THE QBE/SOLID INTERFACE

AUTOMATIC SYSTEMS FOR THE PHYSICAL SCIENCES
REPORT NUMBER 14

Technical Report CSE-89-08

P.A.D. de Maine
M. Edhala

Department of Computer Science and Engineering
Auburn University
Auburn, AL 36849-5347

September 18, 1989
THE QBE/SOLID INTERFACE

M. Edhala and P.A.D. de Maine

Automatic Systems for the Physical Sciences
Report Number 14

Computer Science and Engineering Department
Auburn University
Auburn, Alabama 36849

31 January 1988
THE QBE/SOLID INTERFACE

M. Edhala and P.A.D. de Maine

Report No. 14
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I.</td>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>II.</td>
<td>THE USER MANUAL</td>
<td>1</td>
</tr>
<tr>
<td>II.1</td>
<td>Introduction</td>
<td>1</td>
</tr>
<tr>
<td>II.2</td>
<td>Invoking the Interface</td>
<td>1</td>
</tr>
<tr>
<td>II.3</td>
<td>Entering and Submitting a Query</td>
<td>1</td>
</tr>
<tr>
<td>II.3.1</td>
<td>Type (1) Query</td>
<td>2</td>
</tr>
<tr>
<td>II.3.2</td>
<td>Types (2), (3) and (4) Queries</td>
<td>2</td>
</tr>
<tr>
<td>II.4</td>
<td>The Help Function</td>
<td>3</td>
</tr>
<tr>
<td>II.5</td>
<td>Loading and Storing Dictionaries</td>
<td>4</td>
</tr>
<tr>
<td>III.</td>
<td>NOTES OF THE IMPLEMENTATION OF THE QBE-SOLID INTERFACE</td>
<td>4</td>
</tr>
<tr>
<td>III.1</td>
<td>Data Structures</td>
<td>4</td>
</tr>
<tr>
<td>III.1.1</td>
<td>Dictionary</td>
<td>4</td>
</tr>
<tr>
<td>III.1.2</td>
<td>Domain List</td>
<td>4</td>
</tr>
<tr>
<td>III.1.3</td>
<td>Intermediate Tuple List</td>
<td>5</td>
</tr>
<tr>
<td>III.1.4</td>
<td>Intermediate Print List</td>
<td>5</td>
</tr>
<tr>
<td>III.1.5</td>
<td>Query</td>
<td>5</td>
</tr>
<tr>
<td>III.2</td>
<td>Main Procedure (QBE.FOR)</td>
<td>7</td>
</tr>
<tr>
<td>III.3</td>
<td>Draw Structure Procedure (DRAW_STRUCTURE)</td>
<td>8</td>
</tr>
</tbody>
</table>

I. INTRODUCTION

This report contains two parts. The first part is the QBE/SOLID User Manual and the second part is the QBE-SOLID System Manual, which should be read with the well documented FORTRAN Code stored on Tape D: PROGRAMS in directory PROGCURR\VAX780\JOBLIST\EDHALA.

II. THE USER MANUAL

II.1 Introduction:

The QBE/SOLID interface system provides a QBE interface to the high speed retrieval system SOLID. Queries are entered using a user friendly tabular format. The results from the query are also in a tabular format and the user can browse through the response. This is a dynamic data base system. The user can work on more than one dictionary by loading and storing dictionaries from a file.

II.2 Invoking the Interface:

To invoke the interface, type RUN QBE when the "$" prompt appears. This will take you to the main ENTER-SUBMIT-ENTER loop. The commands that can be performed at this point are:

PF1: Enter all tables in the present dictionary if any
PF2: Draw_structure
PF3: Quit the interface
PF4: Load a dictionary from a file
II.3 Entering and Submitting a Query:

There are different types of queries like:

1. Insert a new table into the dictionary
2. Insert a line/lines into an already existing table
3. Delete a line/lines from a table
4. Print information pertinent to the query

Any of these different types of queries can be invoked by PF2, which is the function key for the Draw_structure routine. The user is asked to enter "T" for a table or "C" for a condition box. To invoke a query, type "T". A skeleton table appears on the screen with a prompt for entering the table name.

II.3.1 Type (1) Query:

Enter "I.TABLENAME" and hit return. Type each column name at the point where the cursor is. As each new column name is typed, a prompt for the domain of that column appears at the bottom of the screen. Enter the domain name and hit return to continue. To stop the process of entering column names, type "E." instead of the column name, after the last column has been entered. The information will be entered into the dictionary and the control is passed to the main loop to start a new query. Note that if the columns do not fit into one screen, the table is split into as many screens as required. Each screen of a table is called a page.

II.3.2 Types (2), (3) and (4) Queries:

These queries might need more than one table. Start with any table. Enter the table name and hit return. If the table exists already, then all the skeleton table is displayed with the names of the columns. If the table does not exist, then that means a tempora-
ry table is being created to print information as required. In that case enter the column names same way as for a type (1) query (no domain name prompt will appear). As before, a table may be spread over more than one page (or screen). Initially only one row is allocated for the query. Move the cursor to the appropriate positions and enter the query in the usual QBE style. If more than one row is required, type the F14 function key. If a part of the query does not fit into the column, use the extend column facility (Function key F7). Now if more tables are required for the query, just use the PF2 key to get another skeleton table and continue the above process. The user can swap the screens to browse or enter query data by using the NEXT_PAGE, PREV_PAGE, NEXT_TABLE, and PREV_TABLE functions. When the user is sure that all the necessary data has been entered correctly, pressing the SUBMIT_QUERY function key, F12, will translate the data into JLI form and then send it to SOLID. SOLID returns information in the JLI form. This is in turn translated and displayed in the tabular (QBE) format. The user can browse through the data using the same function keys as before. To terminate browsing enter the PF3 function key.

II.4 The Help Function

The HELP key will take the user into the help routine. All function keys and the names of the functions are displayed. For more information about a particular function, move the cursor to the appropriate row and hit the ENTER key.
II.5 Loading and Storing Dictionaries

Once a new table has been entered, store the dictionary using the F18 key. The user will be prompted for the file name. If the file exists, it will be rewritten else it will be created. To use an already existing dictionary, use the load function.

III. NOTES ON THE IMPLEMENTATION OF THE QBE-SOLID INTERFACE

III.1 Data Structures

All the major data structures are declared in the file QBE_DEF.FOR.

III.1.1 Dictionary:

The dictionary is an array of records. The name of the data structure is DICTIONARY. The record type is TDICNRY. This record consists of:

1. The table name up to twenty characters long.
2. The M and J array of integers (for storing M and J values)
3. The number of columns in the table
4. For each column:
   4.1 The column name up to twenty characters long
   4.2 A pointer to its domain in the domain list

III.1.2 Domain List:

The domain list is a list of all the domains present in the data base. The maximum number of domains allowed is NDOMAIN_MAX. This is a parameter that can be changed when needed. This list is an array of records called DLIST. The record type is TDLIST. Each element
record consists of:

1. The domain name, up to twenty characters long.

2. A list of variables for that domain, which might appear in the query. The maximum number of variables allowed is given by the parameter NLISTVAR_MAX.

3. For each of the variables above, a list of pointers to tuples which contain these variables (used in JLI algorithms).

III.1.3 Intermediate Tuple List:

Another important data structure in the system is the intermediate tuple list. This is declared as an set of records called TUPLE. Each element in TUPLE is a record consisting of:

1. The pointer to the table from which it is built. If no table exists the pointer is zero.

2. A list of column entries 40 characters long.

3. For each column entry, a pointer to the domain it belongs to.

III.1.4 Intermediate Print List:

The intermediate tuple list is an interface between the input of the query and the JLI algorithms. Similarly, an intermediate print list will be needed to interface between the solution from SOLID and the print display routines. This data structure has not yet been implemented.

III.1.5 Query:

To handle all the input and displays, a new data structure is defined. This will keep tabs of all the tables that are currently displayed, and what information is stored in them etc. The display and input handling routines, deal directly with this data structure and do not manipulate any other data structure. Only the "insert
new table" routine will add the name of the table to the dictionary and all relevant data like the column names and domain pointers. This data structure is a list of tables that are presently being manipulated in the query. Each table may have more than one page (or screen). A page is displayed as a whole entity. So, for each page in the table, there is a virtual display associated with it. Since the positions of the columns in the table and hence in the page are not constant, such information required to make it known, should also be included in the data structure.

This data structure is an array of records called QUERY. The record type is defined as TQUERY. The maximum number of tables that can be manipulated by the query at any time is given by the parameter NQUERY_TABLES_MAX. Each element in this array consists of:

1. The name of the table
2. A pointer to the table in the dictionary. If the table is just a result table, this pointer will be zero.
3. The number of pages in the table
4. The number of rows that are used in the table
5. The type of table.
   T: Used to insert a table or to give information about a query
   C: A condition box
   P: Used by the system to print out results

6. An for each page:
   a. The display ID that corresponds to the virtual display
   b. The page number
   c. The number of columns in that page
   d. Two intermediate integers used during extension of a column
   e. For each column:
      i. The starting position of that column in the display
      ii. The width of the column
Various global variables are declared, which give the information about the current display on the screen and the current query information etc.

III.2 Main Procedure (QBE.FOR):

The main program is as follows:

A. Initialization

B. DO WHILE NOT QUIT_QBE

   a. Read a character from the keyboard

   The allowed characters are:

   PF1: Print all the names of the tables in the present dictionary (PRINT_TAB_NAMES)

   PF2: Draw a structure for a new table or condition box (DRAW_STRUCTURE)

   PF3: Quit the browsing of the results (QUIT_BROWSE)

   F7: Extend the present column in the present page (EXT_COL)

   F8: Display the next page in the current table (NEXT_PAGE)

   F9: Display the previous page of the current table (PREV_PAGE)

   F10: Display the first page of the next table (NEXT_TABLE)

   F11: Display the first page of the previous table (PREV_TABLE)

   F12: Submit the query (SUBMIT_QUERY)

   F13: QUIT_QBE := TRUE

   F14: Add another row to the current table (ADD_ROW)

   F17: Load a new dictionary (LOAD_DICTIONARY)

   F18: Save the current dictionary (STORE_DICTIONARY)
HELP: Call the help routine (QBE_HELP)
Cursors: Reflect appropriate cursor movement on the screen
Printerable Chars: Echo character (ECHO_CHAR)

C. END DO WHILE

III.3 Draw Structure Procedure (DRAW_STRUCTURE)
The algorithm for the DRAW_STRUCTURE procedure follows

Read (CONDITION_BOX or TABLE)
IF condition_box needed THEN
draw_condition_box and return to main procedure loop
ELSE
get_table_name
IF (name in dictionary) THEN
draw the table with all the information
use as many pages as required to fit the columns
each page has its own display_id
return to main procedure loop
ELSE
insert the table in the dictionary
get column information from user
use as many pages as required during the process
after receiving all the information, kill all displays
return to main procedure to start a new query

This procedure is in QBE.FOR which also contains the main program.
All the cursor movements and character echoing procedures are in
CRQBE.FOR. Load and store dictionary are in PRQBE.FOR. The hep
routine is in ORQBE.FOR.

All the routines for the SUBMIT_QUERY have still to be implemented. They are dependent on the SOLID system. All the files are on
tape D: PROGRAMS in directory PROGCURR\VAX780\JOBLIST\EDHALA.