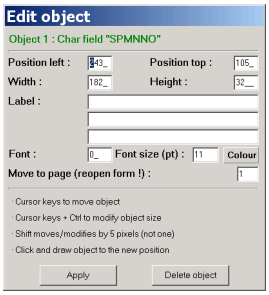


Hdb2Win 2.6

The Data Acquisition Forms – Format and Entry Screen Editor

Version 2.6 – March 2025

Content

Introduction

Data acquisition forms are essential for the work with the database system since they form the interface between the database and the user. Data can (manually) only be appended and modified by using the data acquisition forms. The forms are not part of the programme; they are separate parameter files and interpreted during run time. The forms are powerful and encompasses many functions. On one hand they are used for data recording, but on the other hand, they may provide multiple information about a record.

These parameter files are described here. As a rule, these files should not be manually modified; the comfortable editor for their modification is described below. If a manual modification of a data acquisition form is necessary

(1) Make a copy and work with the copy. Within the application you may assign this copy as default form.

(2) Always use the in-built editor of Hdb2Win, which is available through the Interpreter.

For compatibility reasons the data acquisition form files are in the ASCII format, not ANSI. So you should not edit them using the notepad of your operating system or any other text processor. The data acquisition form files may have comments; a comment starts with a semicolon and any text behind it is ignored. Their standard file extension is FRM.

Modifications in Hdb2Win 2.6

With the new version, the font size for the various objects is not recalculated anymore. There is a standard font size for the programme (text and buttons), and there are four font sizes for the data acquisition forms – labels, data fields, buttons, and lists. This values should be set according to the screen size and resolution. It can be find in Hdb2Win > Application library > Options > Adapt fonts.

The absolute font size for a specified object in a data acquisition form cannot be defined anymore. Although the command FSIZE is still recognised and the font size will be recalculated, it is better not to use this command. In all published application libraries, this command is replaced by FNTSIZE which is not an absolute value but a value relative to the standard font size. The value for FNTSIZE can be positive or negative and means an augmentation or reduction by full points (pt).

The same applies to the height of items in lists. In former versions, this value could be set in the options. This function still exists (Hdb2Win > Application Library > Options > Adapt Fonts > Height of items). The height of the items can be also modified in the data acquisition form description.

The function of showing images has been greatly enhanced. But to make use of it, some manual programming is required.

Table and form

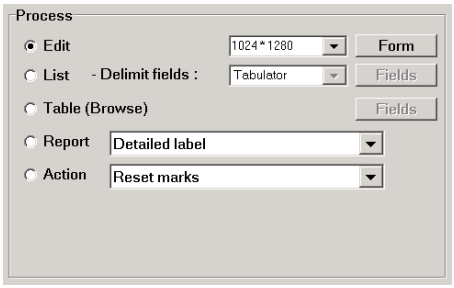
When a record of a table is opened in the Edit mode, the systems looks for the data acquisition form file. This file has normally the same name as the table. But it can be also defined that a different (more simple, more extended) form file is opened. There are two options to change the name of the form file. First, it can be directly defined in the database parameter (file extension DB2) with the command `setform` (please note, this is one word).

```
SETFORM TO <file name>
```

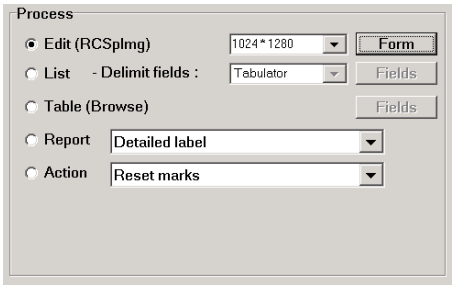
as for instance

```
SETFORM TO Authors-2.frm
```

A more elegant solution is offered in the Search Module of the Application Library. From Hdb2Win V. 2.5 on you will find on the right hand side of the output mode Edit a button 'Form'.



When clicking on this button, a file open dialog offers to select form files that starts with the first three letters of the table name. For the table Authors the files Authors.frm and Authors-2.frm are shown.



When a newly selected form file is different from the standard form file, its name is shown in parentheses right to the word Edit. The selection is persistent because it is stored in the Hdb2Win registry. When opening the table in another occasion, the previously form file will be used. When selecting a table for editing the form file you are asked which form file you want to edit.

Format of the forms

General

The data acquisition form file usually starts with a describing part of up to three lines (for the three languages German, English and Spanish). The describing part starts with a semicolon, the letter C, and a colon, a space, followed by the text.

```
;C: Autoren
;C: Authors
;C: Autores
```

Any line beginning with a semicolon is understood as a comment. So you may add comments to the file.

After the description of the file content follows a general part. The code word DBALIASA assigns an understandable name to the table. The code word DBALIAS in former versions of the database system is not used anymore.

```
DBALIASA  "Autoren","Authors","Autores"
```

The available client area for the smallest data acquisition form has a size of 768 × 489 pixels. This area corresponds to the (formerly) smallest supported screen size of 800 × 600 pixels. If the user has a larger screen, the position and size of the objects are correspondingly recalculated (but not the font size and not the line height). If a data acquisition form is modified using a larger screen, the applied screen size is indicated in the header of the file:

```
AREA      1280,1024
```

All forms of published applications are designed for the screen size of 1280 × 1024 pixels. In the form files still the old values for 800 × 600 pixels are kept and there is therefore no indication of the screen size. Do not modify these values, let only the system set them.

The form can be reduced in size:

```
RESIZE    x,y
```

Both values are percentage values and may vary between 10 and 90%. Please note, that this reduction does not rearrange the objects of the form. Objects beyond the visible area are not shown (except the buttons on the bottom). This function was only introduced to allow smaller forms on small screens.

A data acquisition form can have multiples pages in the form of file cards. A maximum of 16 file cards is possible. When the form has more than one file card, the number of file cards should be indicated:

```
PAGES     4
```

A lacking page number is actually not a problem because the system will automatically set the number of pages according to the page numbers indicated by the objects.

After the general part, the objects are declared and described. Any declaration must be accompanied by an identification number that may reach from 1 to 1000. The objects are grouped and each group has a certain range of numbers.

1 .. 128	data fields	400-409	connections to other tables
129-144	page labels	410-419	lists
150-190	panels, lines, and images	420-439	list actions
198	*enter proc image*	440-459	replace definitions
199	*blue hint in top box*	460-500	unassigned
200/201	*copyright*	501-959	reserved for the user
202-399	labels		

The numbering of the groups is not mandatory. Only the object 198 to 201 (marked above in italics) must not be modified. Any object must be declared with an object number. When attributes are added, the value '1024' can be used as a reference to the last definition. As for instance, the declaration of a data field gives the name of the field and the ID number.

```
FLDID     SOURCE,2
```

A following attribute such as the position refers with the ID 1024 back to the last declaration, in this case to the data field.

```
POSOBJ    1024,180,75,560,1,1
```

Different languages (as for labels) are separated by commas. After German follows English and then Spanish. More languages cannot be added. If you wish to create forms in your own language, it could be a good idea to overwrite the German or Spanish version. You cannot append a fourth language; this will not be understood by the Interpreter since it expects only three languages.

The objects are created according to their ID number, and also placed in this order on the page. This is important to know when images are used. For the declaration of objects, besides the declaration, there are mandatory attributes, and optional attributes. Both the declaration and mandatory attributes are printed in bold in this manual.

Data fields

Data fields of the tables are represented by text areas (for characters, numbers, items of other tables), buttons (large texts), or small boxes (logic fields). Only fields listed in the form are displayed for editing:

```
FLDID     SOURCE,2[,<Enabled/Disabled>[,<with or without Event hand-
           ling>]]
FALIAS    2,"Quelle","Source","Publicación"
POSOBJ    2,180,75,560,1,1
PREVIOUS  2,1
NEXT      2,3
FONT      2,0
FNTSIZE   2,0
COLOUR    2,$000000
BGCOL     2,$F0F0F0
HINT      2,"Auswahl der Quelle des Zitats (Literaturstelle)","Selection
          of the source of the citation (publication)","Selección de la
          referencia"
```

Declaration

FLDID Definition of the object and assignment of an ID number to a data field. The ID is then used to identify the field. The declaration may have two numerical optional controls. The first value indicates whether the field is enabled (default setting, value=0) or disabled (value=1). The second value indicates whether the event handler should be started (value=1) when the content of the field is modified. The default value for the event handling is zero.

Obligatory attributes

POSOBJ Position of the field: x start position, y start position (both in pixels), width of the field in pixels, height of field in lines (not in pixels), number of the file card.

PREVIOUS ID number of the previous field (where to go when the up cursor or Shift+Tab keys are pressed). Not necessary when the fields are in a consecutive order, but necessary for the first data field on the page.

NEXT ID number of the next field (where to go when down cursor or tab keys are pressed). Not necessary when the fields are in a consecutive order, but necessary for the last data field on the page.

FLDLBL To label a button (as a text field) refer to the ID number of the object.

```
FLDLBL      12,"Notiz","Note","Nota"
```

The command can be also used to label a Checkbox, but it is not mandatory.

Optional attributes

FALIAS Alias name.

FONT Font number according to the font table. Can be skipped if standard font is used. A dynamic font is defined by giving the field name of the field that defines the font:

```
FLDID       OTITLE,10
FALIAS      1024,"Orig.Titel","Orig.title","Título original"
POSOBJ      1024,120,215,630,2,1
FONT        1024,TPLANE     ; TPLANE is a numeric field
```

FNTSIZE Font size. This is relative to the global font size and can be also a negative value.

STYLE Font style. The font style is bit wise coded. Bit 1 = bold, bit 2 = italics, bit 3 = underlined, bit 4 = strike out.

COLOUR Text colour. Can be skipped if black. The value is a 3-byte hex number ($BBGGRR). The three bytes corresponds to the value between 0 and 255 for the blue, red and green colour. If you want to paint the background colour of a field with the colour value of this field put as colour value $FFFFFF:

```
FLDID       FACOLOUR,3
FALIAS      1024,"Farbe","Colour","Color"
POSOBJ      1024,400,100,100,1,1
COLOUR      1024,$FFFFFF
```

If the value in the field is modified, the background colour of the field changes.

BGCOL Background colour. Can be skipped if white. The value is a 3-byte hex number: $BBGGRR. The three bytes corresponds to the value between 0 and 255 for the blue, red and green colour.

HINT The hint that is displayed when the mouse is moved over the field.

Short list

A short list is a special form of the data field and refers to a numeric data field and has a limited number of items. Not the numbers are shown, but the entries depending on the number.

```
CBOX 67,"Entry1;Entry2;Entry3",fieldname,1[,0/1]
```

The fourth parameter indicates which item is the default value. Zero shows a question mark. This type should be combined with the data fields and can have all attributes that has a data field. The attribute FLDLBL must be used to list the entries in the various languages. If the entries are the same for all languages, the attribute can be suppressed (because they appear in the declaration). The last optional parameter disables (1) or enables (0) the field.

```
FLDLBL    1024,"sehrselten;selten;gelegentlich;häufig;sehr häufig", "very
          rare;rare;occasional;common;very common","muy raro;
          raro;occasional;común; muy común"
```

Mandatory and optional attributes as above.

Graphics

Graphics are another special form of data fields. They are declared as graphics in the DB2 configuration files. In the form file, they are declared as normal data fields as described above. The position indicates the area for the image, the file name and a select button. Width and height are expressed in pixels. Previous and next refer to the ID number of the same field. The systems adds automatically a field for the name and a button to select the graph. This type should be combined with the data fields and can have all attributes that has a data field.

```
FLDID     SPICT,23
FALIAS    23,"Abbildung","Graphic","Imagen"
POSOBJ    23,28,50,700,425,4
PREVIOUS  23,23
NEXT      23,23
```

Note that graphics are not the same as an image. Graphics should occupy a large area of the form.

Labels

Labels are all captions in the edit form.

```
LABEL     203,"Zitate","Citation","Cita"
POSOBJ    203,28,10,0,0,1
FNTSIZE   203,10
COLOUR    203,$0000FF
```

Declaration

LABEL Definition of the object with the assignation of an ID number followed by the caption, if you wish in multiple languages (German, English, Spanish). It does not make sense to add a fourth language, better replace any of the three languages you do not need.

Obligatory attributes

POSOBJ Position x and y in pixels as above. Width and height may be zero because they are defined by the text size.

Optional attributes

FONT Font number according to the font table.

FNTSIZE As above.

STYLE Font style. The font style is bit wise coded. Bit 1 = bold, bit 2 = italics, bit 3 = underlined, bit 4 = strike out.

BGCOL Labels are normally transparent. This is not the case when a background colour is defined. Can be skipped if the standard form colour. The value is a 3-byte hex number: $BBGGRR. The three bytes corresponds to the value between 0 and 255 for the blue, red and green colour.

COLOUR As above.

Page labels

To label file cards, use a separate ID number for each file card:

```
PAGELBL   131,3,"Abbildung","Figure","Imagen"
```

The ID is followed by the page number and the caption.

Optional attributes

FONT Font number according to the font table.
FNTSIZE As above. The default font size is that of the labels.
STYLE Font style. The font style is bit wise coded. Bit 1 = bold, bit 2 = italics, bit 3 = underlined, bit 4 = strike out.

The attributes should be only assigned to the first page and are valid for all pages. It is not possible to select different attributes for different pages. A font colour cannot be defined.

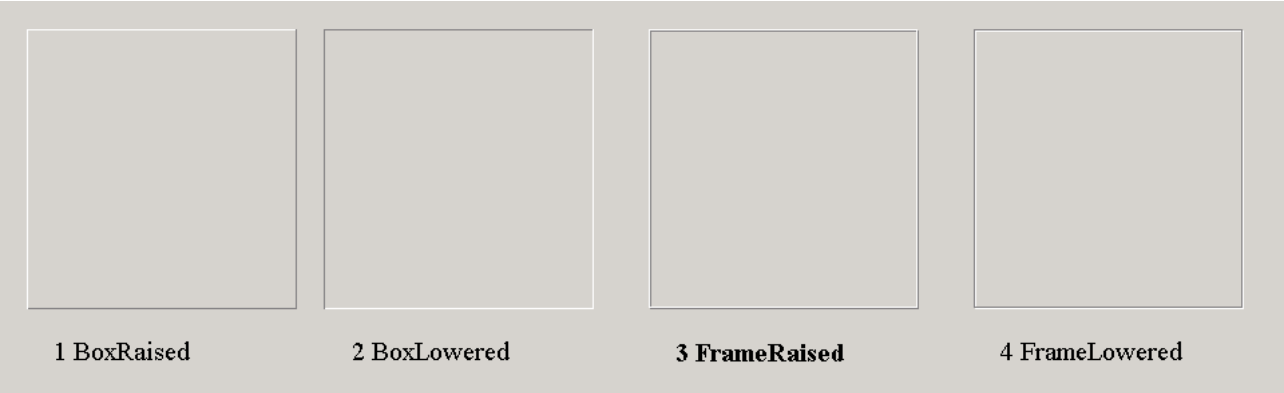
Panels and lines

The ID number is followed by the start value for x and y, the width and height in pixels and the page number:

```
BEVEL     150,8,0,752,400,1
```

Optional attribute

STYLE There are four styles available (see example below) that can be coded as words or as numbers.
1 or BOXRAISED
2 or BOXLOWERED
3 or FRAMERAISED (default value)
4 or FRAMELOWERED.



Images

An image is defined as an area to show a bitmap or vector graphic (BMP, GIF, JPG, WMF). The image file can only be selected when this happen in a programme. The form or a programme have to assign the file. The ID number is followed by the start value for x and y, the available space (width and height) in pixels and the page number.

```
IMAGE     180,25,155,725,325,1
ASSIGN    1024,IMAGES\LOGO.JPG
ASSIGN    1024,Species-Img1.prf
```

Optional attribute

ASSIGN The attribute assigns either an image file to the image or indicates a programme (file extensions PRF and LBA are recognised as programmes) that provides the image.

Assignation of an image

There are three different methods to assign an image.

(1) Direct assignation

Just give the file name in the ASSIGN attribute.

(2) Indirect assignation

Write a programme that assigns the image. Let's assume that the name of the image corresponds to the record number. See the example programme below

```
DEFINE    s,c
MOV       s,'R'+str(recno)+'.JPG'
IMG.TOC   imparam
IMG.SHOW  &s
MOV       LASTIMAGENAME,s
EXIT
```

Be sure to use the IMG commands as given above. If you miss the fifth line, the image cannot be enlarged when clicking on it.

(3) In an ENTER or EVENT procedure

The image is assigned as above but you can only assign an image to the image with the object number 198 because the ENTER or EVENT procedures do not know which image should be addressed.

Clicking on any image in the data acquisition form expands it to the screen size. This works only when the image is not covered by a label, a data field or even a bevel. So always assign images to high object numbers to be sure they are on the top of all objects.

Lists

Lists are shown in a white area. The content of a list is defined by an external program.

```
LBOX      410,RCSITES1.LBA,17
POSOBJ    1024,20,30,355,440,2
```

Declaration

LBOX An ID number is followed by the name of the programme that fills the list and by a mode.
The mode is coded bit wise:
bit 1 = 1 The list is filled when the file card is entered.
bit 2 = 1 The list has a head line for selection.
bit 3 = 1 Works also in the query-by-example mode.
bit 4 = 1 Always rebuild when page is entered.
bit 5 = 1 Do not sort the items in the list.
bit 6 = 1 Show two columns.

Obligatory attributes

POSOBJ The (x, y) position and size of the list. All values in pixels, except the file card number.

Optional attributes

COLOUR As above.
FONT Font number according to the font table.
FNTSIZE Font size relative to the standard font size for lists. This value can be also negative. Can be skipped when the standard value is applied.

STYLE Font style. The font style is bit wise coded. Bit 1 = bold, bit 2 = italics, bit 3 = underlined, bit 4 = strike out.

HINT Hint on the list.

HEIGHT A relative value to decrease or increase the vertical space for one line. There is a global value for the height of list items that is defined in the font settings (Hdb2Win > Application Library > Options > Adapt Fonts > Height of items). This value can be (relatively) modified.

List action

List actions are programmes started to display or modify data. They are represented by a button. Normally (but not necessarily) they are connected to a list.

```
LBACTION  413,410,CITAT13.LBA,0
POSOBJ    1024,610,150,140,32,2
FLDLBL    413,"Zitat &entfernen","Remove &citation","Remover &citación"
HINT      413,"Entfernt das markierte Zitat aus der Synonymliste","Removes
          the marked citation from the list"
```

Declaration

LBACTION The ID number is followed by the ID number of the list to which the action refers, the name of the programme and a mode. The mode is coded bit wise:
bit 1 = 1 The data buffer is modified by the action and has to be restored.
bit 2 = 1 Do not rebuild the list after the action.
bit 3 = 1 Action has to be allowed also when no item from the list is selected.
bit 4 = 1 This action is started by double clicking on a list item.
bit 5 = 1 If bit 1 is set, the status of the Save button is changed.
bit 6 = 1 All lists on the page are rebuilt.
bit 7 = 1 It is necessary to save the current record before applying this action.
bit 8 = 1 This action is started by any item is marked and the DEL (or delete left) key is pressed.
bit 9 = 1 This action is started when the INS (or Ctrl-N) key is pressed.

Obligatory attributes

POSOBJ Position and size of the button in pixels.

FLDLBL Caption of the button. The et sign ('&') before a letter indicates that this letter will be underlined and the action can be started by pressing Alt+letter.

Optional attributes

FONT Font number according to the font table.

FNTSIZE Font size relative to the standard font size for buttons. This value can be also negativ. Can be skipped when the standard value is applied.

STYLE Font style. The font style is bit wise coded. Bit 1 = bold, bit 2 = italics, bit 3 = underlined, bit 4 = strike out.

HINT Hint on the action.

Non-relational interconnection with other tables

Non-relational interconnections with other tables as described above are represented by a button:

```
CONNECT   400,OCCURR,OCITATE,LOCALITY,0
POSOBJ    400,180,410,240,24,1
FLDLBL    400,"Lokalitäten zu diesem Zitat","Localities for this
           citation"
```

```
HINT      400,"Lokalitäten anzeigen, entfernen oder hinzufügen",
          "Indicate, remove or add localities"
```

Declaration

CONNECT The ID number is followed by the name of the interconnected table (here OCCURRENCES), the name of the data field in this table which is refers to the current table (here *ocitate* in OCCURRENCES, which refers to a record in CITATIONS) and the name of the data field to be selected in the interconnected file (here *locality* in the table OCCURRENCES).
The mode is coded bit wise:
bit 1 = 1 A newly created record will be displayed (see description of tables).
bit 2 = 1 The items in the list are treated as graph files and are displayed.
bit 3 = 1 The item is not dedicated to one table but shares it with other tables (OWNER mode).
bit 4 = 1 The item is a document which has to be opened by an external programme.
bit 5 = 1 Lock selection.
bit 6 = 1 Allow only one entry.

Obligatory attributes

POSOBJ Position and size of the button in pixels.
FLDLBL Caption of the button. The et sign ('&') before a letter indicates that the letter will be underlined and the action can be started by pressing Alt+letter.

Optional attributes

FONT Font number according to the font table.
FNTSIZE Font size relative to the standard font size. This value can be also negativ. Can be skipped when the standard value is applied.
STYLE Font style. The font style is bit wise coded. Bit 1 = bold, bit 2 = italics, bit 3 = underlined, bit 4 = strike out.
HINT Optional addition of a hint to the connection.

Replacements

Replacements are non-visual objects used to replace automatically data fields, labels, buttons, or symbols with data (text).

```
F7REPL    440,CREATED,date,3
```

A new ID number is followed by the name of the field (number of label, list action, or name of the symbol), the replace expression and a mode:

1 = field is always automatically replaced

2 = field is only replaced if empty (or if F7 is pressed, from Hdb2Win 2.5.2 on)

3 = field is only replaced when F7 is pressed

4 = as 2 but the field is also replaced when zero

10 = field is replaced when the record is modified

20 = start programme

The name of the field can be also a number that refers to a label, button, or a list action:

```
LABEL     250,"(replaced during run-time)"
POSOBJ    1024,100,200,100,1,1
...
F7REPL    450,250,spmnloc.age.agename+'/'+spmnloc.lithos.luname,1
```

The name of the field can be also a variable:

```
F7REPL    450,lastloc,spmnloc,1
```

There are no attributes. If addressing a field that is related to another table, and you want to replace the field only when empty (mode 2), please use mode 4. Otherwise it will not work.

From version Hdb2Win 2.5.2 on, the replace object can be also used to start a programme:

```
F7REPL    <ID>,<data field>,<programme name>,20
```

as for instance

```
F7REPL    300,image,editimage.prf,20
```

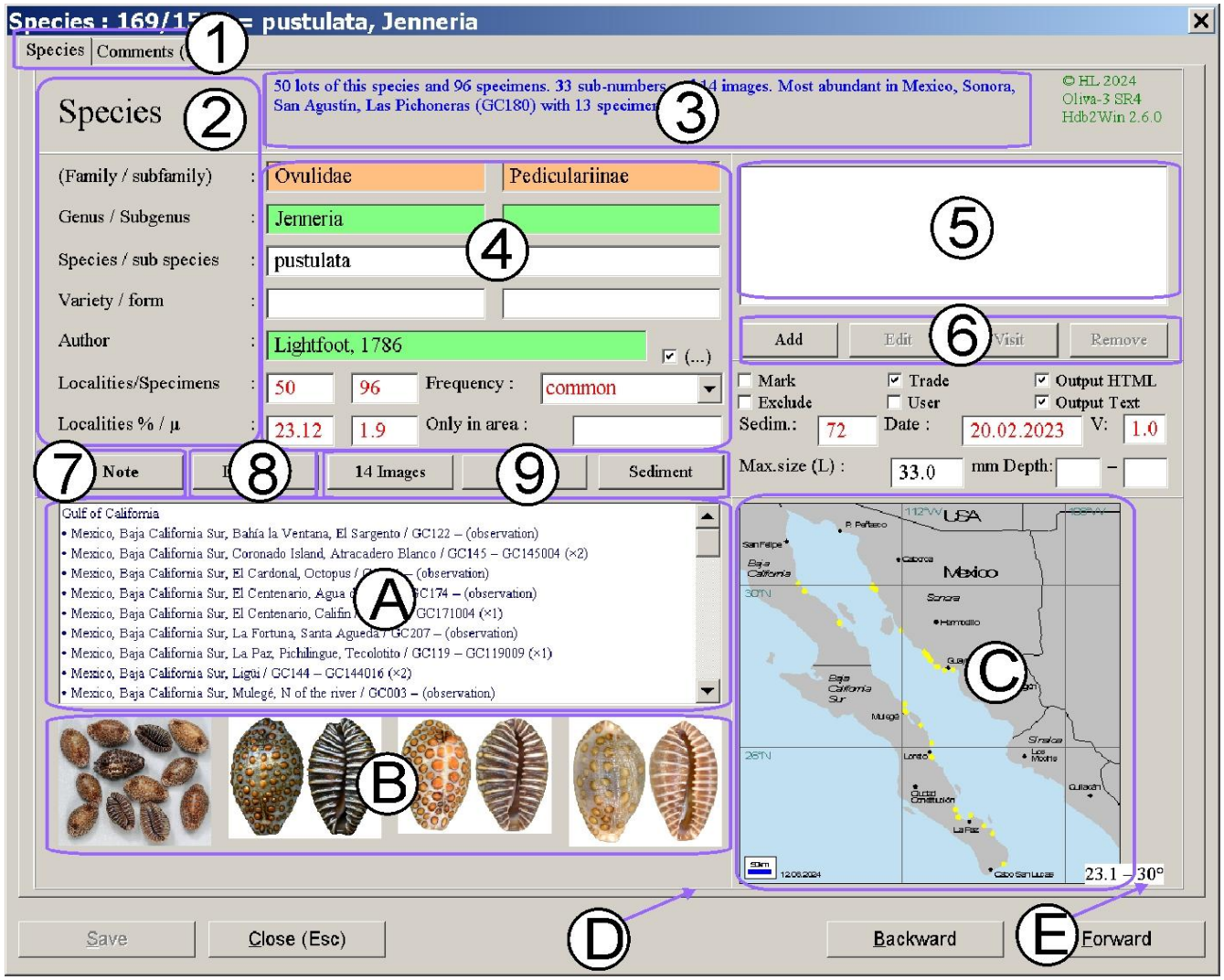
The programme must be related to a data field and the programme is only started when the key F7 is pressed.

Tips

1. Make backup copies before modifying edit data acquisition form files.
2. Skip as much as possible. Many options (PREVIOUS, NEXT, FONT, FNTSIZE, COLOUR) will set to best fits with standard values. The attributes NEXT and PREVIOUS can be skipped when fields are in consecutive order.
3. After the declaration of objects (FLDID, LABEL, LBOX, LBACTION, CONNECT), use '1024' instead of the object number to refer to the previously declared object:

```
FLDID    SOURCE,2
FALIAS   1024,"Quelle","Source","Fuente"
POSOBJ   1024,180,75,560,1,1
HINT     1024,"Auswahl der Quelle des Zitats (Literaturstelle)",
         "Selection of the source of citation (publication)",
         "Selección de la fuenta (referencia)"
```

4. To create a new file card, just increase the number of pages.
5. Complete text attributes (`label`, `fldlbl`, `hint`) only as far as needed. When the Spanish label lacks, the English label is used, lacks the English, the German is taken.
6. After modification activate "Check consistency of forms" and "Display warning errors" in the Hdb2Win options (Hdb2Win > Application Library > Options > Error check). If everything works fine, you should deactivate this options again.
7. In predefined applications (PaleoTax, PalCol, Oliva) the standard part of the form file is finished with the line
   `; --- End of Standard`
   Do not remove this line since it separates published standard items and personal items. If you modify the form, new items will be places after this line. This makes form updates easier.
8. Within the application library, the form file can be changed within the search form. If you want to modify an existing form, you may make a local copy and load this copy in the search form. The assignation of the form file is persistent.



This data acquisition form unifies most objects introduced above.

1 Page labels.

2 Labels.

3 Information on the current record. This must be object 199 when filled in an Enter procedure.

4 Data fields.

5 List with actions.

6 List actions.

7 Text field.

8 Connected table.

9 Actions without list.

A List without actions.

B Small images, addressed with programmes.

C Large image. This is object 198 and is filled in an Enter procedure.

D Panel.

E Label with white background colour that is set by a replace object.

	BEVEL	CBOX	CONNECT	F7REPL	FLDID	IMAGE	LABEL	LBACTION	LBOX	PAGELBL
Declaration		×	×	×	×		×	×	×	×
Declaration with position	×					×				
Obligatory attributes										
POSOBJ		×	×		×		×	×	×	
PREVIOUS		×			×					
NEXT		×			×					
FLDLBL		×	×		×*			×		
Optional attributes										
BGCOL					×		×			
COLOUR		×			×		×		×	
FALIAS		×			×					
FONT		×	×		×		×	×	×	×
FNTSIZE		×	×		×		×	×	×	×
HINT		×	×		×			×	×	
STYLE	×	×	×		×		×	×	×	×
HEIGHT			×						×	

*) A FLDLBL command has only effect when a text (Memo) field or a logical field is addressed.

Lists and list actions

Lists are created by programmes of the interpreter. The name of the programme is indicated in the declaration command for the lists. The programme should always check whether a new recorded is appended:

```
CMP       recno,reccount
JA        exit              ; if true, append mode, so exit
```

In this case, the programme is finished without any action. The programme should also revise whether the table of interest is opened and not empty. Let us assume you want to compile in the data acquisition form of genera a list of species. In order to do so, you need to change the table, but before you need to save the current record number of the current table:

```
POB       species           ; Does "species" form part of the database?
JNE       exit              ; no! so better leave
DEFINE    genrec,i
MOV       genrec,recno      ; save the current record in the Genera table
FILE      species           ; change the table
RESET                       ; very important
CMP       reccount,0        ; are there any records?
JE        onerror           ; no! so better leave
```

The label `onerror` is a reserved label name. In the case of an error, the programme goes to this label. As for instance in the above example, you need to return to the table of genera and set the current record:

```
:oneeror
file      genera
go        genrec            ; set correct record
:exit
exit
```

In order to address the list, the following commands essential:

```
LB.TOC    lbparam
```

It connects the Interpreter with the current list. The list is cleared with

```
LB.CLR
```

The items of the list are added with the following command

```
LB.ADD    <svar>
```

The argument can be any string variable. Let us continue the programme from above:

reset

```
:begin
CMP       genus,genrec
JNE       skip
LB.ADD    genus.gname+#32+sname
:skip
SKIP
JNEOF     begin
```

Please note that per default the items in a list are alphabetically sorted. If you wish the list not to be sorted, you need to set bit 5 in the declaration. If you activate an index, you must close this index in the section following by the `onerror` label.

When list actions are used, it must be possible to identify the selected list items. So the programme that is started by a list action need to now which item was selected. This happens by the help of two symbols, `param1` and

`param1s`. When a list action is called, the corresponding programme has access to the selected items in the list via `param1s`, which is a string variable. This string variable is in the ANSI, not ASCII format. You should convert it into ASCII before using it because it:

```
DEFINE    cs,c
MOV       cs,ASCII(param1s)
```

When the item in the list ends with an asterisk, followed by an index, the index number is stored in `param1` which is an integer.

Let us assume the list the following items

```
Spring * 1
Summer * 2
Fall * 3
Winter * 4
```

When marking the second item and clicking on one of the list actions connected to this list, the variable `param1` contains the integer value 2, whereas `param1s` contains the string Summer. If you need access to the record number, you need to adapt the programme as for instance

```
LB.ADD    name+' * '+str(recno)
```

The record number helps in cases where the list items are very complex and do not allow to identify a record in a table from the item.

Programmes related to forms

There are three programme types that are closely related to the data acquisition forms. All have the same name as the table (not the form!), but distinguish in their extension. In complex applications (such as PalCol or Oliva) these programmes (all written for the database interpreter) are frequently used and form part of the respective standard.

Enter procedure

The enter procedure (file extension ENT) is called when the record is displayed. When switching to another (the next or the previous) record, the procedure is called again. The procedure updates the box in the header area of the first page providing data about the current record and refreshes images.

Event handling

There are no separate Event procedures for data fields. There is one file (extension EVT) that is called when a data field is modified. The field must be marked in having event handling: event handling is only started when programmed in the FLDID command for the data field. To be able to know which field was modified, a symbol `imfield` exists that inform about the field. The number is the current number of the field in the table, not the ID number of the data acquisition form. So the event handling does not work anymore properly when fields were inserted or deleted before the field for which an event handler was written.

Exit procedure

The Exit procedure (extension EXT) is called when the record is left through the save button (or PgDn, F10 or F12 pressed). It generally serves to control the data consistency. The variable return (that may not be declared) informs whether the record was modified (return=1) or not (return=0). In the case, the record was modified, the programme may check any data consistency and then setting the return value. A return value of zero means, that the record cannot be saved. Depending on a return value, the user may return to the same record (return = 0), correct values and leave it again. When the record was not modified, the exit procedure is called in any case but it has

no meaning when the return value is zero. But if the record was modified and the return values is zero, it will not be saved.

```
CMP       return,0    ; this means the record was not modified
JE        exit        ; so no test is necessary
...       ; check something
MOV       return,0    ; 1 means that everything is OK (0 for not)
EXIT
```

Album

The Album is a page with images. Images can be enlarged, one image can be selected. The Album function is complex. Please compare to the manual of the Interpreter.

Creating new tables and forms

When creating a new table (Hdb2Win > Application Library > Options > Tools > Create new table) a data acquisition form file is automatically created. You may edit this form directly afterwards or at any time you want.

Modifying the structure

When modifying the structure of an existing table (Hdb2Win > Application Library > Options > Tools > Modify structure) the data acquisition form is automatically modified. You may edit this form directly afterwards or at any time you want.

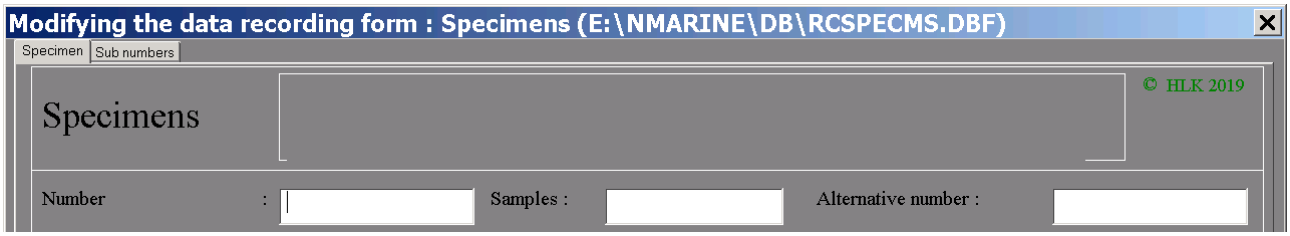
Editor

The data acquisition forms can be edited with the Data Entry Screen Editor (Hdb2Win > Application Library > Options > Tools > Modify form). It is a good idea to make a copy of a form which you wish to modify. When clicking on the button a file select menu is opened that shows all files of the current directory with the extension DB2. This file type describes the table. If you cannot find the file you want to modify, change the extension from *.DB2 to *.DBF. The data acquisition forms file must exist. You are not able to create a new file (but there is a programme that creates a form from a DBF file, see below).

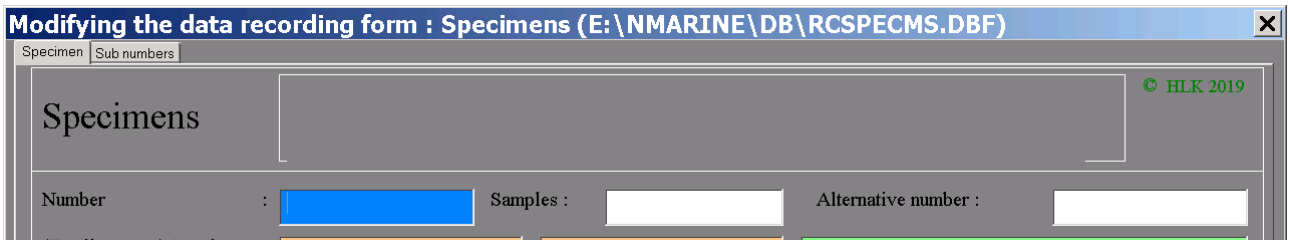
If you want to import data from an external source, it is very probable that you have got a DBF table without a configuration file (DB2) and without a data acquisition form file (FRM). In this case you need to create at least the second one. To do so, open Hdb2Win, select the Interpreter and search in the working directory for the programme makeform.prf ("Creates from a DBF file a format file"). Start it and select the DBF file. A new form is created that must be renamed before use. Now you can modify the from with the editor.

After selecting a table (a DB2 or DBF file) the table is opened with all related and subordinated tables. If one of these tables is missing, the form cannot be modified. When the table was successfully opened, the data acquisition form is shown in the currently selected screen format. It makes sense to select the most used screen format beforehand.

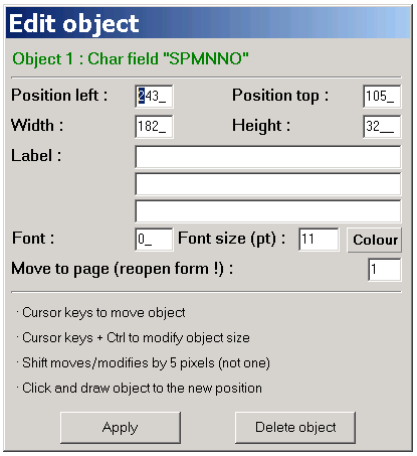
In the editor, the data acquisition form looks similar when opened in the edit mode. Just the background is darker.



When clicking in/at one object, several things happen. First, the colour of the object changes to light blue. Except for buttons that just turn into bold, and panels/lines that cannot change the colour.



Second, a small window opens to the right hand side of the form.



In the top of this windows you may see the object number and the type. There are various fields you may modify as the position and size of the object, the font (in relation to the font table), the font size and colour. If you want to move an object to another page, modify the page number. You need to reopen the file to see the change. Click on "Apply" to apply changes.

With the cursor keys you can also move the objects (Cursor keys) or modify its size (Cursor keys + Ctrl). With the Shift key each step moves/enlarges/reduces the object by five pixels. It is also possible to click on an object, move this object (keeping the mouse key down) to another position and drop it by releasing the mouse key.

When finished, click on Save in the form.

Imprint

Hdb2Win / PaleoTax © H. Löser 1993-2025

The Data Acquisition Forms – Format and Entry Screen Editor

Published	1 March 2025
Version	2.6
History	May 2021 • Version 1 May 2022 • Version 1.1 March 2025 • Version 2.6
Internet	https://www.paleotax.de/
E-Mail	info@paleotax.de
Document	E:\T\DOK\Forms\forms-26.doc