Displaying database files

for IBM i

User Guide

Contents

Introduction

Motivation for this application was the fact that the popular utility DFU (Data File Utility) is unable to display all Unicode characters, especially UTF-8, UTF-16 or UCS-2.

The application serves to displaying data in database files. It works with physical and logical files. Only one file can be processed at a time.

The physical file must contain at least the first member. For example, a reference file without a data member cannot be displayed.

The logical file that refers to a single physical file can be used to enter and update data.

The file (physical or logical) with multiple members can be processed if ALIAS objects corresponding to the members exist in the library.

Traditional AS/400 terms are mainly used in the application. Correspondence of AS/400 and SQL terms are listed in the following table.

Traditional names	***SQL names***
library	schema
physical file	table
field	column
record	row
unique key	primary key unique key
key	index
logical file with selection	view
DDM file for a member	ALIAS object

Programs are written in Java language and require version *Java SE 8* or higher. They cooperate with programs from package *IBM i Toolbox for Java* (or *JTOpen*). Programs were created in the OS X operating system. They were tested in systems OS X 10.9 Mavericks, 10.10 Yosemite, 10.11 El Capitan and Windows 7 with remote internet connection to the IBM i, version 7.1.

Using the application

The application is not installed, it is delivered as a directory that can be placed at a suitable location in the computer (maybe in a flash disk). An alias or shortcut can be created from the file *IBMiSqlDisplay.jar* and located somewhere. The application is launched by double click on the shortcut or the original of this file.

The same copy of the application works the same in systems macOS and Windows.

Objects used in the application

Objects contained in the aplication must be placed in the current directory. This is ensured at installation of the application, that is delivered as a whole directory. When the application is launched the directory becomes current.

The application does not require an installation. It consists of a directory containing other directories and files.

Note: Objects contained in the application directory that are not introduced here serve for managing the project and may be removed without any harm.

Directories

- *columnfiles* - contains text files with column lists for the SELECT statement,
- *helpfiles* - contains this document in Czech and English languages,
- *logfiles* - contains text files *err.txt* and *out.txt*, to which redirected output of the files,System.err a System.out (i.e. console),
- *paramfiles* - contains the file *U_Parameters.txt* with application parameters,
- *selectfiles* - contains text files with data for selection and ordering of records in database files.

Note: Files *err.txt* and *out.txt* serve to find the cause of an error in program.

Direcories must not be deleted or renamed. The files inside directories *selectfiles* and *columnfiles* can be deleted without damage, they are created if needed. Contents of these files should not be changed manually.

Program files

- File *IBMiSqlDisplay.jar* - contains Java classes of the application and launches the application.
- File *jt400Small.jar* - contains a subset of classes from the package IBM i Toolbox for Java.

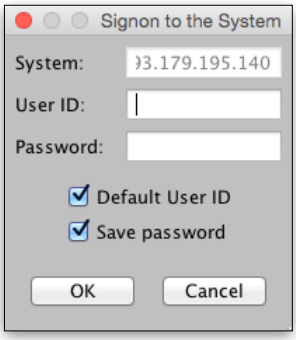
Location and running the application

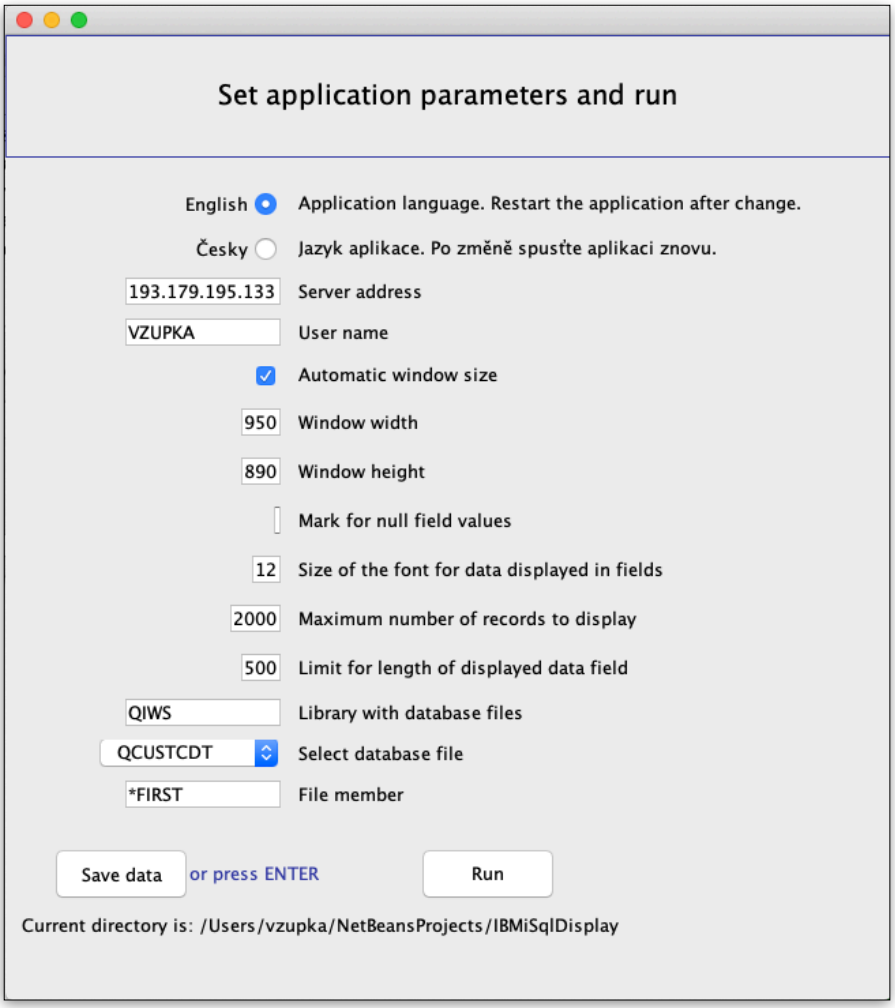
The application directory can be placed anywhere and possibly renamed. Find the file *IBMiSqlDisplay.jar* in the directory. Shortcuts (aliases) can be created from this file and placed somewhere.

The application starts by *double click* on the shortcut of (or original) *IBMiSqlDisplay.jar* file.

Start of the application

After starting the application the window *Set application parameters and run* is displayed. The dialog for signon parameters to access to the IBM i system is displayed at the same time. After signing on the user can adjust parameter values using the *Save data* button or *Enter* key (or leave the values unchangend), and then press the *Run* button.





Parameters

Note: If the user enters some important parameters incorrectly, then after clicking the *Run* button the following message appears

SQL STATEMENT ERROR or CONNECTION TO THE SERVER LOST.

Besides signon entries, the most sensitive entry is the *library name*.

Application language

The application can be processed in English (en_US) or Czech (cs_CZ) localization. The localization concerns titles, messages, button labels. The user can choose the localization by clicking the button. The selected option is applied fully (including the application menu) after ending the application and launching it again.

Server address

The user enters a single IP address in dot or domain form.

User name

The user enters the name of the profile that has the authority to write and change data in database files. This name will be prescribed in the dialog window *Signon to the System*.

Window size

If the box *Automatic window size* is checked the window is accomodated to the size of the displayed results. Otherwise the window will have dimensions specified in input fields *Window width* and *Window height*.

Note: If the value is not a whole number, zero is assumed.

Mark for null field values

The user enters a symbol that will be shown everywhere the field value is NULL. This symbol is also used for entering the NULL value in the table cell or input field. It is *not used* for fields of types CLOB and BLOB.

Size of the font for printing data in print points

This entry represents the number of points on the display. It determines size of letters in titles and cells in table cells and also in input data fields.

Note: If the value is not a whole number, zero is assumed.

Maximum number of records to display

This entry represents maximum number of records that are selected from the database file an that will be shown in the window table.

Limit for length of displayed data fields

This entry represents maximum length of the display field containing the data field value.

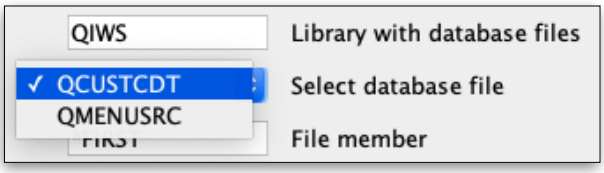
•

Library with database files

The name convention "*system*" is used to process SQL statements. The user enters a library name containing database files to work with of which one file is subsequently chosen.

Select database file

The user presses the button and selects a name from the name list, e. g. QCUSTCDT.



The selected name is then shown on the button and in the input fields *Database file* and input field *File member*.

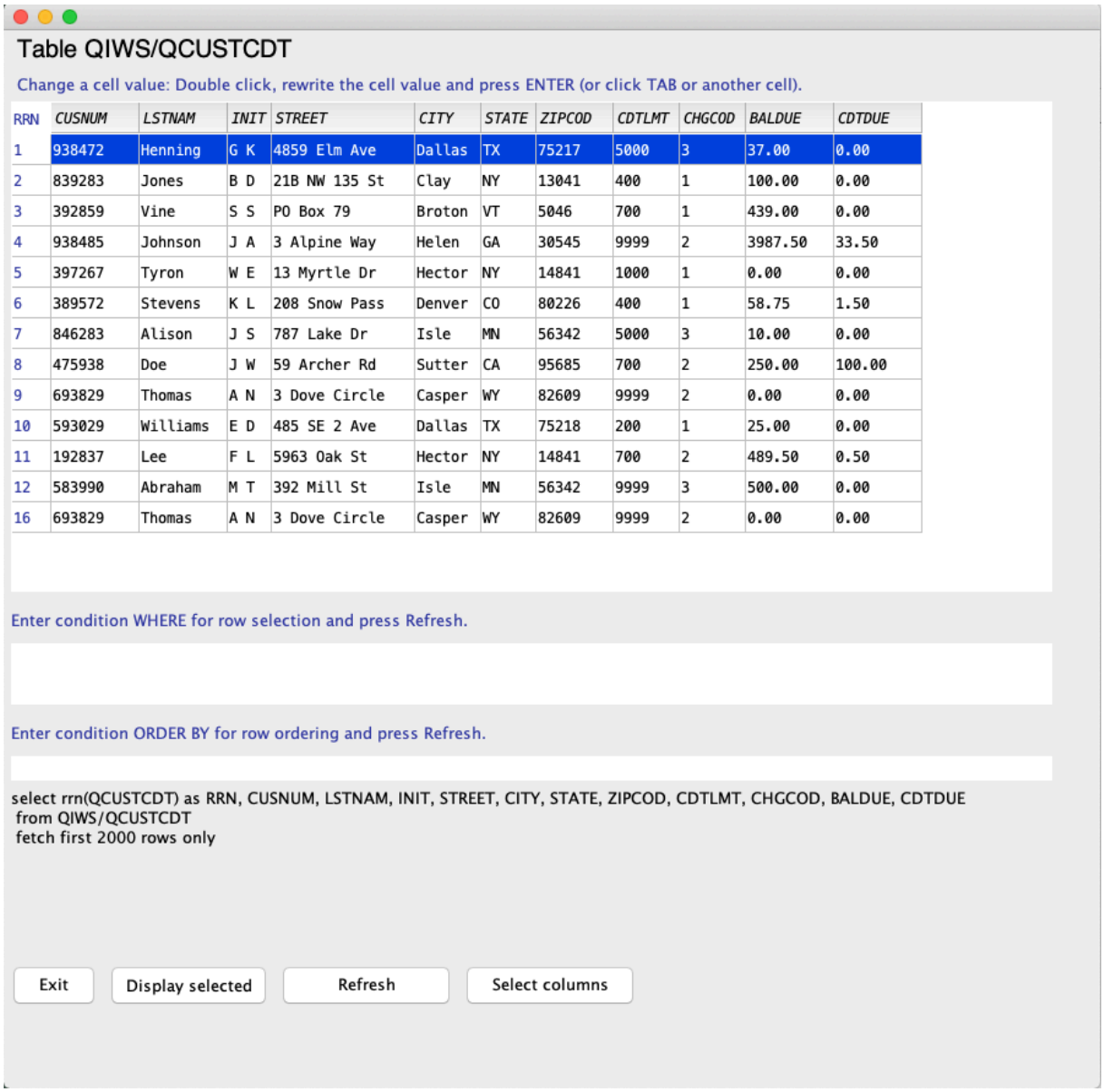
File member

The user may overwrite the default value, usually *FIRST, by a member name of the database file. See the chapter Data members of physical and logical files below.

Run

The button *Run* invokes the window with a table containing maximum number of the first records selected from the file specified in application parameters (see *Parameters* above).

File QCUSTCDT in library QWIS was chosen in this example.



The complete SELECT statement is shown under two text fields (above the button row). The list of columns and maximum number of records is visible from the statement. The expression *rrn(QCUSTCDT) as RRN* represents relative number of the record in the file. It is displayed for information in the first column of the table in blue color and is used for updating or deleting of the record. This value cannot be changed.

Selection and ordering of records

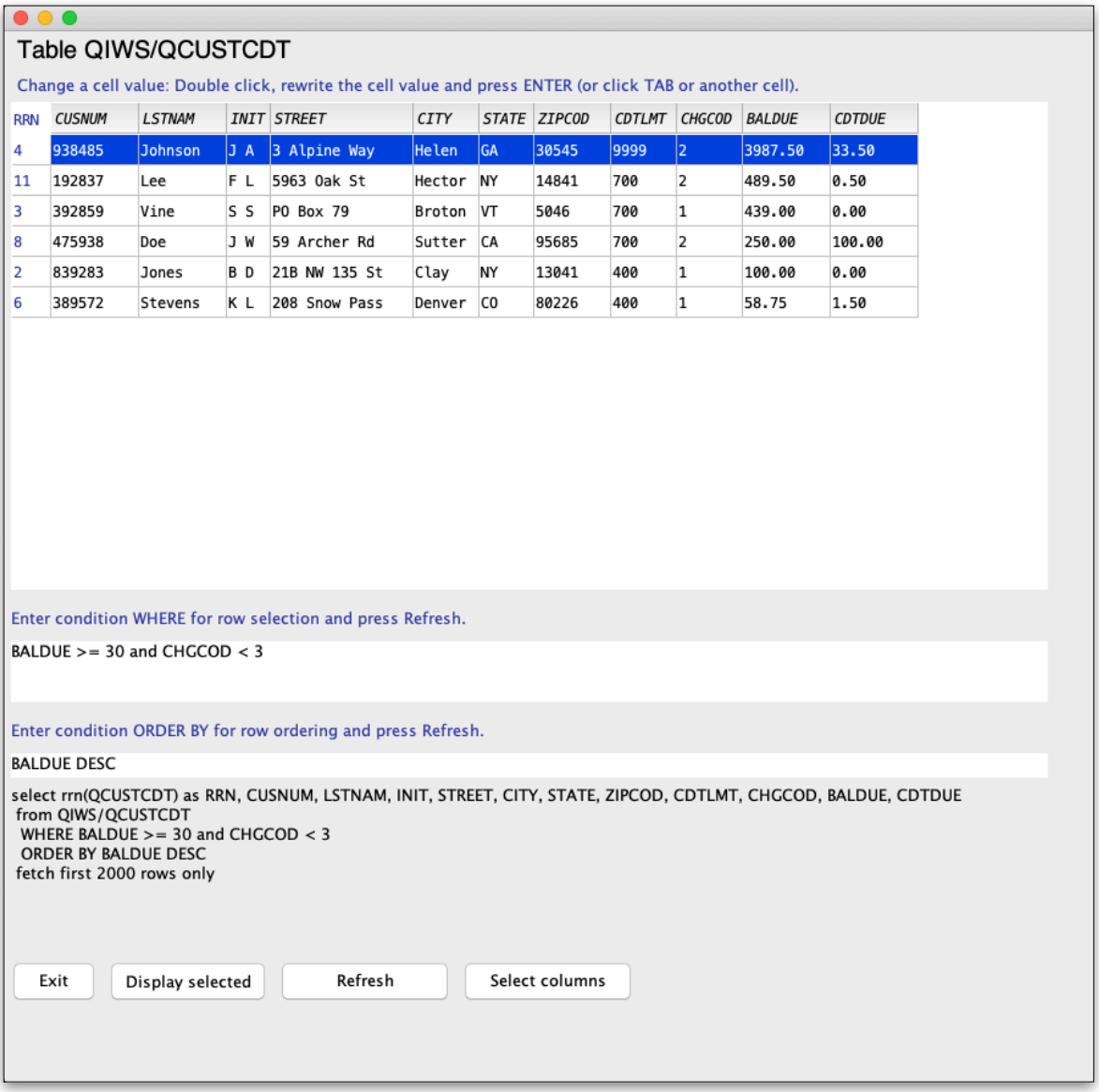
The first display of the file is a result of the SELECT statement without using clauses WHERE an ORDER BY, so the first unordered records are selected in the maximum number specified in Parameters. More accurate, the records are ordered by the relative record number named as RRN. This order corresponds to the arrival sequence in which the records were written to the file.

To display records of interest input fields under the table of records can be used.

- Expression as in the **WHERE** clause or the SELECT statement is entered in the first field.
- Expression as in the **ORDER BY** clause of the SELECT statement is entered in the second field.

After entering sn expression in one or both fields, the user presses the *Refresh* button and the program performs new selection and/or ordering of records.

For example, if the entry in the first field is **BALDUE >= 30 and CHGCOD < 3** and the entry in the second field is, the following window is displayed with selected and ordered records and the modified SELECT statement.



The whole text of the SELECT statement is shown under the both text fields including clauses WHERE and ORDER BY.

The expressions in input fields are stored in a text file named after the library and the file with ending .sel. This file is stored in the directory *selectfiles*.

Contents of the text file has the following form.

```
selection;
ordering
```

where *selection* is either empty text or the expression for WHERE clause and *ordering* is either empty text or the expression for ORDER BY clause. The two parts are separated by semicolon.

The file from the example is named **QIWS-QCUSTCDT.sel** and contains

```
BALDUE >= 30 and CHGCOD < 3;
BALDUE desc
```

Note 1: As an entry in both fields, the expression **RRN(QCUSTCDT)** denoting the relative record number can be used.

Note 2: To select a binary field (type BINARY, VARBINARY), function HEX() must be used because the value must be written as couples of hexadecimal characters. For example, if the field BIN01 of the BINARY type the expression for WHERE is entered as follows.

```
hex(BIN01) like '%cd%'
```

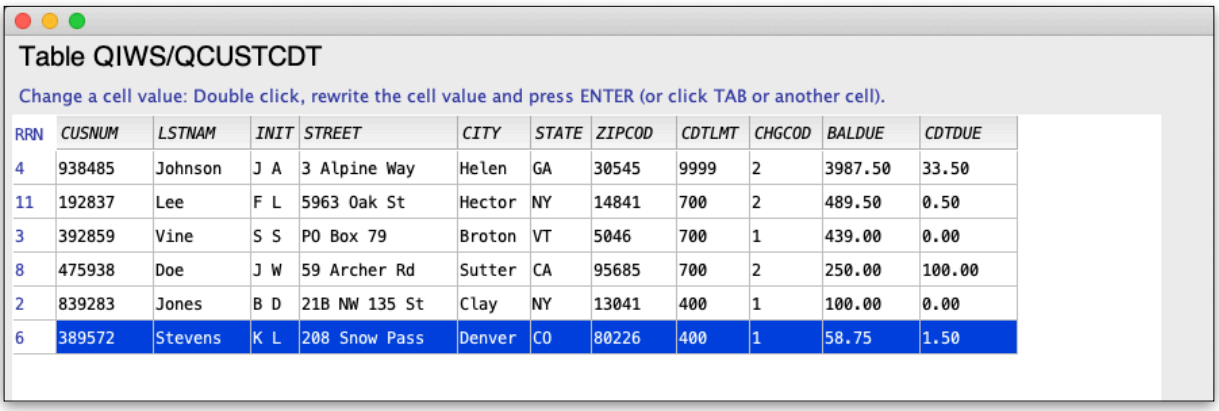
or

```
hex(BIN01) >= 1A
```

and the like.

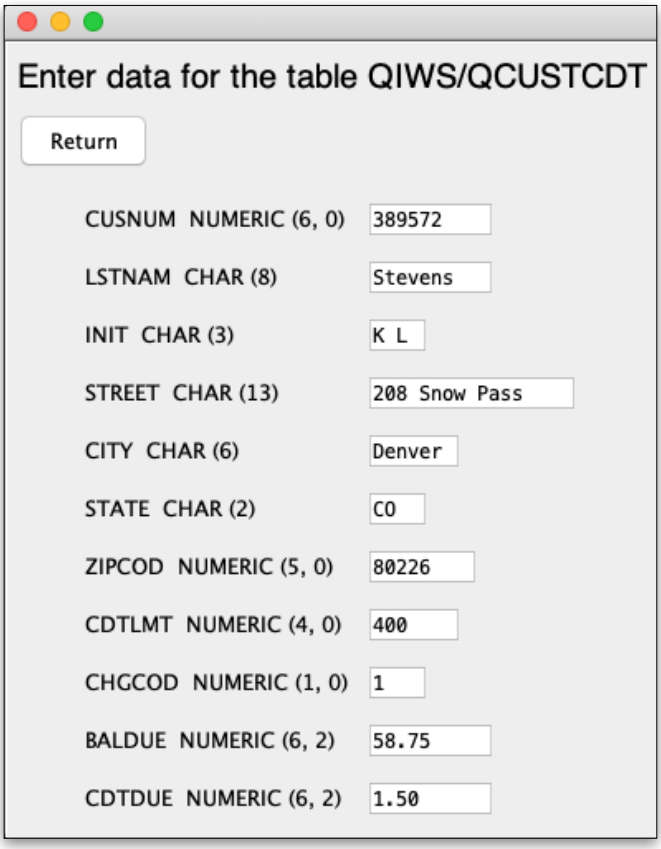
Displaying fields of the record

The user selects a row (record) from the list in the table and presses button *Edit selected*.



RRN	CUSNUM	LSTNAM	INIT	STREET	CITY	STATE	ZIPCOD	CDTLMT	CHGCOD	BALDUE	CDTDUE
4	938485	Johnson	J A	3 Alpine Way	Helen	GA	30545	9999	2	3987.50	33.50
11	192837	Lee	F L	5963 Oak St	Hector	NY	14841	700	2	489.50	0.50
3	392859	Vine	S S	PO Box 79	Broton	VT	5046	700	1	439.00	0.00
8	475938	Doe	J W	59 Archer Rd	Sutter	CA	95685	700	2	250.00	100.00
2	839283	Jones	B D	21B NW 135 St	Clay	NY	13041	400	1	100.00	0.00
6	389572	Stevens	K L	208 Snow Pass	Denver	CO	80226	400	1	58.75	1.50

A window with the current values of the fields is displayed.

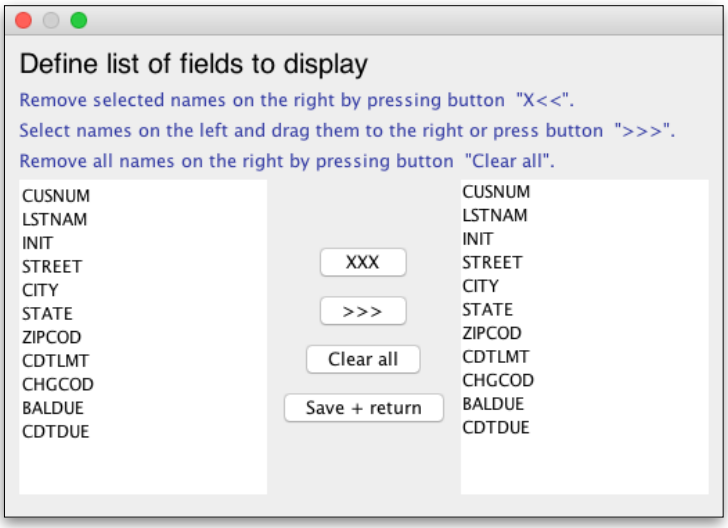


Button *Return* returns the preceding window.

Selection of columns

This function enables reducing and ordering colums in the list of records.

If the user presses the button *Select columns*, a window appears with two frames and buttons between.



The left frame contains always the complete list of fields (columns) of the database file. The right frame contains the same list at the beginning. The user can write a changed list in the right frame, possibly shortend, or reordered.

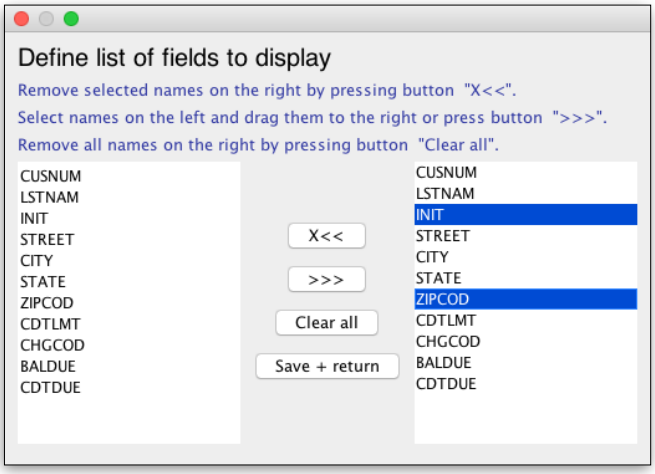
- Button ***X<<*** removes selected names from the right frame.
- Drag selected names from the left frame into a suitable position in the right frame or copy them by pressing button **>>>** to the end of the right frame.
- Button ***Clear all*** clears the whole contents of the right frame.
- Button ***Save + return*** saves the list of field names from the right frame to a text file in the directory "columnfiles" and shows the list with the selected and ordered columns.

The user can select one or more names from the frames. A single name is selected by clicking on it. A group of names is selected by holding *Shift* key. Individual names are selected by holding *Ctrl* key (*Cmd* key in macOS).

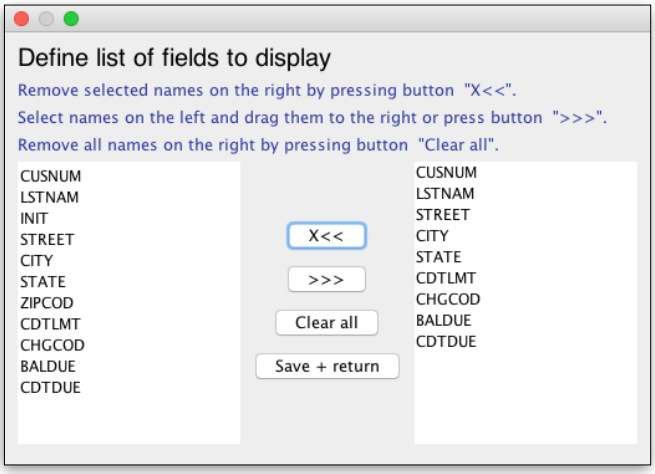
The list of selected fields amended by separating commas is stored in a work text file suffixed with *.col* and named by the library and database file name. The file is placed in the directory "columnsfiles". If this file is deleted, a new file with the complete column list is created when the database file is displayed.

Example

After pressing the button *Select columns* the following window is displayed. Select fields ZIPCOD and INT in the right frame.



Press button X<<, the right field list is shortened.



Press button *Save + return* the resulting list is stored in the text file and the list of records is displayed with a new field configuration.

The fiels list is stored to the text file named QIWS-QCUSTCDT.col and has the following form.

```
, CUSNUM, LSTNAM, INIT, CDTLMT, CHGCOD, BALDUE, CDTDUE
```

Note: The first comma separates the list from the field/column *rrn(QCUSTCDT) as RRN*, which is always displayed.

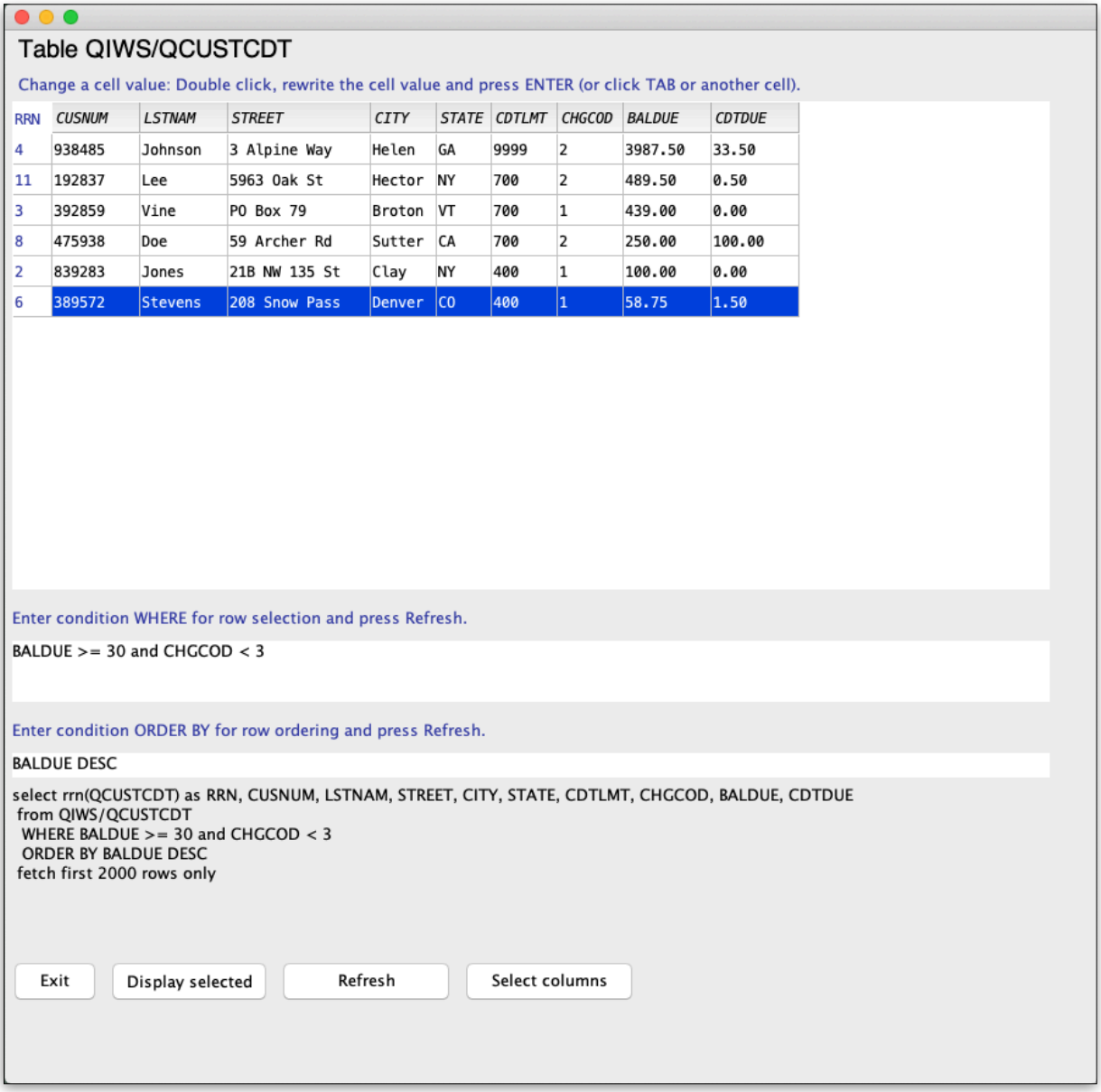
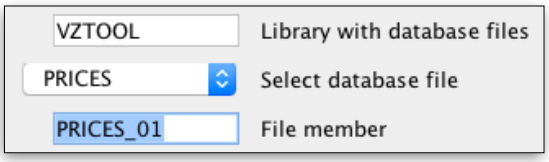


Table QIWS/QCUSTCDT
Change a cell value: Double click, rewrite the cell value and press ENTER (or click TAB or another cell).
RRN CUSNUM LSTNAM STREET CITY STATE CDTLMT CHGCOD BALDUE CDTDUE
4 938485 Johnson 3 Alpine Way Helen GA 9999 2 3987.50 33.50
11 192837 Lee 5963 Oak St Hector NY 700 2 489.50 0.50
3 392859 Vine PO Box 79 Broton VT 700 1 439.00 0.00
8 475938 Doe 59 Archer Rd Sutter CA 700 2 250.00 100.00
2 839283 Jones 21B NW 135 St Clay NY 400 1 100.00 0.00
6 389572 Stevens 208 Snow Pass Denver CO 400 1 58.75 1.50
Enter condition WHERE for row selection and press Refresh.
BALDUE >= 30 and CHGCOD < 3
Enter condition ORDER BY for row ordering and press Refresh.
BALDUE DESC
select rrn(QCUSTCDT) as RRN, CUSNUM, LSTNAM, STREET, CITY, STATE, CDTLMT, CHGCOD, BALDUE, CDTDUE
from QIWS/QCUSTCDT
WHERE BALDUE >= 30 and CHGCOD < 3
ORDER BY BALDUE DESC
fetch first 2000 rows only
Exit
Display selected
Refresh
Select columns

Data members of physical and logical files

The application can process data members of physical and logical files as if there were files. An alias object must be created in the library for the member whose name differs from the default of the file (*FIRST or file name). The *application* creates the alias object *itself* if it does not exist.

For example, the user selects the file VZTOOL/PRICES in *Parameters* and changes the input field *File member* to PRICES_01. If the member exists, the application automatically creates a new alias object named PRICES_01 and displays the list of records contained in the member. As soon as the application exits the data window the corresponding alias is deleted.



Alias objects

The alias objects are physical files (object type *FILE) with attribute DDMF. DDMF (Distributed Data Management File) is an object serving in access from a Local Location to a Remote Location in SNA, APPC architecture. Here, both locations are the same and the alias object serves as a mediator of access to the data file.

The user can also create alias objects using SQL statements using SQL statement CREATE ALIAS. A different means than this application must be used to do it, e.g. STRSQL CL command. For example, if the physical file PRICES has members PRICES_01, PRICES_02, the user creates alias objects using the following SQL statements.

```
CREATE ALIAS PRICES_01 FOR VZTOOL.PRICES(PRICES_01)
CREATE ALIAS PRICES_02 FOR VZTOOL.PRICES(PRICES_02)
```

Note: The alias names need not be the same as member names but it is practical.

The objects in the library look like this:

```
PRICES      *FILE       PF-DTA
PRICES_01   *FILE       DDMF
PRICES_02   *FILE       DDMF
```

The same method can be used for logical file members. If the logical file is named PRICESL and has members PRICESL_01, PRICESL_02, the user creates alias objects using the following SQL statements.

```
CREATE ALIAS VZTOOL.PRICESL_01 FOR VZTOOL.PRICESL(PRICESL_01)
CREATE ALIAS VZTOOL.PRICESL_02 FOR VZTOOL.PRICESL(PRICESL_02)
```