
GL-AMT-JUL
GL-AMT-AUG
GL-AMT-SEP
GL-AMT-OCT
GL-AMT-NOV
GL-AMT-DEC

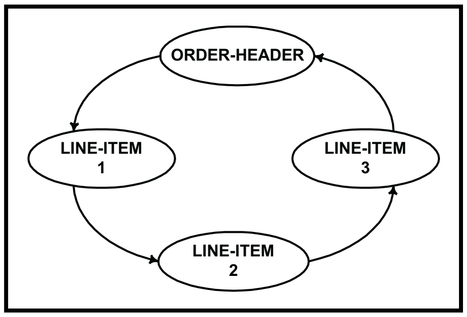
This redefinition provides the end user with easy access to any monthly amount. Monthly amounts can easily be totaled vertically. The user may also use derived item calculations to total monthly amounts horizontally.

6.13.2 Vertical Presentation

The above horizontal representation technique required only some simple changes to the way the data was defined. No special database access logic in the DBM was required. In the second case, where data is to be represented vertically, things get a bit more complicated. Consider the following order record definition:

```
01 ORDER-REC.
            05 OR-ORDER-NO          PIC 9(8).
            05 OR-CUST-ACCOUNT      PIC 9(10).
            05 OR-ENTRY-DATE        PIC 9(6).
            05 OR-SHIP-DATE         PIC 9(6).
            05 OR-CUST-ADDR         PIC X(50).
            05 OR-SHIP-TO-ADDR      PIC X(50).
            05 OR-NUMBER-OF-LINES   PIC 9(10) COMP.
            05 OR-LINES OCCURS 1 TO 100
                  DEPENDING ON OR-NUMBER-OF-LINES.
                  10 OR-PRODUCT     PIC X(6).
                  10 OR-UNIT        PIC X(2).
                  10 OR-UN-PRICE    PIC 9(6)V999.
                  10 OR-TOT-AMT     PIC 9(6)V99.
```

An order record would obviously have many more data elements than were shown here. However, we will keep it simple for the sake of clarity. Because of the occurring items, the problems mentioned above are present here as well. One additional problem is presented by the fact that the OCCURS has the DEPENDING ON option applied. The use of this option means that all entries will not be present for a given order, and we must therefore prevent access to entries beyond the actual number of occurrences. To solve this problem, two actions are required: one involves changing the data definition, while the other involves a feature of InfoQuest's path generation processing.



Logical Relationship of Occurring Elements

First, let's look at how this record may be viewed logically. Each line item in an order can be considered a separate subordinated record owned by the fixed portion of the order record. If this were defined in a DMS1100 database, the line items might be records linked via set to an order header containing the fixed information. The DMS structure would be ideally suited to.

InfoQuest. Therefore, the solution is to make the single record logically resemble the DMS hierarchical structure.

In order to apply this logical hierarchical structure to this record definition, a dummy record definition will be added. This dummy record definition will be used to access elements within the occurring group.

Consider the following definitions as they would be input to the QINDEX processor prior to view generation.

```
FILE ORDERFILE
      01 ORDER-REC.
            05 OR-ORDER-NO           PIC 9(8).
            05 OR-CUST-ACCOUNT       PIC 9(10).
            05 OR-ENTRY-DATE         PIC 9(6).
            05 OR-SHIP-DATE          PIC 9(6).
            05 OR-CUST-ADDR          PIC X(50).
            05 OR-SHIP-TO-ADDR       PIC X(50).
            05 OR-NUMBER-OF-LINES    PIC 9(10) COMP.
            05 OR-LINES OCCURS 1 TO 100
                  DEPENDING ON OR-NUMBER-OF-LINES.
                  10 OR-PRODUCT      PIC X(6).
                  10 OR-UNIT         PIC X(2).
                  10 OR-UN-PRICE     PIC 9(6)V999.
                  10 OR-TOT-AMT      PIC 9(6)V99.
FILE $DUMMY-1
      01 DM-LINES.
            05 DM-PRODUCT            PIC X(6).
            05 DM-UNIT               PIC X(2).
            05 DM-UN-PRICE           PIC 9(6)V999.
            05 DM-TOT-AMT            PIC 9(6)V99.
```

Notice that we have included a file definition for $DUMMY-1. This file contains definitions that match those contained in the occurring group, OR-LINES, of the ORDERFILE definition. The name $DUMMY-n is important in the path generation process. It is recognized by InfoQuest, and no file assignments or read commands will be generated for a $DUMMY-n file. The resulting view will contain the items listed below. Again items to be deleted from the view are marked by an asterisk.

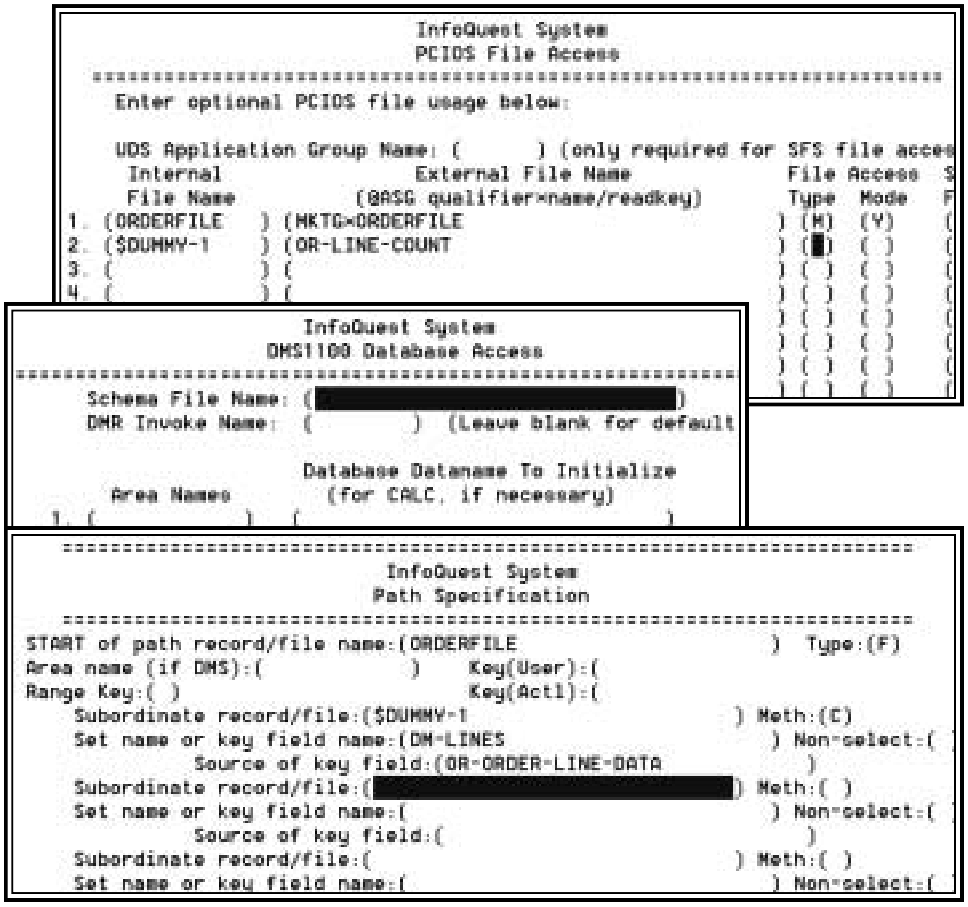
OR-ORDER-NO
OR-CUST-ACCOUNT
OR-ENTRY-DATE
OR-SHIP-DATE
OR-CUST-ADDR
OR-SHIP-TO-ADDR
*OR-NUMBER-OF-LINES **
*OR-LINES **
*OR-PRODUCT **
*OR-UNIT **
*OR-UN-PRICE **
*OR-TOT-AMT **
DM-PRODUCT
DM-UNIT
DM-UN-PRICE
DM-TOT-AMT

The deletions indicated above, remove the occurring items, the occurs counts and group level element of the original record definition from the user's visibility. These elements can still be references within the DBM because they remain defined in the data item index file generated by QINDEX.

Once the data definition and user view have been created, the DBM may be generated. Again, the logic of the DBM used in accessing this record will be very similar to that used in accessing a DMS1100 owner and all its members. The standard InfoQuest path generation process will be used, but with some different parameters. Base DBM generation is performed as usual.

OCCURS Groups

Shown below are the three path generation screens that would be completed to generate the DBM logic to access the ORDERFILE.



```
                         InfoQuest System
                         PCIOS File Access
 =====================================================================
   Enter optional PCIOS file usage below:

   UDS Application Group Name: (      ) (only required for SFS file acces
    Internal             External File Name              File Access  S
    File Name        (@ASG qualifier*name/readkey)       Type  Mode   F
1. (ORDERFILE   ) (MKTG*ORDERFILE                      ) (M)  (V)    (
2. ($DUMMY-1    ) (OR-LINE-COUNT                       ) ( )  ( )    (
3. (            ) (                                    ) ( )  ( )    (
4. (            ) (                                    ) ( )  ( )    (
```

```
                    InfoQuest System
                 DMS1100 Database Access
==========================================================
     Schema File Name: (                              )
     DMR Invoke Name:  (        )  (Leave blank for default

                   Database Dataname To Initialize
      Area Names       (for CALC, if necessary)
  1. (           )   (                              )
```

```
  ===================================================================
                          InfoQuest System
                         Path Specification
  ===================================================================
START of path record/file name:(ORDERFILE                 )  Type:(F)
Area name (if DMS):(            )    Key(User):(
Range Key:( )                        Key(Actl):(
    Subordinate record/file:($DUMMY-1                 ) Meth:(C)
    Set name or key field name:(DM-LINES                  ) Non-select:(
              Source of key field:(OR-ORDER-LINE-DATA          )
    Subordinate record/file:(                          ) Meth:( )
    Set name or key field name:(                          ) Non-select:(
              Source of key field:(                            )
    Subordinate record/file:(                          ) Meth:( )
    Set name or key field name:(                          ) Non-select:(
```

Let's look at the difference from normal entries for each screen. On screen one, the $DUMMY-1 record is entered with the name of the data element referenced by the DEPENDING ON item of the OCCURS clause (OR-NUMBER-OF-LINES). This is used in controlling the loop that accesses each occurrence. If there were a fixed number of occurrences (e.g., OCCURS 12 TIMES), the number of occurrences could be entered in lieu of the data item name. The number must be entered bound by parentheses; e.g., (12).

Notice that there is no "File Type" or "Access Mode" specified.

Screen two in this example is left blank because no DMS1100 access is involved this time.

Screen three shows that the ORDERFILE is the start, or root, of the path. The first, and only, subordinate record in the path is the $DUMMY-1. Note that the method field is set to ‘C’ indicating that this access is to an occurring group. The KEY FIELD and SOURCE KEY FIELD entries are used to indicate the name of the occurring group in the actual record (OR-LINES), and the redefined name of the group in the dummy record (DM-LINES). These names will be used in generating group level moves (SET RDA commands) from each occurrence of the group to the dummy record. This will generate a procedure that will emulate a FETCH NEXT loop for a via set record.

The following illustrates how the actual DBM logic would be generated by the path generation process. Observe how the ACCESS-4098 (the dummy record access) procedure handles each occurrence.

```
. __________________________
*REC0000
*REC4098           $DUMMY-1
*REC4097           ORDERFILE

INDEX BOB*OF-INDEX.

ADD STDDEFS

. ********** File definitions **********
DEF F $DUMMY-1 SEQ 15,1
DEF F ORDERFILE INDEXED 1522,10 (1,11) (12,9) DUPS

. Alternate record area definitions...
DEF RA $DUMMY-1 AFTER STDMY
DEF RA ORDERFILE AFTER $DUMMY-1

. Record selection switches.  These switches are set
. by the generated selection routines to determine
. whether or not a record has been selected.
DEF N SW-SELECT0000
DEF N SW-SELECT4098
DEF N SW-SELECT4097

. Record access switched.  These switches are set by
. the generated SETUP routine to indicate which
. records are actually accessed in the request.
DEF N SW-REC0000
DEF N SW-REC4098
DEF N SW-REC4097

. Selection switches.  These switches, also set by the
. SETUP routine, indicate on which records selections
. are to be made in the request.
DEF N SW-SEL0000
DEF N SW-SEL4098
DEF N SW-SEL4097

BEGIN
           DO SETUP
           DO LOAD-SEL-TAB
           DO SORT-INIT

           DO HEADERS
```

```
. ***************************************************
. ****** Generated Navigation Logic Begins *****
   CSF X '@ASG,AZ ORD-ENT*ORDERFILE'
   IF X  0
     D 'Can''t assign file ORDERFILE'
     STOP 9999
   ENDIF
   CSF X '@USE ORDERFILE,ORD-ENT*ORDERFILE'
   OPEN ORDERFILE INPUT SEQ
   DO
     READ ORDERFILE AT END BREAK
     DO SELECT4097
     IF SW-SELECT4097 = 1
     IF SW-REC4098 = 1
       DO ACCESS-4098
     ELSE
       DO SELECTED
     ENDIF
     ENDIF
   ENDDO
   GO FINALE
KEY-ERR     . *** INVALID KEY ***
   DISPLAY '*** Invalid key error processing stopped'
   GO FINALE

SELECTED PROCEDURE
   DO MOVE-DTL
   IF SW-SORT  1
     DO DISP-DTL
   ENDIF
   ENDPROC

ACCESS-4098 PROCEDURE
DEF SUB SUB4098 OR-LINES
     SUB4098 = 1
     DO WHILE SUB4098 <= RDA OR-NUMBER-OF-LINES
       RDA DM-LINES = RDA OR-LINES :SUB4098
     DO SELECT4098
     IF SW-SELECT4098 = 1
       DO SELECTED
     ENDIF
     SUB4098 = SUB4098 + 1
   ENDDO
   ENDPROC

. ***** End code insertion:
. ***** INVOKE from file name count = 0
. ***** File definitions created = 2
. ***** Path code lines created = 43
. ****** Generated Path Navigation Logic Ends *****
. ***************************************************
```

For clarity, this example used only a single record with one occurs clause. The record may have been anywhere within a logical path as long as the occurring group can be viewed logically as one level of the hierarchy. Additionally, this process can be used to handle nested occurring groups (multidimensional tables). However, multiple non-nested occurring groups cannot be handled automatically. Such structures may be made accessible through manual modifications to a generated DBM.

Chapter 7 Demonstration/Validation

The demonstration application has been shipped with InfoQuest to provide you with a method of validating your InfoQuest installation, and to provide you with a training application. The application can be used for training both InfoQuest Support personnel and end users.

7.1 The Application

The demonstration application is a small sales order entry system. It consists of three PCIOS files. PCIOS files were chosen for this application to avoid complications involved in attempting to integrate other database files (i.e., DMS-1100 areas or RDMS-1100 tables) into the customer's local environment. The files needed for this application can be easily copied onto your system.

The three data files are:

CUSTFILE	This file is the customer master. It contains customer specific information such as name, address and telephone number. It is a multi-keyed Index Sequential (MSAM) file. The primary key is the customer's account number. The secondary key is made up of the customer's 2-character state abbreviation and city. This key structure allows access to the customer master randomly by account, or in groupings by location.
ORDERFILE	This file contains all current sales orders. It is also an MSAM file. Its primary key is a unique order identification number. Its secondary key is the customer's account number. This key structure allows orders to be randomly found by order identification number, or in groupings by customer account.
ORDERHIST	This file is used to perform historical analysis of past sales. It is a sequential file which includes one record containing selected information from each line item of closed sales orders.

COBOL descriptions of these files are shown at the end of this chapter.

Three InfoQuest views and associated DBMs have been set up for this application. The first is a general purpose inquiry which allows the user to access any information on current orders based on any selection criteria.

The second view provides access to the same information as the first, but restricts the user to access by customer location. By using the customer location as a required entry point, the file search time is greatly reduced. The reduction is accomplished by accessing the CUSTFILE using the secondary key (state+city).

The third view provides access to sales historical information contained in the ORDERHIST file. Sequential, the most simple form of InfoQuest access, is used in this view.

7.2 InfoQuest Database Access Paths

Let's look a little more deeply into just how the demonstration data files will be accessed by the InfoQuest database modules (DBMs).

The first two views utilize a logical linkage between the CUSTFILE and the ORDERFILE. The logical link is the customer account number. This field is named CM-ACCOUNT in the CUSTFILE and is its primary key. The customer account number is named OR-CUSTKEY in the ORDERFILE and is its secondary key.

The first view, which provides general access to information in these two files, uses a path which is based on sequential access to the ORDERFILE. Whenever customer specific information is required by the user request, a CUSTFILE record is accessed randomly using the account found in the current ORDERFILE record. While this approach can generate a great deal of I/O, it does provide the capability of delivering any data contained within the two files based on any selection criteria.

While the second view provides the same information, it can produce results more quickly. In this view, access is based on a key, or range of keys, which must be furnished by the end user. Access in the view is based on the secondary key of the CUSTFILE. In the view, the field name LOCKEY is flagged as required and as a range-key. The user will always be prompted to enter selection values for this data item. InfoQuest's physical access uses the MSAM START command to position to the first matching location, then reads forward sequentially until the limit of the range is reached. If order information is required, the secondary key of the ORDERFILE, the account number, is used to access order records using that same technique. This method greatly reduces the number of records that must be read to complete the request.

One thing we haven't looked at yet is how the occurring line item entries within the order records are accessed. Since line items can be viewed as logically subordinate records to static order header information, InfoQuest accesses line items as if they were a chain of separate records attached to each order record.

As mentioned earlier, sequential, the most simple form of access possible, is used in this view. This view will only provide information from the history file. If the user would like to combine customer specific information with the historical data, a more complex path must be set up, which would be an excellent learning exercise for a new customer.

The demonstration files must remain catalogued under the following names in order to be used for InfoQuest.

IQ$DEMO*IQ$DEMO.
IQ$DEMO*CURR-ORD-IDX.
IQ$DEMO*ORD-HIST-IDX.
IQ$DEMO*CUSTFILE.
IQ$DEMO*ORDERFILE.
IQ$DEMO*ORDERHIST.

For a complete description of all files on the release tape, see Chapter 1 of the InfoQuest Installation Guide.

7.3 COBOL File Definitions

The following are the COBOL definitions of the data files used in the demonstration application:

1. ORDERFILE:

```
 01  ORDER-RECORD.
** Secondary key, duplicates allowed **
     05  OR-CUSTKEY.
         10  OR-CUSTDIV            PIC 9.
         10  OR-CUSTNUM            PIC X(5).
         10  OR-CUSTSHIPTO         PIC 999.
** Primary key **
     05  OR-ORDER-IDENT.
         10  OR-ORDER-LOC          PIC 99.
         10  OR-ORDER-KEY          PIC X(7).
         10  OR-SHIP-LOC           PIC 99.
     05  OR-ORDER-TYPE-CODE        PIC X.
     05  OR-PRODLINE-CODE          PIC X.
     05  OR-ENTRY-DATE.
         10  OR-ENTRY-MO           PIC XX.
         10  OR-ENTRY-DA           PIC XX.
         10  OR-ENTRY-YR           PIC XX.
     05  OR-BUYER.
         10  OR-BYPASS             PIC X.
         10  FILLER                PIC X(10).
     05  OR-PURCHASE-ORD           PIC X(8).
     05  OR-REQ-SHIPDATE           PIC X(6).
     05  OR-SHIP-VIA               PIC X(11).
     05  OR-CREDIT-HOLD            PIC X.
     05  OR-HOLD-CODE              PIC X.
     05  OR-INVOICE-CODE           PIC X.
     05  OR-BOL-PRT-CODE           PIC X.
     05  OR-PAY-METHOD-CODE        PIC XXX.
     05  OR-WORKORD-CODE           PIC X.
```

```
          05  OR-SPECIAL-TERMS         PIC X(20).
          05  OR-TERMS.
              10  OR-TERM-CODE         PIC X.
              10  OR-TERM-PER          PIC V9(4) COMP.
              10  OR-TERM-DATE-DAYS    PIC 9(6).
          05  OR-DELETE-FLAG           PIC X.
          05  OR-INPROCESS-HOLD        PIC X.
          05  FILLER                   PIC XX.
          05  OR-ACTUAL-SHIP-DATE.
              10  OR-SHIP-YR           PIC XX.
              10  OR-SHIP-MO           PIC XX.
              10  OR-SHIP-DA           PIC XX.
          05  OR-PIECES                PIC 9(5) COMP.
          05  FILLER                   PIC XX.
          05  OR-WEIGHT                PIC 9(7) COMP.
          05  OR-SHIP-FEE              PIC 9(5)V99 COMP.
          05  OR-TOT-CHARGES           PIC 9(6)V99 COMP.
          05  OR-TOT-CLC               PIC 9(6)V99 COMP.
          05  OR-DISCOUNT              PIC 9(5)V99 COMP.
          05  OR-CREDIT-REL-DATE.
              10  OR-CREL-YR           PIC XX.
              10  OR-CREL-MO           PIC XX.
              10  OR-CREL-DA           PIC XX.
          05  OR-WORKORD-PRT-DATE.
              10  OR-WKORD-PRT-YR      PIC XX.
              10  OR-WKORD-PRT-MO      PIC XX.
              10  OR-WKORD-PRT-DA      PIC XX.
          05  OR-STATE-TAX             PIC S9(4)V99 COMP.
          05  OR-CITY-TAX              PIC S9(4)V99 COMP.
          05  OR-COUNTY-TAX            PIC S9(4)V99 COMP.
          05  OR-CREDIT-USERID         PIC X(8).
          05  OR-NBR-PALLETS           PIC S9(2)  COMP.
          05  OR-PALLET-CHG            PIC S9(3)V99  COMP.
          05  OR-TOT-PALLET-COST       PIC S9(5)V99  COMP.
          05  OR-AUTHDLR-CODE          PIC X.
          05  OR-LINE-COUNT            PIC 9(10) COMP.
          05  OR-ORDER-LINE-DATA OCCURS 1 TO 50.
     *** Order line item data ***
              10  OR-PRODUCT           PIC X(6).
              10  OR-TYPE-ORD-CODE     PIC X.
              10  OR-QUANTITY          PIC S9(5) COMP.
              10  OR-UNIT-PRICE        PIC 9(5)V99 COMP.
              10  OR-DESC              PIC X(25).
              10  OR-WEIGHT            PIC 9(5)V99 COMP.
              10  OR-PACKAGE           PIC X(8).
              10  OR-PRICE-CODE        PIC XX.
              10  OR-BOL-KEY           PIC 999 COMP.
              10  OR-TAX-CODE          PIC X.
              10  OR-REG-CODE          PIC X.
              10  OR-SHIP-QTY          PIC 9(5) COMP.
              10  OR-BILL-ONLY-CODE    PIC X.
              10  OR-PRICE-CHANGE      PIC X.
              10  OR-LAST-DATE         PIC X(6).
              10  OR-PRIORITY          PIC X.
              10  OR-EXCEPTION-SW      PIC X.
              10  OR-SUB-ITEM          PIC X(6).
```

2. CUSTFILE:

```
      01  CUSTOMER-MASTER-REC.
     ** Primary key **
          05  CM-ACCOUNT               PIC X(9).
     ** Secondary key, duplicates allowed **
```

```
        05  CM-LOCKEY.
            10   CM-STATE            PIC XX.
            10   CM-CITY             PIC X(18).
        05  CM-CUSTNAME              PIC X(33).
        05  CM-ADDR1                 PIC X(30).
        05  CM-ADDR2                 PIC X(30).
        05  CM-ADDR3                 PIC X(30).
        05  CM-ZIP                   PIC X(9).
        05  CM-TELEPHONE.
            10  CM-AREACODE          PIC XXX.
            10  CM-EXCHANGE          PIC XXX.
            10  CM-TELNUM            PIC XXXX.
```

3. ORDERHIST:

```
    01  ORDER-HISTORY-RECORD.
        05  OH-ACCOUNT               PIC X(9).
        05  OH-ORDER-IDENT           PIC X(11).
        05  OH-ORDER-TYPE-CODE       PIC X.
        05  OH-PRODLINE-CODE         PIC X.
        05  OH-ENTRY-DATE.
            10  OH-ENTRY-MO          PIC XX.
            10  OH-ENTRY-DA          PIC XX.
            10  OH-ENTRY-YR          PIC XX.
        05  OH-BUYER                 PIC X(19).
        05  OH-PURCHASE-ORD          PIC X(8).
        05  OH-SPECIAL-TERMS         PIC X(20).
        05  OH-ACTUAL-SHIP-DATE.
            10  OH-SHIP-YR           PIC XX.
            10  OH-SHIP-MO           PIC XX.
            10  OH-SHIP-DA           PIC XX.
        05  OH-DISCOUNT              PIC 9(5)V99 COMP.
        05  OH-STATE-TAX             PIC S9(4)V99 COMP.
        05  OH-CITY-TAX              PIC S9(4)V99 COMP.
        05  OH-COUNTY-TAX            PIC S9(4)V99 COMP.
        05  OH-AUTHDLR-CODE          PIC X.
        05  OH-PRODUCT               PIC X(6).
        05  OH-QUANTITY              PIC S9(5) COMP.
        05  OH-UNIT-PRICE            PIC 9(5)V99 COMP.
        05  OH-WEIGHT                PIC 9(5)V99 COMP.
        05  OH-TAX-CODE              PIC X.
```

7.4 QINDEX Definitions

The following are the QINDEX definitions set up for the demonstration application. Note the dummy record definition created to handle the OCCURS DEPENDING ON in the ORDERFILE definition.

1. CURR-ORDER/QINDEX:

```
»@DELETE,C IQ$DEMO*CURR-ORD-IDX.
»@ASG,UPV  IQ$DEMO*CURR-ORD-IDX.,F///50
»@FREE     IQ$DEMO*CURR-ORD-IDX.
»@IQNDXK,LI IQ$DEMO*CURR-ORD-IDX.
»FILE ORDERFILE
»       01  ORDER-RECORD.
»      ** Secondary key, duplicates allowed **
»           05  OR-CUSTKEY.
»               10  OR-CUSTDIV           PIC 9.
»              10  OR-CUSTNUM           PIC X(5).
»              10  OR-CUSTSHIPTO        PIC 999.
»       ** Primary key **
»          05  OR-ORDER-IDENT.
»              10  OR-ORDER-LOC         PIC 99.
»              10  OR-ORDER-KEY         PIC X(7).
»              10  OR-SHIP-LOC          PIC 99.
»          05  OR-ORDER-TYPE-CODE       PIC X.
»          05  OR-PRODLINE-CODE         PIC X.
»          05  OR-ENTRY-DATE.
»              10  OR-ENTRY-MO          PIC XX.
»              10  OR-ENTRY-DA          PIC XX.
»              10  OR-ENTRY-YR          PIC XX.
»          05  OR-BUYER.
»              10  OR-BYPASS            PIC X.
»              10  FILLER               PIC X(10).
»          05  OR-PURCHASE-ORD          PIC X(8).
»          05  OR-REQ-SHIPDATE          PIC X(6).
»          05  OR-SHIP-VIA              PIC X(11).
»          05  OR-CREDIT-HOLD           PIC X.
»          05  OR-HOLD-CODE             PIC X.
»          05  OR-INVOICE-CODE          PIC X.
»          05  OR-BOL-PRT-CODE          PIC X.
»          05  OR-PAY-METHOD-CODE       PIC XXX.
»          05  OR-WORKORD-CODE          PIC X.
»          05  OR-SPECIAL-TERMS         PIC X(20).
»          05  OR-TERMS.
»              10  OR-TERM-CODE         PIC X.
»              10  OR-TERM-PER          PIC V9(4) COMP.
»              10  OR-TERM-DATE-DAYS    PIC 9(6).
»          05  OR-DELETE-FLAG           PIC X.
»          05  OR-INPROCESS-HOLD        PIC X.
»          05  FILLER                   PIC XX.
»          05  OR-ACTUAL-SHIP-DATE.
»              10  OR-SHIP-YR           PIC XX.
»              10  OR-SHIP-MO           PIC XX.
»              10  OR-SHIP-DA           PIC XX.
»          05  OR-PIECES                PIC 9(5) COMP.
»          05  FILLER                   PIC XX.
»          05  OR-WEIGHT                PIC 9(7) COMP.
»          05  OR-SHIP-FEE              PIC 9(5)V99 COMP.
»          05  OR-TOT-CHARGES           PIC 9(6)V99 COMP.
»          05  OR-TOT-CLC               PIC 9(6)V99 COMP.
»          05  OR-DISCOUNT              PIC 9(5)V99 COMP.
»          05  OR-CREDIT-REL-DATE.
»              10  OR-CREL-YR           PIC XX.
»              10  OR-CREL-MO           PIC XX.
»              10  OR-CREL-DA           PIC XX.
»          05  OR-WORKORD-PRT-DATE.
»              10  OR-WKORD-PRT-YR      PIC XX.
»              10  OR-WKORD-PRT-MO      PIC XX.
»              10  OR-WKORD-PRT-DA      PIC XX.
»          05  OR-STATE-TAX             PIC S9(4)V99 COMP.
»          05  OR-CITY-TAX              PIC S9(4)V99 COMP.
»          05  OR-COUNTY-TAX            PIC S9(4)V99 COMP.
»          05  OR-CREDIT-USERID         PIC X(8).
»          05  OR-NBR-PALLETS           PIC S9(2) COMP.
»          05  OR-PALLET-CHG            PIC S9(3)V99 COMP.
»          05  OR-TOT-PALLET-COST       PIC S9(5)V99 COMP.
»          05  OR-AUTHDLR-CODE          PIC X.
»          05  OR-LINE-COUNT            PIC 9(10) COMP.
»       ** Line items field definitions have been moved to
»      ** dummy record.
»           05  OR-ORDER-LINE-DATA OCCURS 1 TO 50
»               DEPENDING ON OR-LINE-COUNT PIC X(72).
»      **
»      ** The following is actually a redefinition of
»      ** OR-ORDER-LINE-DATA.  It has been redefined as a
»      ** separate dummy record so that InfoQuest can handle
»      ** it as a logical record subordinate to the
»      ** ORDER-RECORD.
»      **
»FILE $DUMMY-OL
»       01  ORDER-LINE-DUMMY-REC.
»           05  OL-PRODUCT               PIC X(6).
»           05  OL-TYPE-ORD-CODE         PIC X.
»           05  OL-QUANTITY              PIC S9(5) COMP.
»           05  OL-UNIT-PRICE            PIC 9(5)V99 COMP.
»           05  OL-DESC                  PIC X(25).
»           05  OL-WEIGHT                PIC 9(5)V99 COMP.
»           05  OL-PACKAGE               PIC X(8).
»           05  OL-PRICE-CODE            PIC XX.
»           05  OL-BOL-KEY               PIC 999 COMP.
»           05  OL-TAX-CODE              PIC X.
»           05  OL-REG-CODE              PIC X.
»           05  OL-SHIP-QTY              PIC 9(5) COMP.
»           05  OL-BILL-ONLY-CODE        PIC X.
»           05  OL-PRICE-CHANGE          PIC X.
»           05  OL-LAST-DATE             PIC X(6).
»           05  OL-PRIORITY              PIC X.
»           05  OL-EXCEPTION-SW          PIC X.
»           05  OL-SUB-ITEM              PIC X(6).
»      **
»FILE CUSTFILE
»       01  CUSTORMER-MASTER-REC.
»      ** Primary key **
»           05  CM-ACCOUNT               PIC X(9).
»      ** Secondary key, duplicates allowed **
»           05  CM-LOCKEY.
»               10   CM-STATE            PIC XX.
»               10   CM-CITY             PIC X(18).
»           05  CM-CUSTNAME              PIC X(33).
»           05  CM-ADDR1                 PIC X(30).
»           05  CM-ADDR2                 PIC X(30).
»           05  CM-ADDR3                 PIC X(30).
»           05  CM-ZIP                   PIC X(9).
»           05  CM-TELEPHONE.
»               10  CM-AREACODE          PIC XXX.
»               10  CM-EXCHANGE          PIC XXX.
»               10  CM-TELNUM            PIC XXXX.
»@EOF
```

2. ORDER-HIST/QINDEX:

```
»@DELETE,C IQ$DEMO*ORD-HIST-IDX.
»@ASG,UPV  IQ$DEMO*ORD-HIST-IDX.,F///50
»@FREE     IQ$DEMO*ORD-HIST-IDX.
»@IQNDXK,LI IQ$DEMO*ORD-HIST-IDX.
»FILE ORDERHIST
»       01  ORDER-HISTORY-RECORD.
»           05  OH-ACCOUNT               PIC X(9).
»           05  OH-ORDER-IDENT           PIC X(11).
»           05  OH-ORDER-TYPE-CODE       PIC X.
»           05  OH-PRODLINE-CODE         PIC X.
»           05  OH-ENTRY-DATE.
»              10  OH-ENTRY-MO          PIC XX.
»              10  OH-ENTRY-DA          PIC XX.
»              10  OH-ENTRY-YR          PIC XX.
»          05  OH-BUYER                 PIC X(19).
»          05  OH-PURCHASE-ORD          PIC X(8).
»          05  OH-SPECIAL-TERMS         PIC X(20).
»          05  OH-ACTUAL-SHIP-DATE.
»              10  OH-SHIP-YR           PIC XX.
»              10  OH-SHIP-MO           PIC XX.
»              10  OH-SHIP-DA           PIC XX.
»          05  OH-DISCOUNT              PIC 9(5)V99 COMP.
»          05  OH-STATE-TAX             PIC S9(4)V99 COMP.
»          05  OH-CITY-TAX              PIC S9(4)V99 COMP.
»          05  OH-COUNTY-TAX            PIC S9(4)V99 COMP.
»          05  OH-AUTHDLR-CODE          PIC X.
»          05  OH-PRODUCT               PIC X(6).
»          05  OH-QUANTITY              PIC S9(5) COMP.
»          05  OH-UNIT-PRICE            PIC 9(5)V99 COMP.
»          05  OH-WEIGHT                PIC 9(5)V99 COMP.
»          05  OH-TAX-CODE              PIC X.
»@EOF
```

Chapter 8 Off-line Request Trace/Scheduler

InfoQuest provides three levels of off-line request control: no control, request trace only, and request scheduling with trace. The level of control is specified in the system configuration record. Each user registration entry indicates which system configuration record is to be used for the user and application, thus allowing the level of off-line control to be specified for a user/application.

When no control is used, InfoQuest off-line request runs are @STARTed immediately using the run parameters from the system configuration record (@RUN card image, Account/User-id and start time). Once such a run is started, no further monitoring takes place.

When request trace only is specified, the off-line request run is @STARTed immediately with no control, but log entries are kept tracing each event in the run's progress. The trace log may be queried to find the status of an off-line request. If a run, or its print output is lost, the trace file can be used to determine the print file name or at what point the run erred.

When request trace with scheduling is used (recommended for all off-line requests), the generated run file is saved but not @STARTed immediately. The fact that the run file has been built and saved is recorded. An off-line request monitor run then periodically checks the trace log file for runs ready to be started taking into account any scheduled start times. A maximum number of active requests is specified to the monitor, thus preventing the start up of too many InfoQuest off-line runs simultaneously.

8.1 InfoQuest Trace/Scheduler Files

There are two files maintained when trace only or trace and scheduling are configured: the trace log file and the trace log locking file. These files will be named as follows:

The trace log:

INFOQ[*x*]*INFOQLOG.

The locking file:

INFOQ[*x*]*INFOQLOCK.

x is dependent upon the COMUS mode used to install InfoQuest — IQMON for the default mode or IQMONA through IQMONK for modes INFOQA through INFOQK, respectively.

The trace log is used for tracking the events of each request run. The events tracked are indicated by the following status codes:

B - Run built and ready to execute.
E - Run is in execution.
F - Run has finished normally.
H - Run held due to maximum limit.
L - Run lost but restartable (was in execution, but no longer running).
Q - Run @STARTed.
R - Run has been errored off.
T - Run @START failed.
X - Run lost and not restartable (run file is deleted).

Status code B is only entered by the INFOQBATCH Q-LINK program which is the InfoQuest function that generates off-line request run streams. The B status can only occur if scheduling is configured. If trace only is configured, the INFOQBATCH program will enter the Q status into the log indicating the run has been @STARTed, instead of the B status. The Q status will be logged by the InfoQuest off-line monitor as it starts each run when trace and scheduling is configured.

The E, F and R status codes are logged from within the off-line run when trace only or trace with scheduling is configured. These entries are made by calling the monitor process with a logging option. The same processor is used for both monitoring and event logging depending on processor call options.

The H, L, X and T status codes are only entered into the trace log by the InfoQuest monitor.

The trace log contains the following information:

Unique print file name (primary key)
User-id
Department
Station
Report name
Run black-out start time
Run black-out end time
Scheduled start time
Scheduled start date
Run file name
Batch account and user-id
Current status (above)
Date and time run built

Date and time run @STARTed
Date and time run finished normally
Date and time run erred
Original batch run-id
Generated batch run-id

The locking file is used for two purposes: as a locking mechanism to control concurrent access to the log file, and to keep counts of current activity. Whenever a logging or monitoring function needs to access the trace log file, the EXEC read and lock IOW$ function is first used to lock the lock file. When log processing is complete, the lock file is released with a write IOW$ function.

The locking file is also used to maintain current activity counts. By maintaining these counts in the lock file, the monitor does not have to be active at all times; however, the monitor must be active and off-line scheduling configured for off-line requests to be automatically started. The counts maintained in the lock file are:

Maximum active runs
Runs ready to start (Built)
Starts failed
Runs held for maximum limit
Lost runs, restartable
Lost runs, not restartable
Currently active runs (Currently in execution)
Started runs (@STARTed may be in backlog)
Original batch run-id
Generated batch run-id

8.2 The InfoQuest Monitor Processor

The InfoQuest Monitor Processor, IQMON, is used to perform all event logging and monitoring functions. Its function is determined by processor call options used when it is executed. IQMON is called as follows:

@IQMON[*x*],*options spec-field*[,*x*]

x is dependent upon the COMUS mode used to install InfoQuest — @IQMON for the default mode or @IQMONA through @IQMONK for modes INFOQA through INFOQK, respectively.

The IQMON *options* are:

C	Refresh all IQMON counts at start-up time.
INT	Cause both the trace log and lock files to be initialized.
E	Log the beginning of run in execution.
F	Log the normal finish of the run.
R	Log the error finish of the run.
M	Begin monitoring InfoQuest off-line activity.

The entry in *spec-field* depends on the type of execution of IQMON. If IQMON is being executed with the E, F, or R options, *spec-field* contains the unique print file name which is used to locate the log entry being updated. If IQMON is being run with the M option (begin monitoring), *spec-field* contains the maximum number of runs active.

The second field, *x*, always contains the InfoQuest install mode indicator which is used in determining the name of the log and locking files as described earlier.

8.3 IQMON Operation

If no trace or scheduling is desired, there is no need for the IQMON processor, or the log and lock files. If either trace or trace with scheduling is desired, IQMON and the log and lock files are required.

The logging of events in the life of the off-line request will automatically be generated into the off-line request runstream when the configuration file indicates trace is on. The only requirement is that the log and lock files be created and initialized. The log and locking files must be secured in such a way as to be available to the Q-LINK servers that build InfoQuest requests and the off-line batch run and the off-line monitor run (if scheduling is to be used).

Both the log and lock files are initialized by calling the IQMON processor with the INT options as follows:

@IQMON[*x*],INT [,*x*]

x is dependent upon the COMUS mode used to install InfoQuest — @IQMON for the default mode or @IQMONA through @IQMONK for modes INFOQA through INFOQK, respectively.

If scheduling is configured, off-line requests will only be started via the IQMON processor. It is not necessary to have the monitor running at all times—it is only needed when off-line requests are to be started. Any requests that are built while the monitor is not running will be picked up for scheduling while the monitor is started. Since the monitor run starts off-line requests via an ER to ACSF$, it must be run under a valid account/user-id. The runs started by the monitor will have the same account/user-id as the monitor run, not the account/user-id specified in the configuration file. The monitor runstream is placed in SYS$LIB$*RUN$ when InfoQuest is installed. If the default installation mode was used, the element name will be IQMON — IQMONA through IQMONK for the alternate modes. The default mode runstream contains the following:

```
@RUN,A IQMON,A/U,IQUINS,999,999/0
@ . THIS RUNSTREAM INSTALLED BY KMSMAS USING COMUS DB QKMS
@MSG **** INFOQUEST MONITOR RUN STARTING *****
@ASG,A SYS$LIB$*INFOQUEST-1(25).
@COPY,A SYS$LIB$*INFOQUEST-1(25).IQMON,TPF$.
@FREE SYS$LIB$*INFOQUEST-1(25).
@ . CALL IQMON PROGRAM........................
@.IQMON,M 2,P    . (2=initial-max-active-runs,P=install-mode)
@PMD,LEAP
@FIN
```

When IQMON is started for InfoQuest off-line request monitoring, it first sets the maximum number of active off-line requests allowed. The initial value for the maximum number of active off-line requests is obtained from spec-field one of the IQMON processor call statement (shown above).

The maximum active runs parameter is the number of off-line requests that may be either running or in back log to run. Using the maximum active runs parameter prevents having too many InfoQuest batch runs active in the EXEC but queued to a Q-LINK server class.

When IQMON process begins monitoring, it firsts reads the current counts from the locking file, then makes a pass through the trace log file. When the trace log is read, off-line requests are @STARTed at the allowed maximum. IQMON then reads the locking and log files once every minute. On each pass of the log file, the monitor checks for runs ready to start, runs already active, and runs that may be lost.

Lost runs are determined by the presence of the run stream file which is assigned by the off-line run when is starts and is deleted from within the off-line run on normal completion. If the monitor finds a log entry with a status of E, for executing, and its run file is not currently assigned, it is given a status of L, meaning lost but restartable. If the current status is “Q” or “E”, in back log or executing, and the run file is not catalogued, the log entry is given a status of X which means the run is lost and not restartable.

The IQMON process, when active and in monitor mode, can be given commands via II keyins from the system console. The following commands are currently available:

HELP	Displays a list of available keyins.
LOCK	Lock the IQMON trace log file and prohibit jobs from starting until unlocked. Every 15 minutes, IQMON will display a console warning message that the file has been locked.
NEWMAX	Provides a mechanism to change the maximum number of active runs. A second prompt will solicit the new number.
RESTRT	Restarts all off-line requests that are currently considered lost but restartable (Status code L).
REFRSH	Causes the counts in the locking file to be cleared. IQMON does not allow a refresh of the counts while IQMON scheduled jobs are active. To refresh the counts, first use the LOCK command (see above) to suspend scheduling. Wait until all active jobs have completed before issuing the REFRESH command. Once the counts have been refreshed, use the UNLOCK command to resume scheduling.
STATUS	Lists the current counts from the locking file and display the status (locked or unlocked) of the IQMON trace log file.
STCYCLE	Provides a mechanism to change the frequency of the automatic status display on the system console. The initial value is every 200 minutes. A zero turns the automatic display off.
TERM	Terminates the monitor.
UNLOCK	Unlock IQMON trace log file and allow job scheduling to resume.

8.4 The InfoQuest Log Utility Run

A utility, which is called from the IQMNT main menu, is supplied to give the InfoQuest administrator access to the lock and trace log files also. The log access run provides the following functions:

1. Retrieve current status counts and/or view trace log entries from selected date/time and display status of the trace log file.
2. Delete, or Kill, catalogued print files no longer needed.
3. Purge trace log entries and delete all associated files (off-line runstreams and print files) up to the select date entered.

All log access functions except “S” (retrieve current status counts) allow multiple selections based on status code(s), user-id, department and print file name. For the purge function, “P”, the additional selection criteria, date and time, are required.

When using report name for selection, a search for a particular string match within the report name may be invoked by enclosing the search value within double quotes (“). If the report name is not quoted, an exact match will be assumed.

All print files are created by off-line requests and will be printed, then deleted (unless the user chose to save the print file on the host), upon completion. If for any reason a completed print file does not print, it can be found by searching the trace log and then manually retrieved or printed.

8.5 Steps to Implementing Off-line Trace/Scheduling

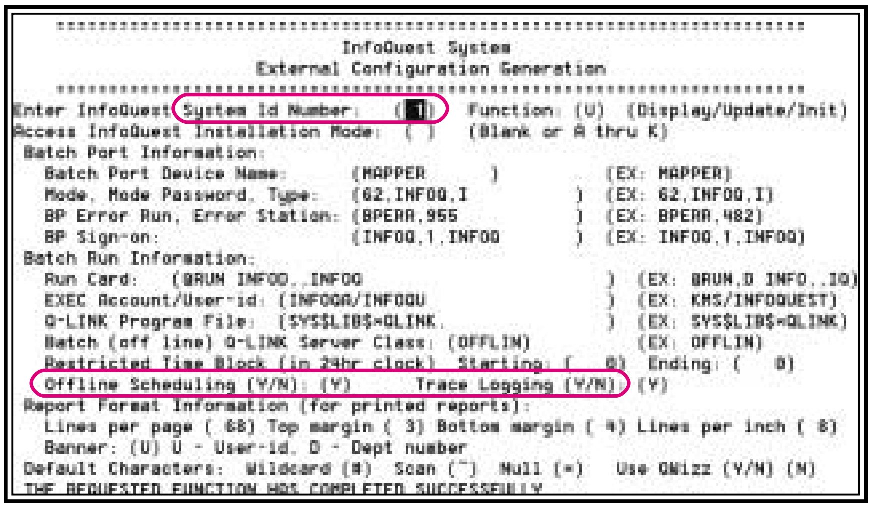
Six basic steps are required to enable off-line requests to be automatically scheduled and traced. The six steps are outlined below and discussed in detail on the following pages:

1. Mark the external configuration record of each system configuration to use trace and scheduling.
2. Assure that each user whose off-line requests are to be scheduled is registered to use one of the enabled system configurations.
3. Initialize the log and trace files. This should only be done the very first time trace/scheduling is enabled.
4. Start the IQMON batch run that is provided. This run must be active in order for the off-line requests to be started.
5. Schedule the actual request when generating the request with the IQ run.
6. Trace the request with the log utility.
7. If it is desirable to run the request automatically on a scheduled basis (daily, weekly, etc.), register the request with the Automatic Scheduling function of InfoQuest.

Steps to Implementing Off-line Trace/Scheduling

8.5.1 Set an External Configuration for Trace/Scheduling

Using the IQMNT run, select the “Security and Registration” function and subsequently the “External Configuration” function to establish a new configuration or modify an existing configuration record. Enable trace and scheduling by entering a “Y” in both fields as shown below.



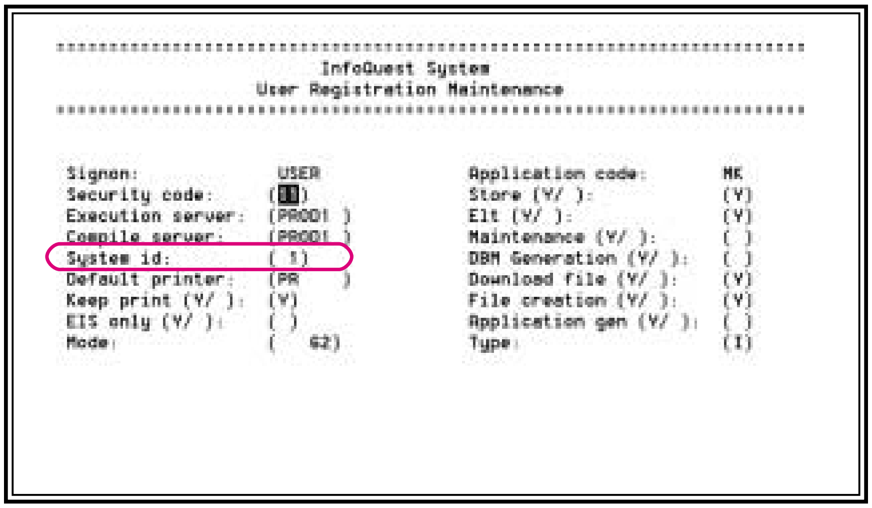
```
    ======================================================================
                           InfoQuest System
                    External Configuration Generation
    ======================================================================
Enter InfoQuest System Id Number:   (1)  Function: (U)  (Display/Update/Init)
Access InfoQuest Installation Mode:  ( )  (Blank or A thru K)
 Batch Port Information:
   Batch Port Device Name:       (MAPPER      )         (EX: MAPPER)
   Mode, Mode Password, Type:    (62,INFOQ,I           ) (EX: 62,INFOQ,I)
   BP Error Run, Error Station: (BPERR,955            ) (EX: BPERR,482)
   BP Sign-on:                   (INFOQ,1,INFOQ        ) (EX: INFOQ,1,INFOQ)
 Batch Run Information:
   Run Card:    (@RUN INFOQ,,INFOQ                      )  (EX: @RUN,D INFO,,IQ)
   EXEC Account/User-id: (INFOQA/INFOQU                 )  (EX: KMS/INFOQUEST)
   Q-LINK Program File:  (SYS$LIB$*QLINK.               )  (EX: SYS$LIB$*QLINK)
   Batch (off line) Q-LINK Server Class: (OFFLIN)          (EX: OFFLIN)
   Restricted Time Block (in 24hr clock)  Starting: (    0)  Ending: (    0)
   Offline Scheduling (Y/N): (Y)       Trace Logging (Y/N): (Y)
 Report Format Information (for printed reports):
   Lines per page ( 68) Top margin ( 3) Bottom margin ( 4) Lines per inch ( 6)
   Banner: (U) U - User-id, D - Dept number
 Default Characters:  Wildcard (#)  Scan (^)  Null (=)   Use QWizz (Y/N) (N)
 THE REQUESTED FUNCTION HAS COMPLETED SUCCESSFULLY
```

The “System Id” chosen will be used to associate one or more users with a particular configuration for off-line request processing.

As stated in Section 3.4.1, “External Configuration Generation”, several configuration records may be established.

8.5.2 Register the User to Use the Configuration

Next, go back to the “Security and Registration” menu and select the “User Registration” function. Each user whose off-line requests are to be scheduled must be registered with a “System Id” that has trace and scheduling enabled.



```
*********************************************************************
                        InfoQuest System
                  User Registration Maintenance
*********************************************************************

 Signon:            USER              Application code:       MK
 Security code:    (  )               Store (Y/ ):            (Y)
 Execution server: (PROD1  )          Elt (Y/ ):              (Y)
 Compile server:   (PROD1  )          Maintenance (Y/ ):      ( )
 System id:        ( 1)               DBM Generation (Y/ ):   ( )
 Default printer:  (PR     )          Download file (Y/ ):    (Y)
 Keep print (Y/ ): (Y)                File creation (Y/ ):    (Y)
 EIS only (Y/ ):   ( )                Application gen (Y/ ):  ( )
 Mode:             (    62)           Type:                   (I)
```

8.5.3 Initialize the Log and Trace Files

The log and trace files must be initialized the first time that scheduling is enabled. This initialization can be accomplished from a DEMAND terminal by using the “INT” options on the IQMON processor as illustrated below:

```
@IQMON[x],INT ,P
```

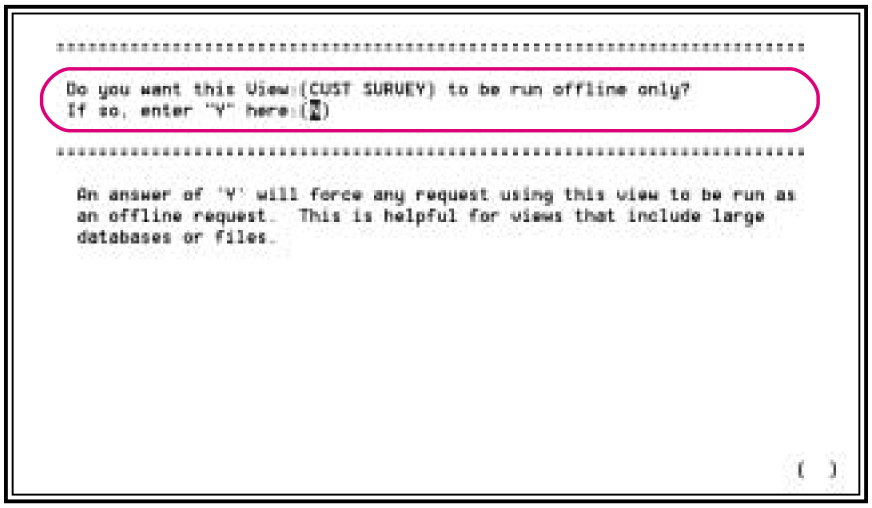
x is dependent upon the COMUS mode used to install InfoQuest — @IQMON for the default mode or @IQMONA through @IQMONK for modes INFOQA through INFOQK, respectively.

8.5.4 Starting the IQMON Batch Run

Start the IQMON batch run from the system console (ST IQMON for production or ST IQMT for test). The runstream is installed into SYS$LIB$*RUN$ and may be started from a demand terminal or another batch runstream if desired.

8.5.5 Scheduling Off-line Requests

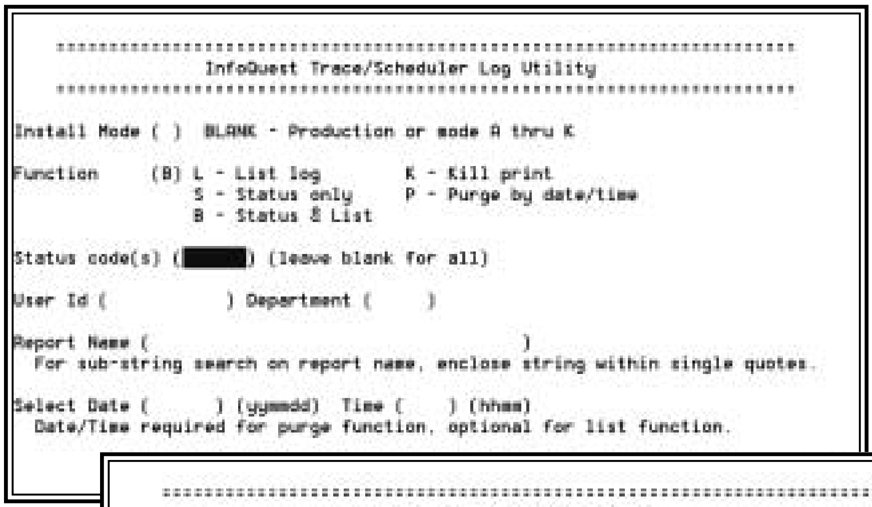
Off-line requests may be scheduled in two ways. The first way is for the user to choose the off-line option when the request is generated from InfoQuest. The second way to cause requests to be scheduled for off-line execution is to set an option with the “View Maintenance” function of the IQMNT run (see below). This option forces all requests that are generated using that view to be run off-line.



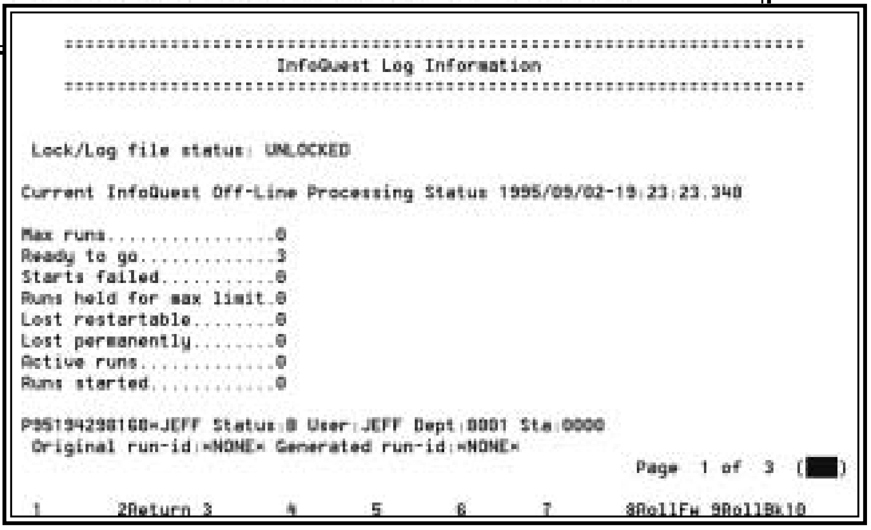
```
========================================================================

Do you want this View:(CUST SURVEY) to be run offline only?
If so, enter "Y" here:(N)

========================================================================

  An answer of 'Y' will force any request using this view to be run as
  an offline request.  This is helpful for views that include large
  databases or files.




                                                                    (  )
```

8.5.6 Tracing Scheduled Requests

As mentioned earlier, a utility is provided to monitor and maintain the log entries and files associated with off-line scheduling. An example of the “InfoQuest Log Utility” function of the IQMNT run is shown below:



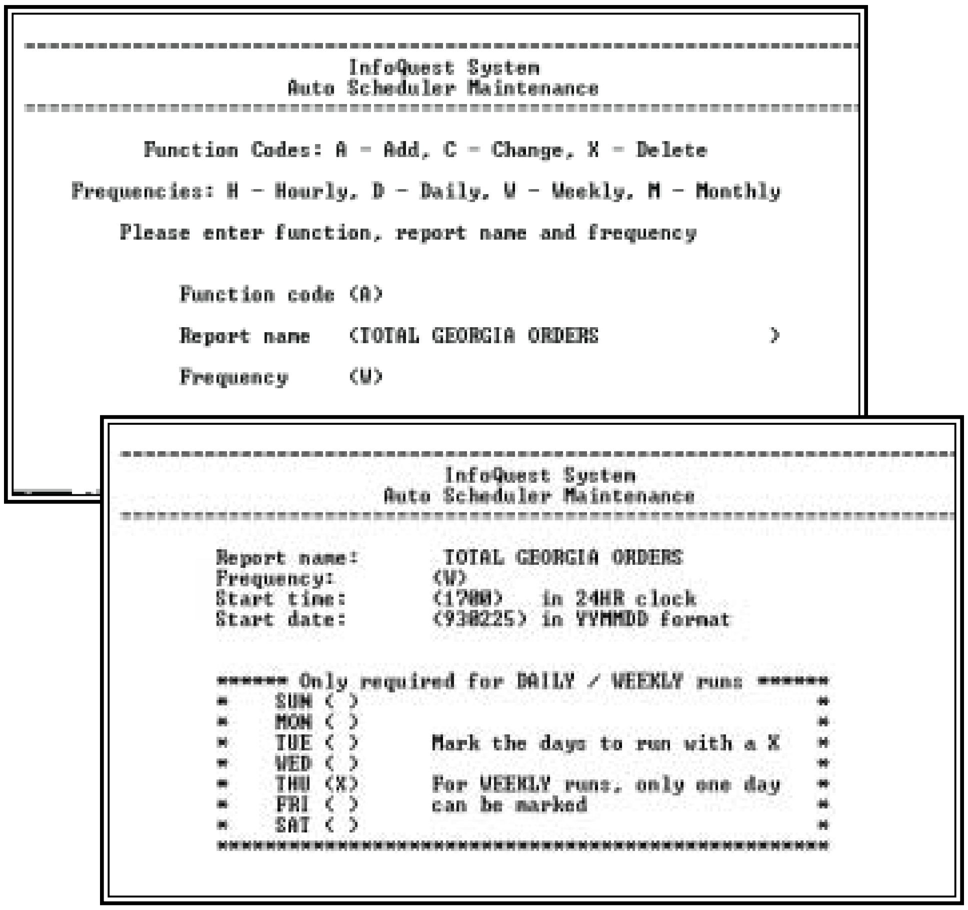
```
    ======================================================================
                  InfoQuest Trace/Scheduler Log Utility
    ======================================================================

Install Mode ( )  BLANK - Production or mode A thru K

Function     (B) L - List log        K - Kill print
                 S - Status only     P - Purge by date/time
                 B - Status & List

Status code(s) (      ) (leave blank for all)

User Id (           ) Department (     )

Report Name (                                    )
  For sub-string search on report name, enclose string within single quotes.

Select Date (      ) (yymmdd)  Time (    ) (hhmm)
  Date/Time required for purge function, optional for list function.
```



```
    ======================================================================
                        InfoQuest Log Information
    ======================================================================

 Lock/Log file status: UNLOCKED

Current InfoQuest Off-Line Processing Status 1995/09/02-19:23:23.340

Max runs.................0
Ready to go..............3
Starts failed............0
Runs held for max limit.0
Lost restartable.........0
Lost permanently.........0
Active runs..............0
Runs started.............0

P95194298160*JEFF Status:8 User:JEFF Dept:0001 Sta:0000
 Original run-id:*NONE* Generated run-id:*NONE*
                                                   Page  1 of  3 (   )

 1       2Return 3       4       5       6       7       8RollFw 9RollBk10
```

8.5.7 Auto-Scheduling of Requests

If a request is to be scheduled to be run on a regular basis (i.e., daily, weekly, etc.), the auto-scheduling feature should be used. The user enters the information on two screens as shown below:



```
======================================================================
                          InfoQuest System
                    Auto Scheduler Maintenance
======================================================================

          Function Codes: A - Add, C - Change, X - Delete

    Frequencies: H - Hourly, D - Daily, W - Weekly, M - Monthly

         Please enter function, report name and frequency

             Function code (A)

             Report name   (TOTAL GEORGIA ORDERS              )

             Frequency     (W)
```

```
======================================================================
                          InfoQuest System
                    Auto Scheduler Maintenance
======================================================================

       Report name:       TOTAL GEORGIA ORDERS
       Frequency:        (W)
       Start time:       (1700)   in 24HR clock
       Start date:       (930225) in YYMMDD format

       ****** Only required for DAILY / WEEKLY runs ******
       *    SUN ( )                                      *
       *    MON ( )                                      *
       *    TUE ( )      Mark the days to run with a X   *
       *    WED ( )                                      *
       *    THU (X)      For WEEKLY runs, only one day   *
       *    FRI ( )      can be marked                   *
       *    SAT ( )                                      *
       ***************************************************
```

The example above was invoked from User Housekeeping of the IQEX program. The same facility is also available in the MAPPER-based InfoQuest and InfoQuest Client.

Steps to Implementing Off-line Trace/Scheduling

Appendix A QINDEX Reference

InfoQuest uses a file called the data item index file in order to obtain the data item definitions for DMS1100 databases and PCIOS files. This file is built by the QINDEX processor executed in demand mode. QINDEX is a processor that is installed on your system when another KMSystems product, Q-LINK (used by InfoQuest), is installed. This appendix describes the purpose and rules for using the QINDEX processor.

A.1 Introduction

The data item index file is used by the Q-LINK processor to automatically obtain the location and format of data items in the record delivery area (RDA). Q-LINK automatically creates a primary data item index file on start-up. When an INVOKE directive is processed, information related to DMS1100 areas, records, sets and database datanames, and information related to data items (fields within records) is added to the primary index. This information is used in editing and encoding DMS1100 DML commands, and in the resolution of RDA names used within various commands and directives. Data item definitions in the primary data item index are obtained directly from the object schema and subschema.

Data redefinitions coded in the subschema are not available via the INVOKE process, because they are not included in the object subschema. They are only present in the S$PROC element created by the SDDL and copied into user programs by the ADMLP processor. To make redefinitions available to Q-LINK, a secondary data item index may be created using QINDEX.

When a data item name is used in a Q-LINK command or directive, the processor will first search its internal data definition table for the item name. Then, if the user has not defined the item (DEFINE RDA), the search will continue to the primary data item index for the correct definition. If the name is not found in the primary index, the secondary index, if present, is searched. The search of either index file will also match the record qualifier, if one was used. If the item name appears in more than one record and no record qualification is used, the first matching definition will be used.

The search order used in resolving a data item name is as follows:

1. Current DEFINE RDA names (always overrides data item index).
2. The primary data item index.
3. The secondary data item index.

A major use of the QINDEX processor is to create a single definition for all data within an entire application. The definition may be composed from a DMS1100 subschema and schema plus any number of non-DMS1100 file definitions. Non-DMS1100 files may include any file types supported by the Q-LINK processor.

A.2 QINDEX, General Information

The QINDEX processor is designed to create compatible data item index files from object schemas and subschemas, and standard COBOL definitions for use in the Q-LINK processor. The data item index file created by QINDEX may be invoked by the Q-LINK processor by using the INDEX directive (see your Q-LINK Programmer Reference Manual).

A.2.1 Input to QINDEX

The input to QINDEX consists of several directives, COBOL data definitions and optionally, an object schema and subschema. COBOL data definitions from many elements may be processed at one time (the processor call may be followed by many @ADD statements). There is no limit to the number of items that can be included in the data item index file. COBOL data definitions are input in standard COBOL source image format, and must follow certain restrictions.

QINDEX will assume that the COBOL source input has been processed through the COBOL compiler at some point, and therefore will perform no COBOL syntax checking.

Each 01 level definition from the source input will begin a new record definition. The generated RDA position pointer will be reset to character position one. The first data item image input to the QINDEX Processor must be an 01 level.

Input images may contain comments, COBOL PROC headings and END statements. These will be ignored by the QINDEX processor.

QINDEX will ignore all 66, 77 and 88 level items. FILLER items will be used to determine data item positions, and will not be output to the data item index file. Any VALUE clauses encountered will also be ignored.

Input must not contain mixed FIELDATA and ASCII definitions. Allocation is based on the processor option settings. QINDEX will not make a distinction between DISPLAY and DISPLAY-1 or COMP and COMP-4. QINDEX will not support or recognize FIELDATA binary alignment (for example, PIC H99999).

On items within an OCCURS clause, QINDEX will generate a definition that refers to the first item occurrence. References to subsequent occurrences, may be made by using either the RDA Indexing or Subscripting features of the Q-LINK Processor.

QINDEX supports exact binary notation for both DMS 1100 records (INVOKE directive) and PCIOS files (FILE directive). QINDEX can properly align data items that are included in record/file definitions (COBOL 01 level) containing exact binary data items. For example:

```
PIC 1(36)
PIC 1(5)
PIC 1(1)
```

This feature is currently supported for field alignment purposes only. A future release of Q-LINK will support access-to-bit definitions.

A.2.2 Output of QINDEX

The final output of QINDEX is a Q-LINK compatible secondary data item index file. In addition to the data item index file, QINDEX may optionally produce a listing of the data item index file's contents. The listing will show each entry in alphabetical sequence, grouped by areas, records, sets, database datanames and data items.

A.3 Running QINDEX

QINDEX must be run as a processor (not an @XQT) using the following format:

@QINDEX,*options index-file-name*
Followed by DIRECTIVES and user-supplied COBOL data definition source images

The processor call name given to QINDEX (the default name) is determined when QINDEX is installed. For the correct processor name at your site, consult with the person responsible for installing QINDEX at your site.

The **index-file-name** must be the name of an existing mass-storage file that will be used as the Data Item Index File.

Following the processor call line are the optional QINDEX directives and user-supplied COBOL source images that will be used in processing the Data Item Index File.

A.3.1 QINDEX Processor Options

Available execute *options* are:

B	Causes the execution of QINDEX to be treated as a batch mode execution.

D	Causes the execution of QINDEX to be treated as a demand mode execution. This option is the converse of the B-Option.
F	Option “F” indicates that the definitions being processed are FIELDATA. The default is ASCII.
I	The “I” option will cause the index file to be initialized. Any existing definitions will be lost. If this option is not present, the index file will be opened for update. The INVOKE directive may only be used when the I option is used.
L	The “L” option will result in the listing of all source input images as they are read, and will cause the entire data item index file contents to be listed upon completion of processing of the source input.
N	Option “N” is used to produce a list of the contents of the index file with no update and no directives or source input. If this option is used, all other options are ignored.
S	Option “S” will cause the listing of all source input images as they are read.
T	For KMSystems debugging only. Use only if directed to by KMSystems’ personnel.
O	The “O” option will cause the processor to overwrite any duplicate definitions encountered. The definition of the last duplicated item name will be used. If this option is not used, a warning message will be displayed for each duplicated item name encountered, and the first definition will be used.
W	Option “W” causes QINDEX to produce a data item list, replacing the “RDA REF” portion of the listing with starting word, starting bit and bit length of each item. The starting word is relative to zero, that is, for the first word in a record/file, starting word = 00000. The starting bit begins at bit one of the word and proceeds from left to right. If the “W” option is not specified, the default RDA reference (starting character, number of characters) will be displayed.

A.3.2 QINDEX Directives

There are several QINDEX directives that control the creation or update of the Data Item Index File. The following describes the function of each.

The QINDEX directives are:

```
INVOKE
S$PROC
RECORD
FILE
```

A.3.2.1 INVOKE

The INVOKE directive for QINDEX works in a similar manner to the INVOKE in the Q-LINK processor in that it accesses the object schema and subschema to build the Data Item Index file. The difference is that no initialization of D$WORK and S$WORK is involved.

The INVOKE will create a data item index file record for each area, record, set, database dataname and data item included in the specified subschema. The data item index file will not include redefinitions of data items in the subschema, because redefinitions are not included in the subschema or schema object. Redefinitions are only present as source images in the S$PROC element for the subschema which is copied into COBOL programs by the ADMLP Processor prior to program compilation (see the S$PROC directive).

The INVOKE may only be used when initializing (“I” option) a new Data Item Index file, and must be the first directive read.

Format:

$$\mathit{subschema-name} \begin{bmatrix} \\ \end{bmatrix} \mathit{schema-name} \begin{Bmatrix} \mathit{filename} \\ \mathit{file-code} \end{Bmatrix};$$

$$[[\quad]\,\mathit{invoke-key}\,]$$

The *filename* or *file-code* must contain both the *subschema* and *schema* object elements.

Example:

```
INVOKE SUB-REQMTS IN MFG-SCHEMA FILE DMS*SCHEMAFILE
```

A.3.2.2 S$PROC

The purpose of the S$PROC directive is to apply data item redefinitions to an initial Data Item Index file created by the INVOKE directive. The S$PROC element contains the COBOL source for the subschema as generated by the SDDL process, including any item redefines statements. INDEX will match the S$PROC source to the records built by the invoke to apply redefinitions to the appropriate records. The S$PROC directive is optional, and must immediately follow the INVOKE directive.

Format:

    [*filename*]

The *filename* is optional, and only needs to be specified if the S$PROC element for the subschema named on the INVOKE is in a different file, or if the INVOKE specified a TIP schema file. If omitted, QINDEX will used the same file name used in the INVOKE. S$PROC can not be processed from a TIP schema file.

A.3.2.3 RECORD

The RECORD directive is used to add additional item definitions to an existing DMS1100 record. In this manner, a record can be redefined in any way desired beyond any definitions that exist in either the schema or subschema. This redefinition may be required if an application uses a generic record definition in the schema with the record defined in many different formats within various application programs. Definitions added via the RECORD directive will be tied directly to the specified record. All RECORD directives must follow the INVOKE and S$PROC directives, if present.

Format:

subschema – record – name
Followed by COBOL description source

The *subschema-record-name* must currently exist in the data item index file. It is placed in the data item index file by the INVOKE when the file is initialized.

If the subschema specifies only selected items of the record via an ITEMS ARE clause, care must be taken to ensure that added definitions match the mapped form of the record. The mapped form is the form containing only those items included in the ITEMS ARE clause.

Examples:

```
RECORD PART-MSTR
        01  PART-MSTR.
       *** THE FOLLOWING DEFINES THE FIRST 80 POSITIONS
       *** OF THE PART MASTER RECORD FOR USE AS A
       *** HEADER (THIS IS NOT DEFINED IN THE SCHEMA).
           05 PART-MSTR-HDR      PIC X(80).
           05 FILLER             PIC X(102).
```

The above definitions will be applied to the PART-MSTR record defined in the currently invoked subschema.

A.3.2.4 FILE

The file directive allows the addition of non-DMS1100 file records. These definitions are not created by the INVOKE, S$PROC or RECORD directives. When a file definition is included in a data item index file, the Q-LINK processor will be able to address file data items by name, and automatically determine the RDA location of the record.

When a file is defined in QINDEX, it is assigned an internal file code beginning with 4097. Each subsequent file is assigned the next higher number. This range is used to distinguish non-DMS1100 files from DMS1100 records (the highest possible record code is 4095). These codes are used in the Q-LINK processor to automatically determine the records RDA position when items within records are referenced. In order for this feature to function correctly in Q-LINK, the INDEX directive must be processed before all file definitions (DEFINE Fs).

Format:

filename [ ]
Followed by COBOL description source

The *filename* must be the same name used when the file is defined in Q-LINK with the DEFINE F directive.

Example:

```
FILE PART-HIST
        01  PART-HIST.
            05 PH-PART-NUMBER      PIC 9(5).
            05 PH-PART-NAME        PIC X(50).
            05 PH-PRICE            PIC 9(5)V999.
            05 PH-PRICE-DATE       PIC 9(6).
```

The file definition may contain REDEFINES and additional 01 levels as needed. Once a file has been defined in this manner, RDA item referencing and automatic record area offset operate the same as for DMS1100 records.

QINDEX can be instructed to treat the COBOL PIC clause as if it had been defined for an IBM 360/370 environment: i.e., allocate/align each data item according to the rules required for the EBCDIC character set by adding the “IBM” option to the QINDEX FILE directive.

The allocation rules are defined below:

Data Type	UNISYS 1100/2200 Allocation	IBM 360/370 Allocation
DISPLAY	9 bits per character (quarter word aligned)	9 bits per character (quarter word aligned)
COMP	1-2 digits, 9 bits 3-5 digits, 18 bits 6-7 digits, 27 bits 8-10 digits, 36 bits 11-13 digits, 45 bits 14-15 digits, 54 bits 16-18 digits, 63 bits (quarter word aligned)	1-4 digits, 18 bits 5-9 digits, 36 bits 10-18 digits, 72 bits (quarter word aligned)
COMP-1	36 bits (word aligned)	36 bits (quarter word aligned)
COMP-2	72 bits (word aligned)	72 bits (quarter word aligned)
COMP-3	9 bits per digit (quarter word aligned)	1 digit 9 bits 2-3 digits 18 bits 4-5 digits 27 bits 6-7 digits 36 bits 8-9 digits 45 bits 10-11 digits 54 bits 12-13 digits 63 bits 14-15 digits 72 bits 16-17 digits 81 bits 18 digits 90 bits (quarter word aligned)

A.4 Building an Application Definition

The following example builds a new application Data Item Index file from the base subschema and schema, applies redefinitions from the S$PROC element, adds more redefinitions to data base records and finally adds PCIOS file definitions. The Data Item Index file built may then be considered an entire application data definition to be used in Q-LINK applications.

```
@ASG,UP MFG*MFGQINDEX.
@QINDEX,IL MFG*MFGQINDEX.
INVOKE REQMTS-SUB IN MFG-SCHEMA FILE DMS*SCHEMAFILE
S$PROC
RECORD PART-MSTR
        01  PART-MSTR.
       *** THE FOLLOWING DEFINES THE FIRST 80
       *** POSITIONS OF THE PART MASTER RECORD FOR
       *** USE AS A HEADER (NOT DEFINED IN SCHEMA).
            05 PART-MSTR-HDR      PIC X(80).
            05 FILLER             PIC X(102).
FILE DLY-TRANS
        01  DLY-TRANS-REC.
            05 DT-ACCOUNT-NUM     PIC 9(10).
            05 DT-ACCOUNT-ALP REDEFINES DT-ACCOUNT-NUM.
               10 DT-DIVISION     PIC 9(3).
               10 DT-REGION       PIC 9(3).
               10 DT-SERIAL       PIC 9(5).
            05 DT-CTGY-CODE       PIC X.
            05 DT-TRANS-CODE      PIC X.
            05 DT-AMOUNT          PIC S9(5)V99 COMP.
@EOF
```

The proper sequence for using the sample data item index in a Q-LINK application program is as follows:

```
INVOKE REQMTS-SUB IN MFG-SCHEMA FOR MT NX
INDEX MFG*MFGQINDEX.
DEFINE F DLY-TRANS SEQ 16,200
DEFINE RA DLY-TRANS AFTER PART-MSTR
. . .
```

The file name used in the DEFINE F directive must match the file name used when the file was defined in QINDEX in order for Q-LINK to automatically detect record area offsets.

Once the application has been defined as shown here, any data item in any DMS1100 record or non-DMS file may be referenced without qualification. The referencing of

DMS1100 records or non-DMS files without qualification is possible because each item definition has been linked to its DMS1100 record or file name internally by QINDEX. When an item is referenced in Q-LINK, the command editor automatically looks up the item's record or file entry to determine if the record or file has been relocated within the RDA via a DEFINE RA directive. If an item name appears within more than one record or file within the application, it will be necessary to qualify its reference.

A.5 A QINDEX Execution

The following example shows the executions of QINDEX to build the data item index file used in the examples shown throughout this guide. The file contains both DMS and PCIOS data item definitions. In the execution, a FILE directive is supplied so that the Q-LINK DEFINE RA (record area) directive can be used in the generated DBM code for InfoQuest.

```
@DELETE,C MKTG*CO-INDEX.
FURPUR  30R1C        (891109 1204:28) 1990 Aug 29 Wed 1104:14
@ASG,UP   MKTG*CO-INDEX.
I:002333 ASG complete.
@QINDXK,LI MKTG*CO-INDEX.
QINDEX QLINK 5R1 (Release 5R1) (900820 1439:57) 1990 Aug 29 Wed 1104:16
(C) Copyright 1985-1990 by KMSystems, Inc.  All Rights reserved.
This program licensed for use by KMSYSTEMS, INC.
File:MKTG*        CO-INDEX.     already assigned
***Index will be INITIALIZED.
(00001)INVOKE DEMOSUB IN DEMOSCH FILE uds$$src*schabs
(00002)RECORD CUST-ADDR-REC
(00003)      *** THE FOLLOWING REDEFINES TO CA-LOCKEY FIELD ***
(00004)       01  CUST-ADDR-REC-RE-DEF.
(00005)           05 CA-LOCKEY-REDEF.
(00006)               10 CA-STATE           PIC XX.
(00007)               10 CA-CITY            PIC X(18).
(00008)               10 CA-ACCOUNT         PIC 9(9).
(00009)FILE ORDERFILE
(00010)       01  ORDER-RECORD.
(00011)      ** Secondary key, duplicates allowed **
(00012)           05  OR-CUSTKEY.
(00013)               10  OR-CUSTDIV           PIC 9.
(00014)               10  OR-CUSTNUM           PIC X(5).
(00015)               10  OR-CUSTSHIPTO        PIC 999.
(00016)      ** Primary key **
(00017)           05  OR-ORDER-IDENT.
(00018)               10  OR-ORDER-LOC         PIC 99.
(00019)               10  OR-ORDER-KEY         PIC X(7).
(00020)               10  OR-SHIP-LOC          PIC 99.
(00021)           05  OR-ORDER-TYPE-CODE       PIC X.
(00022)           05  OR-PRODLINE-CODE         PIC X.
(00023)           05  OR-ENTRY-DATE.
(00024)               10  OR-ENTRY-MO          PIC XX.
(00025)               10  OR-ENTRY-DA          PIC XX.
(00026)               10  OR-ENTRY-YR          PIC XX.
(00027)           05  OR-BUYER.
(00028)               10  OR-BYPASS            PIC X.
(00029)               10  FILLER               PIC X(10).
(00030)           05  OR-PURCHASE-ORD          PIC X(8).
(00031)           05  OR-REQ-SHIPDATE          PIC X(6).
```

```
(00032)          05  OR-SHIP-VIA              PIC X(11).
(00033)          05  OR-CREDIT-HOLD           PIC X.
(00034)          05  OR-HOLD-CODE             PIC X.
(00035)          05  OR-INVOICE-CODE          PIC X.
(00036)          05  OR-BOL-PRT-CODE          PIC X.
(00037)          05  OR-PAY-METHOD-CODE       PIC XXX.
(00038)          05  OR-WORKORD-CODE          PIC X.
(00039)          05  OR-SPECIAL-TERMS         PIC X(20).
(00040)          05  OR-TERMS.
(00041)              10  OR-TERM-CODE         PIC X.
(00042)              10  OR-TERM-PER          PIC V9(4) COMP.
(00043)              10  OR-TERM-DATE-DAYS    PIC 9(6).
(00044)          05  OR-DELETE-FLAG           PIC X.
(00045)          05  OR-INPROCESS-HOLD        PIC X.
(00046)          05  FILLER                   PIC XX.
(00047)          05  OR-ACTUAL-SHIP-DATE.
(00048)              10  OR-SHIP-YR           PIC XX.
(00049)              10  OR-SHIP-MO           PIC XX.
(00050)              10  OR-SHIP-DA           PIC XX.
(00051)          05  OR-PIECES                PIC 9(5) COMP.
(00052)          05  FILLER                   PIC XX.
(00053)          05  OR-WEIGHT                PIC 9(7) COMP.
(00054)          05  OR-SHIP-FEE              PIC 9(5)V99 COMP.
(00055)          05  OR-TOT-CHARGES           PIC 9(6)V99 COMP.
(00056)          05  OR-TOT-CLC               PIC 9(6)V99 COMP.
(00057)          05  OR-DISCOUNT              PIC 9(5)V99 COMP.
(00058)          05  OR-CREDIT-REL-DATE.
(00059)              10  OR-CREL-YR           PIC XX.
(00060)              10  OR-CREL-MO           PIC XX.
(00061)              10  OR-CREL-DA           PIC XX.
(00062)          05  OR-WORKORD-PRT-DATE.
(00063)              10  OR-WKORD-PRT-YR      PIC XX.
(00064)              10  OR-WKORD-PRT-MO      PIC XX.
(00065)              10  OR-WKORD-PRT-DA      PIC XX.
(00066)          05  OR-STATE-TAX             PIC S9(4)V99 COMP.
(00067)          05  OR-CITY-TAX              PIC S9(4)V99 COMP.
(00068)          05  OR-COUNTY-TAX            PIC S9(4)V99 COMP.
(00069)          05  OR-CREDIT-USERID         PIC X(8).
(00070)          05  OR-NBR-PALLETS           PIC S9(2)  COMP.
(00071)          05  OR-PALLET-CHG            PIC S9(3)V99  COMP.
(00072)          05  OR-TOT-PALLET-COST       PIC S9(5)V99  COMP.
(00073)          05  OR-AUTHDLR-CODE          PIC X.
(00074)          05  OR-LINE-COUNT            PIC 9(10) COMP.
(00075)          05  OR-ORDER-LINE-DATA OCCURS 1 TO 50
(CONT.)              DEPENDING ON OR-LINE-COUNT.
(00076)      *** Order line item data ***
(00077)              10  OR-PRODUCT           PIC X(6).
(00078)              10  OR-TYPE-ORD-CODE     PIC X.
(00079)              10  OR-QUANTITY          PIC S9(5) COMP.
(00080)              10  OR-UNIT-PRICE        PIC 9(5)V99 COMP.
(00081)              10  OR-DESC              PIC X(25).
(00082)              10  OR-LINE-WEIGHT       PIC 9(5)V99 COMP.
(00083)              10  OR-PACKAGE           PIC X(8).
(00084)              10  OR-PRICE-CODE        PIC XX.
(00085)              10  OR-BOL-KEY           PIC 999 COMP.
(00086)              10  OR-TAX-CODE          PIC X.
(00087)              10  OR-REG-CODE          PIC X.
(00088)              10  OR-SHIP-QTY          PIC 9(5) COMP.
(00089)              10  OR-BILL-ONLY-CODE    PIC X.
(00090)              10  OR-PRICE-CHANGE      PIC X.
(00091)              10  OR-LAST-DATE         PIC X(6).
(00092)              10  OR-PRIORITY          PIC X.
```

```
(00093)               10  OR-EXCEPTION-SW      PIC X.
(00094)               10  OR-SUB-ITEM          PIC X(6).

(00095)FILE CUSTFILE
(00096)       01  CUSTORMER-MASTER-REC.
(00097)      ** Primary key **
(00098)           05  CM-ACCOUNT               PIC X(9).
(00099)      ** Secondary key, duplicates allowed **
(00100)           05  CM-LOCKEY.
(00101)               10   CM-STATE            PIC XX.
(00102)               10   CM-CITY             PIC X(18).
(00103)           05  CM-CUSTNAME              PIC X(33).
(00104)           05  CM-ADDR1                 PIC X(30).
(00105)           05  CM-ADDR2                 PIC X(30).
(00106)           05  CM-ADDR3                 PIC X(30).
(00107)           05  CM-ZIP                   PIC X(9).
(00108)           05  CM-TELEPHONE.
(00109)               10  CM-AREACODE          PIC XXX.
(00110)               10  CM-EXCHANGE          PIC XXX.
(00111)               10  CM-TELNUM            PIC XXXX.
.....End of QINDEX processing.
**************************************************
***         Date Item Index File List          ***
**************************************************
( Subschema names and codes listed )



<<< Schema/Subschema >>>
DEMOSCH     DEMOSUB              SCHEMA/SUBSCHEMA

******* COUNT 0000000001

<<< Areas >>>
DEMO-ADDR                        AREA CODE:  00001
DEMO-CKEY                        AREA CODE:  00003
DEMO-INDX                        AREA CODE:  00002
DEMO-ORD                         AREA CODE:  00004

******* COUNT 0000000004

<<< Records >>>
CUST-ADDR-REC                    RECORD CODE:00002
CUST-KEY-REC                     RECORD CODE:00001
ORDER-COMMENT-REC                RECORD CODE:00004
ORDER-HEADER-REC                 RECORD CODE:00003
ORDER-LINE-REC                   RECORD CODE:00005

******* COUNT 0000000005

<<< Sets >>>
CUST-KEY-ADDR-SET                SET CODE:   00001
CUST-ORD                         SET CODE:   00004
ORDH-CMT                         SET CODE:   00003
ORDH-LINE                        SET CODE:   00002

******* COUNT 0000000004

<<< Database Datanames >>>
AREA-DEMO-CKEY                   DBDN CODE:  00002
AREA-DEMO-ORD                    DBDN CODE:  00003
SH-MISC-ANAME                    DBDN CODE:  00001
```

```
******* COUNT 0000000003

<<< Files >>>
CUSTFILE                         FILE CODE:  04098
ORDERFILE                        FILE CODE:  04097

******* COUNT 0000000002

<<< Data Items >>>

IX|ITEM NAME                      |  RDA REF    |DATA TYPE|DEC P6S|REC/FILE CODE
—|———————————————-|———————-|—————-|———-|———————-
18:CA-ACCOUNT                      (00021,00009) UN9                 00002
17:CA-ADDR1                        (00063,00030) A9                  00002
17:CA-ADDR2                        (00093,00030) A9                  00002
17:CA-ADDR3                        (00123,00030) A9                  00002
17:CA-AREACODE                     (00162,00003) A9                  00002
16:CA-CITY                         (00003,00018)                     00002
17:CA-CUSTNAME                     (00030,00033) A9                  00002
17:CA-EXCHANGE                     (00165,00003) A9                  00002
17:CA-LOCKEY                       (00001,00029) A9                  00002
16:CA-LOCKEY-REDEF                 (00001,00029)                     00002
16:CA-STATE                        (00001,00002)                     00002
17:CA-TELEPHONE                    (00162,00010) A9                  00002
17:CA-TELNUM                       (00168,00004) A9                  00002
17:CA-ZIP                          (00153,00009) A9                  00002
00:CK-CHG-TIME                     (00043,00004) UB9                 00001
17:CK-CUST-KEY                     (00001,00009) A9                  00001
17:CK-NAME                         (00010,00033) A9                  00001
16:CM-ACCOUNT                      (00001,00009)                     04098
16:CM-ADDR1                        (00063,00030)                     04098
16:CM-ADDR2                        (00093,00030)                     04098
16:CM-ADDR3                        (00123,00030)                     04098
16:CM-AREACODE                     (00162,00003)                     04098
16:CM-CITY                         (00012,00018)                     04098
16:CM-CUSTNAME                     (00030,00033)                     04098
16:CM-EXCHANGE                     (00165,00003)                     04098
16:CM-LOCKEY                       (00010,00020)                     04098
16:CM-STATE                        (00010,00002)                     04098
16:CM-TELEPHONE                    (00162,00010)                     04098
16:CM-TELNUM                       (00168,00004)                     04098
16:CM-ZIP                          (00153,00009)                     04098
17:CUST-ADDR-REC                   (00001,00180) A9                  00002
16:CUST-ADDR-REC-RE-DEF            (00001,00029)                     00002
17:CUST-KEY-REC                    (00001,00048) A9                  00001
16:CUSTORMER-MASTER-REC            (00001,00171)                     04098
17:OC-BOL-TYPE                     (00082,00001) A9                  00004
17:OC-COMMENT                      (00001,00080) A9                  00004
17:OC-TYPE                         (00081,00001) A9                  00004
17:OH-ACTUAL-SHIP-DATE             (00106,00006) A9                  00003
17:OH-AUTHDLR-CODE                 (00168,00001) A9                  00003
17:OH-BOL-PRT-CODE                 (00068,00001) A9                  00003
17:OH-BUYER                        (00029,00011) A9                  00003
17:OH-BYPASS                       (00029,00001) A9                  00003
04:OH-CITY-TAX                     (00148,00003) SB9        02       00003
04:OH-COUNTY-TAX                   (00151,00003) SB9        02       00003
17:OH-CREDIT-HOLD                  (00065,00001) A9                  00003
17:OH-CREDIT-REL-DATE              (00133,00006) A9                  00003
17:OH-CREDIT-USERID                (00154,00008) A9                  00003
17:OH-CREL-DA                      (00137,00002) A9                  00003
17:OH-CREL-MO                      (00135,00002) A9                  00003
```

```
17:OH-CREL-YR                      (00133,00002) A9                   00003
18:OH-CUSTDIV                      (00001,00001) UN9                  00003
17:OH-CUSTKEY                      (00001,00009) A9                   00003
17:OH-CUSTNUM                      (00002,00005) A9                   00003
18:OH-CUSTSHIPTO                   (00007,00003) UN9                  00003
17:OH-DELETE-FLAG                  (00102,00001) A9                   00003
03:OH-DISCOUNT                     (00130,00003) UB9        02        00003
17:OH-ENTRY-DA                     (00025,00002) A9                   00003
17:OH-ENTRY-DATE                   (00023,00006) A9                   00003
17:OH-ENTRY-MO                     (00023,00002) A9                   00003
17:OH-ENTRY-YR                     (00027,00002) A9                   00003
17:OH-HOLD-CODE                    (00066,00001) A9                   00003
17:OH-INPROCESS-HOLD               (00103,00001) A9                   00003
17:OH-INVOICE-CODE                 (00067,00001) A9                   00003
01:OH-NBR-PALLETS                  (00162,00001) COMP                 00003
17:OH-ORDER-IDENT                  (00010,00011) A9                   00003
17:OH-ORDER-KEY                    (00012,00007) A9                   00003
18:OH-ORDER-LOC                    (00010,00002) UN9                  00003
17:OH-ORDER-TYPE-CODE              (00021,00001) A9                   00003
04:OH-PALLET-CHG                   (00163,00002) SB9        02        00003
17:OH-PAY-METHOD-CODE              (00069,00003) A9                   00003
00:OH-PIECES                       (00112,00002) UB9                  00003
17:OH-PRODLINE-CODE                (00022,00001) A9                   00003
17:OH-PURCHASE-ORD                 (00040,00008) A9                   00003
17:OH-REQ-SHIPDATE                 (00048,00006) A9                   00003
17:OH-SHIP-DA                      (00110,00002) A9                   00003
03:OH-SHIP-FEE                     (00119,00003) UB9        02        00003
18:OH-SHIP-LOC                     (00019,00002) UN9                  00003
17:OH-SHIP-MO                      (00108,00002) A9                   00003
17:OH-SHIP-VIA                     (00054,00011) A9                   00003
17:OH-SHIP-YR                      (00106,00002) A9                   00003
17:OH-SPECIAL-TERMS                (00073,00020) A9                   00003
04:OH-STATE-TAX                    (00145,00003) SB9        02        00003
17:OH-TERM-CODE                    (00093,00001) A9                   00003
18:OH-TERM-DATE-DAYS               (00096,00006) UN9                  00003
03:OH-TERM-PER                     (00094,00002) UB9        04        00003
17:OH-TERMS                        (00093,00009) A9                   00003
03:OH-TOT-CHARGES                  (00122,00004) UB9        02        00003
03:OH-TOT-CLC                      (00126,00004) UB9        02        00003
04:OH-TOT-PALLET-COST              (00165,00003) SB9        02        00003
00:OH-WEIGHT                       (00116,00003) UB9                  00003
17:OH-WKORD-PRT-DA                 (00143,00002) A9                   00003
17:OH-WKORD-PRT-MO                 (00141,00002) A9                   00003
17:OH-WKORD-PRT-YR                 (00139,00002) A9                   00003
17:OH-WORKORD-CODE                 (00072,00001) A9                   00003
17:OH-WORKORD-PRT-DATE             (00139,00006) A9                   00003
17:OL-BILL-ONLY-CODE               (00057,00001) A9                   00005
00:OL-BOL-KEY                      (00051,00002) UB9                  00005
17:OL-DESC                         (00013,00025) A9                   00005
17:OL-EXCEPTION-SW                 (00066,00001) A9                   00005
17:OL-LAST-DATE                    (00059,00006) A9                   00005
17:OL-PACKAGE                      (00041,00008) A9                   00005
17:OL-PRICE-CHANGE                 (00058,00001) A9                   00005
17:OL-PRICE-CODE                   (00049,00002) A9                   00005
17:OL-PRIORITY                     (00065,00001) A9                   00005
17:OL-PRODUCT                      (00001,00006) A9                   00005
01:OL-QUANTITY                     (00008,00002) COMP                 00005
17:OL-REG-CODE                     (00054,00001) A9                   00005
00:OL-SHIP-QTY                     (00055,00002) UB9                  00005
17:OL-SUB-ITEM                     (00067,00006) A9                   00005
17:OL-TAX-CODE                     (00053,00001) A9                   00005
17:OL-TYPE-ORD-CODE                (00007,00001) A9                   00005
```

```
03:OL-UNIT-PRICE                   (00010,00003) UB9         02       00005
03:OL-WEIGHT                       (00038,00003) UB9         02       00005
16:OR-ACTUAL-SHIP-DATE             (00106,00006)                      04097
16:OR-AUTHDLR-CODE                 (00168,00001)                      04097
16:OR-BILL-ONLY-CODE               (00229,00001)                      04097
00:OR-BOL-KEY                      (00223,00002) UB9                  04097
16:OR-BOL-PRT-CODE                 (00068,00001)                      04097
16:OR-BUYER                        (00029,00011)                      04097
16:OR-BYPASS                       (00029,00001)                      04097
01:OR-CITY-TAX                     (00148,00003) COMP        02       04097
01:OR-COUNTY-TAX                   (00151,00003) COMP        02       04097
16:OR-CREDIT-HOLD                  (00065,00001)                      04097
16:OR-CREDIT-REL-DATE              (00133,00006)                      04097
16:OR-CREDIT-USERID                (00154,00008)                      04097
16:OR-CREL-DA                      (00137,00002)                      04097
16:OR-CREL-MO                      (00135,00002)                      04097
16:OR-CREL-YR                      (00133,00002)                      04097
18:OR-CUSTDIV                      (00001,00001) UN9                  04097
16:OR-CUSTKEY                      (00001,00009)                      04097
16:OR-CUSTNUM                      (00002,00005)                      04097
18:OR-CUSTSHIPTO                   (00007,00003) UN9                  04097
16:OR-DELETE-FLAG                  (00102,00001)                      04097
16:OR-DESC                         (00185,00025)                      04097
00:OR-DISCOUNT                     (00130,00003) UB9         02       04097
16:OR-ENTRY-DA                     (00025,00002)                      04097
16:OR-ENTRY-DATE                   (00023,00006)                      04097
16:OR-ENTRY-MO                     (00023,00002)                      04097
16:OR-ENTRY-YR                     (00027,00002)                      04097
16:OR-EXCEPTION-SW                 (00238,00001)                      04097
16:OR-HOLD-CODE                    (00066,00001)                      04097
16:OR-INPROCESS-HOLD               (00103,00001)                      04097
16:OR-INVOICE-CODE                 (00067,00001)                      04097
16:OR-LAST-DATE                    (00231,00006)                      04097
00:OR-LINE-COUNT                   (00169,00004) UB9                  04097
00:OR-LINE-WEIGHT                  (00210,00003) UB9         02       04097
01:OR-NBR-PALLETS                  (00162,00001) COMP                 04097
16:OR-ORDER-IDENT                  (00010,00011)                      04097
16:OR-ORDER-KEY                    (00012,00007)                      04097
16:OR-ORDER-LINE-DATA              (00173,00072)                      04097
18:OR-ORDER-LOC                    (00010,00002) UN9                  04097
16:OR-ORDER-TYPE-CODE              (00021,00001)                      04097
16:OR-PACKAGE                      (00213,00008)                      04097
01:OR-PALLET-CHG                   (00163,00002) COMP        02       04097
16:OR-PAY-METHOD-CODE              (00069,00003)                      04097
00:OR-PIECES                       (00112,00002) UB9                  04097
16:OR-PRICE-CHANGE                 (00230,00001)                      04097
16:OR-PRICE-CODE                   (00221,00002)                      04097
16:OR-PRIORITY                     (00237,00001)                      04097
16:OR-PRODLINE-CODE                (00022,00001)                      04097
16:OR-PRODUCT                      (00173,00006)                      04097
16:OR-PURCHASE-ORD                 (00040,00008)                      04097
01:OR-QUANTITY                     (00180,00002) COMP                 04097
16:OR-REG-CODE                     (00226,00001)                      04097
16:OR-REQ-SHIPDATE                 (00048,00006)                      04097
16:OR-SHIP-DA                      (00110,00002)                      04097
00:OR-SHIP-FEE                     (00119,00003) UB9         02       04097
18:OR-SHIP-LOC                     (00019,00002) UN9                  04097
16:OR-SHIP-MO                      (00108,00002)                      04097
00:OR-SHIP-QTY                     (00227,00002) UB9                  04097
16:OR-SHIP-VIA                     (00054,00011)                      04097
16:OR-SHIP-YR                      (00106,00002)                      04097
16:OR-SPECIAL-TERMS                (00073,00020)                      04097
```

```
01:OR-STATE-TAX                    (00145,00003) COMP        02       04097
16:OR-SUB-ITEM                     (00239,00006)                      04097
16:OR-TAX-CODE                     (00225,00001)                      04097
16:OR-TERM-CODE                    (00093,00001)                      04097
18:OR-TERM-DATE-DAYS               (00096,00006) UN9                  04097
00:OR-TERM-PER                     (00094,00002) UB9         04       04097
16:OR-TERMS                        (00093,00009)                      04097
00:OR-TOT-CHARGES                  (00122,00004) UB9         02       04097
00:OR-TOT-CLC                      (00126,00004) UB9         02       04097
01:OR-TOT-PALLET-COST              (00165,00003) COMP        02       04097
16:OR-TYPE-ORD-CODE                (00179,00001)                      04097
00:OR-UNIT-PRICE                   (00182,00003) UB9         02       04097
00:OR-WEIGHT                       (00116,00003) UB9                  04097
16:OR-WKORD-PRT-DA                 (00143,00002)                      04097
16:OR-WKORD-PRT-MO                 (00141,00002)                      04097
16:OR-WKORD-PRT-YR                 (00139,00002)                      04097
16:OR-WORKORD-CODE                 (00072,00001)                      04097
16:OR-WORKORD-PRT-DATE             (00139,00006)                      04097
17:ORDER-COMMENT-REC               (00001,00084) A9                   00004
17:ORDER-HEADER-REC                (00001,00172) A9                   00003
17:ORDER-LINE-REC                  (00001,00072) A9                   00005
16:ORDER-RECORD                    (00001,03772)                      04097

******* COUNT 0000000195
**** End of Data Item Index List ****
```

A QINDEX Execution

Appendix B DEMO Schema

```
       IDENTIFICATION DIVISION
       SCHEMA NAME IS DEMOSCHEMA IN FILE SCHABS
       DATA DIVISION
       DATA NAME SECTION
       77  SH-MISC-ANAME   USAGE IS AREA-NAME CODE 1
       77  AREA-DEMO-CKEY  USAGE IS AREA-NAME CODE 2

AREA SECTION
       AREA CONTROL IS 600 AREAS

AREA NAME IS DEMO-ADDR
           AREA CODE IS 1
           MODE IS DATA AREA
           ALLOCATE 50 PAGES
               10 OVERFLOW PAGES AT END
               EXPANDABLE TO 99999 PAGES
           PAGES ARE 896 WORDS

AREA NAME IS DEMO-INDX
           AREA CODE IS 2
           MODE IS INDEX AREA
           ALLOCATE 5 PAGES
               EXPANDABLE TO 99999 PAGES
           PAGES ARE 896 WORDS

AREA NAME IS DEMO-CKEY
           AREA CODE IS 3
           MODE IS DATA AREA
           ALLOCATE 50 PAGES
               5 OVERFLOW PAGES AT END
               EXPANDABLE TO 99999 PAGES
           PAGES ARE 896 WORDS

AREA NAME IS DEMO-ORD
           AREA CODE IS 4
           MODE IS DATA AREA
           ALLOCATE 500 PAGES
               50 OVERFLOW PAGES AT END
               EXPANDABLE TO 99999 PAGES
           PAGES ARE 896 WORDS

RECORD SECTION

RECORD NAME IS CUST-KEY-REC
           RECORD CODE IS 2
           LOCATION MODE IS CALC DMSCALC
               IN AREA-DEMO-CKEY
               USING CK-CUST-KEY
               DUPLICATES NOT ALLOWED
           WITHIN DEMO-CKEY
           RECORD MODE IS ASCII COBOL
           05  CK-CUST-KEY                        PIC X(9)
           05  CK-NAME                            PIC X(33)
           05  CK-CHG-TIME                        PIC 9(8) USAGE COMP
           05  FILLER                             PIC XX

RECORD NAME IS CUST-ADDR-REC
           RECORD CODE IS 4
           LOCATION MODE IS INDEX SEQUENTIAL
               USING ASCENDING RANGE KEY CA-LOCKEY
               INDEX AREA IS DEMO-INDX
               LINKS ARE NEXT
               DUPLICATES ARE NOT ALLOWED
           WITHIN DEMO-ADDR
           RECORD MODE IS ASCII COBOL
           05  CA-LOCKEY                         PIC X(29)
           05  CA-CUSTNAME              PIC X(33)
           05  CA-ADDR1                 PIC X(30)
           05  CA-ADDR2                 PIC X(30)
           05  CA-ADDR3                 PIC X(30)
           05  CA-ZIP                   PIC X(9)
           05  CA-TELEPHONE
               10  CA-AREACODE          PIC XXX
               10  CA-EXCHANGE          PIC XXX
               10  CA-TELNUM            PIC XXXX
           05  FILLER                    PIC X(9)

RECORD NAME IS ORDER-HEADER-REC
           RECORD CODE IS 11
           LOCATION MODE IS CALC DMSCALC
               IN AREA-DEMO-ORD
               USING OH-ORDER-KEY DUPLICATES NOT ALLOWED
           WITHIN DEMO-ORD
           RESERVE 1 POINTERS
           RECORD MODE IS ASCII COBOL
           05  OH-CUSTKEY
               10  OH-CUSTDIV           PIC 9
               10  OH-CUSTNUM           PIC X(5)
               10  OH-CUSTSHIPTO        PIC 999
           05  OH-ORDER-IDENT
               10  OH-ORDER-LOC         PIC 99
               10  OH-ORDER-KEY          PIC X(7)
               10  OH-SHIP-LOC          PIC 99
           05  OH-ORDER-TYPE-CODE       PIC X
           05  OH-PRODLINE-CODE         PIC X
           05  OH-ENTRY-DATE
               10  OH-ENTRY-MO          PIC XX
               10  OH-ENTRY-DA          PIC XX
               10  OH-ENTRY-YR          PIC XX
           05  OH-BUYER
               10  OH-BYPASS            PIC X
               10  FILLER               PIC X(10)
           05  OH-PURCHASE-ORD          PIC X(8)
           05  OH-REQ-SHIPDATE          PIC X(6)
           05  OH-SHIP-VIA              PIC X(11)
           05  OH-CREDIT-HOLD           PIC X
           05  OH-HOLD-CODE             PIC X
           05  OH-INVOICE-CODE          PIC X
           05  OH-BOL-PRT-CODE          PIC X
           05  OH-PAY-METHOD-CODE       PIC XXX
           05  OH-WORKORD-CODE          PIC X
           05  OH-SPECIAL-TERMS         PIC X(20)
```

```
      ****     IF SPACES, PRINT TERMS CALCULATED
           05  OH-TERMS
               10  OH-TERM-CODE         PIC X
               10  OH-TERM-PER          PIC V9(4) USAGE COMP
               10  OH-TERM-DATE-DAYS    PIC 9(6)
           05  OH-DELETE-FLAG           PIC X
           05  OH-INPROCESS-HOLD        PIC X
           05  FILLER                   PIC XX
           05  OH-ACTUAL-SHIP-DATE
               10  OH-SHIP-YR           PIC XX
               10  OH-SHIP-MO           PIC XX
               10  OH-SHIP-DA           PIC XX
           05  OH-PIECES                PIC 9(5) USAGE COMP
           05  FILLER                   PIC XX
           05  OH-WEIGHT                PIC 9(7) USAGE COMP
           05  OH-SHIP-FEE              PIC 9(5)V99 USAGE COMP
           05  OH-TOT-CHARGES           PIC 9(6)V99 USAGE COMP
           05  OH-TOT-CLC               PIC 9(6)V99 USAGE COMP
           05  OH-DISCOUNT              PIC 9(5)V99 USAGE COMP
           05  OH-CREDIT-REL-DATE
               10  OH-CREL-YR           PIC XX
               10  OH-CREL-MO           PIC XX
               10  OH-CREL-DA           PIC XX
           05  OH-WORKORD-PRT-DATE
               10  OH-WKORD-PRT-YR      PIC XX
               10  OH-WKORD-PRT-MO      PIC XX
               10  OH-WKORD-PRT-DA      PIC XX
           05  OH-STATE-TAX             PIC S9(4)V99 USAGE COMP
           05  OH-CITY-TAX              PIC S9(4)V99 USAGE COMP
           05  OH-COUNTY-TAX            PIC S9(4)V99 USAGE COMP
           05  OH-CREDIT-USERID         PIC X(8)
           05  OH-NBR-PALLETS           PIC S9(2)  USAGE COMP
           05  OH-PALLET-CHG            PIC S9(3)V99  USAGE COMP
           05  OH-TOT-PALLET-COST       PIC S9(5)V99  USAGE COMP
           05  OH-AUTHDLR-CODE          PIC X
           05  FILLER                   PIC X(4)

RECORD NAME IS ORDER-COMMENT-REC
           RECORD CODE IS 13
           LOCATION MODE IS VIA ORDH-CMT SET
           WITHIN DEMO-ORD
           RECORD MODE IS ASCII COBOL
           05  OC-COMMENT               PIC X(80)
           05  OC-TYPE                  PIC X
           05  OC-BOL-TYPE              PIC X
           05  FILLER                   PIC X(2)

RECORD NAME IS ORDER-LINE-REC
           RECORD CODE IS 14
           LOCATION MODE IS VIA ORDH-LINE SET
           WITHIN DEMO-ORD
           RECORD MODE IS ASCII COBOL
           05  OL-PRODUCT               PIC X(6)
           05  OL-TYPE-ORD-CODE         PIC X
           05  OL-QUANTITY              PIC S9(5) USAGE COMP
           05  OL-UNIT-PRICE            PIC 9(5)V99 USAGE COMP
           05  OL-DESC                  PIC X(25)
           05  OL-WEIGHT                PIC 9(5)V99 USAGE COMP
           05  OL-PACKAGE               PIC X(8)
           05  OL-PRICE-CODE            PIC XX
           05  OL-BOL-KEY               PIC 999 USAGE COMP
           05  OL-TAX-CODE              PIC X
           05  OL-REG-CODE              PIC X
```

```
          05  OL-SHIP-QTY              PIC 9(5) USAGE COMP
          05  OL-BILL-ONLY-CODE        PIC X
          05  OL-PRICE-CHANGE          PIC X
          05  OL-LAST-DATE             PIC X(6)
          05  OL-PRIORITY              PIC X
          05  OL-EXCEPTION-SW        PIC X
          05  OL-SUB-ITEM            PIC X(6)
      SET SECTION

 SET NAME IS CUST-KEY-ADDR-SET
          SET CODE IS 5
          MODE IS CHAIN
          ORDER IS NEXT
          OWNER IS CUST-KEY-REC
          MEMBER IS CUST-ADDR-REC AUTOMATIC
          SET SELECTION THRU CURRENT OF SET

 SET NAME IS ORDH-LINE
          SET CODE IS 7
          MODE IS CHAIN LINKED PRIOR
          ORDER IS SORTED
              DUPLICATES ARE LAST
          OWNER IS ORDER-HEADER-REC
          MEMBER IS ORDER-LINE-REC AUTOMATIC
              ASCENDING KEY IS OL-PRODUCT
          SET SELECTION THRU CURRENT OF SET

 SET NAME IS ORDH-CMT
          SET CODE IS 8
          MODE IS CHAIN
          ORDER IS NEXT
          OWNER IS ORDER-HEADER-REC
                  MEMBER IS ORDER-COMMENT-REC AUTOMATIC
                  SET OCCURRENCE SELECTION IS CURRENT OF SET

 SET NAME IS CUST-ORD
          SET CODE IS 27
          MODE IS CHAIN
          ORDER IS NEXT
          OWNER IS CUST-KEY-REC
                  MEMBER IS ORDER-HEADER-REC AUTOMATIC
                  SET OCCURRENCE SELECTION IS CURRENT OF SET
```

Appendix C DBM Components

C.1 The Generated DBM

The following is a listing of the complete DBM with a description of each of the basic sections of the DBM. The example DBM contains access code generated for a DMS1100 path containing range key processing. The first part of the DBM shows the PATHGEN parameters entered when the DBM was generated (see Section 6.3, ”DMS: Range Key, Owner and Via Set Access”).

```
. ****** AUTOPATH 93/06/14 17:59:48.731
. ****BEGIN PATHGEN parameters
. IDX MKTG*ORDERS-INDX                             . Data item index file name
. SCH UDS$$SRC*SCHABS
. DMR
. AREADEMO-ADDR
. AREADEMO-INDX
. AREADEMO-CKEY
. AREADEMO-ORD
. STRTCUST-ADDR-REC                 DDEMO-ADDR   STATE-CITY
CA-LOCKEY              Y
. REL CUST-KEY-REC                  OCUST-KEY-ADDR-SET
Y
. REL ORDER-HEADER-REC              SCUST-ORD
. REL ORDER-LINE-REC                SORDH-LINE
. ****END PATHGEN parameters
. File:MKTG*ORDERS-INDX                        By:                    F001052
* DBM Number:0017 DBM Name:DMS E3     ***
*
*==============================================================================
=================
. InfoQuest Database Module for DMS E3
. ------- !! IMPORTANT !! -----------
. List of records/files/tables accessed by the DBM:
. Columns 1-8 must be maintained in the format: *RECnnnn
. Where nnnn is the subschema record code or QINDEX
. generated file code.  The record/file code must be
. exactly four digits.  The record/file name must be
. maintained, unaltered, beginning in column 20.
. These lines are critical to the code generation process.
. --------------------------
.
*REC0000
*REC0002           CUST-ADDR-REC
*REC0001           CUST-KEY-REC
*REC0003           ORDER-HEADER-REC
*REC0005           ORDER-LINE-REC

INVOKE DEMOSUB IN DEMOSCHEMA FILE UDS$$SRC*SCHABS NX
INDEX MKTG*ORDERS-INDX

ADD STDDEFS


. Alternate record area definitions...
```

```
DEF RA CUST-ADDR-REC AFTER STDMY
DEF RA CUST-KEY-REC AFTER CUST-ADDR-REC
DEF RA ORDER-HEADER-REC AFTER CUST-KEY-REC
DEF RA ORDER-LINE-REC AFTER ORDER-HEADER-REC

. Record selection switches.  These switches are set
. by the generated selection routines to determine
. whether or not a record has been selected.
DEF N SW-SELECT0000
DEF N SW-SELECT0002
DEF N SW-SELECT0001
DEF N SW-SELECT0003
DEF N SW-SELECT0005

. Record access switches.  These switches are set by
. the generated SETUP routine to indicate which
. records are actually accessed in the request.
DEF N SW-REC0000
DEF N SW-REC0002
DEF N SW-REC0001
DEF N SW-REC0003
DEF N SW-REC0005

. Selection switches.  These switches, also set by the
. SETUP routine, indicate on which records selections
. are to be made in the request.
DEF N SW-SEL0000
DEF N SW-SEL0002
DEF N SW-SEL0001
DEF N SW-SEL0003
DEF N SW-SEL0005

BEGIN
           DO SETUP
           DO LOAD-SEL-TAB
           DO SORT-INIT


. *****************************************************
. ****** Generated Navigation Logic Begins *****
   SET NON-FATAL 6 OFF     . Make ERROR-NUM 6 fatal.
   IMPART
   OPEN DEMO-ADDR
   OPEN DEMO-INDX
   OPEN DEMO-CKEY
   OPEN DEMO-ORD
   CUR-SEL-ITEM = 'STATE-CITY'
   DO FIND-SEL-TAB
   IF STSUB <> 0
     KYSUB = STSUB
     DO WHILE RDA ST-TYPE OF STDMY :KYSUB = 'V'
       RDA CA-LOCKEY OF CUST-ADDR-REC = RDA SEL-LO OF STDMY :KYSUB
       FETCH5 CUST-ADDR-REC
       DO WHILE RDA CA-LOCKEY OF CUST-ADDR-REC <= RDA SEL-HI OF STDMY :KYSUB
       AND ERROR-NUM = 0
       DO SELECT0002
       IF SW-SELECT0002 = 1
         IF SW-REC0003 = 1
           OR SW-REC0005 = 1
           DO ACCESS-0001
         ELSE
           DO SELECTED
           ENDIF
       ENDIF
       FETCH5 NEXT CUST-ADDR-REC
     ENDDO
NOSTRT
     KYSUB = KYSUB + 1
   ENDDO
   ENDIF
   GO FINALE

ACCESS-0001 PROCEDURE
```

```
   IF NOT MEMBER CUST-KEY-ADDR-SET
     ERROR-NUM = 999
   ELSE
     FETCH3 OWNER CUST-KEY-ADDR-SET SET SUPPRESS AREA
   ENDIF
   IF ERROR-NUM <> 0
     BREAK
   ENDIF
     DO ACCESS-0003
   ENDPROC

ACCESS-0003 PROCEDURE
   FETCH4 FIRST ORDER-HEADER-REC CUST-ORD SET SUPPRESS AREA
   DO WHILE ERROR-NUM = 0
     DO SELECT0003
     IF SW-SELECT0003 = 1
         IF SW-REC0005 = 1
           DO ACCESS-0005
         ELSE
           DO SELECTED
           ENDIF
     ENDIF
     FETCH4 NEXT ORDER-HEADER-REC CUST-ORD SET SUPPRESS AREA
   ENDDO
   ENDPROC

ACCESS-0005 PROCEDURE
   FETCH4 FIRST ORDER-LINE-REC ORDH-LINE SET SUPPRESS AREA
   DO WHILE ERROR-NUM = 0
     DO SELECT0005
     IF SW-SELECT0005 = 1
       DO SELECTED
     ENDIF
     FETCH4 NEXT ORDER-LINE-REC ORDH-LINE SET SUPPRESS AREA
   ENDDO
   ENDPROC

. ***** End code insertion:
. ***** INVOKE from file name count = 1
. ***** File definitions created = 0
. ***** Path code lines created = 70
. ****** Generated Path Navigation Logic Ends *****
. ***************************************************

ADD STDCODE
```

C.1.1 File and Record Codes

At the beginning of the DBM skeleton, there is a list of files and records accessed by the DBM. Beginning in column 1 is an eight-digit, record access code entry that will be used by the CODEGEN process to determine which records/files are to be accessed by the DBM. Each entry consists of a four-character prefix, "*REC", followed by a four-digit numeric suffix. The suffix is the subschema record code or the QINDEX generated file code for PCIOS files. For PCIOS files, the suffix will begin with 4097 and be incremented by 1 for each additional PCIOS file included in the view. MAPPER DTM logical record codes begin with 4900, RDMS tables with 5000 and DB4 logical records with 5200. *REC0000 is used to maintain any derived items that the end user might generate during an InfoQuest session.

The record code scheme is extremely important and is used throughout InfoQuest.

C.1.2 INVOKE and INDEX

The INVOKE directive is used to dynamically build S$WORK and D$WORK for the DMS1100 interface. The INDEX directive is used to access the secondary data item index file that will be used to reference data items from DMS1100 as well as non-DMS1100 files.

C.1.3 The Switches

There are three banks of switches that the DBM programmer will find essential when adding logic to DBMs. The first bank of switches (prefixed by “SW-SELECT”) are set whenever data in an accessed record meets the selection criteria specified by the value(s) entered by the end user. A switch of this type is interrogated after calling the corresponding select routine. For example:

```
DO SELECT0005
IF SW-SELECT0005 = 1
  DO SELECTED
ENDIF
```

See the section on CODEGEN for a description of the select routines.

The second bank of switches (prefixed by “SW-REC”) are set when the end user has chosen to use a data item from the record in any way (any reference to a record).

The final bank of switches (prefixed by “SW-SEL”) are set if a data item in the record is to be used for record selection; i.e., a pre-set value or range of values will be used to match against each record occurrence, or user-chosen selection.

Note: The SW-REC and SW-SEL switches are interrogated before the record is accessed to determine whether it is required for data selection (i.e., is this record type needed to produce the report?). On the other hand, the SW-SELECT switches are interrogated after a record occurrence has been accessed to determine if the record meets the search criteria (i.e., is it to be used or bypassed?).

C.1.4 The Base Code

The base code consists of only two routines: BEGIN and FINALE. BEGIN contains the code to initialize the DBM environment. It performs the standard routines SETUP, LOAD-SEL-TAB, SORT-INIT and HEADERS. Following the perform of the HEADERS routine and up to the FINALE routine is the file access code (“Navigation Logic”) created when the DBM was generated. FINALE performs the routines DISP-DETAIL, TOT-CTL and CLOSE-RPT. More information on these routines is in the subsection on CODEGEN below.

C.1.5 STDDEFS and STDCODE

The ADD element, “STDDEFS”, contains standard Q-LINK definitions used by the internal InfoQuest routines that determine data item selection. The DBM programmer will not normally have to reference these definitions. The element, “STDCODE”, contains the code for the standard routines: DISP-DETAIL (returns records from the sort processor) and LOAD-SEL-TAB (loads the selection tables that have been set by the end user during the InfoQuest session). Again, the DBM programmer will not normally have to reference these routines.

C.2 CODEGEN1/CODEGEN2 Operational Information

CODEGEN1 and CODEGEN2 are Q-LINK programs called within InfoQuest to generate all of the request code necessary to produce a completed report when combined with the user-developed InfoQuest Database Module (DBM). The input to CODEGEN1 includes parameters created from selected lines from the user’s view, the calculation parameters for derived data items and the DBM. The output of CODEGEN1 is a complete Q-LINK request program ready for execution. CODEGEN2 executes this generated request program.

The first portion of the program generated by CODEGEN1 will include several variable and record area definitions, followed by the contents of the DBM. The original DBM will be followed by several procedures which will be called from the DBM. The author of the InfoQuest DBM needs to be aware of the names and functions of these definitions and procedures, and how they affect report processing. The following subsections will describe the generated definitions and the procedures that must be called from the DBM.

C.2.1 Generated Definitions

The following is an example of the variable and record area definition generated by CODEGEN1.

```
RUN INFOQBATCH FROM SYS$LIB$*INFOQT-1.
SYSTEM 4     USER4R1     4     710
++START 2200
++PRINT  USER4R1*P93175765390             MAPPERU
DEF A IQ-RPT-NAME ‘ ‘ 60 . Rpt name and parm RID
DEF N LPP 054
DEF N SW-BATCH 1
DEF N SW-LVAL 0
DEF N SELECT-LIMIT 0
DEF N PERCENT 0.0000  . For per centage calculation
DEF N LN 999999999 . Line counter
DEF N PG 0 . Page number counter
DEF N TL 0 . Total lines counter
DEF N SYSNUM$ 4 . System id number
DEF A APP$ ‘MK’ . InfoQuest app code.
DEF A CTL1 33
DEF N GRPFLG1
DEF N TOT80 0.00
```

```
19.  DEF N INIT8 0.
20.  DEF N TOT81 0.00
21.  DEF N TOT82 0.00
22.  DEF N TOT83 0.00
23.  DEF N R01-1 0.00
24.  DEF N R02-1 0.00
     .
     .
25.  DEF RDA INTREC (1,1)
26.  DEF RDA ITEM0 (INTREC,1) UN9
27.  DEF RDA ITEM1 (*ITEM0,33) A9
28.  DEF RDA ITEM2 (*ITEM1,18) A9
29.  DEF RDA ITEM3 (*ITEM2,2) A9
30.  DEF RDA ITEM4 (*ITEM3,9) A9
31.  DEF RDA ITEM5 (*ITEM4,6) A9
32.  DEF RDA ITEM6 (*ITEM5,3) UB9 0.00
33.  DEF RDA ITEM7 (*ITEM6,7) SB9 0.00
34.  DEF RDA ITEM8 (*ITEM7,7) SB9 0.00
35.  DEF RDA ITEM9 (*ITEM8,29) A9
```

Generally, the InfoQuest DBM developer will not use any of the above definitions, but it may be helpful to understand how these definitions are used.

The first four lines of the example are generated to force the request to be processed off-line (if the user has requested off-line processing).

Lines 5 through 24 include total accumulators (TOTnn), control fields (CTLn), calculation intermediate results (Rnn-n) and various variables used in report output control.

Lines 25 through 35 include definitions of the intermediate record into which all data elements required to satisfy the report are collected and combined with redefinitions of the selection table to match (data type and length) the actual definitions of data items used in selections. Intermediate record data element definitions are prefixed by “ITEM”.

C.2.2 Procedures Called by the DBM

C.2.2.1 SETUP

The SETUP procedure MUST be called at the beginning of every InfoQuest DBM to establish the current environment for processing the current request. SETUP will set switch variables to indicate which records within the view must be accessed and which records within the view contain data elements used in selection. SETUP also sets a switch indicating whether or not a sort will be performed. The following is an example of SETUP:

```
SETUP PROCEDURE
       PC BRKPT ‘USER4R1*P93175765390’
       DO GET-REPORT-NAME
       SW-SORT = 1
       SW-REC0000 = 1
       SW-REC0002 = 1
       SW-SEL0002 = 1
       SW-REC0001 = 1
       SW-REC0005 = 1
```

```
        SW-SELECT0000 = 1
        SW-SELECT0001 = 1
        SW-SELECT0003 = 1
        SW-SELECT0005 = 1
        ENDPROC
```

The second and third line above will cause the report to be routed to an alternate print file which will be @SYMed to the MAPPER batch port as configured in the InfoQuest External Configuration Generation (see Section 3.4.1).

C.2.2.2 CLOSE-RPT

The CLOSE-RPT procedure MUST be called at the end of the DBM. CLOSE-RPT will contain the Q-LINK code to handle the routing of a report created using the OFF-LINE run option. If the report is run ON-LINE, the CLOSE-RPT procedure will contain no code. Following are two examples of CLOSE-RPT (one for an on-line report and one for an off-line report):

An on-line report (no code is generated):

```
      CLOSE-RPT PROCEDURE
                ENDPROC
```

An off-line report (report routing code generated):

```
  CLOSE-RPT PROCEDURE
        IF SORT-CNT = 0
          DO HEADERS
          DISPLAY '<<< NO DATA SELECTED >>>'
        ENDIF
        PC CLOSE
        PC BRKPT 'USER4R1*N93175765390'
        D 'BPRUN$ INFOBP,R,710,USER4R1,4,MK,USER4R1,P93175765390,' +
        TD TL ' INFOQ,4,INFOQ 68,INFOQ,I BPERR,954'
        D '. InfoQuest report name:' IQ-RPT-NAME +
        D ' Lines:' TL
        PC CLOSE
        ENDPROC
```

C.2.2.3 SORT-INIT

The SORT-INIT procedure MUST be called at the beginning of the DBM to initialize the sort interface based on the user-selected sort parameters. If no sort was specified, the SORT-INIT procedure will contain no Q-LINK code.

```
  SORT-INIT PROCEDURE
        SORT 116,116 ;
          53,2,A9,A ;
          35,18,A9,A ;
          2,33,A9,A ;
          64,6,A9,A ;
            (40000) . Sort core
        ENDPROC
```

C.2.2.4 SELECTED

The SELECTED procedure must be called at the point at which all records required to fulfill the user's request have been collected into the record delivery area (RDA). SELECTED consolidates all required data elements into an intermediate record. At this point, any derived data item calculations are also performed. If a sort was specified, SELECTED will release the intermediate record to the sort.

Warning: CODEGEN1 rebuilds this procedure every time a request is generated. Therefore, **DO NOT** insert any customized code in this procedure!

```
SELECTED PROCEDURE
       RDA ITEM1 = RDA CK-NAME OF CUST-KEY-REC
       RDA ITEM2 = RDA CA-CITY OF CUST-ADDR-REC
       RDA ITEM3 = RDA CA-STATE OF CUST-ADDR-REC
       RDA ITEM4 = RDA CA-ZIP OF CUST-ADDR-REC
       RDA ITEM5 = RDA OL-PRODUCT OF ORDER-LINE-REC
       RDA ITEM6 = RDA OL-UNIT-PRICE OF ORDER-LINE-REC
       RDA ITEM9 = RDA CA-LOCKEY OF CUST-ADDR-REC
       RDA ITEM7 = RDA ITEM6 * .8
       RDA ITEM8 = RDA ITEM6 - RDA ITEM7
       RELEASE 116
       SORT-CNT = SORT-CNT + 1
       ENDPROC
```

C.2.2.5 HEADERS

The HEADERS procedure MUST be called at the beginning of the DBM to output page and column headings for the report. If the user specifies that headings are to appear at the beginning of each page, the HEADERS procedure will automatically be called for the DISP-DTL procedure.

```
. **** GENERATE REQUEST HEADER/FOOTER CODE
HEADERS PROCEDURE
       IF LN = 0
         BREAK
       ELSE
         LN = 0
       ENDIF
       D 1 '.As of date:' +
       TD MONTH '/' DAY '/' +
       TD YEAR ' Time:' TIME
       ITMLEN = 101
       JUSTTYPE = 1
       RDA ITMHLD = "CALCULATE CUSTOMER ORDERS WITH 8% DISCOUNT"
       D 1 '.' +
       DO JUSTIFY-PRINT
       D
       ITMLEN = 101
       JUSTTYPE = 1
       RDA ITMHLD = "FOR STATES"
       D 1 '.' +
       DO JUSTIFY-PRINT
       D
       ITMLEN = 101
       JUSTTYPE = 1
       RDA ITMHLD = "BEGINNING WITH THE LETTER A"
       D 1 '.' +
       DO JUSTIFY-PRINT
       D
       D '.  '
       D `*                                  .                  .  .         .   `
+
       D `   .        .8       .        .’
       D `*                                  .                  .  .         .   `
+
       D `   .UNIT    .PERCENT .PRICE   .’
       D `*NAME                              .CITY              .ST.ZIP      .PRO’
+
       D ‘DUC.PRICE   .DISCOUNT.DISCOUNT.’
       D `*==================================.==================.==.=========.===’
+
       D ‘===.========.========.========.’
       TL = TL + 10
       ENDPROC
```

C.2.2.6 DISP-DTL

The DISP-DTL procedure MUST be called from the DBM only if no sort has been specified (SW-SORT = 0). The DISP-DTL procedure performs the detail report formatting and accumulation of totals. The generated TOT-CTL procedure is also called from here if total and/or subtotals are to be taken. In the following DISP-DTL procedure, subtotals and a grand total are being taken. The TOT-CTL procedure shown following the DISP-DTL procedure would not be directly accessed by the DBM.

```
DISP-DTL PROCEDURE
       DO TOT-CTL
       IF LN > LPP
         DO HEADERS
       ENDIF
       DISPLAY 1 $TAB +
       IF GRPFLG1 = 0
       DISPLAY 2 RDA ITEM1 +
             GRPFLG1 = 1
       ENDIF
       DISPLAY 35 $TAB +
       DISPLAY 36 RDA ITEM2 +
       DISPLAY 54 $TAB +
       DISPLAY 55 RDA ITEM3 +
       DISPLAY 57 $TAB +
       DISPLAY 58 RDA ITEM4 +
       DISPLAY 67 $TAB +
       DISPLAY 68 RDA ITEM5 +
       DISPLAY 74 $TAB +
       EDIT 75 RDA ITEM6’ZZZZZ.99’ +
       DISPLAY 83 $TAB +
       EDIT 84 RDA ITEM7 * `-ZZZZ.99’ +
       DISPLAY 92 $TAB +
       EDIT 93 RDA ITEM8 * `-ZZZZ.99’ +
       TOT80 = TOT80 + RDA ITEM8
       TOT81 = TOT81 + RDA ITEM8
       TOT82 = TOT82 + RDA ITEM8
       TOT83 = TOT83 + RDA ITEM8
       DISPLAY 101 $TAB
       TL = TL + 1
       ENDPROC
. **** GENERATED TOTAL CONTROL CODE
TOT-CTL PROCEDURE
       IF SW-FIRST = 0
         SW-FIRST = 1
         CTL1 = RDA ITEM1
         BREAK
       ENDIF
```

```
        IF SW-GRAND = 1
          SW-LEVEL = 0
          GO TOT-OUT
        ENDIF
        IF RDA ITEM1 <> CTL1
          SW-LEVEL = 1
          GO TOT-OUT
        ENDIF
        BREAK
.
TOT-OUT
        D 1 '*** Subtotal...'
        TL = TL + 3
        ED 93 TOT81 * '-ZZZZ.99' +
        D 1 '*'
        TOT81 = INIT8
        D '*'
        GRPFLG1 = 0
        CTL1 = RDA ITEM1
        IF SW-LEVEL = 1
          GO TOT-OUT-EXIT
        ENDIF
        D 1 '*** Grand Totals ***'
        TL = TL + 3
        ED 93 TOT80 * '-ZZZZ.99' +
        D 1 '*'
        D '*'
TOT-OUT-EXIT
        ENDPROC
```

C.2.2.7 SELECTnnnn

The SELECTnnnn procedures MUST be called from the DBM at each point where a record has been retrieved which may be involved in data selection. The selection procedures return a status indicating whether or not the record has met the selection criteria. The data selection procedures are named by the record type for which they are used. The name format is SELECTnnnn, where nnnn is the record code from the view. If no selections have been specified for a record type, the selection procedure for that record will set the appropriate selection switch to 1, indicating the record has passed selection tests. For records on which selections have been specified, the generated code will be quite complicated in order to cover the wide range of selection methods possible in InfoQuest. The following is an example of the selection routines generated for both records that do and do not have selections specified.

```
SELECT0000 PROCEDURE
        ENDPROC
SELECT0002 PROCEDURE
        IF RDA CA-LOCKEY OF CUST-ADDR-REC >= 'AA                   '
          AND RDA CA-LOCKEY OF CUST-ADDR-REC <= 'AZ                   '
          GO SELNXT29
        ENDIF
        GO SELFALSE0002
SELNXT29
        SW-SELECT0002 = 1
        BREAK
SELFALSE0002
        SW-SELECT0002 = 0
```

```
         ENDPROC
SELECT0001 PROCEDURE
         ENDPROC
SELECT0003 PROCEDURE
         ENDPROC
SELECT0005 PROCEDURE
         ENDPROC
```

CODEGEN1/CODEGEN2 Operational Information

Appendix D A Completed DBM

The following completed DBM is the result of the third path generation example in Section 6.3, ”DMS: Range Key, Owner and Via Set Access”.

```
RUN INFOQBATCH FROM SYS$LIB$*INFOQT-1.
SYSTEM 4     USER4R1     4     710
++START 2200
++PRINT  USER4R1*P93175765390             MAPPERU
DEF A IQ-RPT-NAME ‘ ‘ 60 . Rpt name and parm RID
DEF N LPP 054
DEF N SW-BATCH 1
DEF N SW-LVAL 0
DEF N SELECT-LIMIT 0
DEF N PERCENT 0.0000  . For per centage calculation
DEF N LN 999999999 . Line counter
DEF N PG 0 . Page number counter
DEF N TL 0 . Total lines counter
DEF N SYSNUM$ 4 . System id number
DEF A APP$ ‘MK’ . InfoQuest app code.
DEF A CTL1 33
DEF N GRPFLG1
DEF N TOT80 0.00
DEF N INIT8 0.
DEF N TOT81 0.00
DEF N TOT82 0.00
DEF N TOT83 0.00
DEF N R01-1 0.00
DEF N R02-1 0.00

. SCH UDS$$SRC*SCHABS
. DMR
. AREADEMO-ADDR
. AREADEMO-INDX
. AREADEMO-CKEY
. AREADEMO-ORD
. STRTCUST-ADDR-REC                 DDEMO-ADDR   STATE-CITY
CA-LOCKEY                  Y
. REL CUST-KEY-REC                  OCUST-KEY-ADDR-SET
Y
. REL ORDER-HEADER-REC              SCUST-ORD
. REL ORDER-LINE-REC                SORDH-LINE
. ****END PATHGEN parameters
. File:MKTG*ORDERS-INDX                          By:                    F001052
* DBM Number:0017 DBM Name:DMS E3     ***
*
*==============================================================================
=====================
. InfoQuest Database Module for DMS E3
. ———————- !! IMPORTANT !! ——————————
. List of records/files/tables accessed by the DBM:
. Columns 1-8 must be maintained in the format: *RECnnnn
. Where nnnn is the subschema record code or QINDEX
. generated file code.  The record/file code must be
. exactly four digits.  The record/file name must be
. maintained, unaltered, beginning in column 20.
```

A Completed DBM

```
. These lines are critical to the code generation process.
. __________________________
. *REC0000
. *REC0002           CUST-ADDR-REC
. *REC0001           CUST-KEY-REC
. *REC0003           ORDER-HEADER-REC
. *REC0005           ORDER-LINE-REC

INVOKE DEMOSUB IN DEMOSCHEMA FILE UDS$$SRC*SCHABS NX
INDEX MKTG*ORDERS-INDX

ADD STDDEFS FROM SYS$LIB$*INFOQT
DEF RDA INTREC (1,1)
DEF RDA ITEM0 (INTREC,1) UN9
DEF RDA ITEM1 (*ITEM0,33) A9
DEF RDA ITEM2 (*ITEM1,18) A9
DEF RDA ITEM3 (*ITEM2,2) A9
DEF RDA ITEM4 (*ITEM3,9) A9
DEF RDA ITEM5 (*ITEM4,6) A9
DEF RDA ITEM6 (*ITEM5,3) UB9 0.00
DEF RDA ITEM7 (*ITEM6,7) SB9 0.00
DEF RDA ITEM8 (*ITEM7,7) SB9 0.00
DEF RDA ITEM9 (*ITEM8,29) A9
DEF RDA ITMHLD (*,256) A9
DEF RDA ITMHCHR (ITMHLD,1) A9
DEF RDA XLATA (*ITMHLD,20) A9
DEF RDA XLATN (XLATA,20) MAPNUM
DEF RDA INTREC-ALL (INTREC,116)


. Alternate record area definitions...
DEF RA CUST-ADDR-REC AFTER STDMY
DEF RA CUST-KEY-REC AFTER CUST-ADDR-REC
DEF RA ORDER-HEADER-REC AFTER CUST-KEY-REC
DEF RA ORDER-LINE-REC AFTER ORDER-HEADER-REC

. Record selection switches.  These switches are set
. by the generated selection routines to determine
. whether or not a record has been selected.
DEF N SW-SELECT0000
DEF N SW-SELECT0002
DEF N SW-SELECT0001
DEF N SW-SELECT0003
DEF N SW-SELECT0005

. Record access switches.  These switches are set by
. the generated SETUP routine to indicate which
. records are actually accessed in the request.
DEF N SW-REC0000
DEF N SW-REC0002
DEF N SW-REC0001
DEF N SW-REC0003
DEF N SW-REC0005

. Selection switches.  These switches, also set by the
. SETUP routine, indicate on which records selections
. are to be made in the request.
DEF N SW-SEL0000
DEF N SW-SEL0002
DEF N SW-SEL0001
DEF N SW-SEL0003
DEF N SW-SEL0005

BEGIN
           DO SETUP
           DO LOAD-SEL-TAB
           DO SORT-INIT


. ***************************************************
. ****** Generated Navigation Logic Begins *****
   SET NON-FATAL 6 OFF     . Make ERROR-NUM 6 fatal.
   IMPART
   OPEN DEMO-ADDR
   OPEN DEMO-INDX
   OPEN DEMO-CKEY
   OPEN DEMO-ORD
   CUR-SEL-ITEM = 'STATE-CITY'
   DO FIND-SEL-TAB
   IF STSUB <> 0
     KYSUB = STSUB
     DO WHILE RDA ST-TYPE OF STDMY :KYSUB = 'V'
       RDA CA-LOCKEY OF CUST-ADDR-REC = RDA SEL-LO OF STDMY :KYSUB
       FETCH5 CUST-ADDR-REC
       DO WHILE RDA CA-LOCKEY OF CUST-ADDR-REC <= RDA SEL-HI OF STDMY :KYSUB
       AND ERROR-NUM = 0
       DO SELECT0002
       IF SW-SELECT0002 = 1
         IF SW-REC0003 = 1
           OR SW-REC0005 = 1
           DO ACCESS-0001
         ELSE
           DO SELECTED
           ENDIF
       ENDIF
       FETCH5 NEXT CUST-ADDR-REC
     ENDDO
NOSTRT
     KYSUB = KYSUB + 1
   ENDDO
   ENDIF
   GO FINALE

ACCESS-0001 PROCEDURE
   IF NOT MEMBER CUST-KEY-ADDR-SET
     ERROR-NUM = 999
   ELSE
     FETCH3 OWNER CUST-KEY-ADDR-SET SET SUPPRESS AREA
   ENDIF
   IF ERROR-NUM <> 0
     BREAK
   ENDIF
     DO ACCESS-0003
   ENDPROC

ACCESS-0003 PROCEDURE
   FETCH4 FIRST ORDER-HEADER-REC CUST-ORD SET SUPPRESS AREA
   DO WHILE ERROR-NUM = 0
     DO SELECT0003
     IF SW-SELECT0003 = 1
         IF SW-REC0005 = 1
           DO ACCESS-0005
         ELSE
           DO SELECTED
           ENDIF
     ENDIF
     FETCH4 NEXT ORDER-HEADER-REC CUST-ORD SET SUPPRESS AREA
   ENDDO
   ENDPROC

ACCESS-0005 PROCEDURE
   FETCH4 FIRST ORDER-LINE-REC ORDH-LINE SET SUPPRESS AREA
   DO WHILE ERROR-NUM = 0
     DO SELECT0005
     IF SW-SELECT0005 = 1
       DO SELECTED
     ENDIF
     FETCH4 NEXT ORDER-LINE-REC ORDH-LINE SET SUPPRESS AREA
   ENDDO
   ENDPROC

. ***** End code insertion:
. ***** INVOKE from file name count = 1
. ***** File definitions created = 0
. ***** Path code lines created = 70
. ****** Generated Path Navigation Logic Ends *****
. *************************************************

ADD STDCODE FROM SYS$LIB$*INFOQT
FINALE
       DEPART
       IF SORT-CNT > 0
         DO
           RETURN AT END BREAK
           DO DISP-DTL
         ENDDO
       ELSE
         DO CLOSE-RPT
         STOP TL
       ENDIF
       SW-GRAND = 1
       SW-FIRST = 1
       DO TOT-CTL
       DO CLOSE-RPT
       STOP TL
. **** GENERATED TOTAL CONTROL CODE
TOT-CTL PROCEDURE
       IF SW-FIRST = 0
         SW-FIRST = 1
         CTL1 = RDA ITEM1
         BREAK
       ENDIF
       IF SW-GRAND = 1
         SW-LEVEL = 0
         GO TOT-OUT
       ENDIF
       IF RDA ITEM1 <> CTL1
         SW-LEVEL = 1
         GO TOT-OUT
       ENDIF
       BREAK
.
TOT-OUT
       D 1 '*** Subtotal...'
       TL = TL + 3
       ED 93 TOT81 * '-ZZZZ.99' +
       D 1 '*'
       TOT81 = INIT8
       D '*'
       GRPFLG1 = 0
       CTL1 = RDA ITEM1
       IF SW-LEVEL = 1
         GO TOT-OUT-EXIT
       ENDIF
       D 1 '*** Grand Totals ***'
       TL = TL + 3
       ED 93 TOT80 * '-ZZZZ.99' +
       D 1 '*'
       D '*'
TOT-OUT-EXIT
       ENDPROC
. **** GENERATE REQUEST HEADER/FOOTER CODE
HEADERS PROCEDURE
       IF LN = 0
         BREAK
       ELSE
         LN = 0
       ENDIF
       D 1 '.As of date:' +
       TD MONTH '/' DAY '/' +
       TD YEAR ' Time:' TIME
       ITMLEN = 101
       JUSTTYPE = 1
       RDA ITMHLD = "CALCULATE CUSTOMER ORDERS WITH 8% DISCOUNT"
       D 1 '.' +
       DO JUSTIFY-PRINT
       D
       ITMLEN = 101
       JUSTTYPE = 1
       RDA ITMHLD = “FOR STATES”
       D 1 ‘.’ +
       DO JUSTIFY-PRINT
       D
       ITMLEN = 101
       JUSTTYPE = 1
       RDA ITMHLD = “BEGINNING WITH THE LETTER A”
       D 1 ‘.’ +
       DO JUSTIFY-PRINT
       D
       D ‘.  ‘
       D ‘*                                 .                  .  .         .   ‘
+
       D ‘   .        .8       .        .’
       D ‘*                                 .                  .  .         .   ‘
+
       D ‘   .UNIT    .PERCENT .PRICE   .’
       D ‘*NAME                             .CITY              .ST.ZIP      .PRO’
+
       D ‘DUC.PRICE   .DISCOUNT.DISCOUNT.’
       D ‘*=================================.==================.==.=========.===’
+
       D ‘===.========.========.========.’
       TL = TL + 10
       ENDPROC
SETUP PROCEDURE
       PC BRKPT ‘USER4R1*P93175765390’
       DO GET-REPORT-NAME
       SW-SORT = 1
       SW-REC0000 = 1
       SW-REC0002 = 1
       SW-SEL0002 = 1
       SW-REC0001 = 1
       SW-REC0005 = 1
       SW-SELECT0000 = 1
       SW-SELECT0001 = 1
       SW-SELECT0003 = 1
       SW-SELECT0005 = 1
       ENDPROC
.
CLOSE-RPT PROCEDURE
       IF SORT-CNT = 0
         DO HEADERS
         DISPLAY ‘<<< NO DATA SELECTED >>>’
       ENDIF
       PC CLOSE
       PC BRKPT ‘USER4R1*N93175765390’
       D ‘BPRUN$ INFOBP,R,710,USER4R1,4,MK,USER4R1,P93175765390,’ +
       TD TL ‘ INFOQ,4,INFOQ 68,INFOQ,I BPERR,954’
       D ‘. InfoQuest report name:’ IQ-RPT-NAME +
       D ‘ Lines:’ TL
       PC CLOSE
       ENDPROC
SORT-INIT PROCEDURE
       SORT 116,116 ;
         53,2,A9,A ;
         35,18,A9,A ;
         2,33,A9,A ;
         64,6,A9,A ;
           (40000) . Sort core
       ENDPROC
SELECTED PROCEDURE
       RDA ITEM1 = RDA CK-NAME OF CUST-KEY-REC
       RDA ITEM2 = RDA CA-CITY OF CUST-ADDR-REC
       RDA ITEM3 = RDA CA-STATE OF CUST-ADDR-REC
       RDA ITEM4 = RDA CA-ZIP OF CUST-ADDR-REC
       RDA ITEM5 = RDA OL-PRODUCT OF ORDER-LINE-REC
       RDA ITEM6 = RDA OL-UNIT-PRICE OF ORDER-LINE-REC
       RDA ITEM9 = RDA CA-LOCKEY OF CUST-ADDR-REC
       RDA ITEM7 = RDA ITEM6 * .8
       RDA ITEM8 = RDA ITEM6 - RDA ITEM7
       RELEASE 116
       SORT-CNT = SORT-CNT + 1
       ENDPROC

DISP-DTL PROCEDURE
       DO TOT-CTL
       IF LN > LPP
         DO HEADERS
       ENDIF
       DISPLAY 1 $TAB +
       IF GRPFLG1 = 0
       DISPLAY 2 RDA ITEM1 +
             GRPFLG1 = 1
       ENDIF
       DISPLAY 35 $TAB +
       DISPLAY 36 RDA ITEM2 +
       DISPLAY 54 $TAB +
       DISPLAY 55 RDA ITEM3 +
       DISPLAY 57 $TAB +
       DISPLAY 58 RDA ITEM4 +
       DISPLAY 67 $TAB +
       DISPLAY 68 RDA ITEM5 +
       DISPLAY 74 $TAB +
       EDIT 75 RDA ITEM6'ZZZZZ.99' +
       DISPLAY 83 $TAB +
       EDIT 84 RDA ITEM7 * '-ZZZZ.99' +
       DISPLAY 92 $TAB +
       EDIT 93 RDA ITEM8 * '-ZZZZ.99' +
       TOT80 = TOT80 + RDA ITEM8
       TOT81 = TOT81 + RDA ITEM8
       TOT82 = TOT82 + RDA ITEM8
       TOT83 = TOT83 + RDA ITEM8
       DISPLAY 101 $TAB
       TL = TL + 1
       ENDPROC
SELECT0000 PROCEDURE
       ENDPROC
SELECT0002 PROCEDURE
       IF RDA CA-LOCKEY OF CUST-ADDR-REC >= 'AA                          `
         AND RDA CA-LOCKEY OF CUST-ADDR-REC <= 'AZ                          `
         GO SELNXT29
       ENDIF
       GO SELFALSE0002
SELNXT29
       SW-SELECT0002 = 1
       BREAK
SELFALSE0002
       SW-SELECT0002 = 0
       ENDPROC
SELECT0001 PROCEDURE
       ENDPROC
SELECT0003 PROCEDURE
       ENDPROC
SELECT0005 PROCEDURE
       ENDPROC
RUN
NCA-LOCKEY                                STATE-CITY
VAA                                       AZ
:
```

Appendix E Customized DBMs

In this appendix, we will discuss the generation of customized path navigational logic that involves manually altering the generated DBM. In such circumstances, mistakes can be made and run-time errors occur. If errors occur, KMSystems provides a run called “IQDEBUG” to assist you in correcting your tailored DBM. Please see the following appendices: Appendix F, “DBM Error Detection”, Appendix I, “QLK External Function Errors”, and Appendix JAppendix K, “Q-LINK Common Bank Errors”.

E.1 Using DMS 1100 Multi-field Keys

In DMS 1100 database applications, it is not uncommon to have a record defined with a key that consists of several parts. An application program would then be required to initialize each of these parts prior to retrieving the record. As stated earlier, InfoQuest generates such a program (DBM) automatically but only allows for a single key entry on the path generation screen.

The following key definition will be used for both examples shown below:

```
05  CK-CUST-KEY.
    10  CK-CUSTDIV           PIC 9(01).
    10  CK-CUSTNUM           PIC X(05).
    10  CK-CUSTSHIPTO        PIC 9(03).
```

E.1.1 Multi-field Key as Start of Path

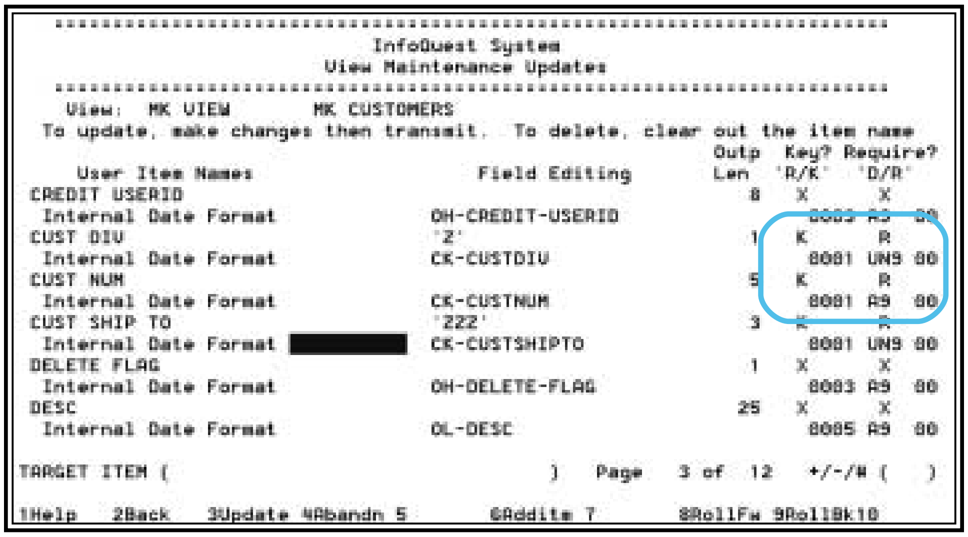
The first example shows how to use a multi-field key as “Start of path” in InfoQuest.

The CK-CUST-KEY field could be flagged in the view as the key field required for selection. Marking the field in this manner would cause the user to enter a value for the entire field, thus requiring the user to know exactly in which column each of the three parts begin.

An alternative to the above method would be to allow the user to enter each part individually, thus eliminating the column restrictions.

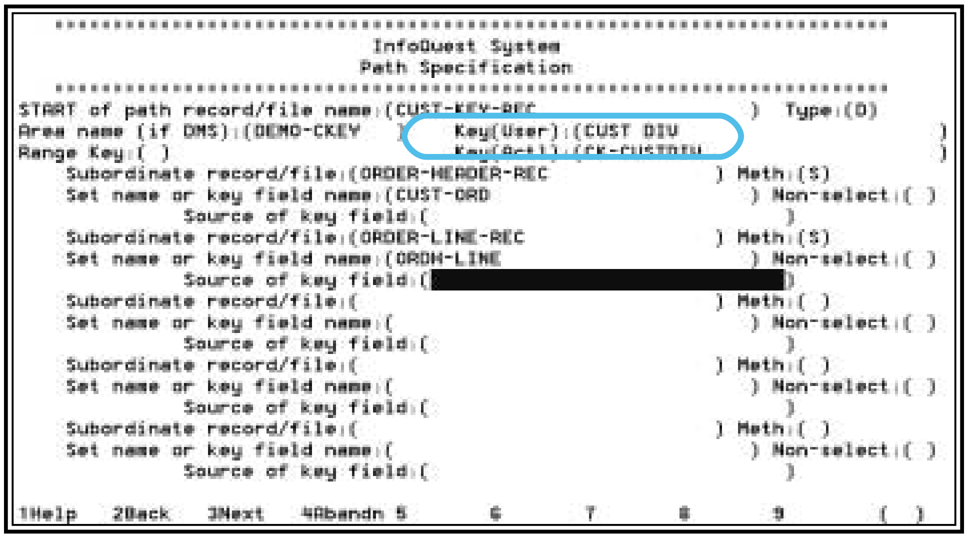
Using DMS 1100 Multi-field Keys

To assure that the user will be prompted for and required to enter a value for each part of the key, it is necessary to flag each as being “A Key Required for Selection”. From the “View Maintenance Updates” screen, mark each item in the view that is a part of the key, with a “K” under the “Key?” column and a corresponding “R” under the the “Require?” column. See the figure below:



```
  ***********************************************************************
                          InfoQuest System
                      View Maintenance Updates
  ***********************************************************************
   View:  MK VIEW       MK CUSTOMERS
  To update, make changes then transmit.  To delete, clear out the item name
                                                          Outp  Key? Require?
    User Item Names                    Field Editing      Len   'R/K'  'D/R'
 CREDIT USERID                                              8    X      X
  Internal Date Format             OH-CREDIT-USERID               8003 A9   00
 CUST DIV                          'Z'                      1    K      R
  Internal Date Format             CK-CUSTDIV                     8001 UN9  00
 CUST NUM                                                   5    K      R
  Internal Date Format             CK-CUSTNUM                     8001 A9   00
 CUST SHIP TO                      'ZZZ'                    3    K      R
  Internal Date Format             CK-CUSTSHIPTO                  8001 UN9  00
 DELETE FLAG                                                1    X      X
  Internal Date Format             OH-DELETE-FLAG                 8003 A9   00
 DESC                                                      25    X      X
  Internal Date Format             OL-DESC                        8005 A9   00

TARGET ITEM (                               )   Page   3 of  12   +/-/# (    )

1Help   2Back   3Update 4Abandn 5       6Additm 7       8RollFw 9RollBk10
```

Once the key fields are properly marked in the view, the DBM can be generated. On the "Path Specification" screen, enter any "User" and "Actl" key pair defined in the view as a key required for selection. In the following example, "CK-CUSTDIV" is the actual key and "CUST DIV" is the corresponding user key:



```
   ****************************************************************
                         InfoQuest System
                        Path Specification
   ****************************************************************
START of path record/file name:(CUST-KEY-REC                 )  Type:(D)
Area name (if DMS):(DEMO-CKEY    )      Key(User):(CUST DIV                  )
Range Key:( )                           Key(Actl):(CK-CUSTDIV                )
    Subordinate record/file:(ORDER-HEADER-REC               ) Meth:(S)
    Set name or key field name:(CUST-ORD                      ) Non-select:( )
              Source of key field:(                          )
    Subordinate record/file:(ORDER-LINE-REC                 ) Meth:(S)
    Set name or key field name:(ORDH-LINE                     ) Non-select:( )
              Source of key field:(                          )
    Subordinate record/file:(                               ) Meth:( )
    Set name or key field name:(                              ) Non-select:( )
              Source of key field:(                          )
    Subordinate record/file:(                               ) Meth:( )
    Set name or key field name:(                              ) Non-select:( )
              Source of key field:(                          )
    Subordinate record/file:(                               ) Meth:( )
    Set name or key field name:(                              ) Non-select:( )
              Source of key field:(                          )

1Help   2Back   3Next   4Abandn 5       6       7       8       9       ( )
```

After the path is created, find where the generated DBM was stored and modify the code to include a search for the remaining portions of the key. The first block of code shown below shows what will be generated, and the second shows what it will look like after the changes have been made.

The DBM may be retrieved/returned by selecting the "DBM Copy Utility" of IQMNT.

The original code for the example (minus the lowercase comments) would appear as follows:

```
CUR-SEL-ITEM = 'CUST DIV'
DO FIND-SEL-TAB
IF STSUB <> 0
  KYSUB = STSUB         .     (we will delete this line)
  . (we will insert 19 lines here)
  DO WHILE RDA ST-TYPE OF STDMY :KYSUB = 'V'
    RDA CK-CUSTDIV OF CUST-KEY-REC = RDA-LO OF STDMY :KYSUB
    . (we will insert 4 lines here)
    FETCH5 CUST-KEY-REC
    IF ERROR-NUM <> 0
      GO NOSTRT
    ENDIF
    DO SELECT0001
    IF SW-SELECT0001 = 1
      IF SW-SELECT0003 = 1
        OR SW-REC0005 = 1
        DO ACCESS-0003
      ELSE
        DO SELECTED
        ENDIF
```

```
         ENDIF
  NOSTRT
        . (we will insert 4 lines here)
        KYSUB = KYSUB + 1   .     (we will delete this line)
      ENDDO
      ELSE                  .     (we will delete this line)
        GO KEY-ERR          .     (we will delete this line)
      ENDIF                 .     (we will delete this line)
      GO FINALE
```

In the above generated code, the select table is only searched for the key selected on the "Path Specification" screen. The code needs to be altered to include a search for the additional two items. The generated program, after the select table is searched, saves the location of the variable in the subscript STSUB, then moves it to the key subscript KYSUB. Since we are using multiple fields, each STSUB value must be saved in a separate variable until all table searches are complete. We used the variables "X", "Y", and "Z" because they are already predefined numeric variables. If you want to use other names or need additional variables, remember to define them first with the DEFINE N directive.

The newly modified code will appear as follows:

```
      CUR-SEL-ITEM = 'CUST DIV'
      DO FIND-SEL-TAB
      IF STSUB <> 0
        x = stsub
      else
        go key-err
      endif
      cur-sel-item = 'CUST NUMBER'
      do find-sel-tab
      if stsub <> 0
        y = stsub
      else
        go key-err
      endif
      cur-sel-item = 'CUST SHIP TO'
      do find-sel-tab
      if stsub <> 0
        z = stsub
      else
        go key-err
      endif
        kysub = x
        DO WHILE RDA ST-TYPE OF STDMY :KYSUB = 'V'
          RDA CK-CUSTDIV OF CUST-KEY-REC = RDA-LO OF STDMY :KYSUB
  kysub = y
          rda ck-custnum of cust-key-rec = rda-lo of stdmy :kysub
          kysub = z
          rda ck-custshipto of cust-key-rec = rda-lo of stdmy :kysub
          FETCH5 CUST-KEY-REC
          IF ERROR-NUM <> 0
            GO NOSTRT
          ENDIF
          DO SELECT0001
          IF SW-SELECT0001 = 1
            IF SW-SELECT0003 = 1
              OR SW-REC0005 = 1
              DO ACCESS-0003
```

```
           ELSE
             DO SELECTED
             ENDIF
         ENDIF
  NOSTRT
       X = X + 1
       Y = Y + 1
       Z = Z + 1
       KYSUB = X
  ENDDO
     GO FINALE
```

Now all the changes are made and the complete DBM is ready for testing. The best way to do this is to create a request in InfoQuest and save the completed program. The saved program can be run through IQDEBUG to locate any possible errors.

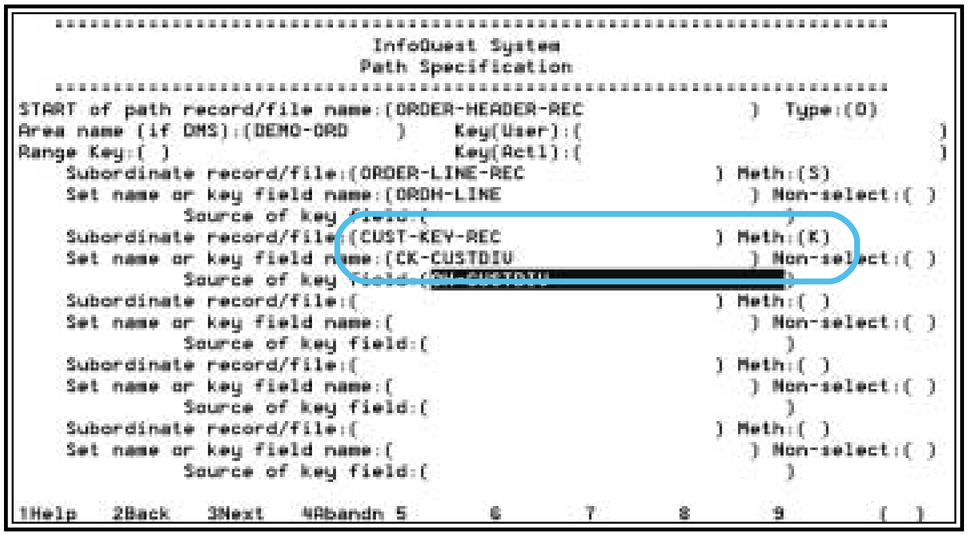
The following multi-field variable table is an example resulting from the creation of a request using the modified DBM above. The "V" entries are the variable values entered by the user:

```
NCK-CUSTDIV                   CUST DIV
V8                                 =
V7                                 =
V4                                 =
V1                                 =
NCK-CUSTNUM                   CUST NUMBER
V23456                             =
V67117                             =
V89710                             =
V23439                             =
NCK-CUSTSHIPTO                CUST SHIP TO
V888                               =
V321                               =
V678                               =
V999                               =
+EOS+
```

E.1.2 Multi-field Key as Subordinate

The other possible setup that may be desired requires only two modifications. This scenario enables the multi-field keyed record to be accessed once the path has been started through another record. In this example, the values for the CK-CUSTDIV and CK-CUSTNUM fields will be obtained from one record, and the value for CK-CUSTSHIPTO from a second record.

In this example, as in the first, only one of the key fields will be entered on the "Path Specification" screen. However, unlike the first example, a "Key field name" for a subordinate record will receive one of the key names. The "Source of key field" parameter will contain the name of the field (in a record earlier on the path) whose value will be used to set the key (in this example we entered the ORDER-HEADER-REC's customer division number "OH-CUST-DIV" field). See the following figure:



```
   ========================================================================
                            InfoQuest System
                           Path Specification
   ========================================================================
START of path record/file name:(ORDER-HEADER-REC              )  Type:(O)
Area name (if DMS):(DEMO-ORD     )     Key(User):(                        )
Range Key:( )                          Key(Actl):(                        )
    Subordinate record/file:(ORDER-LINE-REC                  ) Meth:(S)
    Set name or key field name:(ORDH-LINE                       ) Non-select:( )
              Source of key field:(                               )
    Subordinate record/file:(CUST-KEY-REC                    ) Meth:(K)
    Set name or key field name:(CK-CUSTDIV                      ) Non-select:( )
              Source of key field:(OH-CUSTDIV                     )
    Subordinate record/file:(                                ) Meth:( )
    Set name or key field name:(                                ) Non-select:( )
              Source of key field:(                               )
    Subordinate record/file:(                                ) Meth:( )
    Set name or key field name:(                                ) Non-select:( )
              Source of key field:(                               )
    Subordinate record/file:(                                ) Meth:( )
    Set name or key field name:(                                ) Non-select:( )
              Source of key field:(                               )

1Help   2Back   3Next   4Abandn 5       6       7       8       9       [  ]
```

Next, the RID where the DBM is stored requires a few modifications to the generated code prior to testing. In this example, one line needs to be modified and two lines inserted into the "ACCESS-0001" procedure shown below:

```
ACCESS-0001 PROCEDURE
   RDA CK-CUSTDIV = RDA OH-CUST-DIV  . (we will modify this line)
   . (we will insert 2 lines here)
   FETCH5 CUST-KEY-REC
   IF ERROR-NUM <> 0
     GO TO INVKEY-0001
   ENDIF
     DO SELECT0001
     IF SW-SELECT0001 = 1
       DO SELECTED
     ENDIF
     GO END-0001
INVKEY-0001
```

```
  END-0001
     ENDPROC
```

The following code shows the modifications made to complete the DBM.   Notice that the last part of the key ("ck-custshipto") will be initialized from data in a second record type.

```
  ACCESS-0001 PROCEDURE
     RDA CK-CUSTDIV of cust-key-rec    = RDA OH-CUST-DIV of order-header-rec
     rda ck-custnum of cust-key-rec    = rda oh-cust-num of order-header-rec
     rda ck-custshipto of cust-key-rec = rda oh-cust-shipto of order-line-rec
     FETCH5 CUST-KEY-REC
     IF ERROR-NUM <> 0
       GO TO INVKEY-0001
     ENDIF
       DO SELECT0001
       IF SW-SELECT0001 = 1
         DO SELECTED
       ENDIF
       GO END-0001
  INVKEY-0001
  END-0001
     ENDPROC
```

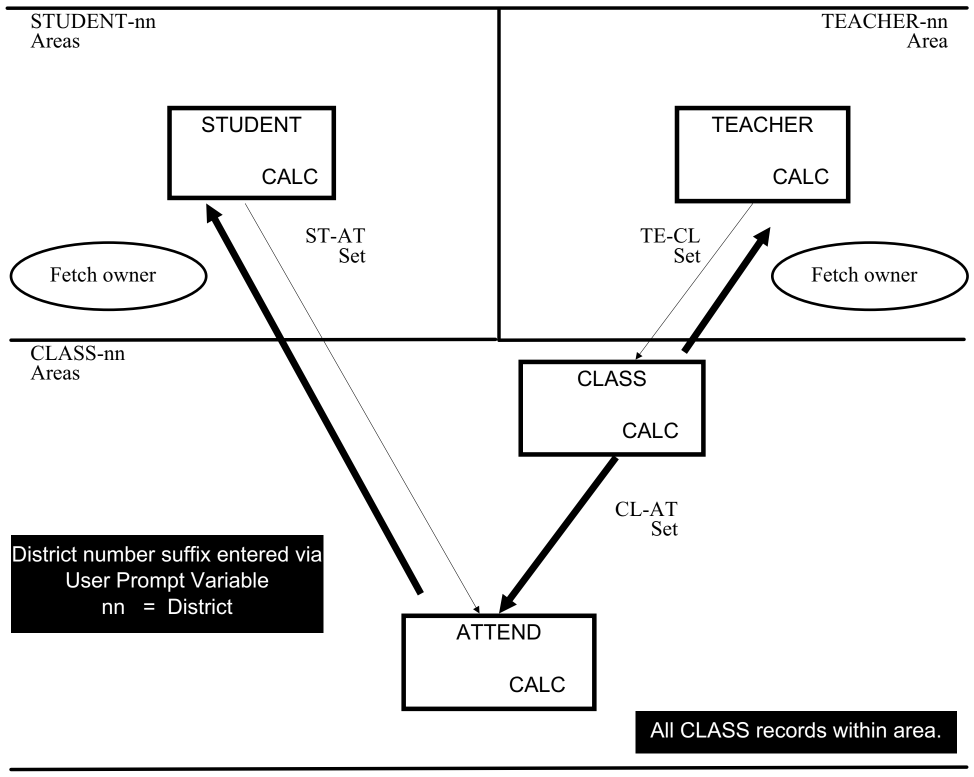
Once the above modifications have been made, the DBM is ready for testing.   The path can be tested by either creating a report using the DBM, or by using the "DBM Path Validation" feature in IQMNT.

E.2 User Prompt Used with Variable Area Name

In this example, a user prompt variable will be defined for the view. The variable will be used to customize a DBM allowing the user to select the area(s) of the database to be accessed.

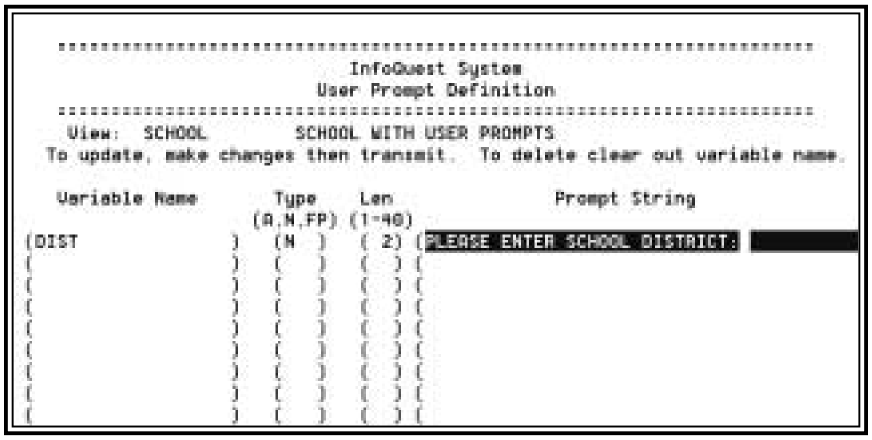
User prompt is a feature of InfoQuest that allows the introduction of variables which are not in the initial generation of the view. In most cases, this feature is used in conjunction with custom coding. The user prompt feature provides the functionality required when there is a need to pass information to the DBM that is not part of a PCIOS record description, DMS schema, RDMS table, etc. The following is a scenario of one possible use of user prompts:

Let's say we have a school system database (DMS) for a large city. Within this city there are over 100 school districts. The database areas are organized by school district. For example, the information for district 55 is in areas CLASS-55, STUDENT-55, TEACHER-55. The district number is not an integral part of any key on most records. 99% of the programs are developed by district. There is no need for the InfoQuest DBM to open the areas for all 100 districts, but rather only areas needed for a specific district.



User Prompt Used with Variable Area Name

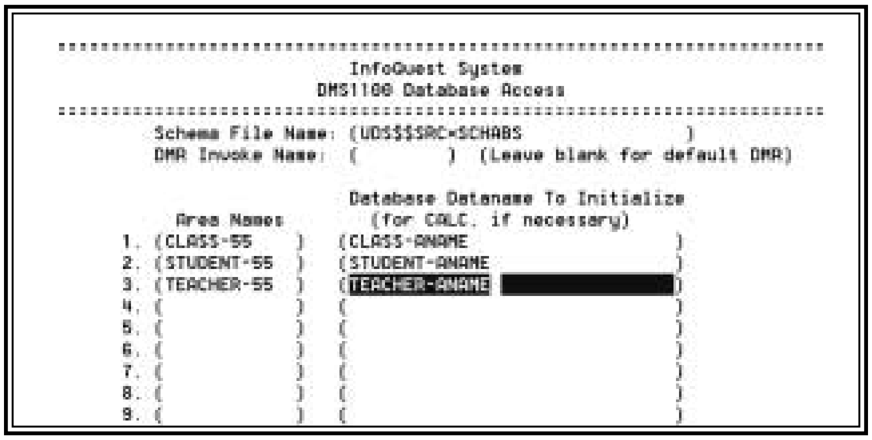
To begin the customization process, select the User Prompt Definition function of View Maintenance as shown below:



```
   ========================================================================
                              InfoQuest System
                           User Prompt Definition
   ========================================================================
    View:  SCHOOL         SCHOOL WITH USER PROMPTS
  To update, make changes then transmit.  To delete clear out variable name.

   Variable Name       Type    Len              Prompt String
                     (A,N,FP) (1-40)
(DIST              )   (N  )   ( 2) (PLEASE ENTER SCHOOL DISTRICT:
(                  )   (   )   (  ) (
(                  )   (   )   (  ) (
(                  )   (   )   (  ) (
(                  )   (   )   (  ) (
(                  )   (   )   (  ) (
(                  )   (   )   (  ) (
(                  )   (   )   (  ) (
(                  )   (   )   (  ) (
```

In our example, we will define a numeric variable called DIST. When the end user generates a request using this view, they will be prompted to enter the district number prior to entering any search values (Value/Thru entries). The value entered will be maintained in this variable. Subsequently, we will use this value in order to construct the name of each area to be accessed.

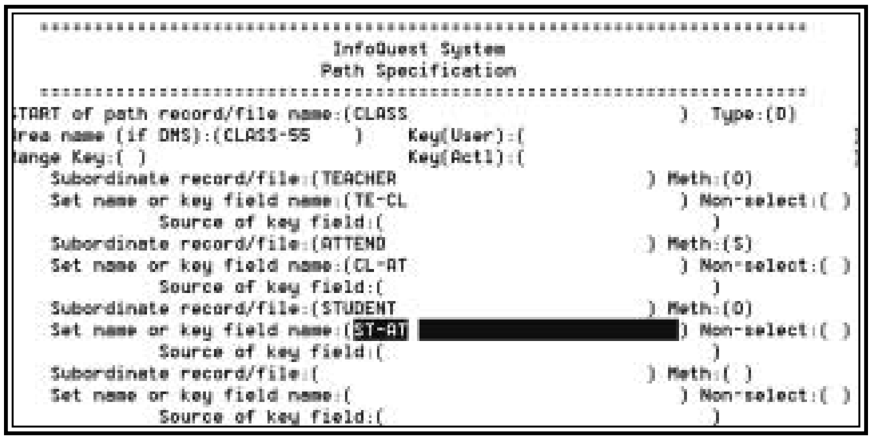
In addition to defining the user prompt variable, a DBM must be generated selecting the areas for one of the districts as shown below. It doesn't matter which district is chosen when entering the area names, as we will alter the area names on the OPEN commands once the DBM is generated. Database datanames are not required as no CALC access will be performed.



```
   ========================================================================
                              InfoQuest System
                           DMS1100 Database Access
   ========================================================================
          Schema File Name: (UDS$$SRC*SCHABS                )
          DMR Invoke Name:  (        )  (Leave blank for default DMR)

                              Database Dataname To Initialize
            Area Names          (for CALC, if necessary)
        1. (CLASS-55     )   (CLASS-ANAME                   )
        2. (STUDENT-55   )   (STUDENT-ANAME                 )
        3. (TEACHER-55   )   (TEACHER-ANAME                 )
        4. (             )   (                              )
        5. (             )   (                              )
        6. (             )   (                              )
        7. (             )   (                              )
        8. (             )   (                              )
        9. (             )   (                              )
```

User Prompt Used with Variable Area Name

The path generation information shown on the following screen indicates that the CLASS record will serve as the root of the path. Notice that, once again, the CLASS-55 area name is entered. In this case, the area name will be used by the DBM generation process to format the FETCH commands for accessing CLASS records through the area. Here again, an area name for a single district is entered as these FETCH commands will also be altered.



```
  ====================================================================
                          InfoQuest System
                         Path Specification
  ====================================================================
TART of path record/file name:(CLASS                       )  Type:(D)
rea name (if DMS):(CLASS-55    )    Key(User):(                       ]
ange Key:( )                        Key(Actl):(                       ]
   Subordinate record/file:(TEACHER                     ) Meth:(O)
   Set name or key field name:(TE-CL                       ) Non-select:( )
            Source of key field:(                              )
   Subordinate record/file:(ATTEND                      ) Meth:(S)
   Set name or key field name:(CL-AT                       ) Non-select:( )
            Source of key field:(                              )
   Subordinate record/file:(STUDENT                     ) Meth:(O)
   Set name or key field name:(ST-AT                       ) Non-select:( )
            Source of key field:(                              )
   Subordinate record/file:(                            ) Meth:( )
   Set name or key field name:(                            ) Non-select:( )
            Source of key field:(                              )
```

The following is a portion of the generated code with a few comments added to show which commands require alteration.

```
. Selection switches.  These switches, also set by the
. SETUP routine, indicate on which records selections
. are to be made in the request.
DEF N SW-SEL0000
DEF N SW-SEL0004
DEF N SW-SEL0003
DEF N SW-SEL0001
DEF N SW-SEL0002
                                . (we will insert 3 lines here)
BEGIN
           DO SETUP
           DO LOAD-SEL-TAB
           DO SORT-INIT

           DO HEADERS

. *****************************************************
. ****** Generated Navigation Logic Begins *****
   SET NON-FATAL 6 OFF     . Make ERROR-NUM 6 fatal.
   IMPART
   OPEN CLASS-55                   .    (we will delete this line)
   OPEN STUDENT-55                 .    (we will delete this line)
   OPEN TEACHER-55                 .    (we will delete this line)
.   (we will insert 9 lines here)
   FETCH4 FIRST CLASS CLASS-55 AREA  .    (we will alter this line)
   DO WHILE ERROR-NUM = 0
     DO SELECT0003
     IF SW-SELECT0003 = 1
         IF SW-REC0002 = 1
           OR SW-REC0004 = 1
           OR SW-REC0001 = 1
           DO ACCESS-0002
         ELSE
           DO SELECTED
           ENDIF
     ENDIF
     FETCH4 NEXT CLASS CLASS-55 AREA    .    (we will alter this line)
ENDDO
   GO FINALE
KEY-ERR     . *** INVALID KEY ***
   DISPLAY '*** Invalid key error processing stopped'
   GO FINALE
```

In the following customized code, three variables have been added that will be set to the three areas to be accessed. The area names are staged in the reserved special name, $PBUFF. Each area name consists of a fixed prefix (e.g., "CLASS-") followed by the value (e.g., "56") entered from the user prompt discussed earlier. $PBUFF is used to set each new variable. These variables are used on the three OPEN commands and the two FETCH commands that are highlighted below.

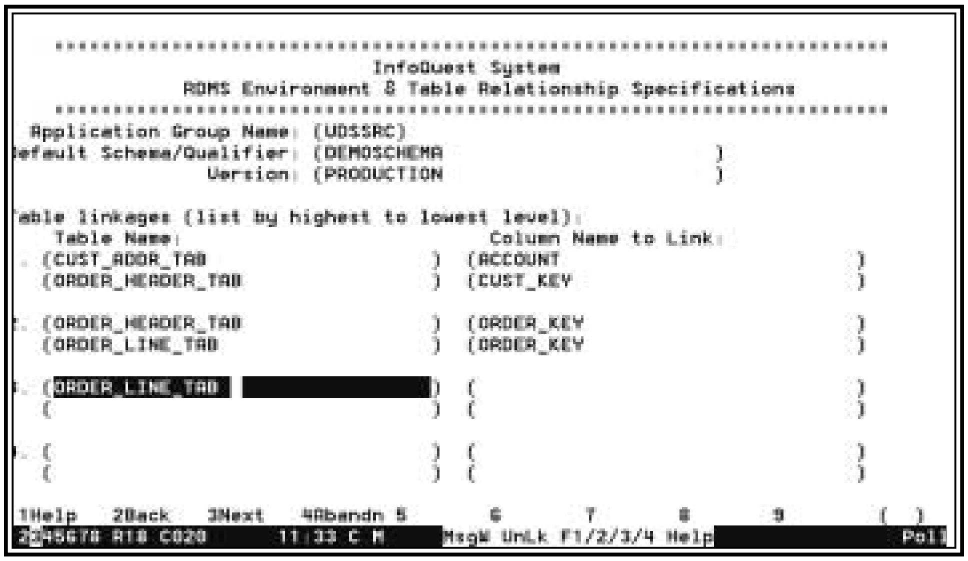
```
  . Selection switches.  These switches, also set by the
  . SETUP routine, indicate on which records selections
  . are to be made in the request.
  DEF N SW-SEL0000
  DEF N SW-SEL0004
  DEF N SW-SEL0003
  DEF N SW-SEL0001
  DEF N SW-SEL0002

  . DEFINE variable(s) to contain area name(s).
  DEF A CLASS-VAR 12
  DEF A STUDENT-VAR 12
  DEF A TEACHER-VAR 12

  BEGIN
             DO SETUP
             DO LOAD-SEL-TAB
             DO SORT-INIT

             DO HEADERS

  . ***************************************************
  . ****** Generated Navigation Logic Begins *****
     SET NON-FATAL 6 OFF     . Make ERROR-NUM 6 fatal.
     IMPART
  TRIMDISP  'CLASS-' DIST +      . Append User Prompt Value to area name.
     CLASS-VAR = $PBUFF             . Place complete area name in variable.
     OPEN CLASS-VAR                 . Open CLASS-?? area for a selected district.
     TD  'STUDENT-' DIST +
     STUDENT-VAR = $PBUFF           . And for STUDENT-??.
     OPEN STUDENT-VAR
     TD  'TEACHER' DIST +
     TEACHER-ANAME = $PBUFF         . And for TEACHER-??.
     OPEN TEACHER-VAR
     FETCH4 FIRST CLASS CLASS-VAR AREA  . Retrieve 1st from selected district.
     DO WHILE ERROR-NUM = 0
       DO SELECT0003
       IF SW-SELECT0003 = 1
           IF SW-REC0002 = 1
             OR SW-REC0004 = 1
             OR SW-REC0001 = 1
             DO ACCESS-0002
           ELSE
             DO SELECTED
             ENDIF
       ENDIF
       FETCH4 NEXT CLASS CLASS-VAR AREA . Retrieve rest from selected district.
     ENDDO
     GO FINALE
  KEY-ERR     . *** INVALID KEY ***
     DISPLAY '*** Invalid key error processing stopped'
     GO FINALE
```

E.3 Using Variables on the RDMS Permanent WHERE

In some instances, it might be desirable to restrict RDMS table access to a specific range or set of values. In Section 6, an RDMS example was shown that joined three tables via an RDMS WHERE clause. As stated, the WHERE clause can link one table to another by comparing values found between tables. It is also possible to compare table values to variables in the DBM. These variables can be reserved variables, user-defined variables or user prompt variables (see the previous example).

In the following example, the range of items selected will be limited to accounts numbers with values less than “170000000”. A user-defined variable will be created to hold this value.

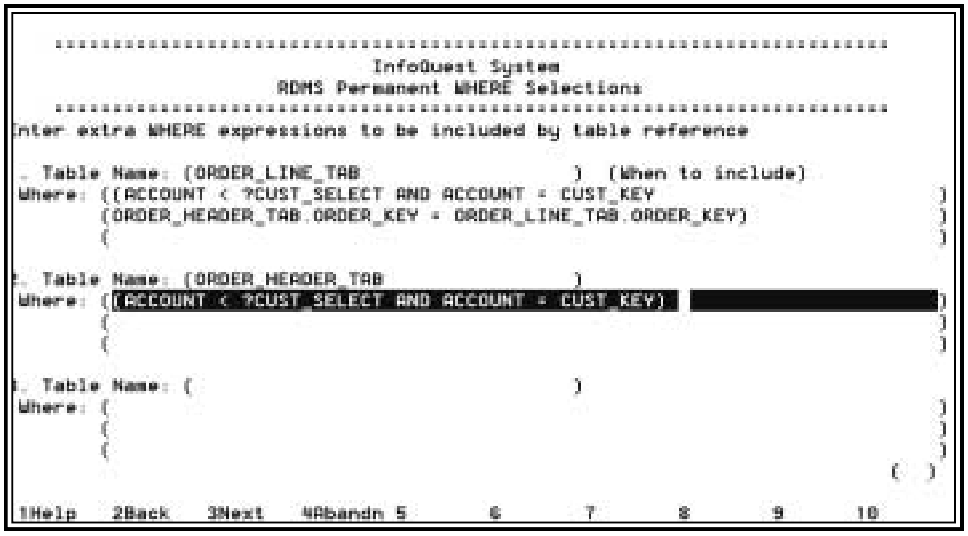
The DBM is generated in the same manner as described in Section 6. ACCOUNT and CUST_KEY will link the CUST_ADDR_TAB to the ORDER_HEADER_TAB, while ORDER_KEY will link the ORDER_HEADER_TAB to the ORDER_LINE_TAB.



```
   *******************************************************************
                            InfoQuest System
              RDMS Environment & Table Relationship Specifications
   *******************************************************************
 Application Group Name: (UDSSRC)
efault Schema/Qualifier: (DEMOSCHEMA                    )
                Version: (PRODUCTION                    )

able linkages (list by highest to lowest level):
   Table Name:                              Column Name to Link:
.  (CUST_ADDR_TAB                  )  (ACCOUNT                          )
   (ORDER_HEADER_TAB               )  (CUST_KEY                         )

.  (ORDER_HEADER_TAB               )  (ORDER_KEY                        )
   (ORDER_LINE_TAB                 )  (ORDER_KEY                        )

.  (ORDER_LINE_TAB                 )  (                                 )
   (                               )  (                                 )

.  (                               )  (                                 )
   (                               )  (                                 )

1Help   2Back   3Next   4Abandn 5      6      7      8      9        (  )
2345678 R18 C020        11:33 C M      MsgW UnLk F1/2/3/4 Help            Poll
```

Using Variables on the RDMS Permanent WHERE

Notice that when a variable is to be used on a permanent WHERE clause parameter, it must be prefixed with a question mark (?). The variable reference must be repeated for each table that is linked via a WHERE.



```
  ==================================================================
                          InfoQuest System
                   RDMS Permanent WHERE Selections
  ==================================================================
Enter extra WHERE expressions to be included by table reference

. Table Name: (ORDER_LINE_TAB              )  (When to include)
Where: ((ACCOUNT < ?CUST_SELECT AND ACCOUNT = CUST_KEY             )
       (ORDER_HEADER_TAB.ORDER_KEY = ORDER_LINE_TAB.ORDER_KEY)      )
       (                                                            )

2. Table Name: (ORDER_HEADER_TAB           )
Where: ((ACCOUNT < ?CUST_SELECT AND ACCOUNT = CUST_KEY)             )
       (                                                            )
       (                                                            )

3. Table Name: (                           )
Where: (                                                            )
       (                                                            )
       (                                                            )
                                                               (  )
1Help   2Back   3Next   4Abandn 5      6      7      8      9      10
```

Once the initial DBM has been generated, the code only needs to be changed to DEFINE a variable to contain the constant, “170000000”. If a user prompt variable had been defined in the view, no alteration whatsoever would be required.

The following is a portion of the generated code with a few comments added to show which commands require alteration:

```
          . (We will insert one line here)
BEGIN
           DO SETUP
           DO LOAD-SEL-TAB
           DO SORT-INIT
           DO HEADERS
. ***************************************************
. ****** Generated Navigation Logic Begins *****
```

The code is changed to include a user-defined variable which will be referenced later in the DBM:

```
def a cust_select ‘170000000’
BEGIN
           DO SETUP
           DO LOAD-SEL-TAB
           DO SORT-INIT
           DO HEADERS
. ***************************************************
. ****** Generated Navigation Logic Begins *****
```

Using Variables on the RDMS Permanent WHERE

At the bottom of the DBM, the permanent WHERE clause does not require any alteration to include the user-defined variable, “cust_select”, since the WHERE condition using the variable was entered on the RDMS Path Generation Screen.

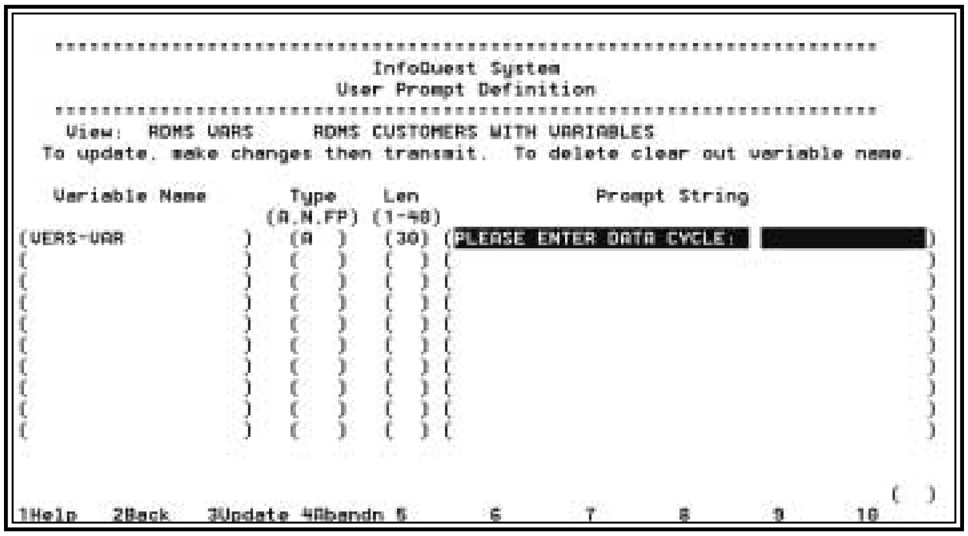
```
. ****** Generated Path Navigation Logic Ends *****
. ************************************************
RDMSERR PROCEDURE
           IF RDMS-STAT <> ‘0000’
             AND RDMS-STAT <> ‘6001’
             DISPLAY ‘*** RDMS error encountered, processing stopped.’
             DISPLAY ‘*** STAT:’ RDMS-STAT ‘ AUX:’ RDMS-AUX
             DO UNTIL RDMS-ERR = $SPACES
               RDMS ‘GETERROR INTO $P1 ;’ RDMS-STAT RDMS-AUX RDMS-ERR
               DISPLAY RDMS-ERR
             ENDDO
             STOP 9999
           ENDIF
           ENDPROC
ADD STDCODE.
*WHERE         . Permanent where clauses
:5002     ORDER_LINE_TAB
(ACCOUNT < ?CUST_SELECT AND ACCOUNT = CUST_KEY AND
ORDER_HEADER_TAB.ORDER_KEY = ORDER_LINE_TAB.ORDER_KEY)
:5001     ORDER_HEADER_TAB
(ACCOUNT < ?CUST_SELECT AND ACCOUNT = CUST_KEY)
*WHEREEND      . End of permanent where clauses
```

E.4 Using Variables on RDMS USE Commands

In this example, a user prompt variable will be defined for the view. The variable will be used to customize a DBM allowing the user to choose which RDMS database VERSION is to be accessed. In this manner, the user can select any valid database version established in the UREP 1100 data dictionary.

For this example, assume that a new version of the database is created on a cyclical or annual basis. The customer order information for 1989 might be in version ORDERS89; while, orders for 1990 are in ORDERS90.

To begin the customization process, select the User Prompt Definition function of View Maintenance as shown below:

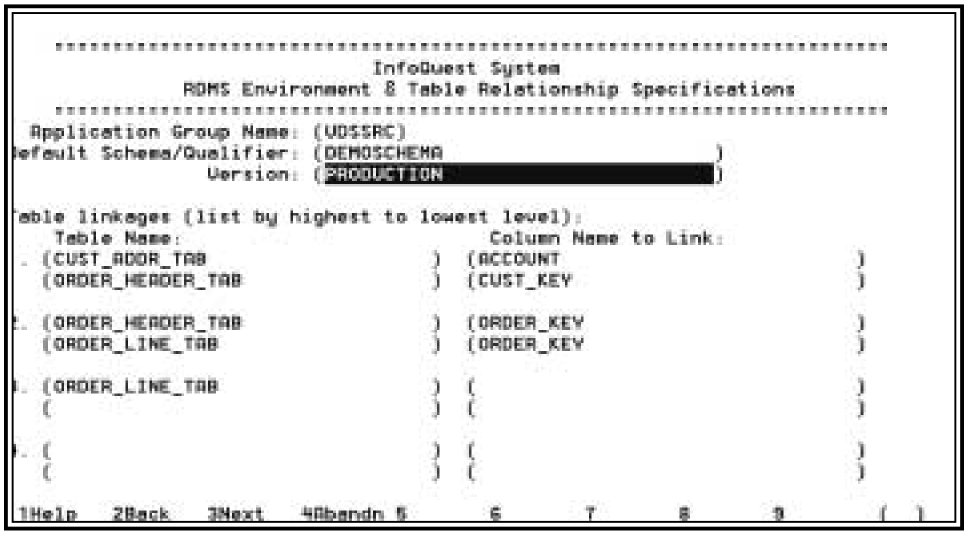


```
   ***************************************************************************
                                   InfoQuest System
                                 User Prompt Definition
   ***************************************************************************
    View:  RDMS VARS     RDMS CUSTOMERS WITH VARIABLES
  To update, make changes then transmit.  To delete clear out variable name.

   Variable Name        Type     Len              Prompt String
                      (A,N,FP) (1-40)
(VERS-VAR           )   (A  )   (30) (PLEASE ENTER DATA CYCLE:                  )
(                   )   (   )   (  ) (                                          )
(                   )   (   )   (  ) (                                          )
(                   )   (   )   (  ) (                                          )
(                   )   (   )   (  ) (                                          )
(                   )   (   )   (  ) (                                          )
(                   )   (   )   (  ) (                                          )
(                   )   (   )   (  ) (                                          )
(                   )   (   )   (  ) (                                          )
(                   )   (   )   (  ) (                                          )

                                                                          (  )
1Help   2Back   3Update 4Abandn 5       6       7       8       9       10
```

In our example, we will define an alpha variable called VERS_VAR. When the end user generates a request using this view, they will be prompted to enter the “Data Cycle” (really an RDMS version: ORDERS89 or ORDERS90) prior to entering any search values (Value/Thru entries). The value entered will be maintained in this variable. Subsequently, we will use this value in order to construct the name of each area to be accessed.

Using Variables on RDMS USE Commands

On the RDMS Path Specification screen, any name may be entered for version. The DBM generation process will use this name to generate the RDMS "USE DEFAULT VERSION" command.



```
   **********************************************************************
                              InfoQuest System
              RDMS Environment & Table Relationship Specifications
   **********************************************************************
  Application Group Name: (UDSSRC)
efault Schema/Qualifier: (DEMOSCHEMA                  )
                 Version: (PRODUCTION                  )

able linkages (list by highest to lowest level):
    Table Name:                              Column Name to Link:
. (CUST_ADDR_TAB                    )  (ACCOUNT                      )
  (ORDER_HEADER_TAB                 )  (CUST_KEY                     )

. (ORDER_HEADER_TAB                 )  (ORDER_KEY                    )
  (ORDER_LINE_TAB                   )  (ORDER_KEY                    )

. (ORDER_LINE_TAB                   )  (                             )
  (                                 )  (                             )

. (                                 )  (                             )
  (                                 )  (                             )
1Help   2Back   3Next   4Abandn 5      6      7      8      9       (  )
```

The following is a portion of the generated code with a few comments added to show which commands require alteration.

```
. ***************************************************
. ****** Generated Navigation Logic Begins *****
        RDMS 'BEGIN THREAD FOR APPLICATION UDSSRC RETRIEVE ; ' ;
          RDMS-STAT RDMS-AUX
        DO RDMSERR
        RDMS 'USE DEFAULT QUALIFIER DEMOSCHEMA ; ' ;
          RDMS-STAT RDMS-AUX
        DO RDMSERR

    . (we will replace the following line with 2 new lines)
        RDMS 'USE DEFAULT VERSION PRODUCTION ; ' ;
          RDMS-STAT RDMS-AUX
        DO RDMSERR
        DO RDMSCURSOR
        DO RDMSFETCH
        DO WHILE RDMS-STAT  '6001'
          DO SELECTED
          DO RDMSFETCH
        ENDDO
    GO FINALE
 KEY-ERR     . *** INVALID KEY ***
    DISPLAY '*** Invalid key error processing stopped'
    GO FINALE

. ***** End code insertion:
. ***** INVOKE from file name count = 0
. ***** File definitions created = 0
. ***** Path code lines created = 20
. ****** Generated Path Navigation Logic Ends *****
. ***************************************************
```

Using Variables on RDMS USE Commands

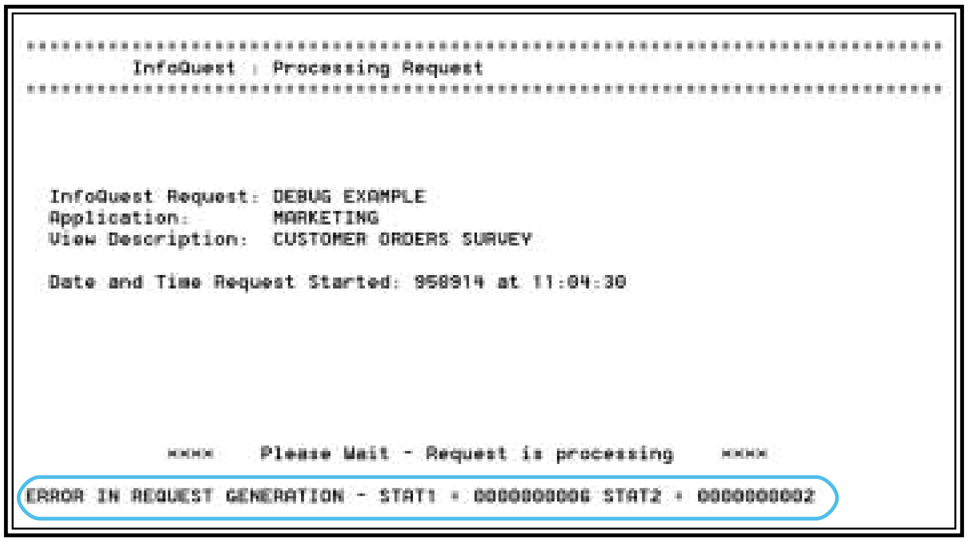
Since RDMS does not allow program variables to be used in place of the version name on the "USE DEFAULT VERSION" command, it will be necessary to construct the command prior to releasing it to RDMS. In the following customized code, the command is staged in the reserved special name, $PBUFF. The user prompt variable, VERS_VAR, will contain the actual version name. Once the command is built in $PBUFF, $PBUFF can be referenced on the RDMS command in place of the string literal.

```
. ****************************************************
. ****** Generated Navigation Logic Begins *****
       RDMS 'BEGIN THREAD FOR APPLICATION UDSSRC RETRIEVE ; ' ;
         RDMS-STAT RDMS-AUX
       DO RDMSERR
       RDMS 'USE DEFAULT QUALIFIER DEMOSCHEMA ; ' ;
         RDMS-STAT RDMS-AUX
       DO RDMSERR
.                                                  . Get version from
       TD 'USE DEFAULT VERSION ' VERS_VAR ' ; ' + .  user prompt
       RDMS $PBUFF ;                               . Send RDMS command
         RDMS-STAT RDMS-AUX                        .  from print buffer
.
       DO RDMSERR
       DO RDMSCURSOR
       DO RDMSFETCH
       DO WHILE RDMS-STAT  '6001'
         DO SELECTED
         DO RDMSFETCH
       ENDDO
   GO FINALE
KEY-ERR     . *** INVALID KEY ***
   DISPLAY '*** Invalid key error processing stopped'
   GO FINALE

. ***** End code insertion:
. ***** INVOKE from file name count = 0
. ***** File definitions created = 0
. ***** Path code lines created = 20
. ****** Generated Path Navigation Logic Ends *****
. ****************************************************
```

Using Variables on RDMS USE Commands

Appendix F DBM Error Detection

If you are testing tailored DBMs (see Appendix E “Customized DBMs”), there will be times that the server will return a status 6 (see below). Status 6 usually means that the customized DBM has a program error:



```
******************************************************************************
          InfoQuest : Processing Request
******************************************************************************



  InfoQuest Request: DEBUG EXAMPLE
  Application:       MARKETING
  View Description:  CUSTOMER ORDERS SURVEY

  Date and Time Request Started: 958914 at 11:04:30





            ****    Please Wait - Request is processing    ****
ERROR IN REQUEST GENERATION - STAT1 = 0000000006 STAT2 = 0000000002
```

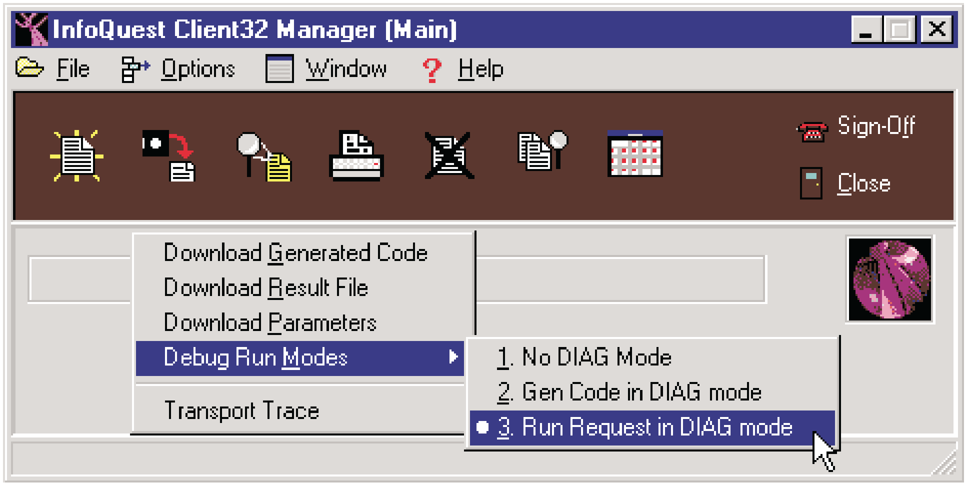
The procedure on the following pages can be used to determine which Q-LINK commands are in error in the DBM. The chapter includes procedures for debugging with InfoQuest Client32, the MAPPER-based InfoQuest and InfoQuest/EX.

Note: You may debug any request (regardless of whether it was created by InfoQuest Client32, InfoQuest/MAPPER or InfoQuest/EX) with the Debug Request function of the IQMNT program; however, this function requires InfoQuest Administrator privileges (see "Debug Request" on page 3-15).

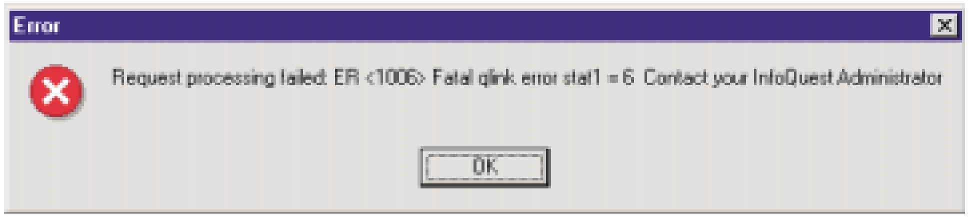
DBM Error Detection

F.1 Debugging Using InfoQuest Client

For InfoQuest Client, the debugging function is invoked by clicking the right mouse button over the grey area behind the buttons on the button bar.

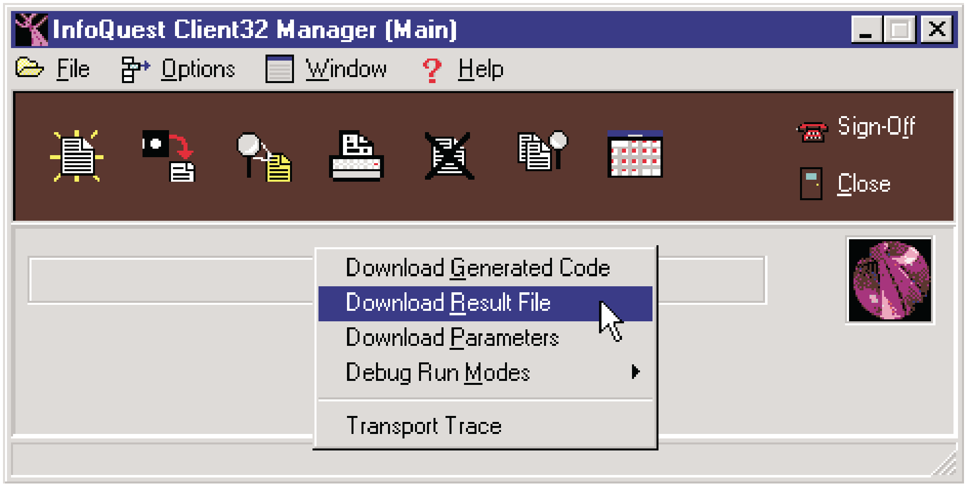


Choose **Run Request in DIAG Mode** from the **Debug Run Modes** menu. Next run the request. The request will once again return the following error:

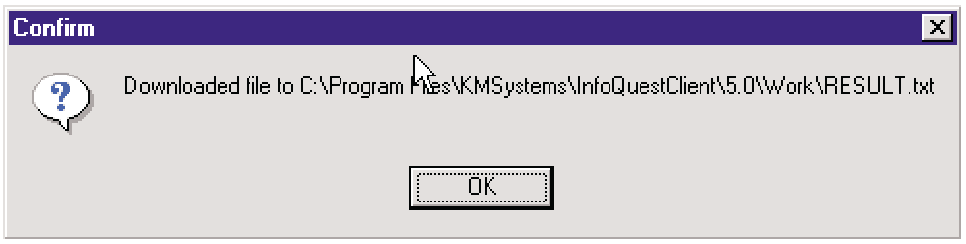


Request processing failed: ER <1006> Fatal qlink error stat1 = 6. Contact your InfoQuest Administrator

At this point you are ready to download the result file as shown below:



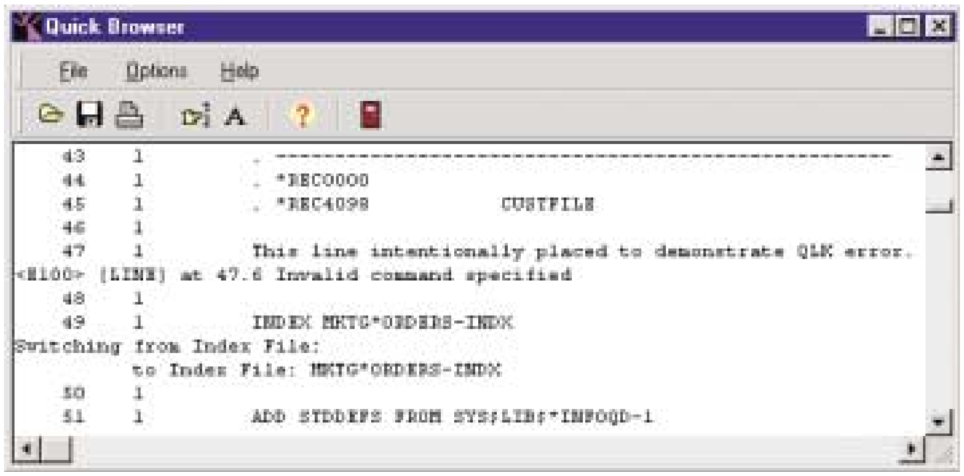
When the down load is complete, the following message will appear pointing to the location of the download result:


Confirm

Downloaded file to C:\Program Files\KMSystems\InfoQuestClient\5.0\Work\RESULT.txt

DBM Error Detection

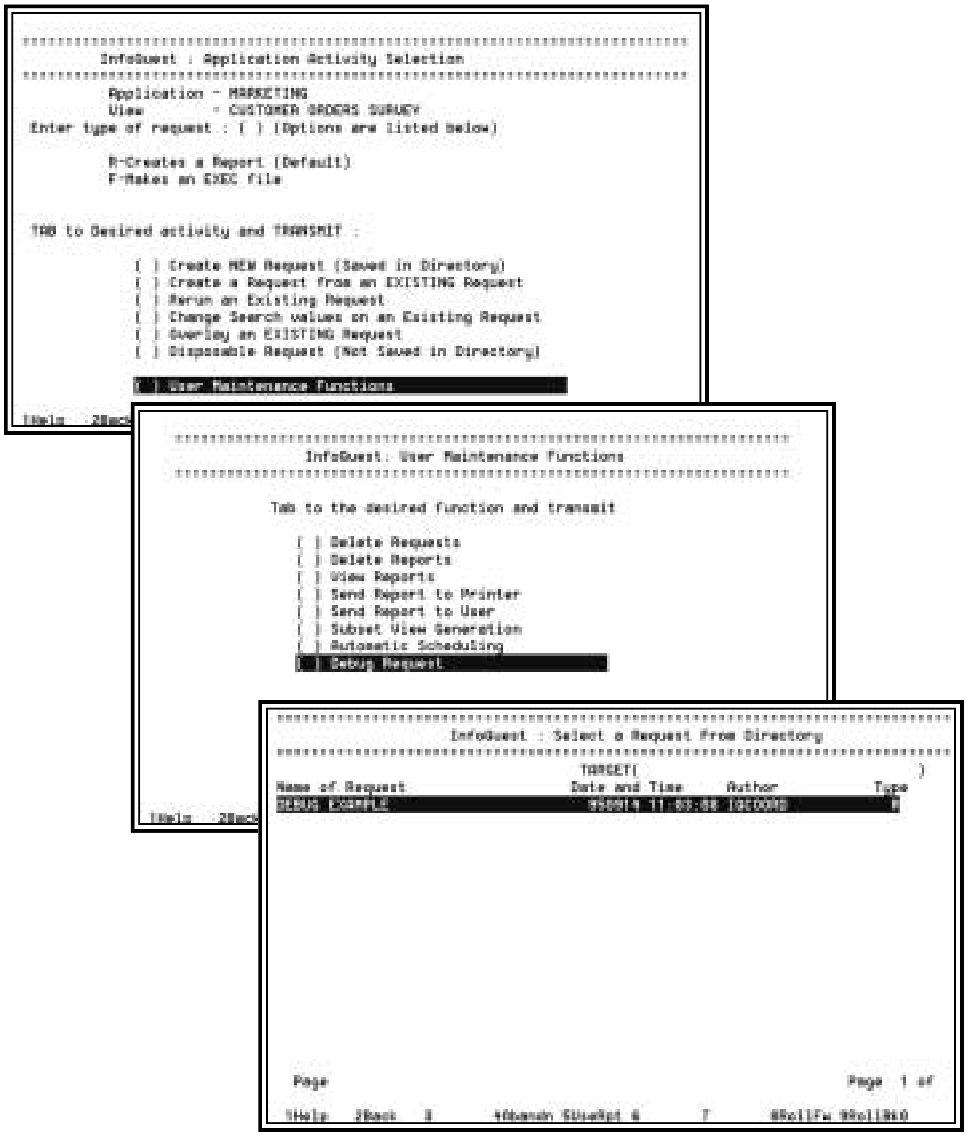
Edit the result with the Quick Browser, NotePad or other text editor to view the result:



```
    43    1          . ------------------------------------------------------
    44    1          . *REC0000
    45    1          . *REC4098            CUSTFILE
    46    1
    47    1          This line intentionally placed to demonstrate QLK error.
<E100> [LINE] at 47.6 Invalid command specified
    48    1
    49    1          INDEX MKTG*ORDERS-INDX
Switching from Index File:
          to Index File: MKTG*ORDERS-INDX
    50    1
    51    1          ADD STDDEFS FROM SYS$LIB$*INFOQD-1
```

F.2 Debugging Using InfoQuest and InfoQuest/EX

KMSystems provides a special function to be used to investigate InfoQuest execution problems. This function, Debug Request, is a selection on the User Maintenance Functions screen (see below) for both the MAPPER-based InfoQuest and InfoQuest/EX.



```
*****************************************************************************
         InfoQuest : Application Activity Selection
*****************************************************************************
         Application - MARKETING
         View        = CUSTOMER ORDERS SURVEY
 Enter type of request : [ ] (Options are listed below)

         R-Creates a Report (Default)
         F-Makes an EXEC file


 TAB to Desired activity and TRANSMIT :

             [ ] Create NEW Request (Saved in Directory)
             [ ] Create a Request from an EXISTING Request
             [ ] Rerun an Existing Request
             [ ] Change Search values on an Existing Request
             [ ] Overlay an EXISTING Request
             [ ] Disposable Request (Not Saved in Directory)

             [ ] User Maintenance Functions

 1Help   2Back
```

```
*****************************************************************************
                InfoQuest: User Maintenance Functions
*****************************************************************************

           Tab to the desired function and transmit

              [ ] Delete Requests
              [ ] Delete Reports
              [ ] View Reports
              [ ] Send Report to Printer
              [ ] Send Report to User
              [ ] Subset View Generation
              [ ] Automatic Scheduling
              [ ] Debug Request

 1Help   2Back
```

```
*****************************************************************************
               InfoQuest : Select a Request From Directory
*****************************************************************************
                                   TARGET(                                  )
Name of Request                    Date and Time     Author           Type
DEBUG EXAMPLE                       980814 11:03:08 [illegible]        [illegible]



  Page                                                           Page  1 of

 1Help   2Back   3        4Abandn 5UseRpt 6        7        8RollFw 9RollBk0
```

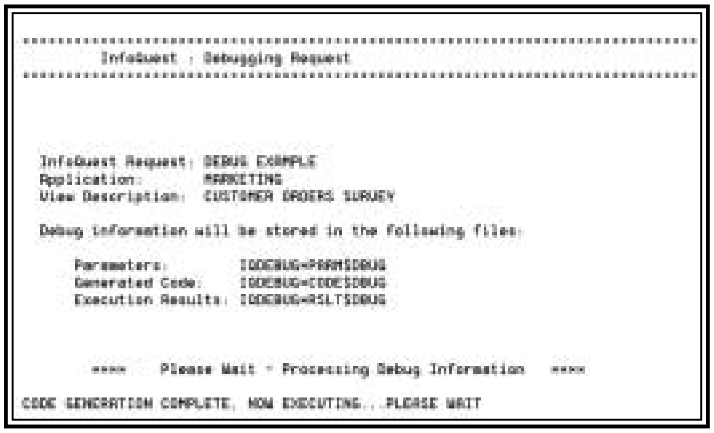
After selecting the Debug Request function, tab to the desired request and transmit.

DBM Error Detection

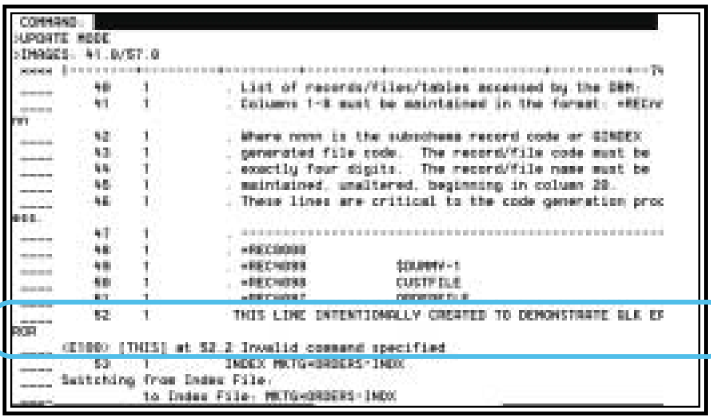
The Debugging Request screen shows three cycled files that are created as a result of the code generation and execution process. The three files are described below and may be examined with any editor:

Physical File Name	@USE File Name	Description
IQDEBUG*PARM$DEBUG.	IQPARM.	The InfoQuest parameters and DBM that are input to the code generation process.
IQDEBUG*CODE$DEBUG.	IQCODE.	The generated request.
IQDEBUG*RSLT$DEBUG.	IQRSLT.	The result of the execution of the generated request.

When the debugging process completes, exit IQEX and edit the third file, IQRSLT.



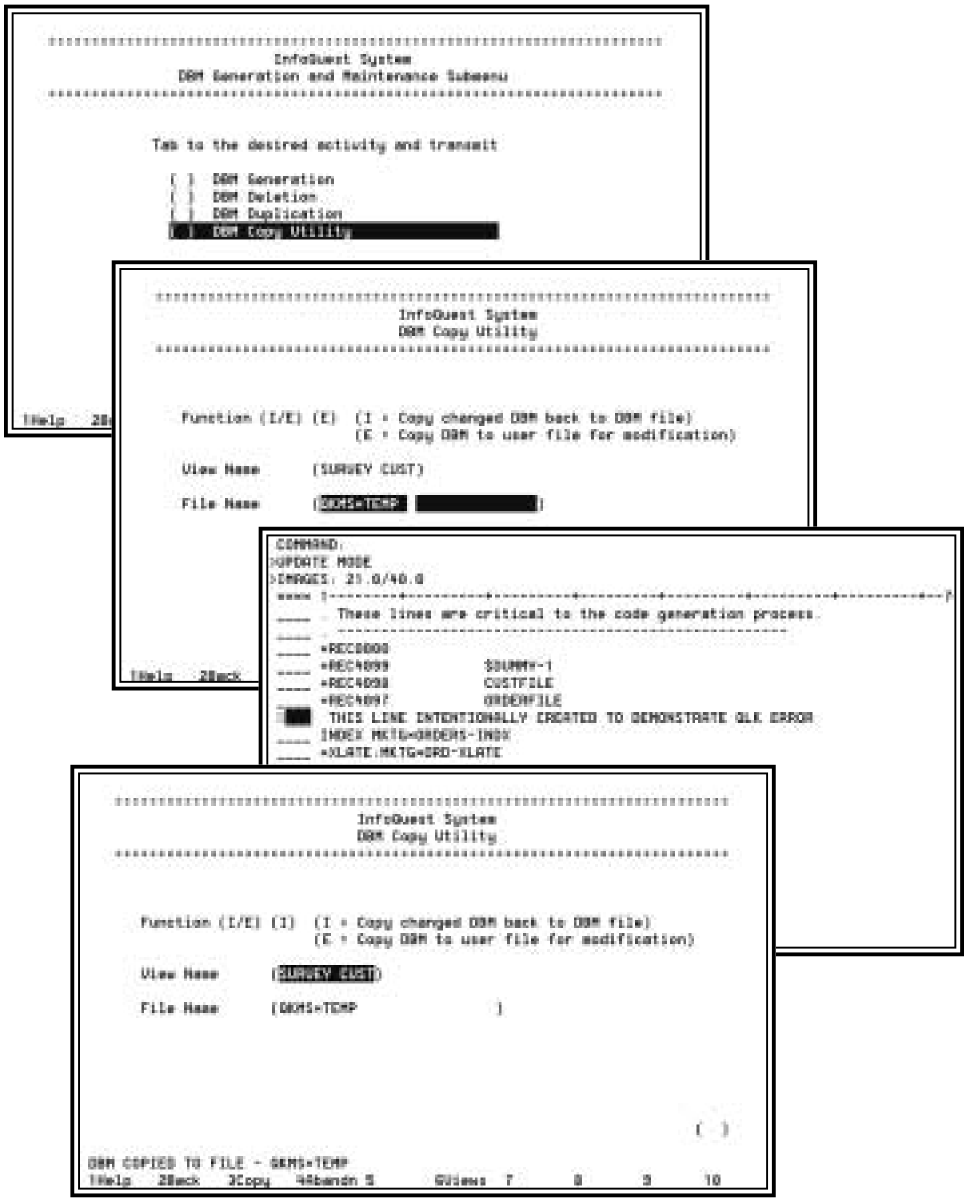
```
********************************************************************************
        InfoQuest : Debugging Request
********************************************************************************



  InfoQuest Request: DEBUG EXAMPLE
  Application:       MARKETING
  View Description:  CUSTOMER ORDERS SURVEY

  Debug information will be stored in the following files:

      Parameters:        IQDEBUG*PARM$DBUG
      Generated Code:    IQDEBUG*CODE$DBUG
      Execution Results: IQDEBUG*RSLT$DBUG


       ****    Please Wait - Processing Debug Information   ****

CODE GENERATION COMPLETE, NOW EXECUTING...PLEASE WAIT
```



```
 COMMAND:
:UPDATE MODE
:IMAGES: 41.0/57.0
 ***** |---------+---------+---------+---------+---------+---------+---------+--7\
 _____     40    1          . List of records/files/tables accessed by the DBM:
 _____     41    1          . Columns 1-8 must be maintained in the format: *RECnn
nn
 _____     42    1          . Where nnnn is the subschema record code or $INDEX
 _____     43    1          . generated file code.  The record/file code must be
 _____     44    1          . exactly four digits.  The record/file name must be
 _____     45    1          . maintained, unaltered, beginning in column 20.
 _____     46    1          . These lines are critical to the code generation proc
ess.
 _____     47    1          . ****************************************************
 _____     48    1          . *REC0000
 _____     49    1          . *REC4099          $DUMMY-1
 _____     50    1          . *REC4098          CUSTFILE
 _____     51    1          . *REC4097          ORDERFILE
 _____     52    1          THIS LINE INTENTIONALLY CREATED TO DEMONSTRATE SLK ER
ROR
 _____ <E100> [THIS] at 52.2 Invalid command specified
 _____     53    1         INDEX MKTG*ORDERS-INDX
 _____ Switching from Index File:
               to Index File: MKTG*ORDERS-INDX
```

F.3 Correcting the DBM

To correct the DBM, use the DBM Copy Utility in IQMNT to export the DBM to a flat file where the error(s) may be corrected and returned to InfoQuest via the import option of the DBM Copy Utility.



```
                              InfoQuest System
                   DBM Generation and Maintenance Submenu

          Tab to the desired activity and transmit

            [ ]  DBM Generation
            [ ]  DBM Deletion
            [ ]  DBM Duplication
            [ ]  DBM Copy Utility

1Help   2B
```

```
                              InfoQuest System
                              DBM Copy Utility

   Function (I/E) (E)  (I = Copy changed DBM back to DBM file)
                       (E = Copy DBM to user file for modification)

   View Name      (SURVEY CUST)

   File Name      (DKMS=TEMP                 )

1Help   2Back
```

```
COMMAND:
>UPDATE MODE
>IMAGES: 21.0/40.0
**** |---------+---------+---------+---------+---------+---------+---------+--|
____ . These lines are critical to the code generation process.
____ . ------------------------------------------------------
____ *REC0000
____ *REC4099           $DUMMY-1
____ *REC4098           CUSTFILE
____ *REC4097           ORDERFILE
____  THIS LINE INTENTIONALLY CREATED TO DEMONSTRATE QLK ERROR
____ INDEX MKTG=ORDERS-INDX
____ *XLATE:MKTG=ORD-XLATE
```

```
                              InfoQuest System
                              DBM Copy Utility

   Function (I/E) (I)  (I = Copy changed DBM back to DBM file)
                       (E = Copy DBM to user file for modification)

   View Name      (SURVEY CUST)

   File Name      (DKMS=TEMP                 )

                                                                  ( )
DBM COPIED TO FILE - DKMS=TEMP
1Help   2Back   3Copy   4Abandn 5      6Views  7      8      9      10
```

DBM Error Detection

Appendix G RDMS Tables

This appendix lists the RDMS tables and the data items used in the RDMS examples shown throughout this guide. The data item index file shown below was built with the "Create RDMS QINDEX" function of the InfoQuest maintenance run, IQMNT.

```
Listing contents of QINDEX file: mktg*rdmsindx.
CUST_ADDR_TAB                    5000 RDMS-1100
 Item descriptions:
  05 ACCOUNT                          RDA reference (22,9) A9
  05 ADDR1                            RDA reference (64,30) A9
  05 ADDR2                            RDA reference (94,30) A9
  05 ADDR3                            RDA reference (124,30) A9
  05 AREACODE                         RDA reference (163,3) A9
  05 CITY                             RDA reference (4,18) A9
  05 CUSTNAME                         RDA reference (31,33) A9
  01 CUST_ADDR_TAB                    RDA reference (5,172) A9
  05 EXCHANGE                         RDA reference (166,3) A9
  05 STATE                            RDA reference (1,2) A9
  05 TELNUM                           RDA reference (169,4) A9
  05 ZIP                              RDA reference (154,9) A9

ORDER_HEADER_TAB                 5001 RDMS-1100
 Item descriptions:
  05 AUTHDLR_CODE                     RDA reference (183,1) A9
  05 BOL_PRT_CODE                     RDA reference (73,1) A9
  05 BUYER                            RDA reference (34,11) A9
  05 CITY_TAX                         RDA reference (155,4) SB9 .02
  05 COUNTY_TAX                       RDA reference (159,4) SB9 .02
  05 CREDIT_HOLD                      RDA reference (70,1) A9
  05 CREDIT_USERID                    RDA reference (163,8) A9
  05 CREL_DA                          RDA reference (143,2) A9
  05 CREL_MO                          RDA reference (141,2) A9
  05 CREL_YR                          RDA reference (139,2) A9
  05 CUST_KEY                         RDA reference (9,9) A9
  05 DELETE_FLAG                      RDA reference (107,1) A9
  05 DISCOUNT                         RDA reference (135,4) SB9 .02
  05 ENTRY_DA                         RDA reference (30,2) A9
  05 ENTRY_MO                         RDA reference (28,2) A9
  05 ENTRY_YR                         RDA reference (32,2) A9
  05 HOLD_CODE                        RDA reference (71,1) A9
  05 INPROCESS_HOLD                   RDA reference (108,1) A9
  05 INVOICE_CODE                     RDA reference (72,1) A9
  05 NBR_PALLETS                      RDA reference (171,4) SB9
  01 ORDER_HEADER_TAB                 RDA reference (157,180) A9
  05 ORDER_KEY                        RDA reference (1,7) A9
  05 ORDER_LOC                        RDA reference (18,4) SB9
  05 ORDER_TYPE_CODE                  RDA reference (26,1) A9
```

```
    05 PALLET_CHG                       RDA reference (175,4) SB9 .02
    05 PAY_METHOD_CODE                  RDA reference (74,3) A9
    05 PIECES                           RDA reference (115,4) SB9
    05 PRODLINE_CODE                    RDA reference (27,1) A9
    05 PURCHASE_ORD                     RDA reference (45,8) A9
    05 REQ_SHIPDATE                     RDA reference (53,6) A9
    05 SHIP_DA                          RDA reference (113,2) A9
    05 SHIP_FEE                         RDA reference (123,4) SB9 .02
    05 SHIP_LOC                         RDA reference (22,4) SB9
    05 SHIP_MO                          RDA reference (111,2) A9
    05 SHIP_VIA                         RDA reference (59,11) A9
    05 SHIP_YR                          RDA reference (109,2) A9
    05 SPECIAL_TERMS                    RDA reference (78,20) A9
    05 STATE_TAX                        RDA reference (151,4) SB9 .02
    05 TERM_CODE                        RDA reference (98,1) A9
    05 TERM_DATE_DAYS                   RDA reference (103,4) SB9
    05 TERM_PER                         RDA reference (99,4) SB9 .04
    05 TOT_CHARGES                      RDA reference (127,4) SB9 .02
    05 TOT_CLC                          RDA reference (131,4) SB9 .02
    05 TOT_PALLET_COST                  RDA reference (179,4) SB9 .02
    05 WEIGHT                           RDA reference (119,4) SB9
    05 WKORD_PRT_DA                     RDA reference (149,2) A9
    05 WKORD_PRT_MO                     RDA reference (147,2) A9
    05 WKORD_PRT_YR                     RDA reference (145,2) A9
    05 WORKORD_CODE                     RDA reference (77,1) A9

  ORDER_LINE_TAB                   5002 RDMS-1100
   Item descriptions:
    05 BILL_ONLY_CODE                   RDA reference (73,1) A9
    05 BOL_KEY                          RDA reference (63,4) SB9
    05 DESC                             RDA reference (24,25) A9
    05 EXCEPTION_SW                     RDA reference (82,1) A9
    05 LAST_DATE                        RDA reference (75,6) A9
    05 ORDER_KEY                        RDA reference (8,7) A9
    01 ORDER_LINE_TAB                   RDA reference (157,88) A9
    05 PACKAGE                          RDA reference (53,8) A9
    05 PRICE_CHANGE                     RDA reference (74,1) A9
    05 PRICE_CODE                       RDA reference (61,2) A9
    05 PRIORITY                         RDA reference (81,1) A9
    05 PRODUCT                          RDA reference (1,6) A9
    05 QUANTITY                         RDA reference (16,4) SB9
    05 REG_CODE                         RDA reference (68,1) A9
    05 SHIP_QTY                         RDA reference (69,4) SB9
    05 SUB_ITEM                         RDA reference (83,6) A9
    05 TAX_CODE                         RDA reference (67,1) A9
    05 TYPE_ORD_CODE                    RDA reference (15,1) A9
    05 UNIT_PRICE                       RDA reference (20,4) SB9 .02
    05 WEIGHT                           RDA reference (49,4) SB9 .02
```

Appendix H InfoQuest Security

This appendix provides a discussion on the administration of InfoQuest security.

H.1 Determining Security Requirements

Security in InfoQuest is based on each application, that is, that each application has its own security. A particular user may have different security requirements depending on which application he is in. An example of this would be a site that has 4 different applications and 20 users. Each user would have to be evaluated to determine which application(s) he would be allowed to access and then to determine what level of security he would need in those applications.

In order for a user to have access to an application, he must have a registration entry for each application he is to access. Even if the site has multiple applications, users can be limited to which applications they are allowed to access. Once it has been determined to which applications the user will be allowed access, it will then be necessary to determine the level of security needed in each application. Security levels are based on a 2 digit numeric value with 01 being the most secure level and 99 being the least secure.

To determine which security level to use, the following items must be taken into consideration:

1) The first consideration is the existing report level the user will be allowed to access. Access to any existing reports is based on the security level value, which means that a user can view any reports generated with a value equal to or greater than the value used to register the user. For example, a user at security level 05 can view any reports created at level 05 through 99. The report security level is the same as the security level of the person who created the report.

2) The second consideration is the view level the user will be allowed to access within an application. An application can have many views, some of which may contain sensitive data while others may only contain general information data. Each view that is created is given a security level at the time it is created. The access to the views works in the same manner as the access to the reports. Any user can access a view with a security level value that is equal to or greater than

that of the user. For example, a view generated at level 15 can be accessed by any user at level 01 through 15, but not by users at levels 16 throuth 99.

H.2 Maintenance Considerations for Security

The maintenance portion of InfoQuest is where all of the views are created, where all of the access paths are created and where all of the users are registered. Access to the maintenance works in the same manner as access to the report creation. Users can be limited to maintenance functions for specific applications, or they can have no maintenance functions at all (except for the user housekeeping).

In a standard site setup, there is one person who is known as the InfoQuest Administrator. This person has access to all maintenance functions for all applications. Each application then has its own person, or persons, who is responsible for maintenance for that application only. The InfoQuest Administrator is the only person who can register users for application access. Based on a request from the user department, the administrator can register users for specific applications and assign the security codes based on each user's data needs.

To be registered as an InfoQuest Administrator, you must be registered for application "00" with a security code of "02". The person who installs InfoQuest is automatically registered as a administrator. This person can then register other users to perform maintenance on given applications. In order for a user to have the maintenance functions, he must be registered in the application with a security code of 02.

Appendix I QLK External Function Errors

The following is a list of error codes, explanations of errors and possible error resolutions for all errors reported to MAPPER runs by the @QLK external function and the QLKSIM (Q-LINK/InfoQuest only) program.

ERROR CODE	DESCRIPTION
1	SPECIFIED SERVER NOT ACTIVE Explanation: The server class passed via the QLK function or the third specification field on the QLKSIM processor does not have any servers active and no servers were successfully activated by the server auto-start process (if configured). Resolution: If not using the server auto-start, you might configure it. If batch servers are taking too long to start, consider using TIP servers. If you do not use the server auto-start, consider having an initial number of servers started when the system is initialized by QMON (R-option in conjunction with the L-option or the RESET command), and carefully consider the effects of using the terminate idle server timer configuration. Investigate why auto-started servers are not able to start. Check the QMON STATUS keyin to verify that the requested class is in the CDB tables. If the PEND or FAIL or QMON columns of the class status are non-zero, investigate why auto-start requests are being delayed or rejected. Remember if using the QMON auto-start configuration (versus BATCH or TIP) that auto-start requests will be held by the CDB until a QMON or a QMON,S run is activated.
2	CDB ID CONSTANT VALIDATION FAILED Explanation: A MAPPER caller could not access all of the CDB due to an addressing error, or the Q-LINK CDB has been corrupted. Resolution: Check your MAPPER main buffer pool address limits versus your CDB configuration to determine if an address conflict should occur. If the problem persists see the action for error 1.
3	*** Error code not currently in use. *** Explanation: Resolution:

QLK External Function Errors

ERROR CODE	DESCRIPTION
4	MAPPER RELEASED BY QMON Explanation: A MAPPER run which was waiting for a server to act was released because the server aborted processing the request. Resolution: Check for server aborts either due to E-keyins or other operational events. Report unexpected server aborts via the procedure outlined for common bank error 1.
5	MAX REQUESTS QUEUED FOR SERVER CLASS Explanation: More than the configured maximum number of requests for this server class would have been outstanding had this request been queued. Resolution: Investigate why requests are not completing faster. Consider increasing the number of servers which can be active. The auto-start and auto-terminate configuration can make this painless, and minimize the impact on operations at other times of the day. Ensure that the number of servers which can be auto-started are actually starting promptly and successfully. Consider moving long running requests from this class into another class to allow short requests to complete. Consider the priority and execution type of the server. TIP requests generally (based on VALTAB) have higher priority than BATCH requests and TIP transaction type servers would usually be configured for higher priority than TIP batch-transaction type servers. Check other system bottlenecks for resource problems including CPU/memory/IO saturation and DMS run-unit queuing problems.
6	SERVER REQUESTED ABORT OF REQUEST Explanation: A run-time error in the Q-LINK program was detected by the server (other than maximum reply lines, maximum time or a security error). Resolution: Retrieve the server log (if not already returned for DEVELOPMENT mode servers) and inspect it for error messages defining the type and cause of the error.
7	SERVER CLASS QUEUE NOT IN MCT Explanation: The server class passed via the QLK function or the third specification field on the QLKSIM processor is not currently configured in the CDB. Resolution: Check the spelling and configuration of the specified class name. Use the QMON CONFIG command to verify that the proper configuration is resident in the CDB.
8	USER ABORTED FUNCTION WITH MESSAGE WAIT KEY Explanation: The Q-LINK server has detected that the user has depressed the message wait key (or the F4 key when running through MCB) and has terminated processing of this request. When running through the QLSIM program, the @@X C and the operator II-keyin will also terminate a request in this fashion. Resolution: None.

ERROR CODE	DESCRIPTION
9	ALL SERVER CLASSES LOCKED Explanation: The QMON program has been used to lock all request classes and prevent the queuing or acceptance of new requests. Resolution: Contact your system administrator to determine why Q-LINK access has been terminated.
10	THIS SERVER CLASS IS LOCKED Explanation: The QMON program has been used to lock this specific class and prevent the queuing or acceptance of new requests. Resolution: Contact your system administrator to determine why Q-LINK access has been terminated for this specific class.
11	ASSIGNED SERVER VANISHED! Explanation: The assigned server was no longer in the CDB tables when MAPPER returned for/with the next block of lines. Resolution: See error code 1.
12	BAD RID - OPEN I/P TO READ Explanation: A nonexistent input RID was specified on the @QLK function or QLKSIM utility call line. Resolution: Verify the desired MODE/TYPE/RID combination for the request.
13	RID # TOO BIG FOR TYPE ON OPEN Explanation: The input RID number specified is larger than the largest number configured in the MAPPER startup parameters. Resolution: Verify the desired MODE/TYPE/RID combination for the request.
14	LINE # TOO BIG ON OPEN I/P RID Explanation: The input RID did not contain even one line of data. Resolution: Verify the desired MODE/TYPE/RID combination for the request. Verify that the desired data is in the requested RID.
15	SYSTEM ERROR OPENING I/P RID Explanation: A MAPPER system error occurred trying to open the input RID. Resolution: Refer this problem to your MAPPER coordinator.

QLK External Function Errors

ERROR CODE	DESCRIPTION
16	ERROR OPENING O/P RID Explanation: A MAPPER system error occurred trying to open the output RID. Resolution: Refer this problem to your MAPPER coordinator.
17	ERROR WRITING OUTPUT RID Explanation: A MAPPER system error occurred trying to write to the output RID. Resolution: Refer this problem to your MAPPER coordinator.
18	ERROR CLOSING OUTPUT RID Explanation: A MAPPER system error occurred trying to finish writing the output RID. Resolution: Refer this problem to your MAPPER coordinator.
19	ERROR WRITING DATE LINE Explanation: A MAPPER system error occurred trying to write the date line to the output RID. Resolution: Refer this problem to your MAPPER coordinator.
20	ERROR WRITING HEADER LINE TO OUTPUT RID Explanation: A MAPPER system error occurred trying to write the header lines to the output RID. Resolution: Refer this problem to your MAPPER coordinator.
21	SYS MAX LINES REACHED Explanation: The Q-LINK user program was terminated when it tried to generate more lines of output than are permitted for this class. Resolution: Investigate how much data the request should be designed to return. Occasional large output can be allowed via the Q-LINK password on the @QLK function or QLSIM program call.
22	USER-SPECIFIED MAX LINES REACHED Explanation: The Q-LINK user program was terminated when it tried to generate more lines of output than were specified on the @QLK function or QLKSIM program call. Resolution: Investigate why the program attempted to return more data than the designer intended.

ERROR CODE	DESCRIPTION
23	SYSTEM ERROR READING I/P RID Explanation: A MAPPER system error occurred trying to read from the input RID. Resolution: Refer this problem to your MAPPER coordinator.
24	CDB ADDRESSING WINDOW PROBLEMS Explanation: The combination of the MAPPER main buffer pool sizing and the CDB installation selected base address has produced an overlap of the CDB address space and the MAPPER buffer pool address space. Resolution: Either MAPPER must be reconfigured, or the CDB must be reinstalled in such a way as to eliminate the addressing overlap. See also the action for common bank errors 11 and 12.
25	UNABLE TO TRMRG$ MAPPER ACTIVITY Explanation: The ER TRMRG$ performed by the CDB when registering a MAPPER activity has been rejected by the EXEC. Resolution: If the problem persists, submit an EXEC dump immediately following the error occurrence, along with the information requested for an internal Q-LINK error, to KMSystems, Inc., for analysis.
26	CDB FAILED INITIAL STATUS CHECKS Explanation: The CDB is improperly installed, is from an incompatible level of Q-LINK or MAPPER was improperly generated with regard to the CDB bank name, type and/or BDI. Resolution: Investigate the type and status of the bank installed for the Q-LINK CDB and the configuration of the MAPPER TCF and the status and completion log of the MAPPER generation used to install the @QLK function. For MAPPER 34 and higher, there is no QLK CDB parameter card defining the CDB BDI.. If the problem persists, submit an EXEC dump immediately following the error occurrence along with the information requested for an internal Q-LINK error to KMSystems, Inc., for analysis.
27	ERROR OPENING #INSERT RID Explanation: The Q-LINK function in MAPPER was unable to open the RID specified on the #insert directive. Resolution: Verify the desired MODE/TYPE/RID combination for the request.
28	ERROR READING #INSERT RID Explanation: An error occurred trying to read from the input RID. Resolution: Refer this problem to your MAPPER coordinator.

QLK External Function Errors

ERROR CODE	DESCRIPTION
29	SYNTAX ERROR ON #INSERT DIRECTIVE Explanation: The format of the #insert directive is incorrect. Resolution: Verify the format of the #insert command from the Q-LINK documentation. Any comment on the directive must be preceded by at least one space, a period and at least one additional space, and must follow all required fields.
30	SERVER CLASS MAXIMUM XQT TIME EXCEEDED Explanation: The Q-LINK user program was terminated when it tried to execute and consume more resources (SUPS) than are permitted for this class. Resolution: Investigate how long the request program should run. Consider processing less records during the request. Consider more efficient database location paths for the desired data. As a last resort, the execution time limit configured for the class can be increased, but this allows requests which might loop to consume unwarranted amounts of system resources. If requests requiring differing magnitudes of system resources are currently lumped together in one class, consider creating an additional class(es) each of which has an appropriate time limit for the type of processing required.
31	ACCESS NOT ALLOWED (#INSERT MODE EDITING) Explanation: The MAPPER run function attempted to access data within a MODE which is not allowed by the run registration RID for that run. Resolution: Verify the desired MODE/TYPE/RID combination for the #insert request. Request that your MAPPER coordinator update the run registration log for this run if you feel your need to access the data is legitimate. Alternatively, move the data to a MODE which the run can access.
32	TOO MANY CHAINED #INSERTS ENCOUNTERED Explanation: When a #INSERT is encountered in a RID being read with #INSERT, the first RID is closed, and the new RID is opened and read. Q-LINK does not keep a history of which RIDs have been processed in a chain of #INSERT commands and so cannot explicitly detect a situation where a RID contains a #INSERT which directly or indirectly references itself. To prevent MAPPER from looping forever in such a case, Q-LINK counts the number of #INSERT directives encountered while reading from a RID selected by another #INSERT. If the count of chained #INSERT commands exceeds ten, this error is issued. Resolution: Check the #INSERT commands in the data stream, and ensure that the maximum chaining of RIDs is not greater than ten. Also, ensure that the chain of #INSERT commands never references a RID present earlier in the chain.
33	SECURITY ERROR - NO ACCESS TO CLASS Explanation: Your MAPPER user group (configured in Q-LINK security) is not allowed access to the server class that you have selected. Resolution: Contact your site security officer to determine which server classes you can access.

ERROR CODE	DESCRIPTION
34	SECURITY ERROR - COMPILE NOT ALLOWED Explanation: Your MAPPER user group (configured in Q-LINK security) is not allowed to compile requests with this server class. Resolution: Contact your site security officer to determine which server class you can use to compile requests.
35	SECURITY ERROR - RUN NOT ALLOWED Explanation: Your MAPPER user group (configured in Q-LINK security) is not allowed to run requests with this server class. Resolution: Contact your site security officer to determine which server classes you can use to run requests.
36	QLKFUN IMAGE LENGTH ERROR Explanation: A bad ACW was transferred to QLKFUN (MAPPER) from the server. Resolution: Contact KMSystems, Inc.
51 through 81	COMMON DATA BANK DIRECTED ERRORS Resolution: For error codes 51 through 81, subtract 50 from the error code and use the result to reference the errors in Appendix JAppendix K, "Common Bank Errors"; i.e., @QLK error code 68 can be found in Appendix JAppendix K as error code 18 (68 - 50 = 18), "INSUFFICIENT PRIVILEGE FOR REQUEST".

QLK External Function Errors

Appendix J Q-LINK Common Bank Errors

The following is a list of error codes, explanations of errors and possible error resolutions for all errors reported by QMON, Q-LINK servers and other Q-LINK utility programs.

ERROR CODE	DESCRIPTION
1	SERVER TABLE NOT FOUND Explanation: Internal error. A server run-id was not found in the CDB when it was expected to be there. Resolution: Take a dump of the common bank immediately following the error using the QMON DUMP command. Take a dump of the utility program reporting the error using @PMD. Initialize the common data bank using the QMON INIT command and try again. Submit the dumps and a listing of your configuration (both generation and run-time) to KMSystems, Inc., for analysis.
2	UNKNOWN FUNCTION REQUESTED Explanation: Internal error. An unused or out of range function code was passed to QCOM (part of the Q-LINK server program) on a request call. Resolution: See error code 1.
3	INIT REQ AND SERVER ALREADY ACTIVE Explanation: Internal error. A server initialization function was received by the CDB from a run-id which was already marked as active in the CDB tables. Resolution: See error code 1.
4	MAXIMUM NUMBER OF SERVERS ACTIVE Explanation: Configuration or operational error. The number of servers allowed to be active concurrently for all classes is too small for the number of servers attempting to connect to the CDB. The run-time configuration for maximum servers active needs to be as large as the sum of the maximum active servers for all classes. Resolution: Change your run-time configuration to reduce the number of servers allowed for some classes, or increase the maximum number of servers globally allowed active by the CDB.

Q-LINK Common Bank Errors

ERROR CODE	DESCRIPTION
5	QMON IS SHUTTING DOWN THIS SERVER -OR- SERVER WAS ABORTED (E-KEY, ...) -OR- Q-LINK TIMER ACTIVITY IS SHUTTING DOWN SERVER Explanation: This condition is expected by Q-LINK and, as such, should not appear as an error; i.e., the CDB is requesting a server to exit. Resolution: Investigate QMON EX and TERM command usage at your site. Ensure that the run-time configuration for the time a server is to remain idle before exiting (DURATION SGS) is as large as you would like it to be.
6	MAXIMUM SERVERS ALREADY ACTIVE FOR CLASS Explanation: Configuration or operational error. The number of servers allowed to be active concurrently for this class is too small for the number of servers attempting to connect to the CDB. The only time this should happen is if someone is manually starting servers, or if auto-started servers were held for some reason while one or more QMON RESET or INIT commands were done. Resolution: If you really want more servers active, update the run-time configuration through COMUS. If not, investigate why servers are being delayed in starting, or who is manually starting extra servers.
7	NO CLASS QUEUE SLOT FOUND Explanation: A server attempted to initialize with a class name which was not in the CDBs current configuration. This could be caused by a misspelled server’s class name on the @QLK command in a manually started batch server. This could also occur if the CDB was initialized with a configuration not containing this class name. Since class names cannot be dynamically added to the CDB, you can successfully process a configuration update but still not have access to the new server class. Resolution: Verify that the server name requested is really in the current COMUS configuration. Verify what classes are currently available in the CDB by using the QMON CONFIG command. If you need to add the class to the CDB, you must shut down the Q-LINK system using the following commands to QMON: LOCK, ALL, TERM and then initialize the Q-LINK system using the QMON INIT command.
8	ABORT OF A REQUEST SIGNALED REQUESTED FROM MAPPER -OR- MAPPER HAS ABORTED (E-KEY, . . .) Explanation: This error is not fatal! Someone has externally requested the termination of the current request for this server. Possible cause includes a shutdown of the MAPPER system which submitted this request to the server. Resolution: Investigate the MAPPER down/purge times as well as any manual shutdowns of MAPPER or any cases of MAPPER aborting.

ERROR CODE	DESCRIPTION
9	**** Error code not currently in use. **** Explanation: Resolution:
10	ERROR LOCATING OR READING CONFIG ELT Explanation: All the Q-LINK utility programs and Q-LINK batch servers locate the run-time configuration by finding an absolute element called QLNK$CFIG in the file from which they were executed. The TIP servers are passed the filename from which the CDB configuration was initialized, and they expect the QLNK$CFIG absolute to be in that file. MAPPER has the CDB type and specified by a parameter card and doesn't need access to the configuration absolute. If the configuration element cannot be found or if it is badly formatted or if an I/O error occurs when attempting to read it, then this error is reported. This can be caused by a hardware or software damaged execution file, by copying a Q-LINK program into a different file (i.e., TPF$) and not copying the configuration element also, or by attempting to process a configuration element produced by a different level of Q-LINK. Note that immediately following a Q-LINK generation and/or install, the installation file (Q5 or SYS$LIB$*QLINKx) contains a dummy QLNK$CFIG element which cannot be loaded by the Q-LINK programs.
11	HIGHEST ADDRESS REQUIRED FOR CDB IS GREATER THAN 0777000 (1ST PCT BLK ADDR) Explanation: The run-time configuration of maximum servers active and number of classes defined, combined with the base address for the CDB chosen at install time, has produced a CDB whose highest address exceeds 0777000. Resolution: Reduce the upper address limit of the CDB by reducing one (or more) of the following: CDB base address (INSTALL), maximum servers concurrently active (CONFIGURE), or number of defined classes (CONFIGURE). Remember not to reduce the base address of the CDB below the highest address of MAPPER's main buffer pool or intermittent addressing status errors may be returned to @QLK users. If you are running a very large configuration and also running multiple MAPPER's or InfoQuest batch servers, you can split these to use a second (or more) CDB although it seems extremely unlikely that anyone would have a configuration this large.
12	Q-LINK UNKNOWN ERROR Explanation: Resolution: Contact KMSystems, Inc..
13	UNABLE TO DO AN ER TO TRMRG$ Explanation: The ER TRMRG$ performed by the CDB when registering a server or QMON run has been rejected by the EXEC. Resolution: If the problem persists, submit an EXEC dump immediately following the error occurrence, along with the information requested for an internal Q-LINK error, to KMSystems, Inc., for analysis.

Q-LINK Common Bank Errors

ERROR CODE	DESCRIPTION
14	ALL CLASSES LOCKED/UNLOCKED Explanation: A QMON LOCK (UNLOCK), ALL directive was performed when the global request lock was already set (clear). Resolution: Perform a QMON STATUS keyin prior to issuing the LOCK/UNLOCK command. Investigate the usage of the QMON LOCK/UNLOCK command at your site.
15	QUEUE CLASS NOT FOUND Explanation: A QMON LOCK (UNLOCK) class directive was performed for a class whose name is not currently in the CDBs configuration tables. Or, an attempt was made to update the configuration by adding a class name to the CDB dynamically. Or, by some strange timing quirk, a class name was removed from the CDB (by a QMON INIT command) while a second QMON was auto-starting a TIP MCB connected server. Resolution: Verify the currently active configuration using the QMON CONFIG command. If you have added a class by the configure process, but the CDB was already initialized, you must use the QMON TERM and INIT commands to allow the reloading of the configuration data into the CDB.
16	SERVER INDEX OUT OF RANGE Explanation: An invalid server index was used on a QMON EX keyin. Resolution: See error code 1.
17	QMON ALREADY ACTIVE Explanation: Only one run can be registered for the QMON privileged commands at any one time. Resolution: Use the ‘S’ option on one or all of the QMON executions. The QMON copy which received this error automatically acts as if the ‘S’ option were specified. The QMON STATUS keyin will display the run-id and start time of the currently registered privileged QMON run.
18	INSUFFICIENT PRIVILEGE FOR REQUEST Explanation: Certain CDB requests require that you be signed on under one of a specific list of user-ids before you are allowed the privilege associated with that request. Resolution: Check the list of user-ids configured in the security SGSs for utility security. Check the user-id of the offending run to make sure that it is configured. If the user is a TIP transaction server, its pseudo user-id will consist of the program name prefixed with at ‘$’ character; i.e., the QLINK transaction program requires configuration of a $QLINK user-id.

ERROR CODE	DESCRIPTION
19	MUST HAVE QMON UNLIMITED PRIVILEGE FOR REQUEST Explanation: Certain CDB requests require that the caller be the privileged QMON user registered with the CDB. If a non-privileged user requests such a function (reload the CDB, etc.), it will be rejected. Resolution: The ‘S’ option on QMON prevents registration as the privileged user and can lead to this error if privileged commands are then attempted. As pointed out for error 17, if a user is not the first to attempt to register for these privileges, it is as if he used the ‘S’ option when starting QMON, and that can lead to this error. Check the QMON privileges for the user in the security SGSs.
20	BANK NOT YET LOADED Explanation: Only the QMON program can initialize the CDB following a COMUS installation, a system reboot or a console reload bank keyin. If any other caller attempts to link to the CDB prior to QMON beginning the bank initialization, the caller will be rejected. Resolution: The QMON program must be executed in some fashion before starting servers or other utility programs. MAPPER can be started before the CDB is initialized, but cannot access Q-LINK until the CDB is initialized. An @SYS$LIB$*QLINKx.QMONx ,L in either the system startup runstream or the MAPPER startup runstream or the server startup runstream will allow the common bank to be loaded in an automatic fashion. Be sure to use the QMON and the QLNK$CFIG element from the current installation file as whatever configuration is in the execution file will be the one which is loaded.
21	TEMPLATE BANK ILL-FORMED -OR- REQUESTER NOT IN QUARTER-WORD MODE Explanation: Either the CDB absolute has been damaged by some software or hardware malfunction, or the CDB does not match the level of the Q-LINK utility/server program being executed. Resolution: Verify the level of Q-LINK used when installing the CDB and the level of Q-LINK being used for execution. If all else fails, deinstall the Q-LINK software and then repeat the installation of the Q-LINK software with COMUS.
22	CDB CONTINGENCY (INCL RELOAD) Explanation: Some contingency has occurred which the CDB is not prepared to handle. This will occur if someone reloads the CDB using the console keyin or a privileged reload program. Other errors constitute an internal error in the Q-LINK system. The contingency packet information is returned to the caller in A1 and A2. Resolution: Investigate the usage of reload keyins/programs affecting the Q-LINK CDB. Check for installation of other software using COMUS which uses the same BDI or bank name as the Q-LINK CDB. If the problem persists, see the action for error 1.

Q-LINK Common Bank Errors

ERROR CODE	DESCRIPTION
23	ILLEGAL CB FUNCTION Explanation: Internal error. An unused or out of range function code was passed to the CDB in A0. Resolution: See error code 1.
24	SERVER(S) ARE ACTIVE Explanation: Attempt to terminate Q-LINK with the QMON TERM command while servers were processing a request. Resolution: Use the QMON LOCK, ALL command to prevent new requests from starting. Once all active requests have finished, retry the TERM command.
25	A MAPPER RUN WAS RELEASED Explanation: Attempt to initialize the Q-LINK CDB with the QMON INIT command while a MAPPER run was waiting for a server activity to process. The MAPPER run has been released. Resolution: The INIT command should be retried after first using the QMON LOCK, ALL and allowing active requests to terminate.
26	NO MAPPER RUN TO RELEASE Explanation: A MAPPER run was to be released by the CDB and no MAPPER run was waiting for the specified server. Resolution: See error code 1.
27	SERVER NOT CLOSED; I.E., IS BUSY Explanation: The server specified on an EX/TERM keyin could not be shutdown since it was not idle. Resolution: Lock the class queue using the QMON LOCK command and then wait till the server is idle before terminating it.
28	SERVER NOT CLOSED; I.E., IS TERMINATED/ABORTED Explanation: The server specified on an EX/TERM keyin could not be shutdown since it is in a termination or abort state. Resolution: If this condition does not clear shortly, then this is an internal error and should be reported as for error code 1.

ERROR CODE	DESCRIPTION
29	AN IDLE SERVER WAS RELEASED Explanation: This status is used between the CDB and QMON as part of the TERM keyin processing. A message is issued to the user and the request will be retried. Once all idle servers are terminated, QMON will finish. Resolution: None.
30	BAD BASE/RETURN ON LBJ Explanation: A caller to the MCB had invalid information captured by the hardware related to the BDR on which the CDB is based and the return address portion of the linkage register. Resolution: See error code 1 if this error persists.
31	USER BUFFER ACW INVALID Explanation: A caller to the CDB passed an invalid buffer length/address for a function requiring data transfer between the program's DBANK and the CDB data areas. Resolution: See error code 1.

Q-LINK Common Bank Errors

Appendix K InfoQuest Messages

The following is a list of messages produced by InfoQuest, their explanations and level of severity:

RID Nbr	M/W Nbr	Message or Warning / Description	Level of Severity
4	0001	**Tab to your choice and use TRANSMIT key.**	INFO
		This message is informational. Some of the menus require tabbing to a position and transmitting.	
4	0002	**Option XX is not authorized - Check registration.**	INFO
		This message results from a user trying to make a file. A user needs to be authorized in InfoQuest to generate a file. A user is authorized to make a file via user registration.	
4	0003	**Qualifier is missing.**	Error-Non fatal
		This message results from the user not entering a name of a qualifier in the field <Enter Qualifier> on the File Creation Menu. If a user is creating a file, InfoQuest does not continue unless a qualifier with up to 12 characters is entered.	
4	0004	**FILENAME is missing.**	Error-Non fatal
		This message results from the user not entering a name of a file in the field <Enter FILENAME> on the File Creation Menu. If a user is creating a file, InfoQuest does not continue unless a filename with up to 12 characters is entered.	
4	0005	**File is being created - Please Stand By.**	INFO
		This message indicates to user that a file is being made on the operating system. When the file is made, InfoQuest continues to the next step.	
4	0006	**Q-LINK error while making file.**	Error-Non Fatal
		This message results from the system having problems making the file. The user needs to contact the administrator and give the him/her the QUALIFIER*FILENAME that the user was trying to create. Q-LINK encountered a problem trying to create the file.	
4	0007	**Output file is unavailable.**	Error-Non Fatal
		This message results from the system having problems making the file. The user needs to contact the administrator and give the him/her the QUALIFIER*FILENAME that the user was trying to create. Q-LINK either encountered a rollout, file in use or not enough system.	

RID Nbr	M/W Nbr	Message or Warning / Description	Level of Severity
4	0008	**No items were selected to preview headings.**	INFO
		The message results from a user depressing the F8 function that displays the headings when the user has not selected any items for a report.	
4	0009	**Problem in generating headings.**	Error-Non Fatal
		A problem was encountered while generating the headings. The user should note the items selected and contact the administrator to report the problem.	
4	0010	**NO items were selected - Next Step (F3) not allowed.**	Error-Non Fatal
		The user did not select any items for the request while in the item Add/Delete menu and depressed (F3) to go to the next step. InfoQuest does not go to the next step without selecting any items for the request.	
4	0011	**Derived item(s) dropped due to existing user name.**	Warning-Non Fatal
		The user defined a derived item with the same user name that already exists. The original item remains but the latest duplicate is dropped.	
4	0012	**Report is X columns too BIG-256 or < is acceptable.**	Warning-Non Fatal
		The user marked too many items for the report request. Reports can be up to 256 characters wide. This message is displayed when a user makes a report in excess of 256 characters.	
4	0013	**Sort Orders exceeds 9 levels.**	Error-Non Fatal
		The user has up to nine levels of sort 1 through 9. Any other level is not supported.	
4	0014	**Sort needed to Subtotal.**	Error-Non Fatal
		If the user tabs to the Subtotal selection on the Review Menu (RMENU), this message will display at the bottom and not display the Subtotal screen because there was no sort selected on any item in the request.	
4	0015	**A,T,H,L,P,1,2,3,4 options are for numeric only.**	Error-Non Fatal
		This error message appears when the user marks an alphanumeric item with one of these numeric options.	
4	0016	**User is not authorized for File Creation.**	Error-Non Fatal
		This error message appears when the user marks the Report Print Options field on the Special Print and Run Options Menu with a Y or K and the user is not authorized to create files via the User Registration. The Y and K entries generate print files on the system.	
4	0017	**Option Valid only with Subtotal Page Break.**	Error-Non Fatal
		This error message appears when the user marks the Prt Subtotal Non-Print Item in header field with a Y and does NOT mark the Subtotal Page Break field.	
4	0018	**Time is not within 0001-2359 hours.**	Error-Non Fatal
		This error message appears when a user enters an out-of-range hour on the Special Report and Run Options Menu.	

RID Nbr	M/W Nbr	Message or Warning	Level of Severity
		Description	
4	0019	**Request name exists - Please enter another.**	Error-Non Fatal
		This error message appears when a user tries to name a request that uses that name in the directory. User must enter a unique name.	
4	0020	**Can not ASG config file.**	Error-Fatal QLK
		Q-LINK returned a status 9001. The SYS$LIB$*INFOQ$CONF file is unavailable either assigned by another session or not on system. See the InfoQuest technical contact.	
4	0021	**Can not ASG RUN control file.**	Error-Fatal QLK
		Q-LINK returned a status 9002.	
4	0022	**Configuration nn not found.**	Error-Fatal QLK
		The system number for InfoQuest has not been established. The user must contact the InfoQuest Administrator to verify its existence. Q-LINK returned a status 9005.	
4	0023	**No Requests available for view.**	Warning-Non Fatal
		This message appears on the bottom line of the Activity Menu. A user selected a view and wanted to access an existing request. But there are no requests for that user to access. The only activity the user can do is create a new request.	
4	0024	**Occurs Item not in OCCURS TBL.**	Error-Fatal QLK
		The user selected an item that is an occurs item, but the internal definition does not have it defined as an occurs item. The user must contact the InfoQuest technical support. The technical support needs to examine the VIEW and the QINDEX file. Q-LINK returned a status 9003.	
4	0025	**Occurring Item not in OCCURS TBL.**	Error-Fatal QLK
		The user selected an item that is within an occurs item, but the table definition does not have it defined. The user must contact the InfoQuest technical support. The technical support needs to examine the VIEW and the QINDEX file. Status 9004 is returned from Q-LINK. Q-LINK returned a status 9004.	
4	0026	**Request is in error.**	Error-Fatal
		The user selected a request to reuse, and there is a problem in accessing it. The user needs to contact the InfoQuest Administrator. The administrator needs to examine the parameter RID.	
59	0027	**This item is a KEY. No ranges allowed.**	Warning-Non Fatal
		The user entered a range into the search values. The item is a KEY which only allows TARGET entry. InfoQuest ignores the value in the THRU field and processes the field in the VALUE field for TARGET entry.	
59	0028	**Missing Conditional - Correct and Retry.**	Error-Non Fatal
		The user entered more than one conditional and left out the AND or OR.	

RID Nbr	M/W Nbr	Message or Warning / Description	Level of Severity
59	0029	**No Predefined List found for <value>.**	Warning-Non Fatal
		The user was using a predefined search list from the search values screen. The list that the user accessed did not have the item in its list. The user has two options at that point which are either to enter the values manually or back out and not select on item.	
62	0030	**Invalid edit code - Please re-enter and xmit.**	Error-Non Fatal
		The user was in the user reporting section (via Special Reporting) and tried to change an edit code to an invalid edit code. See Section 12.1.20, “Modify Report Output Format”, in the InfoQuest User Handbook or Section 4.6, “View Maintenance”, in the InfoQuest System User Guide for a complete list and description of valid edit codes.	
62	0031	**Argument is undefined.**	Error-Non Fatal
		This message appears when a user does not enter a valid item in any calculation or conditional. An argument can be either an item selected for a request, reserved word, number, or result item (R1-R6).	
4	0032	**Error in Free-Format code.**	Error-Fatal QLK
		Q-LINK returned a status 9000.	
4	0033	**Can not ASG TEMP prompt file.**	Error-Fatal QLK
		Q-LINK returned a status 9006.	

Index

Index

If you would like to help us make our documentation better, please take a few moments to complete this form and return it to KMSystems. We are always looking for ways to improve our products and your feedback will go a long way toward helping us reach our goal.

Name ____________________

Company ____________________

Address ____________________

____

____

City __________ State/Province __________

Country __________ Zip/Mail Code __________

Document Name __________ Operating System Level __________

KMSystems Product __________ Level __________

Please rate the documentation on a scale of 1 to 5:

	5	4	3	2	1	
Complete	O	O	O	O	O	Incomplete
Accurate	O	O	O	O	O	Inaccurate
Usable	O	O	O	O	O	Unusable
Readable	O	O	O	O	O	Unreadable
Understandable	O	O	O	O	O	Unintelligible
Attractive	O	O	O	O	O	Unattractive
Excellent	O	O	O	O	O	Poor

What information did you expect to find that was omitted?

Is more information needed? O yes O no. If yes, on what topic?

Did you find factual errors in the documentation? O yes O no. If yes, please give page number and description of the error.

If the documentation is difficult to understand, please specify page number and problem description.

Is the documentation intimidating? O yes O no.

Are the manuals: O too long? O too short? O about the right length?

Other suggestions or comments? (Use back of form if necessary.)

KMSystems, Inc.

(Additional Comments)

. . . . *Fold along dotted line.* . . . .

____

____

____

____

To:

KMSystems, Inc.

3225 Shallowford Road
Suite 1000
Marietta, GA 30062
U.S.A.

Attn: Technical Documentation Section