SUPERDEX
PowerHouse Interface

Demonstration Manual

Version 3.1

Released March, 1992

Table of contents

About this manual

This manual, used in conjunction with the demonstration database and programs supplied, gives you an introduction to SUPERDEX's interface to QUICK[TM] and QUIZ[TM], both components of the PowerHouse[TM] 4GL package. Although an interface to QTP[TM] is also available, it is implemented in the same way as the QUIZ interface and therefore is not demonstrated in this manual.

These demos let you experience various SUPERDEX retrievals in conjunction with QUICK and QUIZ using search criteria you provide.

No knowledge of the SUPERDEX package nor QUICK or QUIZ is assumed for this demonstration.

This manual is arranged in the following format:

Section 1 gives an *Introduction* to the demonstration package and explains how to set up the demo environment.

Section 2 describes SUPERDEX and its interface to QUICK by leading you through the interactive *QUICK interface demo* programs provided, allowing you to experience SUPERDEX's powerful retrieval capabilities and amazing speed first-hand. Data values are suggested, but you are given complete freedom in what search values you supply. An explanation of how the demo works, including data structures and program operation, is given for each demo. These QUICK demos emulate the SUPERDEX COBOL demo programs.

Section 3 reviews the *QUIZ interface demo* and illustrates its capabilities using QUIZ programs that perform the same types of retrievals as the QUICK demo programs.

Appendix A shows the *Data dictionary layout* used for the QUICK demos.

Appendix B contains listings of the *QUICK source programs* used in the QUICK interface demos in Section 2.

Appendix C contains listings of the *QUIZ source programs* used in the QUIZ interface demos in Section 3.

Section 1	Introduction

Features

This demonstration facility gives you the ability to interactively experience SUPERDEX's enhanced data retrieval capabilities from QUICK and QUIZ programs, including:

- multiple keys in master and detail datasets
- concatenated keys containing multiple fields
- sorted sequential retrieval
- automatic keywording and keyword retrieval
- generic and partial-key retrieval
- grouping of functionally-equivalent fields
- multiple value lookup
- relational access across multiple fields, datasets, and databases

QUICK demonstration programs

These features are shown by use of five QUICK programs for which **POSTPATH** procedures have been written for SUPERDEX access. They use the same methods that you would use in your SUPERDEX'ed QUICK programs.

Listings of the demonstration source programs are included in Appendix B of this manual as well as in files in DEMO.SUPERDEX.

QUIZ demonstration programs

SUPERDEX capabilities are also shown by use of five QUIZ programs for which each **SELECT** command has been replaced by a new command that performs SUPERDEX access. They use the same methods that you would use in your SUPERDEX'ed QUIZ programs.

Listings of the demonstration source programs are included in Appendix C of this manual as well as in files in DEMO.SUPERDEX.

Demonstration database

A small demonstration database is provided to facilitate the interactive demos. It is a partial order entry database (called **OEDB**) which contains only four datasets that are used as follows:

CUSTOMERS   Standalone manual master containing 1000 customer entries; IMAGE search item is CUSTOMER-NUMBER.

ORDER-HEADERS Manual master containing 2620 order headers; IMAGE search item is ORDER-NUMBER.

ORDER-LINES   Detail dataset, related to ORDER-HEADERS, containing 10245 line items related to order headers; IMAGE search item is ORDER-NUMBER.

SI   Standalone detail dataset in which all SUPERDEX index structures are maintained. Contains only the item **SI**.

A complete database layout is contained in Appendix A of the Superdex demonstration manual.

Demonstration data dictionary

The data dictionary has been modified to facilitate SUPERDEX access via QUICK by adding a dummy standalone master dataset FILE and its ELEMENTs, RECORDs, and ITEMs. No changes were required to any other file, record, or element statements.

No data dictionary changes are required for the QUIZ demos.

A complete data dictionary listing is contained in Appendix B of this manual.

Database access

Although the entries in this database could be accessed by their IMAGE search items using conventional IMAGE access techniques, this demonstration utilizes SUPERDEX access techniques only.

Loading the software

First, load the SUPERDEX software from the installation tape, following the separate *SUPERDEX loading instructions*.

Log on:

```
:HELLO MGR.SUPERDEX,DEMO
```

Then, place a copy of the regular QUICK and QUIZ programs into DEMO.SUPERDEX:

```
:FCOPY FROM=QUICK.CURRENT.COGNOS;TO=QUICK;NEW
:FCOPY FROM=QUIZ.CURRENT.COGNOS;TO=QUIZ;NEW
```

☞ **These program copies are used for demonstration purposes only; for normal operation of the SUPERDEX PowerHouse interface, these programs are run from the COGNOS account with added capabilities.**

Finally, add DS capability to both the QUICK and QUIZ programs in DEMO.SUPERDEX using one of the following methods:

■ Use ALTPROG.PUB.SUPERDEX
■ If you have VESOFT's MPEX/3000™ utility, use the **ALTFILE** command (e.g. **ALTFILE @.@.account (CODE="PROG"); CAP=+DS**)

Once you have completed these steps, you are ready to run through the demonstrations.

Running the demonstrations

To run the QUICK demonstration programs, type

```
:QUICKDEMO
```

at the MPE colon prompt (do not type the **:**).

The following menu is displayed:

```
MODE:  ACTION:_____

                      Bradmark Technologies, Inc.

                          S U P E R D E X

                           Demonstration


                  01  Simple Key Demo

                  02  Concatenated Key Demo

                  03  Keyworded Key Demo

                  04  Grouped Key Demo

                  05  Relational Access Demo




SUPERDEX is a trademarked product name of Bradmark Technologies for the SI IMAGE
package developed and implemented by Dr. Wolfgang Matt
```

Five separate demonstration QUICK programs may be run from this main menu, and are described on the following pages.

Simple key demo

About the demo

A *simple* SUPERDEX key is very much like an IMAGE search item except that its capabilities are extended in various ways, including the support of

- sorted sequential retrieval
- generic and partial key retrieval
- less-than or equal, greater-than or equal, not equal, and range retrieval

The Simple Key Demo illustrates the use of a simple *Sl-key* (SUPERDEX key) to locate customer entries in the CUSTOMERS master dataset.

Running the demo

Type

1

in the ACTION field and press RETURN to proceed to the Simple Key Demo.

The following screen is displayed:

```
MODE:F ACTION:                Simple Key Demo

Enter customer name _____         Direction (F,B) _
===============================================================================
Customer Name            Customer #
```

The highlighted field is for the customer name to search for; type

UNITED AIRLINES

and press RETURN.

Now, the highlight moves to the Direction field, which specifies whether entries should be returned in ascending or descending alphabetical order (**F** for forward or **B** for backward). Enter

F

to return entries in ascending order, and press RETURN.

The specified customer is displayed:

```
UNITED AIRLINES             2112949
```

The ACTION field is now highlighed. Enter

   F

to perform another **F**ind, and press RETURN.

This time, specify a customer of

   UNITED@

and press RETURN. Then, enter

   F

again in the Direction field, and press RETURN. All entries that start with "UNITED" are displayed:

```
UNITED AIRLINES            2112949
UNITED ALLOYS & STEEL      2100649
UNITED BUSINESS EQUIPMENT  2100652
UNITED CEREBRAL PALSY ASSN 2100400
UNITED CHURCH HOME         2100304
UNITED FUND BUFF & ERIE    2100401
UNITED IMPORT MOTORS INC   2100700
UNITED PRESB CHURCH        2100509
```

The @ character tells SUPERDEX to match zero or more characters in the position where specified, just like MPE's :LISTF command; (unlike :LISTF, characters following the @ are ignored). If you specified a customer name of just @, SUPERDEX would retrieve 1000 entries -- all the entries in the CUSTOMERS dataset.

Note that entries are displayed in ascending alphabetical order. SUPERDEX automatically returns entries in sorted sequential order, either ascending or descending.

Let's look at a descending order retrieval using a new wildcard. Type

   F

and RETURN again (and from now on) to do another **F**ind and specify a customer name of

   UNI?E@

and enter in the Direction field:

   B

and press RETURN.

All the entries that start with "UNI" and contain an "E" in the fifth position followed by any number of any character are displayed:

```
UNIVERSITY BOOKSTORE          2100606
UNITED PRESB CHURCH           2100509
UNITED IMPORT MOTORS INC      2100700
UNITED FUND BUFF & ERIE       2100401
UNITED CHURCH HOME            2100304
UNITED CEREBRAL PALSY ASSN    2100400
UNITED BUSINESS EQUIPMENT     2100652
UNITED ALLOYS AND STEEL       2100649
UNITED AIRLINES               2112949
```

The **?** matchcode is used as a place-holder, representing a single alphanumeric character (like in :LISTF). Note that entries are returned in descending order, as specified.

Now, go back into **F**ind mode and specify a **customer name** of

>**=UN@<=UNI@**

and a **Direction** of

B

and press **RETURN**. This locates a range of entries starting with "UN" through "UNI", inclusive:

```
UNIVERSITY BOOKSTORE
UNITED PRESB CHURCH
UNITED IMPORT MOTORS INC
UNITED FUND BUFF & ERIE
UNITED CHURCH HOME
UNITED CEREBRAL PALSY ASSN
UNITED BUSINESS EQUIPMENT
UNITED ALLOYS & STEEL
UNITED AIRLINES
UNITARIAN CHURCH
UNDERWRITERS SALVAGE CO
```

Further demonstration

You are beginning to see the power, flexibility, and ease-of-use of SUPERDEX SI-keys. You may try out additional values to further experiment with simple SI-keys, and also use the following new operators in your values:

>=*value*   greater-than or equal-to retrieval
<=*value*   less-than or equal-to retrieval
<>*value*   not-equal-to retrieval

There are two methods which you may choose from to exit this (and the following) demostration screens and return to the Main Menu.

The first (which will be considered the default for this demo manual) is to type in "**KL**" and then press RETURN to set the default Quick function keys. This will default f8 to exit the current screen.

The second method is to place a "**^**" in the entry field and press RETURN to exit the current screen.

How the demo works

Although SUPERDEX offers amazingly fast and powerful retrievals, it is surprisingly easy to implement into QUICK programs.

A simple **POSTPATH** procedure is added to set up the SUPERDEX access parameters and causes an indexed-sequential SUPERDEX retrieval to be performed instead of an IMAGE serial read. (As mentioned in the last section, it was first necessary to add a dummy master dataset and its elements into the data dictionary.)

The retrievals in this demonstration were accomplished by accessing SUPERDEX index structures contained in a special standalone detail dataset named **SI** in the database. Each unique relationship is referred to as an SI-path, and is accessed in very much the same way as accessing an IMAGE path. In this demo, an SI-path exists for customer name.

The program contains the statement **FILE SIPROC DESIGNER** to facilitate SUPERDEX access and imposes SEQUENTIAL access on the dataset FILE because no IMAGE keys--only SUPERDEX SI-keys--are being used for access.

The program **ACCEPT**s the customer name and direction values and then uses them to set the four parameters for a new item called **SIPARMS**. These parameters are used to describe the SUPERDEX access performed by the GET. For **SIPARMS** in this program, the *SIDSET* parameter was set to **CUSTOMERS** (the name of the dataset), the *SIPATH* to **CUSTOMER-NAME** (the name of the SI-path), and the *SIARG* to the customer name specified. Depending on the direction specified, the *SIMODE* was set either to **1** (ascending order) or **2001** (descending order).

SUPERDEX's *SIMODE* 1 accepts *SIARG* values that contain special operators, such as @ and ?. In this program, the customer specified is being passed as the *SIARG* used for the GET. The entries are retrieved in ascending or descending sorted order, as specified by the *SIMODE*, where *SIMODE* 1 returns entries in ascending order and, by adding 2000 (therefore *SIMODE* 2001), changes to descending order.

After the **SIPARMS** is established, **GET SIPROC USING SIPARMS OPTIONAL** is called in the POSTPATH procedure, and the subsequent serial retrieval done by **GET...SEQUENTIAL** in the default QDESIGN-generated FIND procedure is automatically redefined as a SUPERDEX retrieval.

A complete copy of the source program appears in *Appendix B*, as well as in the file QSK.DEMO.SUPERDEX.

Concatenated key demo

About the demo

A *concatenated* SI-key consists of the values of two or more fields concatenated together. This not only permits entries to be located by the combination of values for the various concatenated fields-- thereby avoiding lengthy chained reads--but also imposes extended sorting capabilities.

This demo shows

- concatenated keys containing multiple fields
- extended sorted sequential retrieval

The Concatenated Key Demo illustrates the use of a concatenated SI-key to locate order line items in the ORDER-LINES detail dataset.

Running the demo

Select option **2** from the Main Menu and press **RETURN** to proceed to the Concatenated Key Demo.

The following screen is displayed:

```
MODE:F ACTION:                    Concatenated Key Demo

Order Number _____   Part Number _____
==================================================================================
Order Number  Part Number     Part Description




```

The highlighted field is for the Order Number to search for; type

701257

and press **RETURN**.

Now, the highlight moves to the Part Number, which specifies the part number contained in each order line item. Entries must match on both fields in order to qualify. Specify the Part Number

SCM1511

in the second field. When you press **RETURN**, SUPERDEX'ed QUICK returns the corresponding entry:

```
701257  SCM1511         COPYSETS CANARY CA9B  1065
```

With the capability of specifying values for both fields, we were able to avoid a lengthy chained read of the order's chain.

As seen in the Simple Key Demo, SUPERDEX'ed QUICK supports partial key retrievals, but in this case, no @ is required (the reason is explained later).

Go into **Find** mode and specify the same `Order  Number` as before (**701257**) but change the value in the second field to

　　SCM

and press RETURN. All entries with the specified order number and part numbers starting with "SCM" are displayed:

```
     701257    SCM1312        FOLDER MANILA LTR 1/  1120
     701257    SCM1511        COPYSETS CANARY CA9B  1065
     701257    SCM153-ST      PADS TELEPHONE MESSA  1250
     701257    SCM835-ST      PAD SCRATCH 3X5 9120  1230
     701257    SCM858-ST      PAD SCRATCH 5X8  912  1235
     701257    SCM870         PAD STENO GREGG RULE  1240
     701257    SCM8784        PAD STENO PITMAN RUL  1245
     701257    SCM9014-ST     PAD LEGAL CANARY PER  1215
     701257    SCM911-ST      PAD LETTER CANARY 8-  1210
     701257    SCMA1312       FOLDER MANILA LGL 1/  1125
```

Note that entries are displayed in ascending alphabetical order by both order number and part number. This is because all values contained in a concatenated SI-key are used for sorting purposes, permitting extended sorting by multiple fields to be accomplished without the use of sorted chains. In fact, SUPERDEX concatenated SI-keys permit sorted chains to be eliminated entirely, averting the potential performance problems while permitting more flexible sorting.

Further demonstration

You may try out additional order number and part number combinations. Because of the way the program is written, you must specify a full order number in the first field but may specify partial part numbers of any length in the second field or leave it blank.

Specify same the order number, **701257**, and leave the `Part  Number` field blank and press RETURN. Then, try varying the part number and see the results. Also try the order numbers **915066**, **711155**, and **929461** with various part numbers.

Press **f8** when you are done to return to the Main Menu.

How the demo works

This program accesses an SI-path that represents a concatenated SI-key, which is comprised of the order number concatenated with the part number.

This program performs a partial-key retrieval on part number without the use of an @ in the *SIARG* (the method used in the Simple Key Demonstration). Instead, the partial-key retrieval is accomplished using a special *SIMODE* that restricts the number of characters on which SUPERDEX matches.

In the demonstration database, the order number is stored in an I2 item and the part number is stored in an X14 item, having a combined length of 18 bytes. Accordingly, two separate parameters in **SIPROC** are used to represent the two values: *SIARG1* and *SIARG2*, rather than *SIARG*. (Both were also configured in the data dictionary.)

For the last retrieval performed, it was necessary to match on the specified value for the first item but only the first 4 bytes of the second item; therefore, a special *SIMODE* was calculated in the program. The *SIMODE* reflects the base value of **-100**, with the number of significant bytes in the first item (in this case 4) subtracted from it (therefore **-104**), and the truncated length of the second item dynamically subtracted from it.

The program is hard-coded to impose an *SIMODE* of at least -104 (the full length of the order number) and to dynamically subtract the length in bytes of the part description specified from it. This permits retrievals using the full order number and either no part number or any number of leading characters of the part number.

A complete copy of the source program appears in *Appendix B*, as well as in the file QCAT.DEMO.SUPERDEX.

Keyworded key demo

About the demo

A *keyworded* SI-key is just like a simple SI-key except that every significant word contained in the key may be searched on. For example, the customer "BRADMARK TECHNOLOGIES" could be located by **BRADMARK** or **TECHNOLOGIES**.

This demo shows

- ■ keyword retrieval
- ■ generic and partial-keyword retrieval

The Keyworded Key Demo illustrates the use of a keyworded SI-key to locate customers stored in the CUSTOMERS master dataset. It is the same type of retrieval as in the Simple Key Demo using the same CUSTOMER-NAME field, but this time configured as a keyworded SI-key.

Running the demo

Select option **3** from the Main Menu and press **RETURN** to proceed to the Keyworded Key Demo.

The following screen is displayed:

```
MODE:F ACTION:                    Keyworded Key Demo

 Enter any word from a customer's name
_____

========================================================================
Customer Name                 Customer #




















```

You may specify any word contained in any customer name. Type

FRANK

and press **RETURN**. All the customers that contain the word "FRANK" are displayed:

```
CIMINELLI FRANK CONST           300057
RIPPLE J FRANK                 1800510
```

It does not matter where in the field the keyword occurs; just as long as it is separated by spaces or special characters.

Just as SUPERDEX'ed QUICK supports partial-key retrieval, it also supports partial-keyword retrieval. Go back into **Find** mode and append an **@** to the specified value

FRANK@

and press **RETURN**. All entries that contain words that start with "FRANK" are displayed:

```
CIMINELLI FRANK CONST            300057
RIPPLE J FRANK                  1800510
FRANKENSTEIN WM D                600628
```

As in the Simple Key Demonstration, you may use **@** and/or **?** to perform partial-keyword or generic keyword searches.

Further demonstration

You may try additional keyword values to further experiment with keyworded SI-keys, including the **@**, **?**, **>=**, **<=**, and **<>** operators described in the Simple Key Demo.

☞ You will not have much success using the values **ASSN**, **ASSOC**, **CO**, **COMPANY**, **CORP**, or **INC** -- these common words have been excluded from keywording (by entering them in a special file named **KWEXCLUD**) to conserve disk space and optimize retrieval speed.

Press **f8** when you are done to return to the Main Menu.

How the demo works

This program is almost identical to the Simple Key Demo program, with the main difference being that a keyworded SI-path is referenced and therefore all access against the SI-path is treated as keyworded.

In SUPERDEX, an SI-path may be configured as keyworded or not keyworded. This is strictly a configuration option specified when the SI-path is established, and does not impact any subsequent processing. Keywording is performed automatically when entries are DBPUT, DBUPDATEed, and DBDELETEed, and accessed whenever SUPERDEX access is performed. There is no difference in handling a keyworded SI-path versus a non-keyworded SI-path.

A complete copy of the source program appears in *Appendix B,* as well as in the file QKW.DEMO.SUPERDEX.

Grouped key demo

About the demo

A *grouped* SI-key permits multiple fields in a dataset to be handled as if they were a single field. For example, if three fields contain people's names and you need to locate a specific person, all three fields would be searched simultaneously in a single operation.

This demo shows

- grouping of functionally equivalent fields
- multiple keys in master and detail datasets
- generic and partial-key retrieval

The Grouped Key Demo illustrates the use of a grouped SI-key to locate customers stored in the CUSTOMERS master dataset by either address or city. These two fields are combined to form a group, and the group is configured as keyworded to allow access to any word in either field.

Running the demo

Select option **4** from the Main Menu and press **RETURN** to proceed to the Grouped Key Demo.

The following screen is displayed:

```
MODE:F ACTION:                      Grouped Key Demo

Enter any word from the address field or the city field
_____
==========================================================================
Customer Name                   Address                   City




```

You may specify any word contained in any address or city. Type

KENMORE

and press **RETURN**. All the customers that have a address on "KENMORE" Avenue or are in the city of "KENMORE" are displayed:

```
BARBER-COLMAN CO            1249 MILITARY RD        KENMORE
CASSETTA AGENCY CO INC      810 KENMORE AVE         BUFFALO
C B N                       3174 DELAWARE AVE       KENMORE
CEGLIA LAWRENCE             2070 SHERIDAN DR        KENMORE
C S F DESIGNS INC           61 GARDENWOOD LANE      KENMORE
CENTURY 21 GOLD JACKET      3411 DELAWARE AVE       KENMORE
CECOS ENVIRONMENTAL INC     2321 KENMORE AVENUE     BUFFALO
CHECKERCAR CLUB OF AMERICA  4693 TERMAINE AVE.      KENMORE
CHECKPOINT FOREIGN CAR      487 KENMORE AVE         BUFFALO
F B L ASSOCIATED AGENCIES   860 ENGLEWOOD AVE       KENMORE
FASO CHARLES P. AGENCY      860 ENGLEWOOD AVE       KENMORE
HOOD COMPANY INC            2225 KENMORE AVENUE     BUFFALO
IMMCO DIAGNOSTICS INC       963 KENMORE AVE         BUFFALO
KOCH RICHARD J CPA          1026 ENGLEWOOD AVE.     KENMORE
LAKELAND AUTOMOTIVE         536 NIAGARA FALLS BLVD  KENMORE
LOEFFLER F.H. COMPANY INC.  328 KENMORE AVE.        BUFFALO
OTIS ELEVATOR               1175 MILITARY RD        KENMORE
```

It does not matter where in either field the specified keyword or partial-keyword occurs; just as long as it occurs in either field. Note that the customer name is displayed for information only--it is not included in the group and may therefore not be searched on.

Further demonstration

You may try additional values to further experiment with grouping, and may include the @, ?, >=, <=, and <> operators already described.

Try the values **AMHERST, NIAG@**, and **WILLIAM@** for interesting results.

Press **18** when you are done to return to the Main Menu.

How the demo works

In SUPERDEX, an SI-path may be configured as grouped or not grouped. A grouped SI-path may be keyworded or not keyworded. In this example, the SI-path is configured as both grouped and keyworded, and is comprised of the address and city fields. A second-line address or other fields could also have been included in the group, if desired.

It is completely transparent to programs as to whether or not an SI-path is configured as grouped. Grouping is performed automatically when entries are DBPUT, DBUPDATEed, and DBDELETEed, and accessed whenever SUPERDEX access is performed. There is no difference in handling a grouped SI-path vs a non-grouped SI-path, just as keywording is also transparent.

A complete copy of the source program appears in _Appendix B_, as well as in the file QGRP.DEMO.SUPERDEX.

Relational access demo - multiple criteria

About the demo

Before proceeding to the last demo program, another very powerful concept must be introduced that applies to the demo programs run thus far:

■ *relational access* using multiple values for a field

Whereas we've seen how SUPERDEX'd QUICK permits both generic and partial-key retrievals using the @, ?, >=, <=, and <> operators, these capabilities are sometimes not sufficient for qualifying the entries you require. Rather, it is necessary to select entries that qualify by multiple values in various combinations.

Running the demo

To illustrate this concept, referred to as *Relational Access*, go back to the Simple Key Demo (option 1) and type the following (including the trailing vertical bar)

[UNITED@] [CENTRAL@] |

in the customer name field and

F

in the Direction field, and press RETURN. The following entries are displayed:

```
CENTRAL BFLO PROJECT CORP.       300209
CENTRAL PK UNITED METH           300236
CENTRAL AUTO WRECKING            300394
CENTRAL CITY RESTORATN           300427
CENTRAL ANESTHESIA SVCE          300527
CENTRAL ORGAN SERVICE            300559
UNITED CHURCH HOME              2100304
UNITED CEREBRAL PALSY ASSN      2100400
UNITED FUND BUFF & ERIE         2100401
UNITED PRESB CHURCH             2100509
UNITED ALLOYS & STEEL           2100649
UNITED BUSINESS EQUIPMENT       2100652
UNITED IMPORT MOTORS INC        2100700
UNITED AIRLINES                 2112949
```

As you can see, SUPERDEX'ed QUICK selected all the entries that begin with either "CENTRAL" or "UNITED". This was accomplished by specifying both values enclosed in square brackets ([ ]) and following the two bracketed values with a |, which is treated by SUPERDEX as an **OR** boolean operator.

In addition to OR, SUPERDEX'ed QUICK can also perform boolean AND and AND NOT operations against multiple values. SUPERDEX'ed QUICK uses the following boolean operators in performing relational retrievals:

| boolean **OR**
& boolean **AND**
!& boolean **AND NOT**

Values must be enclosed in square brackets ([ ]) and followed by the boolean operator in Reverse Polish Notation (RPN) syntax, as shown:

[*value1*] [*value2*] *boolean operator*

To further illustrate the Relational Access concept, exit this demo and go to the Keyworded Key Demo (option **3**) and specify

FRANK@

and press **RETURN**. The following entries are displayed:

```
CIMINELLI FRANK CONST        300057
RIPPLE J FRANK              1800510
FRANKENSTEIN WM D            600628
```

Now, change the value to

[FRANK@] [FRANKENSTEIN] !&

and press **RETURN**. This displays all the entries that contain a word starting with "FRANK" and not "FRANKENSTEIN".

```
CIMINELLI FRANK CONST        300057
RIPPLE J FRANK              1800510
```

To further demonstrate the power and flexibility of Relational Access within an SI-key, exit this demo and go to the Grouped Key Demo (option **4**) and specify

[KENMORE] [BUFFALO] &

to display all the entries that contain both "KENMORE" and "BUFFALO" in either the address or city field. (Note that in this example it is only coincidental that all the "KENMORE"s appear in addresses and "BUFFALO"s as cities; customers on "BUFFALO STREET" in the city of "KENMORE" would also qualify.)

```
CASSETTA AGENCY CO INC      810 KENMORE AVE        BUFFALO
CECOS ENVIRONMENTAL INC     2321 KENMORE AVENUE    BUFFALO
CHECKPOINT FOREIGN CAR      487 KENMORE AVE        BUFFALO
HOOD COMPANY INC            2225 KENMORE AVENUE    BUFFALO
IMMCO DIAGNOSTICS INC       963 KENMORE AVE        BUFFALO
LOEFFLER F.H. COMPANY INC   328 KENMORE AVE.       BUFFALO
```

Several values with corresponding boolean operators may be specified at one time or in multiple consecutive operations. Type

[KENMORE]

and press RETURN. Note that 17 entries are displayed.

Now, specify a value of

[BUFFALO] &

and press RETURN. SUPERDEX'ed QUICK remembers the qualifying entries that were found previously and uses them for comparison in the next operation. Now only six entries qualify. Using this technique, you may use successive operations to refine the selected entries by additional criteria.

Further demonstration

Experiment with the Simple Key, Keyworded Key, and Grouped Key demo programs using boolean operations to get a greater understanding of Relational Access between values in an SI-key.

Several values may be specified with their corresponding boolean operators. Remember to use Reverse Polish Notation syntax; for example, the combination

[*value1*] [*value2*] & [*value3*] | [*value4*] ! &

is interpreted as "all the entries that contain *value1* AND *value2* OR *value3* AND NOT *value4*."

Press **f8** when you are done to return to the Main Menu.

How the demos work

The three demo programs used in these Relational Access examples are the very same programs that were run before when illustrating *indexed* (non-relational) access. They are accessing the very same SI-paths as before. It is transparent to the programs whether the value specified is a single value or multiple values delimited by square brackets. Both types of retrievals are supported by the same SI-paths with the same code.

In writing programs for relational access, you may prefer to impose the square brackets and/or boolean operators programmatically and instead present the user with an individual field for each value and function keys to specify the boolean operators. There are many methods for forming the complete value with the required delimiters and operators.

Regardless of how the delimited value is formed, it is passed as the *SIARG* using *SIMODE* 1, exactly as shown. SUPERDEX locates the corresponding entries, just as with non-relational access.

Other features are available for further managing the results of multiple GETs, including the ability to refine and undo the results of successive GETs.

Relational access demo - multiple datasets

About the demo

As we've seen, *relational access* may be performed within a single field by specifying multiple values for the field and combining them by use of boolean operators.

Relational access can also be used to compare against multiple fields, datasets, and even multiple databases using similar methods and boolean operators.

This demo shows

■ relational access across multiple datasets

This example finds all the order line items containing a specified part number that exist for a specified customer. This is no trivial task since there is no direct relationship between the CUSTOMERS master and ORDER-LINES detail; therefore, a logical relationship must be formed via the ORDER-HEADERS master dataset. To add even greater flexibility, this program permits a partial-key or generic value to be specified for either field.

Running the demo

Select option **5** from the Main Menu and press **RETURN** to proceed to the Relational Access Demo.

The following screen is displayed:

```
MODE:F ACTION:                    Relational Access Demo

Customer Name _____    Part Number _____
=======================================================================
Order #      Part #    Part Description        Quan   Price
```

The first input field is for the Customer Name, and the second for the Part Number contained in each order line item for the specified customer. Entries must match on both fields in order to qualify.

Type

   UNITED CHURCH@

in the first field and press **RETURN**; then specify

   @

in the second field, and press **RETURN**. This specifies that SUPERDEX'ed QUICK should locate all the order line items for the customer whose name begins with "UNITED CHURCH".

Several entries are found, starting with:

```
    701193 A626765N      BNDR,POST,11 X 17,GN        4 107.80
    701193 Y4403CR       PUNCH,1 HOLE,1/4 DIA        1   1.69
    701193 R9530609      TAPE,EMBOSS,1/2X144 RL,BK   6  16.50
    701193 SRA           SR-B STAPELE REMOVER        1   0.68
    701193 BCMRC21BE     REFILL,F/CLIC,MED,2PK,BE    2  23.52
    701193 G27-12        COL SHEET                   1  29.61
    701193 C15-BLK       DISPENSER                   1   4.22
    701193 BCMRC21BK     REFILL,F/CLIC,MED,2PK,BK    2  23.52
    701193 WES40290      90-CLASP 9X12 ENVELOPES     1   6.01
    701193 710-01        JUST FOR COPIES             2   3.12
    701193 482-2         #100080 MONGOL PENCIL       2   2.46
    701193 332-01-RED-M  WRITE BROS                 12   1.08
    701193 334-01-GRN-M  PEN                        12   1.08
    701193 331-01-BLU-M  WRITE BROS                 24   2.16
    701193 SCM1312       21-1/3 LTR  FILE FOLDERS    1   3.82
    701193 CLI-PC        1-GEM PAPER CLIPS           1   1.02
    701193 SCM800        RULED CARDS   51-5X8        4   7.84
    719117 A615724       BNDR,POST,11X17,GN          4  78.40
```

Now, let's perform a more specific selection. Specify the same Customer Name but a Part Number of

33@

and press RETURN. This specifies that only the line items whose part numbers begin with "33" for the customer whose name begins with "UNITED CHURCH" should be displayed. SUPERDEX'ed QUICK now returns only the four following entries:

```
0000701193 332-01-RED-M  WRITE BROS              12  1.08
0000701193 334-01-GRN-M  PEN                     12  1.08
0000701193 331-01-BLU-M  WRITE BROS              24  2.16
0000928312 334-01-GRN-M  PEN,BALLPOINT,MED PT,GN 12  1.08
```

Further demonstration

You may try out additional customer name and part number combinations, using a full, generic, or partial key for each value.

Note that this demo program automatically encloses values in both fields in square brackets, so do not include square brackets in the values specified because an additional set of brackets would be imposed and entries would therefore not be found. Also note that this program disallows retrievals against more than one customer at a time, so the Customer Name specified must qualify only one entry.

Press f8 when you are done to return to the Main Menu.

How the demo works

The program must perform three **GET SIPROC...**s against three separate SI-paths in the POSTPATH procedure to accomplish the retrieval before performing the **GET SEQUENTIAL** to retrieve the entries in the default FIND procedure.

This requires that SUPERDEX first locate the specified customer in the CUSTOMERS master dataset by name and retain the corresponding customer number, which is stored in the item CUSTOMER-NUMBER. This is done via the simple customer SI-path using *SIMODE* 1 against the *SIDSET* CUSTOMERS with the specified customer name as the *SIARG*, surrounded by square brackets ([ ]).

Next, the retained customer number must be looked up in the ORDER-HEADERS master dataset to locate the corresponding order number(s). This is done via a special operation called a *projection*, which is accomplished simply by performing a **GET SIPROC...** using *SIMODE* 1 against the *SIDSET* ORDER-HEADERS specifying an *SIARG* of **[*]**.

The final **GET SIPROC...** performs a boolean AND between the entries located in the ORDER-HEADERS dataset and the order line items in the ORDER-LINES dataset, using the SI-path PART-ORDER in the *SIPATH* parameter and the part number, surrounded by square brackets, as the *SIARG*.

These same techniques may be used to perform relational retrievals against multiple databases, simply by using a different **SIPROC** definition for each one, e.g. **SIPROC1**, **SIPROC2**, etc. (they must also be defined this way in the data dictionary).

A complete copy of the source program appears in *Appendix B*, as well as in the file QPRJ.DEMO.SUPERDEX.

Section 3	**QUIZ interface demo**

Running the demonstrations

Four SUPERDEX'ed QUIZ programs have been defined to provide the same functionality as the QUICK demos and access the same SI-paths as the QUICK interface demo programs.

Type

```
:QUIZDEMO
```

at the MPE colon prompt (do not type the **:**) to invoke the QUIZ interface demo. This runs a special front-end program called **SIQUIZ**, which prompts with a familiar

```
SIQUIZ>
```

Simple key demo

About the demo

This demo is equivalent to the QUICK interface Simple Key Demo, and illustrates the use of a *simple* SI-key to locate customers in the CUSTOMERS master dataset.

Running the demo

Respond to the prompts as shown below in boldface, pressing **RETURN** after each response:

```
SIQUIZ> USE SIMPLE
Enter customer name: UNITED@
SIQUIZ> GO
```

After a moment, the following qualifying entries are displayed:

```
92/02/06                      SUPERDEX DEMO                   PAGE 1

CUSTOMER-NAME                  CUSTOMER-NUMBER

UNITED AIRLINES                   2112949
UNITED ALLOYS & STEEL             2100649
UNITED BUSINESS EQUIPMENT         2100652
UNITED CEREBRAL PALSY ASSN        2100400
UNITED CHURCH HOME                2100304
UNITED FUND BUFF & ERIE           2100401
UNITED IMPORT MOTORS INC          2100700
UNITED PRESB CHURCH               2100509




?
```

As you can see, SUPERDEX'ed QUIZ can perform a partial-key retrieval and returns entries in sorted sequential order, just like in the QUICK interface demo.

Press RETURN, and note the summary statistics:

```
?RETURN

Records selected:   8
Lines printed:      10
Pages printed:      1
```

Now, rerun the demo to try another type of retrieval:

```
SIQUIZ> USE SIMPLE
Enter customer name: UNI?E@
SIQUIZ> GO
```

This retrieves all the customers that start with with "UNI" followed by any alphanumeric character followed by an "E" followed by any number of any character. The following entries are displayed:

```
UNITED AIRLINES              2112949
UNITED ALLOYS & STEEL        2100649
UNITED BUSINESS EQUIPMENT    2100652
UNITED CEREBRAL PALSY ASSN   2100400
UNITED CHURCH HOME           2100304
UNITED FUND BUFF & ERIE      2100401
UNITED IMPORT MOTORS INC     2100700
UNITED PRESB CHURCH          2100509
UNIVERSITY BOOKSTORE         2100606
```

Press **RETURN** to display the summary statistics and to try another retrieval construct.

```
SIQUIZ> USE SIMPLE
Enter customer name: >=UN@<=UNI@
SIQUIZ> GO
```

This performs a range retrieval of all the entries that begin with "UN" through "UNI", inclusive. These 11 entries qualify:

```
UNDERWRITERS SALVAGE CO      2100347
UNITARIAN CHURCH             2100207
UNITED AIRLINES              2112949
UNITED ALLOYS & STEEL        2100649
UNITED BUSINESS EQUIPMENT    2100652
UNITED CEREBRAL PALSY ASSN   2100400
UNITED CHURCH HOME           2100304
UNITED FUND BUFF & ERIE      2100401
UNITED IMPORT MOTORS INC     2100700
UNITED PRESB CHURCH          2100509
UNIVERSITY BOOKSTORE         2100606
```

Further demonstration

You may try out additional values to further experiment with SUPERDEX'ed QUIZ's handling of a simple SI-key. As in the QUICK interface demonstrations, you may include the following operators in your values to tailor your selections:

<=value	less-than or equal-to retrieval
>=value	greater-than or equal-to retrieval
<>value	not-equal-to retrieval

How the demo works

SUPERDEX's fast and powerful retrievals are very simple to use in QUIZ programs.

The retrievals in this demonstration were accomplished by accessing SUPERDEX index structures contained in a special standalone detail dataset called **SI** in the database. Each unique relationship is referred to as an SI-path, and is accessed in very much the same way as accessing an IMAGE path. In this demo, an SI-path exists for customer name.

The only difference between this program and a regular QUIZ program is that the **SELECT** command was replaced by the new command **SIPATH**, which defines both the SI-path to access and the prompt to issue to the user. Because QUIZ does not interpret this command, it was prefixed by a semicolon (;) so QUIZ sees it as a comment.

In this program, the dataset CUSTOMERS is **ACCESS**ed, and the SI-path CUSTOMER-NAME and prompt `Enter customer name:` are defined in the **SIPATH** command.

The value entered in response to the prompt may contain special operators, such as @ and ?. In this program, the customer you enter is used for the lookup on the SI-path, and entries are by default returned in ascending sorted order.

A complete copy of the source program appears in _Appendix C,_ as well as in the file SIMPLE.DEMO.SUPERDEX.

Keyworded key demo

About the demo

This demo is equivalent to the QUICK interface Keyworded Key Demo, and illustrates the use of a *keyworded* SI-key to locate customers stored in the CUSTOMERS master dataset. It is the same type of retrieval as in the Simple Key Demo, but against a keyworded SI-key.

Running the demo

Respond to the prompts as shown below in boldface, pressing **RETURN** after each response:

```
SIQUIZ> USE KEYWORD
Enter any word in customer name: FRANK
SIQUIZ> GO
```

After a moment, the following qualifying entries are displayed:

```
 92/02/06                    SUPERDEX DEMO                    PAGE 1

 CUSTOMER-NAME               CUSTOMER-NUMBER

 CIMINELLI FRANK CONST            300057
 RIPPLE J FRANK                  1800510











?
```

Press RETURN, and note the summary statistics:

```
?RETURN

Records selected:   2
Lines printed:      4
Pages printed:      1
```

Further demonstration

Re-run the demo to try other types of retrievals.

How the demo works

This program is almost identical to the Simple Key Demo program, with the only difference being that a keyworded SI-path is referenced and therefore all access against the SI-path is treated as keyworded. Accordingly, a different SI-path and prompt are referenced in the **SIPATH** command.

In SUPERDEX, an SI-path may be configured as keyworded or not keyworded. This is strictly a configuration option specified when the SI-path is established, and does not impact any subsequent processing. Keywording is performed automatically when entries are DBPUT, DBUPDATEed, and DBDELETEed, and accessed whenever a retrieval is performed. There is no difference in handling a keyworded SI-path v.s. a non-keyworded SI-path.

A complete copy of the source program appears in *Appendix C*, as well as in the file KEYWORD.DEMO.SUPERDEX.

Grouped key demo

About the demo

This demo is equivalent to the QUICK interface Grouped Key Demo, and illustrates the use of a *grouped* SI-path to locate customers stored in the CUSTOMERS master dataset by either address or city. These two fields are combined to form a group, and the group is configured as keyworded to allow access to any word in either field.

Running the demo

Specify the values shown:

```
SIQUIZ> USE GROUPED
Enter any word in address or city: KENMORE
SIQUIZ> GO
```

The results are shown:

```
92/02/06                     SUPERDEX DEMO                     PAGE 1

CUSTOMER-NAME               ADDRESS-1              CITY

BARBER-COLMAN CO           1249 MILITARY RD       KENMORE
CASSETTA AGENCY CO INC     810 KENMORE AVE        BUFFALO
C B N                      3174 DELAWARE AVE      KENMORE
CEGLIA LAWRENCE            2070 SHERIDAN DR       KENMORE
C S F DESIGNS INC          61 GARDENWOOD LANE     KENMORE
CENTURY 21 GOLD JACKET     3411 DELAWARE AVE      KENMORE
CECOS ENVIRONMENTAL INC    2321 KENMORE AVENUE    BUFFALO
CHECKERCAR CLUB OF AMERICA 4693 TERMAINE AVE.     KENMORE
CHECKPOINT FOREIGN CAR     487 KENMORE AVE        BUFFALO
F B L ASSOCIATED AGENCIES  860 ENGLEWOOD AVE      KENMORE
FASO CHARLES P. AGENCY     860 ENGLEWOOD AVE      KENMORE
HOOD COMPANY INC           2225 KENMORE AVENUE    BUFFALO
IMMCO DIAGNOSTICS INC      963 KENMORE AVE        BUFFALO
KOCH RICHARD J CPA         1026 ENGLEWOOD AVE.    KENMORE
LAKELAND AUTOMOTIVE        536 NIAGARA FALLS BLVD KENMORE
LOEFFLER F.H. COMPANY INC. 328 KENMORE AVE.       BUFFALO
OTIS ELEVATOR              1175 MILITARY RD       KENMORE




?
```

Further demonstration

Press RETURN to see the summary statistics and restart the demo to perform another retrieval, or move on to the next demo.

How the demo works

In SUPERDEX, an SI-path may be configured as grouped or not grouped. A grouped SI-path may be keyworded or not keyworded. In this example, the SI-path is configured as both grouped and keyworded, and is comprised of the address and city fields. A second-line address or other fields could also have been included in the group, if desired.

It is completely transparent to programs as to whether or not an SI-path is configured as grouped. Grouping is performed automatically when entries are DBPUT, DBUPDATEed, and DBDELETEed, and accessed whenever a retrieval is performed. There is no difference in handling a grouped SI-path v.s. a non-grouped SI-path, just as keywording is also transparent.

A complete copy of the source program appears in _Appendix C_, as well as in the file GROUPED.DEMO.SUPERDEX.

Relational access demo - multiple criteria

About the demo

Before proceeding to the last demo program, another very powerful concept must be introduced that applies to the demo programs run thus far:

■ *relational access* using multiple values for a field

Whereas we've seen how SUPERDEX'ed QUIZ permits both generic and partial-key retrievals using the @, ?, >=, <=, and <> operators, these capabilities are sometimes not sufficient for qualifying the entries you require. Rather, it is necessary to select entries that qualify by multiple values in various combinations.

Running the demo

To illustrate this concept, referred to as *Relational Access*, go back to the Simple Key Demo and type the following:

```
SIQUIZ> USE SIMPLE
Enter customer name: UNITED,CENTRAL
SIQUIZ> GO
```

☞ **The reply to prompts cannot contain embedded spaces, as this is used as a terminator.**

The following entries are displayed:

```
CENTRAL BFLO PROJECT CORP.        300209
CENTRAL PK UNITED METH            300236
CENTRAL AUTO WRECKING             300394
CENTRAL CITY RESTORATN            300427
CENTRAL ANESTHESIA SVCE           300527
CENTRAL ORGAN SERVICE             300559
UNITED CHURCH HOME               2100304
UNITED CEREBRAL PALSY ASSN       2100400
UNITED FUND BUFF & ERIE          2100401
UNITED PRESB CHURCH              2100509
UNITED ALLOYS & STEEL            2100649
UNITED BUSINESS EQUIPMENT        2100652
UNITED IMPORT MOTORS INC         2100700
UNITED AIRLINES                  2112949
```

As you can see, SUPERDEX'ed QUIZ interface selected all the entries that begin with either "CENTRAL" or "UNITED". This was accomplished by specifying both values separated by a comma (**,**) which is treated by SUPERDEX as an **OR** boolean operator.

In addition to OR, SUPERDEX'ed QUIZ can also perform boolean AND and AND NOT operations against multiple values. SUPERDEX'ed QUICK uses the following boolean operators in performing relational retrievals:

- **,**   boolean **OR**
- **+**   boolean **AND**
- **–**   boolean **AND NOT**

The boolean operator is specified between values. There is no default order of evaluation (e.g. AND is not performed before OR), so parentheses may be used to enforce a desired order of evaluation, as shown (except without spaces):

> (*value1 operator value2*) *operator* (*value3 operator value4*)

If an operator appear as a valid character in an entry being sought, the value may be enclosed in double quotes (**"**) to resolve the ambiguity.

To further illustrate the Relational Access concept, exit this demo and go to the Keyworded Key Demo and type

```
SIQUIZ> USE KEYWORD
Enter any word in customer name: FRANK@
SIQUIZ> GO
```

The following entries are displayed:

```
   CIMINELLI FRANK CONST          300057
   RIPPLE J FRANK                1800510
   FRANKENSTEIN WM D              600628
```

Now, rerun instead like:

```
SIQUIZ> USE KEYWORD
Enter any word in customer name: FRANK@-FRANKENSTEIN
SIQUIZ> GO
```

This displays all the entries that contain a word starting with "FRANK" and not "FRANKENSTEIN".

```
   CIMINELLI FRANK CONST          300057
   RIPPLE J FRANK                1800510
```

To further demonstrate the power and flexibility of Relational Access within an SI-key, go to the Grouped Key Demo and specify:

```
SIQUIZ> USE GROUPED
Enter any word in customer name: KENMORE+BUFFALO
SIQUIZ> GO
```

to display all the entries that contain both "KENMORE" and "BUFFALO" in either the address or city field. (Note that in this example it is only coincidental that all the "KENMORE"s appear in addresses and "BUFFALO"s as cities; customers on "BUFFALO STREET" in the city of "KENMORE" would also qualify.)

```
CASSETTA AGENCY CO INC      810 KENMORE AVE       BUFFALO
CECOS ENVIRONMENTAL INC      2321 KENMORE AVENUE   BUFFALO
CHECKPOINT FOREIGN CAR       487 KENMORE AVE       BUFFALO
HOOD COMPANY INC             2225 KENMORE AVENUE   BUFFALO
IMMCO DIAGNOSTICS INC        963 KENMORE AVE       BUFFALO
LOEFFLER F.H. COMPANY INC    328 KENMORE AVE.      BUFFALO
```

Further demonstration

Experiment with the Simple Key, Keyworded Key, and Grouped Key demo programs using boolean operations to get a greater understanding of Relational Access between values in an SI-key.

Several values may be specified with their corresponding boolean operators. Parentheses may be used to group values to enforce a desired order of execution; for example, the combination:

> (value1+value2) , (value3–value4)

will select entries that meet either or both of two conditions: either

1. they contain both *value1* and *value2*   or
2. they contain *value3* but not both *value3* and *value4*

If an operator appears as a valid character in an entry being sought, the value may be enclosed in double quotes (") to resolve the ambiguity.

How the demos work

The three demo programs used in these Relational Access examples are the very same programs that were run before when illustrating non-relational access. They are accessing the very same SI-paths as before. It is transparent to the programs whether the value specified is a single value or multiple values with embedded operators optionally delimited by parentheses. Both types of retrievals are supported by the same SI-paths with the same code.

Relational access demo - multiple datasets

About the demo

This demo is equivalent to the QUICK interface multiple dataset Relational Access Demo, and finds all the order line items containing a specified part number that exist for a specified customer. As before, a logical relationship must be formed via the ORDER-HEADERS master dataset.

Running the demo

Type

```
SIQUIZ> USE RELATION
Enter customer name: UNITED CHURCH@+
Enter part number: 33@
SIQUIZ> GO
```

Be sure to specify the + at the end of the first value.

The following entries are displayed:

```
92/02/06                    SUPERDEX DEMO                  PAGE 1

ORDER-NUMBER    PART-NUMBER    PART-DESCRIPTION       QUANTITY-ORDERED

   701193       332-01-RED-M   WRITE BROS                  12
   701193       334-01-GRN-M   PEN,BALLPOINT,MED PT,GN     12
   701193       331-01-BLU-M   WRITE BROS                  24
   928312       334-01-GRN-M   PEN,BALLPOINT,MED PT,GN     12














?
```

Further demonstration

Try additional values, if desired.

How the demo works

The program must perform three distinct lookups against three different datasets and SI-paths to accomplish the retrieval. This is facilitated by specifying one **ACCESS** command and three **SIPATH** commands.

It requires that SUPERDEX first locate the specified customer in the CUSTOMERS master dataset by name and retain the corresponding customer number, which is stored in the item CUSTOMER-NUMBER. The **+** appended to the search value instructs SUPERDEX to retain the qualifying customer. Internally, the simple customer SI-path is utilized, and is specified using an **SIPATH** command that references both the dataset and SI-path.

Next, the retained customer number must be looked up in the ORDER-HEADERS master dataset to locate the corresponding order number(s). This is done via a special operation called a *projection*, which is accomplished simply by using an **SIPATH** command that references the dataset and SI-path, and which specifies the value [*] in place of the prompt.

The final operation performs a boolean AND between the entries located in the ORDER-HEADERS dataset and the order line items in the ORDER-LINES dataset, using the common item ORDER-NUMBER. This is specified by an **SIPATH** command that references the SI-path *PART-ORDER*; the dataset ORDER-LINES is specified in the **ACCESS** command.

A complete copy of the source program appears in *Appendix C*, as well as in the file RELATION.DEMO.SUPERDEX.

The data dictionary used for the QUICK demonstrations is listed on the following pages. The definitions which were added to support SUPERDEX access are shown in boldface.

These modifications were necessary for the QUICK interface only; no modifications are required for the QUIZ interface.

```
TITLE "SUPERDEX DEMO"
;----------------------------------------------------------------------------

FILE CUSTOMERS         ORGANIZATION MASTER              &
   OF OEDB                         PASSWORD "ACCT     "  &
                                   CAPACITY 1103

FILE ORDER-HEADERS     ORGANIZATION MASTER              &
   OF OEDB                         PASSWORD "ACCT     "  &
                                   CAPACITY 2801

FILE ORDER-LINES       ORGANIZATION DETAIL              &
   OF OEDB                         PASSWORD "ACCT     "  &
                                   CAPACITY 10256

FILE SIPROC            ORGANIZATION MASTER              &
   OF OEDB                         PASSWORD "ACCT     "  &
                                   NOCREATE
;----------------------------------------------------------------------------
ELEMENT ADDRESS-1         X(026)
ELEMENT ADDRESS-2         X(026)
ELEMENT ATTENTION-CODE    9(005)
ELEMENT BACK-ORDER-CODE   X(002)
ELEMENT BACK-ORDER-NEED   X(002)
ELEMENT BACKORDER-STATUS  9(005)
ELEMENT BILL-CODE         9(005)
ELEMENT BILLED-VALUE      9(010)
ELEMENT BILLING-CODE      9(005)
ELEMENT BKORD-INDICATOR   9(005)
ELEMENT BRANCH-LOCATION   9(005)
ELEMENT CARRIER-USED      9(005)
ELEMENT CARTON-QUANTITY   9(005)
ELEMENT CITY              X(016)
ELEMENT COMMERCIAL-STAT   9(010)
ELEMENT CONFIRM-DATE      9(005)
ELEMENT CONSOLIDATE-CODE  9(005)
```

```
ELEMENT  CUSTOMER-ABBR      X(004)
ELEMENT  CUSTOMER-NAME      X(030)
ELEMENT  CUSTOMER-NUMBER    9(010)
ELEMENT  ENTRY-DATE         9(005)
ELEMENT  ENTRY-VALUE        9(010)
ELEMENT  FREIGHT-CHARGE     9(010)
ELEMENT  FREIGHT-TRUCK      9(010)
ELEMENT  HOLD-CODE          9(005)
ELEMENT  INVOICE-CODE       9(005)
ELEMENT  INVOICE-LINE-NO    9(005)
ELEMENT  INVOICE-REF-NO     9(010)
ELEMENT  LAST-INVOICE-DTE   9(010)
ELEMENT  LINE-ITEM-PRICE    9(008)V9(2)
ELEMENT  LINE-ITEM-STATUS   9(005)
ELEMENT  NEXT-LINE-NUMBER   9(005)
ELEMENT  ORDER-NUMBER       9(010)
ELEMENT  ORDER-STATUS       9(005)
ELEMENT  ORDER-TYPE         X(002)
ELEMENT  ORDER-WEIGHT       9(005)
ELEMENT  PACKAGE-WEIGHT     X(002)
ELEMENT  PART-DESCRIPTION   X(026)
ELEMENT  PART-ENTRY-DATE    9(005)
ELEMENT  PART-HOLD-CODE     9(005)
ELEMENT  PART-NUMBER        X(014)
ELEMENT  PART-PRICE-CODE    X(002)
ELEMENT  PART-TYPE-CODE     X(002)
ELEMENT  PAYMENT-TERMS      9(005)
ELEMENT  PHONE-AREA-CODE    9(005)
ELEMENT  PHONE-PREFIX       9(005)
ELEMENT  PHONE-SUFFIX       9(005)
ELEMENT  PICKING-CODE       9(005)
ELEMENT  PICKING-LIST       9(005)
ELEMENT  PO-NUMBER          X(014)
ELEMENT  PREV-QTY-SHIPPED   9(005)
ELEMENT  PRICE-CODE         X(002)
ELEMENT  PRICE-DIFFERENTL   9(005)
ELEMENT  PRICE-DISCOUNT     9(005)
ELEMENT  QTY-PER-PACKAGE    9(005)
ELEMENT  QUANTITY-ORDERED   9(005)
ELEMENT  QUANTITY-SHIPPED   9(005)
ELEMENT  SALES-REP-CODE     X(002)
ELEMENT  SALES-TAX-PCT      9(010)
ELEMENT  SHIP-DATE          9(005)
ELEMENT  SHIP-TO-NUMBER     9(005)
ELEMENT  SHIPMENT-DATE      9(005)
```

```
ELEMENT SIPARMS          X(200)
ELEMENT SIDSET           X(16)
ELEMENT SIPATH           X(34)
ELEMENT SIMODE           9(005)
ELEMENT SIARG            X(148)
ELEMENT SIARG1           9(010)
ELEMENT SIARG2           X(144)
ELEMENT SICOUNT          9(010)
ELEMENT STATE            X(002)
ELEMENT STOCK-LOCATION   X(008)
ELEMENT TAX-PAYABLE      9(005)
ELEMENT UNIT-COST        9(008)V9(2)
ELEMENT UNIT-OF-MEASURE  X(002)
ELEMENT UNIT-PRICE       9(008)V9(2)
ELEMENT VALUE-CODE       X(002)
ELEMENT ZIP-CODE         9(010)


;-------------------------------------------------------------------
; OEDB
;-------------------------------------------------------------------

RECORD CUSTOMERS
  ITEM CUSTOMER-NUMBER    INTEGER SIGNED SIZE 4   KEY
  ITEM CUSTOMER-ABBR      CHARACTER SIZE 4
  ITEM CUSTOMER-NAME      CHARACTER SIZE 30
  ITEM ADDRESS-1          CHARACTER SIZE 26
  ITEM ADDRESS-2          CHARACTER SIZE 26
  ITEM CITY               CHARACTER SIZE 16
  ITEM STATE              CHARACTER SIZE 2
  ITEM ZIP-CODE           INTEGER SIGNED SIZE 4
  ITEM PHONE-AREA-CODE    INTEGER SIGNED SIZE 2
  ITEM PHONE-PREFIX       INTEGER SIGNED SIZE 2
  ITEM PHONE-SUFFIX       INTEGER SIGNED SIZE 2

RECORD ORDER-HEADERS
  ITEM ORDER-NUMBER       INTEGER SIGNED SIZE 4   KEY
  ITEM ORDER-TYPE         CHARACTER SIZE 2
  ITEM ENTRY-DATE         INTEGER SIGNED SIZE 2
  ITEM PO-NUMBER          CHARACTER SIZE 14
  ITEM CUSTOMER-NUMBER    INTEGER SIGNED SIZE 4
  ITEM SHIP-TO-NUMBER     INTEGER UNSIGNED SIZE 2
  ITEM BRANCH-LOCATION    INTEGER SIGNED SIZE 2
  ITEM NEXT-LINE-NUMBER   INTEGER SIGNED SIZE 2
  ITEM PAYMENT-TERMS      INTEGER SIGNED SIZE 2
  ITEM ATTENTION-CODE     INTEGER SIGNED SIZE 2
  ITEM TAX-PAYABLE        INTEGER SIGNED SIZE 2
  ITEM SALES-TAX-PCT      INTEGER SIGNED SIZE 4
  ITEM BILLED-VALUE       INTEGER SIGNED SIZE 4
  ITEM ENTRY-VALUE        INTEGER SIGNED SIZE 4
  ITEM SHIPMENT-DATE      INTEGER SIGNED SIZE 2
  ITEM ORDER-WEIGHT       INTEGER SIGNED SIZE 2
```

```
    ITEM FREIGHT-CHARGE    INTEGER SIGNED SIZE 4
    ITEM CARRIER-USED      INTEGER SIGNED SIZE 2
    ITEM CARTON-QUANTITY   INTEGER SIGNED SIZE 2
    ITEM PRICE-CODE        CHARACTER SIZE 2
    ITEM CONFIRM-DATE      INTEGER SIGNED SIZE 2
    ITEM LAST-INVOICE-DTE  INTEGER SIGNED SIZE 4
    ITEM BACK-ORDER-CODE   CHARACTER SIZE 2
    ITEM PICKING-CODE      INTEGER SIGNED SIZE 2
    ITEM BILLING-CODE      INTEGER SIGNED SIZE 2
    ITEM CONSOLIDATE-CODE  INTEGER SIGNED SIZE 2
    ITEM SALES-REP-CODE    CHARACTER SIZE 2            OCCURS 2
    ITEM BACKORDER-STATUS  INTEGER SIGNED SIZE 2
    ITEM HOLD-CODE         INTEGER SIGNED SIZE 2
    ITEM FREIGHT-TRUCK     INTEGER SIGNED SIZE 4
    ITEM VALUE-CODE        CHARACTER SIZE 2
    ITEM ORDER-STATUS      INTEGER SIGNED SIZE 2

RECORD ORDER-LINES
    ITEM ORDER-NUMBER      INTEGER SIGNED SIZE 4       PRIMARY KEY &
                                                      LINKS TO    &
                                                      ORDER-HEADERS
    ITEM INVOICE-LINE-NO   INTEGER UNSIGNED SIZE 2
    ITEM PART-TYPE-CODE    CHARACTER SIZE 2
    ITEM PART-NUMBER       CHARACTER SIZE 14
    ITEM PART-DESCRIPTION  CHARACTER SIZE 26
    ITEM PART-ENTRY-DATE   INTEGER SIGNED SIZE 2
    ITEM QUANTITY-ORDERED  INTEGER SIGNED SIZE 2
    ITEM UNIT-OF-MEASURE   CHARACTER SIZE 2
    ITEM QTY-PER-PACKAGE   INTEGER SIGNED SIZE 2
    ITEM LINE-ITEM-PRICE   INTEGER SIGNED SIZE 4
    ITEM UNIT-PRICE        INTEGER SIGNED SIZE 4
    ITEM UNIT-COST         INTEGER SIGNED SIZE 4
    ITEM PRICE-DISCOUNT    INTEGER SIGNED SIZE 2
    ITEM QUANTITY-SHIPPED  INTEGER SIGNED SIZE 2
    ITEM BACK-ORDER-NEED   CHARACTER SIZE 2
    ITEM SHIP-DATE         INTEGER SIGNED SIZE 2
    ITEM PICKING-LIST      INTEGER SIGNED SIZE 2
    ITEM BILL-CODE         INTEGER SIGNED SIZE 2
    ITEM PREV-QTY-SHIPPED  INTEGER SIGNED SIZE 2
    ITEM PART-HOLD-CODE    INTEGER SIGNED SIZE 2
    ITEM INVOICE-REF-NO    INTEGER SIGNED SIZE 4
    ITEM PRICE-DIFFERENTL  INTEGER SIGNED SIZE 2
    ITEM STOCK-LOCATION    CHARACTER SIZE 8
    ITEM BKORD-INDICATOR   INTEGER SIGNED SIZE 2
    ITEM PART-PRICE-CODE   CHARACTER SIZE 2
    ITEM COMMERCIAL-STAT   INTEGER SIGNED SIZE 4       OCCURS 2
    ITEM INVOICE-CODE      INTEGER SIGNED SIZE 2
    ITEM PACKAGE-WEIGHT    CHARACTER SIZE 2
    ITEM LINE-ITEM-STATUS  INTEGER SIGNED SIZE 2
```

```
RECORD SIPROC
  ITEM SIPARMS            CHARACTER SIZE 200        KEY
  REDEFINED BY
    ITEM SIDSET           CHARACTER SIZE 16
    ITEM SIPATH           CHARACTER SIZE 34
    ITEM SIMODE           INTEGER SIGNED SIZE 2
    ITEM SIARG            CHARACTER SIZE 148
    REDEFINED BY
      ITEM SIARG1         INTEGER SIGNED SIZE 4
      ITEM SIARG2         CHARACTER SIZE 144
      END
    END
  ITEM SICOUNT            INTEGER SIGNED SIZE 4

;

BUILD
```

Appendix B QUICK source programs

The sources for the QUICK demonstration programs appear on the following pages.

In reviewing these programs, you will see references to the file **SIPROC** using the item **SIPARMS**. These parameters are defined as follows:

SIDSET   The **OPEN** name of the dataset (which may differ from the **FILE** name).

SIPATH   The name of the SI-path to access.

SIMode   The type of retrieval to perform (e.g. how many characters to match on, whether to return entries in ascending or descending order, etc...).

SIARG   The value(s) to search for.

Simple Key Demo

```
SCREEN QCSK ACTIVITIES FIND ON 25
FILE CUSTOMERS PRIMARY OCCURS 17
  ACCESS SEQUENTIAL
FILE SIPROC DESIGNER
  ACCESS VIA SIPARMS
TEMP NAME-PROMPT CHAR*30 RESET AT MODE
TEMP DIR-PROMPT CHAR*1
TITLE "Simple Key Demo" AT ,40
SKIP 1
ALIGN (,1,21) (,54,70)
FIELD NAME-PROMPT UPSHIFT REQUIRED LABEL "Enter customer name " &
  HELP "Enter full or abbreviated name (e.g. UNITED@)"
FIELD DIR-PROMPT UPSHIFT VALUES "F","B" LABEL "Direction (F,B)" &
  HELP "Enter F for ascending order, B for descending order"
DRAW "=" FROM 4,1 TO 4,80
SKIP 1
TITLE "Customer Name                    Customer #" AT ,1
ALIGN (,,1) (,,33)
CLUSTER OCCURS WITH CUSTOMERS
FIELD CUSTOMER-NAME
FIELD CUSTOMER-NUMBER
CLUSTER

PROCEDURE POSTPATH
  BEGIN
    ACCEPT NAME-PROMPT
    ACCEPT DIR-PROMPT
    LET SIDSET = "CUSTOMERS "
    LET SIPATH = "CUSTOMER-NAME "
    IF DIR-PROMPT = "B"
      THEN LET SIMODE = 2001
      ELSE LET SIMODE = 1
    LET SIARG = NAME-PROMPT
    GET SIPROC USING SIPARMS OPTIONAL
    IF NOT ACCESSOK
      THEN
      ERROR "No records were found matching key/selection values."
    INFORMATION = ASCII(SICOUNT) + " records found"
    END
```

Concatenated Key Demo

```
SCREEN QCCAT ACTIVITIES FIND ON 25
FILE ORDER-LINES PRIMARY OCCURS 17
  ACCESS SEQUENTIAL
FILE SIPROC DESIGNER
  ACCESS VIA SIPARMS
TEMP ORDER-PROMPT NUM*10 RESET AT MODE
TEMP PART-PROMPT CHAR*14
TITLE "Concatenated Key Demo" AT ,40
SKIP 1
ALIGN (,1,14) (,27,40)
FIELD ORDER-PROMPT REQUIRED LABEL "Order Number" &
  HELP "Enter order number (e.g. 701257)"
FIELD PART-PROMPT UPSHIFT LABEL "Part Number" &
  HELP "Enter full or abbreviated part number (e.g. SCM)"
DRAW "=" FROM 4,1 TO 4,80
SKIP 1
TITLE "Order Number  Part Number      Part Description" AT ,1
ALIGN (,,1) (,,15) (,,31)
CLUSTER OCCURS WITH ORDER-LINES
FIELD ORDER-NUMBER
FIELD PART-NUMBER
FIELD PART-DESCRIPTION
CLUSTER

PROCEDURE POSTPATH
  BEGIN
    ACCEPT ORDER-PROMPT
    ACCEPT PART-PROMPT
    LET SIDSET = "ORDER-LINES "
    LET SIPATH = "ORDER-PART "
    LET SIMODE = -104-SIZE(TRUNC(PART-PROMPT))
    LET SIARG1 = ORDER-PROMPT
    LET SIARG2 = PART-PROMPT
    GET SIPROC USING SIPARMS OPTIONAL
    IF NOT ACCESSOK
      THEN
      ERROR "No records were found matching key/selection values."
    END
```

Keyworded Key Demo

```
SCREEN QCGRP ACTIVITIES FIND ON 25
FILE CUSTOMERS PRIMARY OCCURS 17
  ACCESS SEQUENTIAL
FILE SIPROC DESIGNER
  ACCESS VIA SIPARMS
TEMP ADDRESS-PROMPT CHAR*50 RESET AT MODE
TITLE "Grouped Key Demo" AT ,40
SKIP 1
TITLE "Enter any word from the address field or the city field"
ALIGN (,,1)
FIELD ADDRESS-PROMPT REQUIRED &
HELP "Specify any word contained in any customer's name(e.g. FRANK)"
DRAW "=" FROM 5,1 TO 5,80
SKIP 1
TITLE "Customer Name                    Customer #" AT ,1
ALIGN (,,1) (,,33)
CLUSTER OCCURS WITH CUSTOMERS
FIELD CUSTOMER-NAME
FIELD CUSTOMER-NUMBER
CLUSTER

PROCEDURE POSTPATH
  BEGIN
    ACCEPT NAME-PROMPT
    LET SIDSET = "CUSTOMERS "
    LET SIPATH = "CUSTOMER-NAME-KW "
    LET SIMODE = 1
    LET SIARG = NAME-PROMPT
    GET SIPROC USING SIPARMS OPTIONAL
    IF NOT ACCESSOK
      THEN
      ERROR "No records were found matching key/selection values."
    INFORMATION = ASCII(SICOUNT) + " records found"
    END
```

Grouped Key Demo

```
SCREEN QCGRP ACTIVITIES FIND ON 25
FILE CUSTOMERS PRIMARY OCCURS 17
  ACCESS SEQUENTIAL
FILE SIPROC DESIGNER
  ACCESS VIA SIPARMS
TEMP ADDRESS-PROMPT CHAR*50 RESET AT MODE
TITLE "Grouped Key Demo" AT ,40
SKIP 1
TITLE "Enter any word from the address field or the city field"
ALIGN (,,1)
FIELD ADDRESS-PROMPT REQUIRED HELP &
"Specify any word contained in any address or city (e.g. KENMORE)"
DRAW "=" FROM 5,1 TO 5,80
SKIP 1
TITLE &
   "Customer Name                    Address            City" &
AT ,1
SKIP
ALIGN (,,1) (,,33)  (,,61)
CLUSTER OCCURS WITH CUSTOMERS
FIELD CUSTOMER-NAME
FIELD ADDRESS-1
FIELD CITY
CLUSTER

PROCEDURE POSTPATH
  BEGIN
    ACCEPT ADDRESS-PROMPT
    LET SIDSET = "CUSTOMERS "
    LET SIPATH = "ADDRESS1-CITY-KW "
    LET SIMODE = 1
    LET SIARG = ADDRESS-PROMPT
    GET SIPROC USING SIPARMS OPTIONAL
    IF NOT ACCESSOK
      THEN
      ERROR "No records were found matching key/selection values."
    INFORMATION = ASCII(SICOUNT) + " records found"
    END
```

```
SCREEN QCPRJ ACTIVITIES FIND ON 25
FILE ORDER-LINES PRIMARY OCCURS 17
  ACCESS SEQUENTIAL
FILE SIPROC DESIGNER
  ACCESS VIA SIPARMS
TEMP NAME-PROMPT CHAR*30 RESET AT MODE
TEMP PART-PROMPT CHAR*14
DEFINE TOTAL-PRICE NUM = QUANTITY-ORDERED * UNIT-PRICE
TITLE "Relational Access Demo" AT ,40
SKIP 1
ALIGN (,1,15) (,49,61)
FIELD NAME-PROMPT REQUIRED UPSHIFT LABEL "Customer Name" &
  HELP "Enter full or abbreviated name (e.g. UNITED@)"
FIELD PART-PROMPT UPSHIFT LABEL "Part Number" &
  HELP "Enter full or abbreviated part number (e.g. @)"
DRAW "=" FROM 4,1 TO 4,80
SKIP 1
TITLE &
"Order #      Part #           Part Description" AT ,1
TITLE "Quan    Price" AT ,57
ALIGN (,,1) (,,13) (,,29) (,,57) (,,64)
CLUSTER OCCURS WITH ORDER-LINES
FIELD ORDER-NUMBER
FIELD PART-NUMBER
FIELD PART-DESCRIPTION
FIELD QUANTITY-ORDERED
FIELD TOTAL-PRICE PICTURE "^^^^.^^" SIGNIFICANCE 4
CLUSTER
PROCEDURE POSTPATH
  BEGIN
    ACCEPT NAME-PROMPT
    LET SIDSET = "CUSTOMERS "
    LET SIPATH = "CUSTOMER-NAME "
    LET SIMODE = 1
    LET SIARG = "["+TRUNC(NAME-PROMPT)+"] "
    GET SIPROC USING SIPARMS OPTIONAL
    IF ACCESSOK
      THEN BEGIN
        IF SICOUNT = 1
          THEN BEGIN
```

```
                    LET SIDSET = "ORDER-HEADERS "
                        LET SIPATH = "CUSTOMER-NUMBER "
                        LET SIARG = "[*] "
                        GET SIPROC USING SIPARMS OPTIONAL
                        IF ACCESSOK
                          THEN BEGIN
                            ACCEPT PART-PROMPT
                            LET SIDSET = "ORDER-LINES "
                            LET SIPATH = "PART-ORDER "
                            LET SIARG = "["+TRUNC(PART-PROMPT)+"]& "
                            GET SIPROC USING SIPARMS OPTIONAL
                            IF NOT ACCESSOK
                              THEN ERROR "No qualifying parts"
                            END
                          ELSE ERROR "No qualifying orders"
                        END
                      ELSE ERROR "More than one customer qualified"
                    END
                  ELSE ERROR "No qualifying customers"
                END
```

Appendix C   QUIZ source programs

The sources for the QUIZ demonstration programs appear on the following pages.

In these examples, the new **SIPATH** command is used to define both the SI-path to access and the prompt to issue to the user. It is prefixed by a semicolon (**;**) to designate it as a comment, since QUIZ does not interpret it.

As shown in the last example, multiple **SIPATH** commands may be used to access multiple fields and datasets, with the dataset specified as an optional parameter.

Simple Key Demo

```
;SIPATH CUSTOMER-NAME PROMPT "Enter customer name:"
ACCESS CUSTOMERS
REPORT CUSTOMER-NAME CUSTOMER-NUMBER
GO
EXIT
```

Keyworded Key Demo

```
;SIPATH CUSTOMER-NAME-KW PROMPT "Enter any word in customer name:"
ACCESS CUSTOMERS
REPORT CUSTOMER-NAME CUSTOMER-NUMBER
GO
EXIT
```

Grouped Key Demo

```
;SIPATH ADDRESS1-CITY-KW PROMPT "Enter any word in address or city:"
ACCESS CUSTOMERS
REPORT CUSTOMER-NAME ADDRESS-1 CITY
GO
EXIT
```

Relational Access Demo - multiple datasets

```
;SIPATH CUSTOMERS.CUSTOMER-NAME PROMPT "Enter customer name:"
;SIPATH ORDER-HEADERS.CUSTOMER-NUMBER "[*]"
;SIPATH PART-ORDER PROMPT "Enter part number:"
ACCESS ORDER-LINES
REPORT ORDER-NUMBER PART-NUMBER PART-DESCRIPTION QUANTITY-ORDERED
GO
EXIT
```