## #18 Items and Components Recursive Linking Example

Written by Todd Carper September 20, 1987

Published October 14, 1987

---

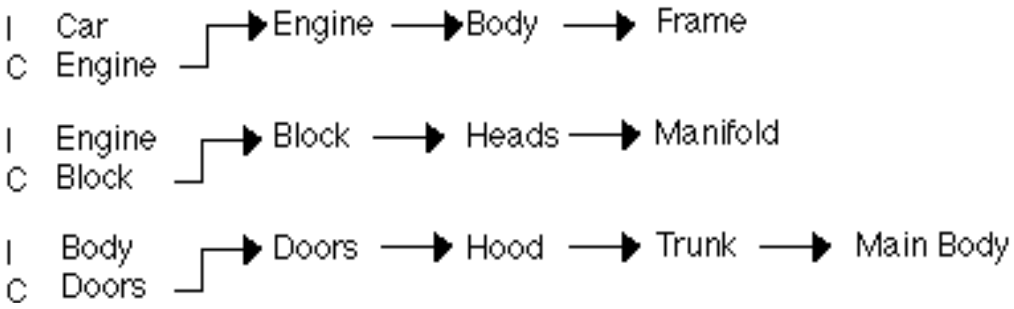
An approach to the items and components database design.

---

Here is an approach to the items and components database.


Make a recursive field link on Item.
Make a recursive file link from Component to Item.

Here's the data Structure. 'I' = Item 'C'=Component




Notice that each component has a record where it is the item.

To use this approach we will need:

1) A main global procedure to run everything.
2) A Dialog (Dialog1) to request with which record they want to start.
3) A Dialog (Dialog2) to allow record modification and to show the components.

The components will be displayed thru a scrollable area.

Here's what happens:

Start the Global.
Call Dialog1 to get the item name that they want.
Search by index for the item.

A. Doing links, load the scrollable area with the components

Call Dialog2 to allow modification and to display the info

In Dialog2 have a get button so that they can click on a component and then click on the get button, (which is an accept), this will send you back to the global.

In the global check if the button was pressed.

B. If Get was pressed then Load linked record until you get the right one.

The above operations would go in a while loop that would wait for a cancel button to be pressed.

A. We will assume that we have a current record.

Variable vItem will be assigned the item name of the current record
Now we need to pack the scrollable area with components

```
Push Record([File1])                        `This is our current record
Load Linked Record([File1]Component)        `Get record of 1st component if one exists
sComp{0}:=0                                 `Initialize scrollable area
While([File1]Item#"")                       `Check for end of components list
 sComp{0}:=sComp{0}+1                       `Increment scrollable variable
 sComp{sComp0}:=[File1]Item                 `Put item in array
 Load Linked Record([File1]Item)            `Get next component
End While
Pop Record([File1])                         `Retreive current record
One Record Select([File1])                  `Ensure that it is the selection
```

B. We will assume that an item was chosen and Get was pressed.

```
Load Linked Record([File1]Component)        `Get first component
Count:=1                                    `Initialize count variable
While(Count<sComp)                          `sComp contains item that was selected
 Load Linked Record([File1]Item)            `Get next component
 Count:=Count+1                             `Increment count variable
End While
```

Now for how the structure is built:

Enter an item that doesn't exist, in Dialog1.
The requested name will be in vItem.

```
Create Record([File1])
[File1]Item:=vItem
Save Record([File1])
```

Now call Dialog2.
The scrollable area will be empty because we have no components specified.
Have an accept button for add components.
Click on accept, you are sent back to the global, check for the button.

```
Push Record([File1])                    `This is our current record
nComp:=Request("Component Name")        `Get component name
Search By Index([File1]Item=nComp)      `Look for component
If(Records in Selection([File])=0)              `Does it exist
 Create Record([File1])                 `No. So, create the record
  [File1]Item:=nCmop
  Save Record([File1])
End If
Pop Record([File1])                             `This is old current record
One Record Select([File1])                      `Select the record
Push Record([File1])                    `Push it back on stack
If([File1]Component="")                 `Is this the first component
 [File1]Component=nComp                         `Assign component name to component field
 Activate Link([File1]Component)                `Create the link
 Save Record([File1])                   `Save the record
Else
 Search By Index([File1]Item=sComp{sComp{0}))       `Find the last linked component
  oItem:=[File1]Item                            `Save it's item name
  [File1]Item:="Junk"                           `Put "Junk" in Item field
  Save Record([File1])                  `Save it back
  Push Record([File1])                  `Push it on the stack
  Search By Index([File1]Item=nComp)    `Get the new component
  [File1]Item:=oItem                            `Give it the previous records Item name
  Save Record([File1])                  `Save it back
  Pop Record([File1])                   `Retreive last linked record
  One Record Select([File1])            `Select the record
  [File1]Item:=oItem                            `Reassign the old field name
  Activate Link([File1]Item)                    `Activate the link
  Push Record([File1])                  `Push the record (Don't save it yet) ***
  Search By Index([File1]Item=oItem)    `Search for the new component
  [File1]Item:=nComp                            `Put the component name back
  Save Record([File1])                  `Save the record
  Pop Record([File1])                   `Retreive the old last linked record
  One Record Select([File1])            `Select it
  Save Record([File1])                  `Now save the record

  Pop Record([File1])                   `Retreive our original current record
  One Record Select([File1])            `Select it
End If
```

*** NOTE: The record was pushed onto the stack. If the record is saved at this point then when the search is performed, we will have a two record selection and be unable to distinguish which record is the record to be added.