Synon is known as CA2E

There are 3 Important Menu options in Synon.  
1. Designer   
2. Programmer   
3. User

Designer option is usually used by admin for creation of New files. This option is often not accessible to everyone.

Programmer menu is used by application developer to Add/update/delete functions and Screens. This is what you will be using most of the time as a developer.

User menu is a good alternative when you just want to browse code and you do not want to update it. Its recommended to use this option if you are not planning to change code.

There are 4 types of files in Synon.  
1. Capture(CPT)  
2. Record(RCD)   
3. Reference(REF)  
4. Structure(STR)

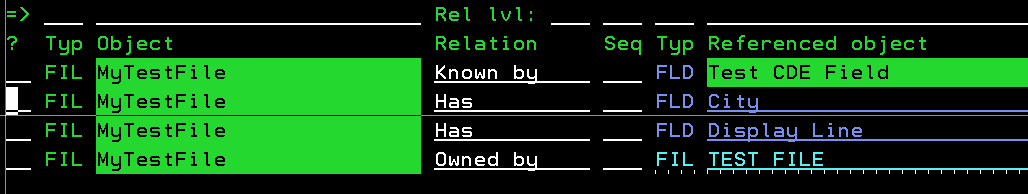
Each type of file has its different role.

Capture files are used where you want to capture all the transactions of your sytem ie..sales table or purchase table

Reference files are used where you want information which is used mostly for reference and not mass updated daily like employee address etc...

The RCD file attribute should be given to files that define data structures for use by programs in the implemented system. Permitted relations are Refers to, Has and Includes.

Structure files usually holds the infomation like Add date, Add time, Change Date, Change time. In a system these fields are required by almost all the files to capture record add and change infromation. Now once you have this structure ready then it can be used by other file by using INCLUDES which is file-to-file relation.



On above screen press F10 to add objects. This will take you to a menu where you can choose your file type like CPT, REF, STR etc. Once you define your file type then next screens will ask you to define fields if the fields you are using is not already defined.

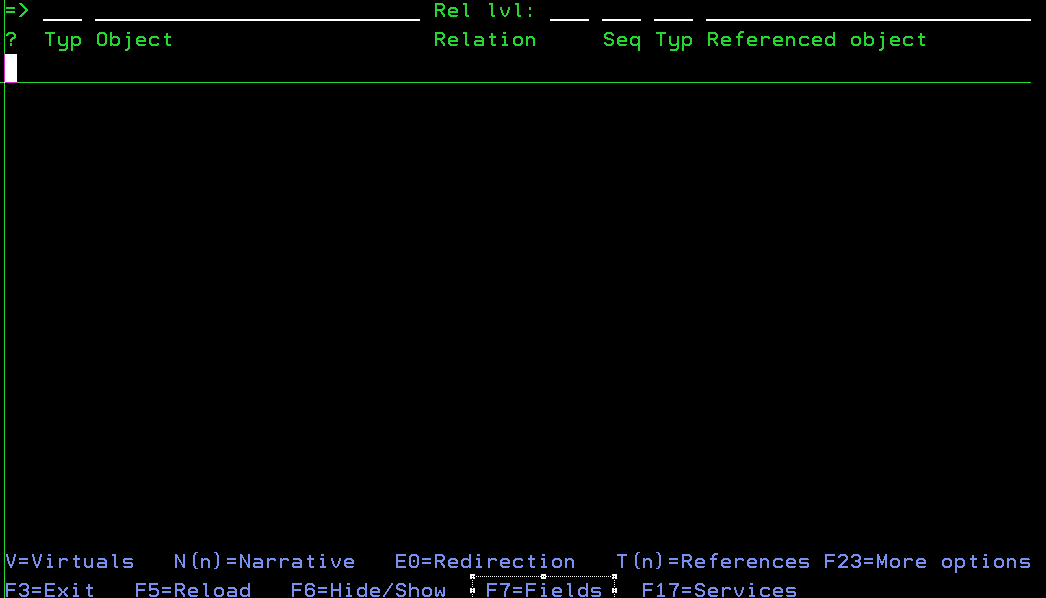
There are 2 kinds of relation.  
1. File-to-File.   
2. File-to-Field.

File to file relation is a relation in which your new file is using fields from the existing file. Like in example above MyTestFile is using Owned by relation to use key fields of "Test File" file. Following are the list of all relation which Synon offers

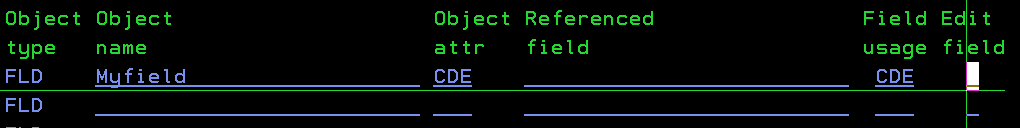
1. Has - Its a way to add just any field to your file.(file-to-field relation)   
2. Includes - Its a way to add all fields of structure file in your file.(file-to-file relation)   
3. Known by - Its a way to add primary keys.A key field has to be of type CDE. (file-to-field relation)   
4. Owned by - Its a way to add all keys of a file as keys in your file.(file-to-file relation)   
5. Qualified by - Qualified by is similar to known by to add key fields but unlike key fields its retrievals get nearest-match not the exact-match. (file-to-field relation)   
6. Refers to - Its a way to add all keys of a file as regular field in your file.(file-to-file relation)   
7. Defined as -   
8. Denoted by - The 'Denoted by' relation specifies that A CA 2E file may be defined by a special unique identifier: 'a surrogate'. Its rarely used.   
9. Extended by - Its a way to use all fields(Key Fields + Regular fields) of a file as it is into another file. Its used when you want to create a file A which should have all the fields of B plus some other fields. (file-to-file relation)

You can define fields as Designer or Developer

Press F7 to all fields in following screen



It will open a screen where you can see all the fields. Press F10 to define Field. That will open a screen like following where you can define field.



Following are the types of fields you can choose at the time of creation

CDE - Alphameric code value  
DT# - Date in \*ISO format (YYYY-MM-DD internally)  
DTE - Date in system date format - (YYMMDD internally)  
D8# - Date in pure numeric format (YYYYMMDD internally)   
IGC - Ideographic text  
NAR - Narrative text  
NBR - Pure numeric value (eg line number).  
PCT - Percentage or market index.  
PRC - Price or tarrif  
QTY - Quantity  
REF - Field is based on another field.  
SGT - Surrogate (number representing an object)  
STS - Status.  
TM# - Time in \*ISO format (HH.MM.SS internally)  
TME - Time in HHMMSS format.   
TS# - Timestamp (YYYY-MM-DD-HH.MM.SS.NNNNNN)  
TXT - Object text.  
VAL - Monetary value.  
VNM - Valid System name.

Following are the Field attributes you can choose at the time of creation

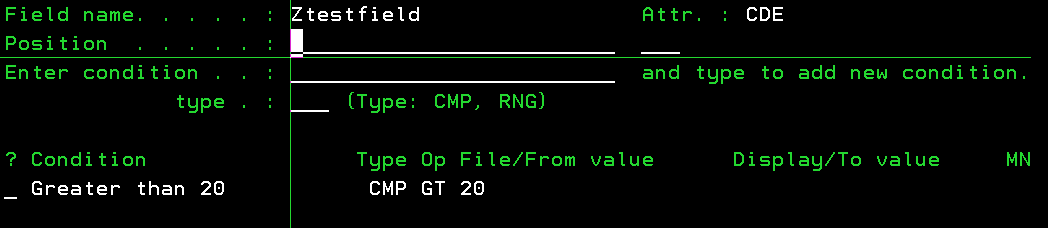
CDE - Code database field. Used in Known by and Qualified by relations only.  
ATR - Attribute database field. Used for filed fields which are non-keys.  
CNT - Count function field  
DRV - Derived function field  
MAX - Maximum function field  
MIN - Minimum function field  
SUM - Sum function field  
USR - User function field

Conditions are used for comparison purpose. Following screens will elaborate in detail.

On Designer/Programmer/User Menu Press F7 and following screen will open and take option Z.

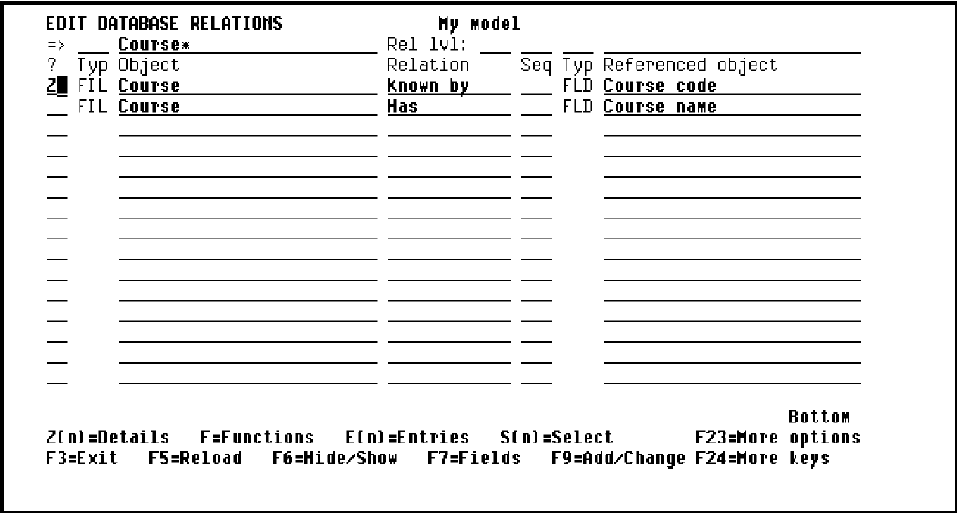
File types

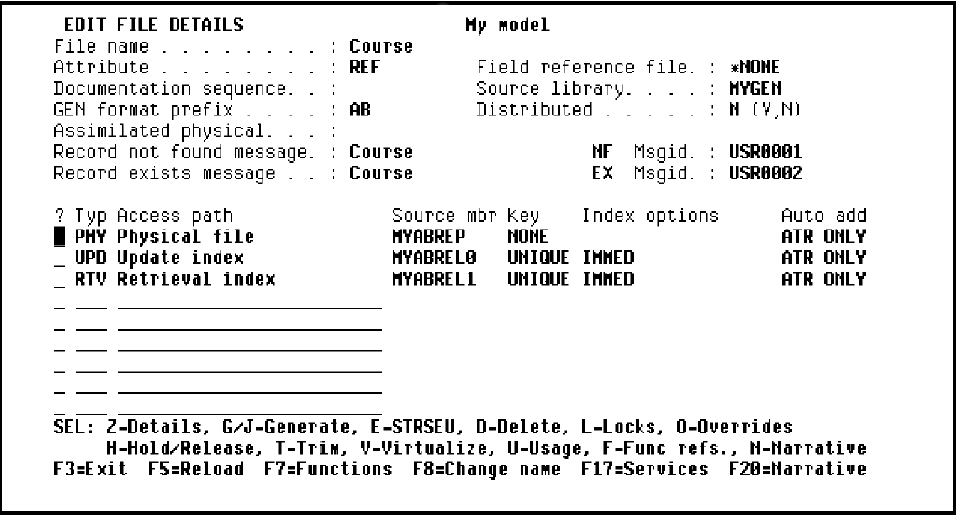
It will open another screen. Press F9 to open condition screen. Here you can add conditions.



Access paths are logical files which provide access to physical file.

On Designer/ProgrammerMenu take Z against the file to see all access path.





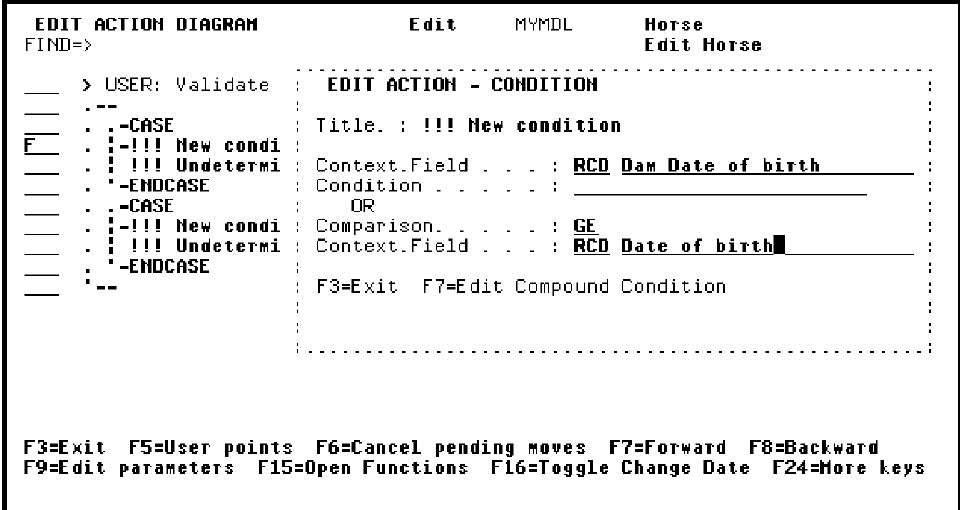
There are folloiwng types of Access Path. PHY,UPD and RTV access paths are created by default for every file

PHY - Physical access path refers to physical file.  
UPD - Update access path used to add/update/delete recrods from the file.  
RTV - Retrieve access path to read record from the file using default keys.  
QRY -   
RSQ - Resequence access path to change the keys and Ascending/Descending order of logical file.  
SPN - Span access path in order to obtain a logical view that combines two different access path formats.

**Access paths are used by Synon functions. For example Display record/File function would use RTV or RSQ access path but to update/Delete or Add record in file UPD access path is used.**

Contexts are used to pass values amoung variables.

Following screen shows use of context in synon.



There are following types of context in synon

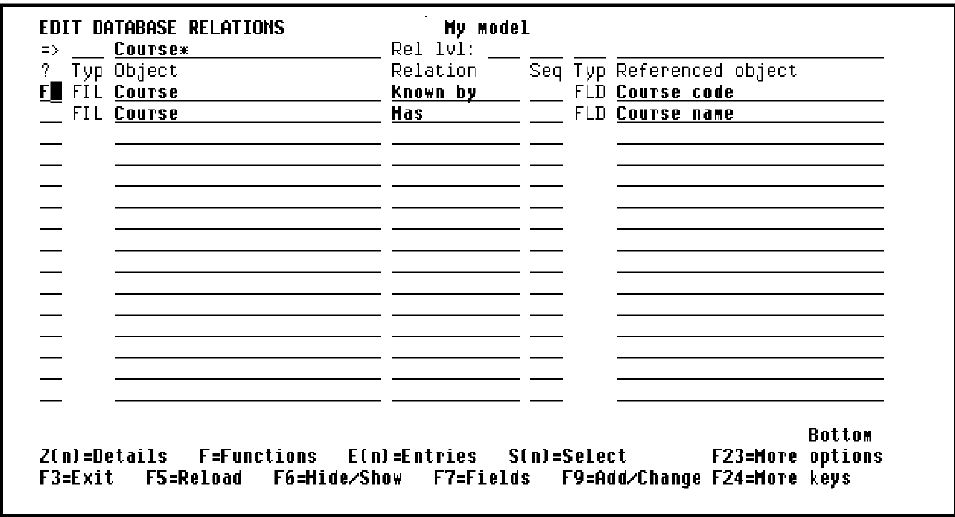
PAR - Parameter context refers to parameters of the current program.  
PGM - Program context is used to pull program information like return code, cursor field etc..  
JOB - The JOB context contains system fields that supply run time information about the job executing the HLL program which implements a CA 2E function, for example, the user name, the job name and the job start time. The JOB context is fixed and additional fields may not be added to it.  
LCL - refers to local variable defined within the program.  
WRK - Work variable are visible through out the External function.  
RCD - Record context is available in screen functions and refers to Records on the screen.  
CTL - Control context is available in screen functions and refers to control field of the screen.   
DTL - The DTL context is available in the action diagrams of function types 'Prompt record', 'Edit record' and 'Display record'. The context is only available after the key has been successfully validated.  
2ND - 2nd refers to 2nd screen of Edit/Display record function(2 Screen).  
3RD - 3rd refers to 3rd screen of Edit/Display record function(3 Screen).  
NXT - Next context is used in printer file to refer next printable variables.  
CUR - Current context is used in printer file to refer curretn printable variables. KEY - The KEY context contains the fields which are on the key screen display of CA 2E device functions that have key screens; for example 'Edit record', 'Display Record'.  
DB1 - The DB1 context contains the file fields which are in the first or only format of CA 2E access path.  
DB2 - The DB2 context contains the file fields which are in the second format(if any). The DB2 context is only available in the action diagram of functions which attach to a Span (SPN) access path, namely 'Edit transaction' (EDTTRN) and 'Display transaction' (DSPTRN).  
NLL - The NLL context is available at all points in the action diagram of all function types. Its used when you dont want to pass any value to a parameter. Only output parameters may be directed to the NLL context.

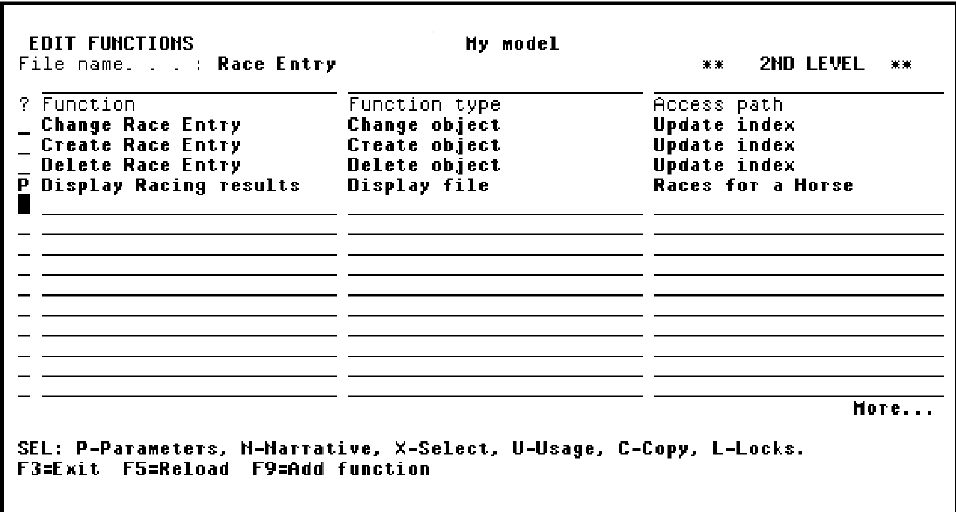
SYNON Parameters

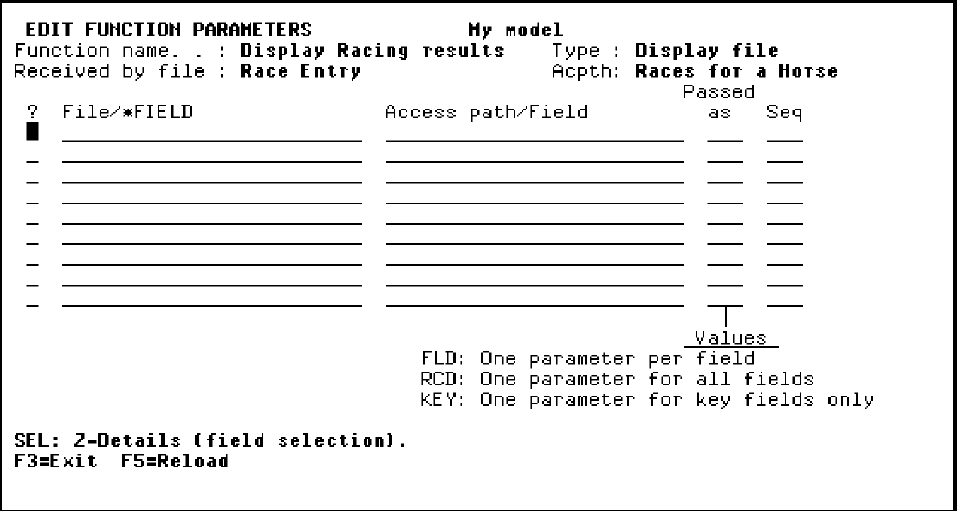
Parameters are used to pass value from one function to another.

You can add or change parameters in Designer or Programmer menu.

Lets go step by step. So far you have a file with access paths on it. If you type F against a file you will see all functions built on that file



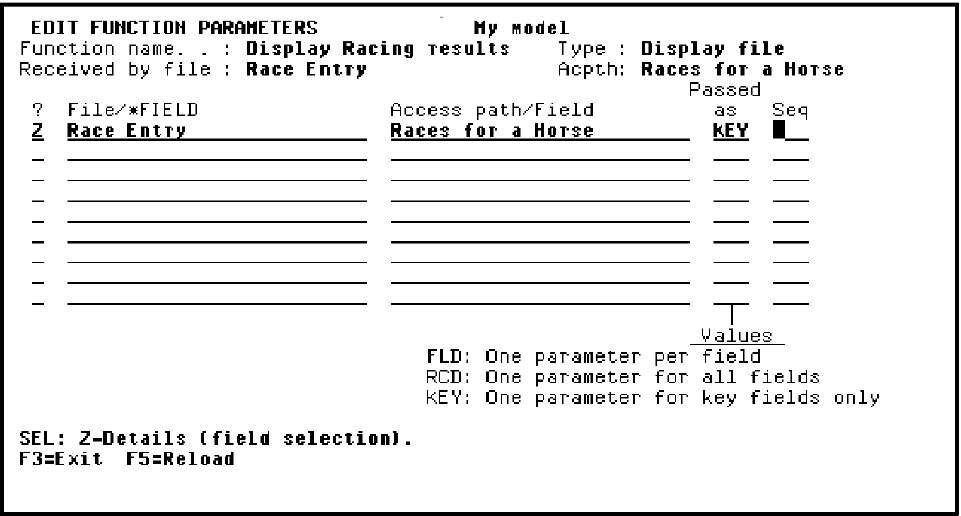




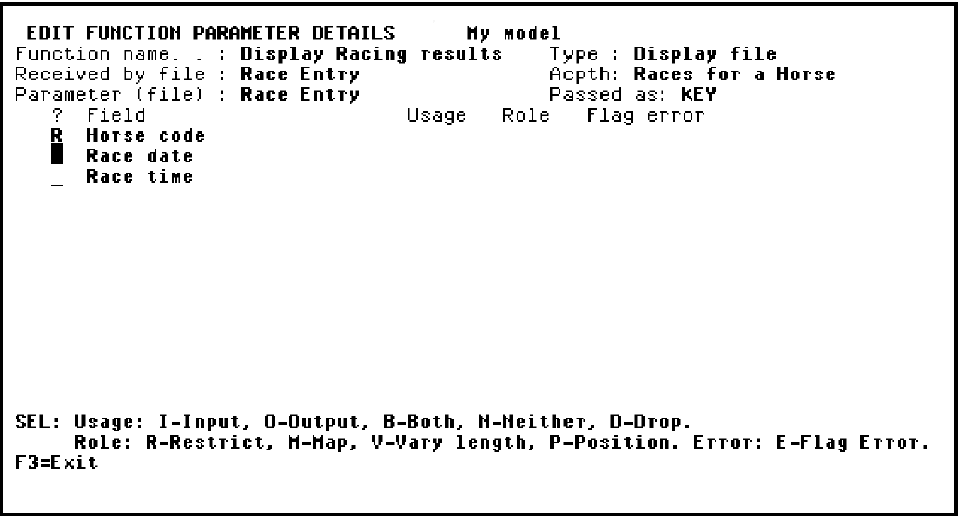
There are following 3 ways to pass parameters.

1. FLD - Provide the field name of the parameter of the function.  
2. RCD - Provide the file name and then choose which field of the file will be used as a parameter.  
3. KEY - Provide the file name and keys of the file will be passed as parameter.

Please see an example below.



Now we have Roles and Usage for Parameters. For our example we passed parameters as KEY but i only want to use Horse code as a parameter. Here usage and role helps.



Types of Usage.

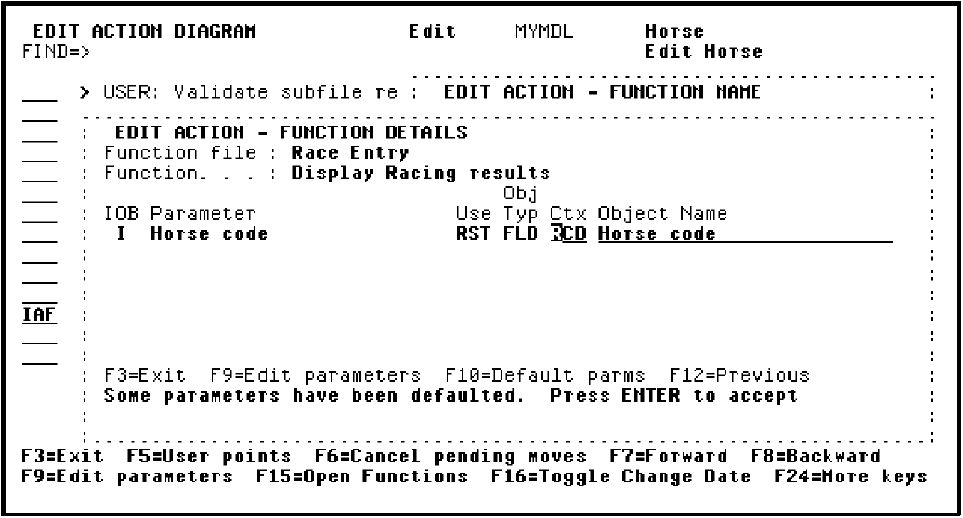
1. I - Input. Parameter will be passed as input.  
2. O - Output. Parameter will be used as output means this "Display racing results" function will return value in this parameter.  
3. B - Both. Parameter will used to pass value and return value both.  
4. N - Neither. Parameter will not be used.  
5. D - Drop. Parameter is dropped and can not be used.

Types of Role.

1. R - Restricted. Function will read access path to find exact matching record.  
2. M - Map. Function will read access path from the beginning.  
3. P - Position. Function will read access path to find closedst matching record.

# SYNON Editor

Typing F against a synon function will open the function in editor which looks like following.



Press F5 anytime to see a list of User points. <<< indicates that user point is updated by a programmer.

Press Shift+F5 to search functions/files/fields within the action diagram.

Press Shift+F7 to open screens of screen functions like edit file, edit record, display file etc..

Press Shift+F6 to open notepad.

Following options are available to edit code in synon

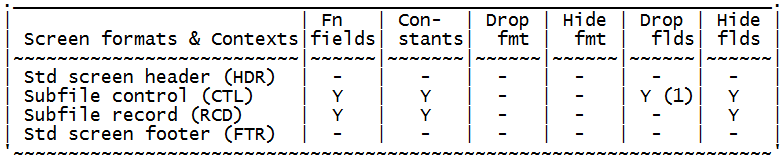
\* - Activate/Inactivate construct (Comment out)  
\*\* - Place block Activate/Inactivate boundary  
A - Place copied or moved construct after this line  
B - Place copied or moved construct before this line  
C - Copy construct to a point indicated by 'A' or 'B'  
CC - Place block Copy boundary  
D - Delete this construct  
DD - Place block Delete boundary  
F - Edit action or condition details for line  
FF - Edit action parameters  
H - Hide construct  
I+ - Insert \*ADD built-in function  
I+F - Insert and Prompt \*ADD built-in function  
I\* - Insert Comment  
I\*F - Insert and Prompt Comment  
I- - Insert \*SUB built-in function  
I-F - Insert and Prompt \*SUB built-in function  
I= - Insert \*MOVE built-in fuction  
I=A - Insert and Prompt \*MOVE ALL built in function  
I=F - Insert and Prompt \*MOVE built-in function  
I=M - Insert and Prompt \*MOVE ARRAY built-in function  
IA - Insert Action  
IAF - Insert and Prompt Action  
IC - Insert Case condition  
ICF - Insert and Prompt Case condition  
IE - Insert \*EXIT PROGRAM function  
II - Insert Iteration  
IIF - Insert and Prompt Iteration  
IM - Insert message function  
IMF - Insert and Prompt Message function  
IO - Insert \*OTHERWISE condition  
IOF - Insert and Prompt \*OTHERWISE condition  
IQ - Insert \*QUIT function  
IS - Insert Sequence  
ISF - Insert and Prompt Sequence  
IX - Insert New condition within case  
IXF - Insert and Prompt New condition within Case  
M - Move construct to a point indicated by 'A' or 'B'  
MM - Place block Move boundary  
N - Display narrative of selected object  
NA - Copy to notepad & append to contents of notepad  
NAA - Place block notepad append boundary  
NI - Insert entire contents of notepad after this line  
NR - Copy to notepad & append to contents of notepad  
NRR - Place block notepad replace boundary  
PR - Display construct protection details  
R - Display model references of selected object  
S - Show construct  
T - Return to top level of action diagram  
U - Display model usages of selected object  
V - View summary  
Z - Zoom into construct

# Screen Design

A typical synon screen looks like following image

This image shows you that a synon Screen has Header, Control, Detail and a Footer.

Following screen show the contexts you have to address fields on the screen.



In design screen mode you can always press Shift+F5 to Add/drop/hide fields from your screen. You will know once you get into that menu.

Suppose you want to change fields display name etc..Put cursor on any field on screen and hit enter to see Field properties. To do magic with a field like color, Highlight, Blink etc..press Shift+F7 from Field properties and next screen will open. You will know what to do and how to do when new screen opens

Suppose you want that a list of value should open when F4 is pressed on a field. In Screen design mode hit enter on the field and Field properties screen will open. Supply your list values in "Check Condition" option.

**Device Design function keys**

**F1** moves the field 40 positions to the left.  
**F2** animates the panel using Toolkit.  
**F3** exits the panel.  
**F4** moves the field 40 columns to the right.  
**F5** edits device format details of the format where the cursor is positioned.  
**F6** cancels the pending operations.  
**F7** displays the Edit Device Design Format Relations panel.  
**F8** moves the selected field to the cursor position.  
**F9** wraps text onto the next line starting from the field on which the cursor is positioned.  
**F10** moves text one column to the right.  
**F11** removes the line on which the cursor is positioned.  
**F12** aligns text below the cursor position.  
**F13** fast exits the panel.  
**F15** moves panel window to the left margin.  
**F16** moves window to the right margin.  
**F17** displays a list of device formats.  
**F18** displays the Edit Field Attributes panel.  
**F19** adds new function fields to the device design.  
**F20** edits the function field on the device design.  
**F21** adds a line above the cursor position.  
**F22** moves text one column to the left.  
**F23** adds a constant field to the device design.  
**F24** aligns all fields under the cursor position.

Change Object

Every file(CPT, REF) once created in synon gets Create object, Change object and delete object function by default

As its name suggest Change object allows you to Change/Update a existing record in the file. It does not have any object name and is used as a subroutine with in the calling functions.

All the fields from the Update access path must be declared as parameters to the 'Change object' function, they cannot be dropped. Any fields which are not included in updating the database file record should be defined as 'Neither' (N) parameters.

**Action diagram user exit points**  
**- Processing before DBF update**  
- USER: Processing before DBF read   
- USER: Processing if DBF record not found  
- USER: Processing after DBF read   
- USER: Processing before DBF update   
**- Processing after DBF update**  
- USER: Processing after DBF update

# Create Object

Every file(CPT, REF) once created in synon gets Create object, Change object and delete object function by default

As its name suggest Create object allows you to add a new record into the file. It does not have any object name and is used as a subroutine with in the calling functions.

Default parameters are all the fields of the update access path.

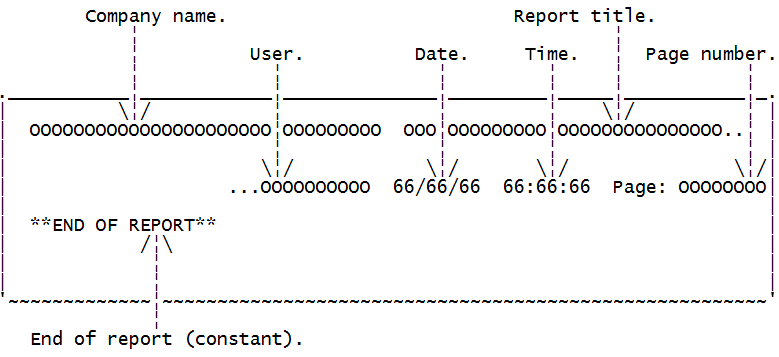
**Action diagram user exit points**   
**- Processing before DBF write**   
- USER: Processing before DBF update   
- USER: Processing if DBF already exists   
**- Processing after DBF write**   
- USER: Processing if DBF update error   
- USER: Processing after DBF update

Define Report format

The CA 2E 'Define report format' function type allows you to define a standard report header and footer for your CA 2E 'Print file' report functions.

The shipped standard report page header is for a report 132 characters wide. If you wish to define a DFNRPTFMT function with a different report width, you would specify a different value for the FORMSIZE parameter on the overrides to the IBM i 'Create Print file' command CRTPRTF) for the function. This may be achieved using the overrides prompt available from the 'EDIT FUNCTION DETAILS' screen (line option 'O').

A typical header & footer for report looks like following which can be customized using "Define Report Format" function.

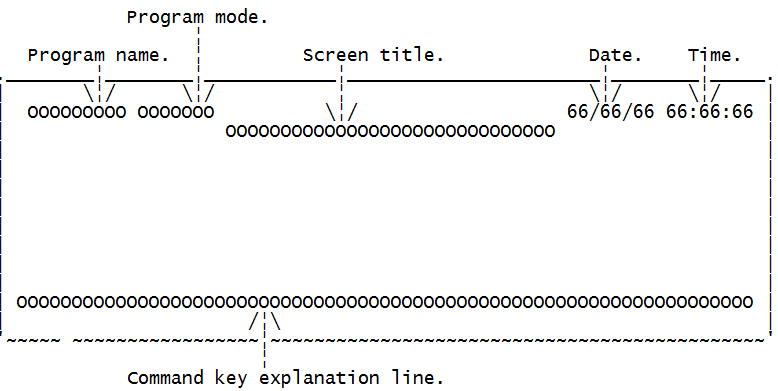


Define Screen Format

The CA 2E 'Define screen format' function type allows you to define a standard screen header and footer for use by other CA 2E functions which have screen designs attached to them.

Four default 'Define screen format' functions are shipped as standard header/footer formats for CA 2E device function screen designs  
•\*STD SCREEN HEADINGS (CUA)' function follows the SAA CUA standards.  
•\*STD CUA WINDOW function follows CUA standards for window panels.   
•\*STD CUA ACTION BAR function follows CUA standards for action bar panels.  
•\*STANDARD SCREEN HEADINGS function follows the CA 2E standards for the System/38.

A typical header & footer for report looks like following which can be customized using "Define Screen Format" function.



# Delete Object

Every file(CPT, REF) once created in synon gets Create object, Change object and delete object function by default

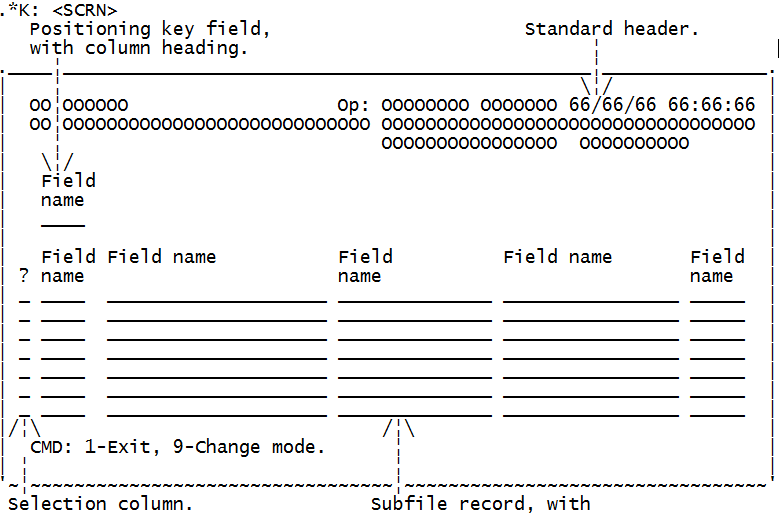
As its name suggest Delete object allows you to delete an existing record in the file. It does not have any object name and is used as a subroutine with in the calling functions.

All the key fields from the Update access path must be declared as parameters to the 'Delete object' function, they cannot be dropped.

**Action diagram user exit points**  
**- Processing before delete**  
- USER: Processing before DBF update   
**- Processing after delete**  
- USER: Processing after DBF update

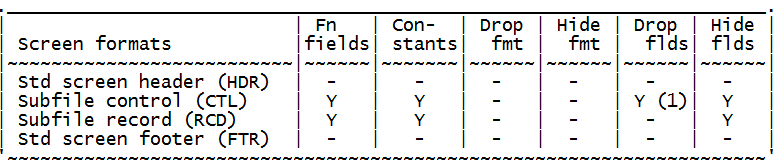
Display File

Suppose you need to provide a screen to display records of a file then you should use display file function. Once you create this function the screen looks like following image.

Display file as its name suggest does not provide any update/delete/add records function. Use Edit file if you need to perform Add/Update/delete file records operations.

Following are the screen formats available for display file.



**Action diagram user exit points**  
**- Initialisation**  
- USER: Initialise program   
**- Load subfile**   
- USER: Initialise subfile control   
- USER: Initialise subfile record from DBF record   
**- Process subfile (Pre-confirm)**  
- CALC: Subfile control function fields   
- USER: Process subfile control (Pre-confirm)  
- CALC: Subfile record function fields   
- USER: Process subfile record (Pre-confirm)   
- USER: Final processing (Pre-confirm)  
**- Process subfile (Post-confirm)**  
- (Only available if Post-confirm pass option selected.)  
- USER: Process subfile control (Post-confirm)   
- USER: Process subfile record (Post-confirm)  
- USER: Final processing (Post-confirm)   
**- Command keys**  
- USER: Process command keys   
**- Closedown**  
- USER: Exit program processing

Display Record

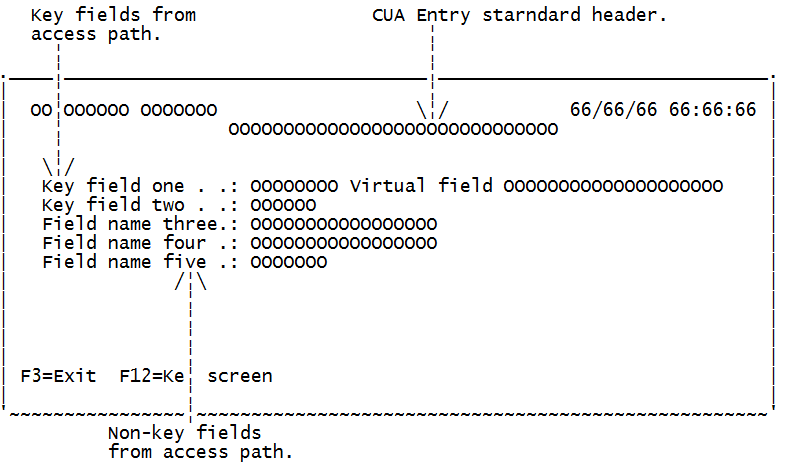
The CA 2E 'Display record' program defines a program to display a single record from a specified database file. If no key is supplied, a key screen prompts for a key.

There are 3 different types of display records.

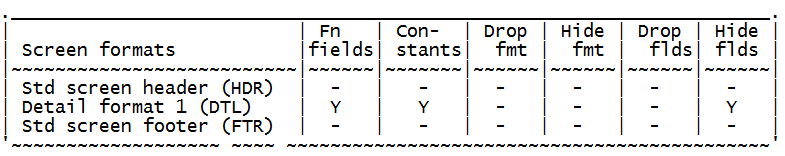
1. Display record(1 Screen)  
2. Display record(2 Screen)  
3. Display record(3 Screen)

They are all same except 2,3 Screen provides extra screen so you can fit more fields. A typical display record screen looks like following.

Now for example there are 100 fields that you want to put on a screen but you can not fit all of them in one screen then you can choose to use Display records 2 Screen or 3 Screen function.



Following are the screen formats available for Display record.

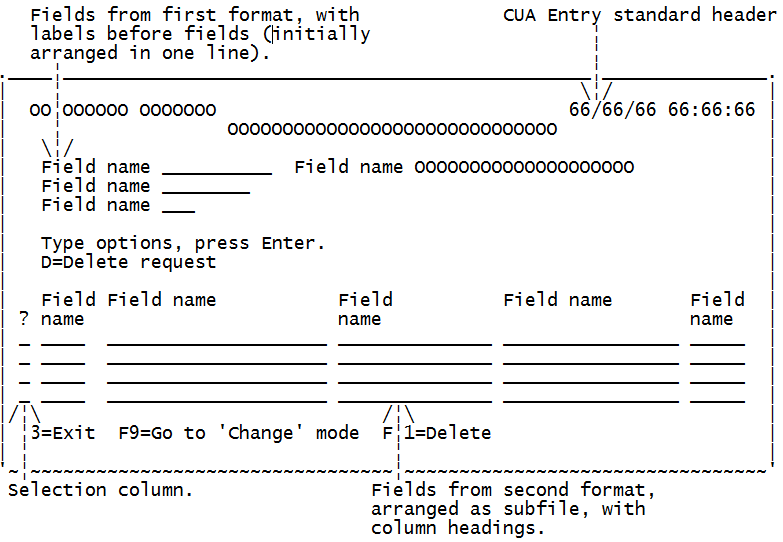
Display record as its name suggest does not provide any update/delete/add record function. Use Edit record if you need to perform Add/Update/delete file record operation.

**Action diagram user exit points**   
**- Initialisation**  
- USER: Initialise program   
**- Key screen**  
- USER: Initialise key screen   
- USER: Validate key screen   
**- Detail screen**  
- USER: Load detail screen from DBF record  
- USER: Process key screen request   
- CALC: Detail screen function fields   
- USER: Validate detail screen   
- USER: Perform confirmed action  
**- Command keys**  
- USER: Process command keys   
**- Closedown**  
- USER: Exit program processing

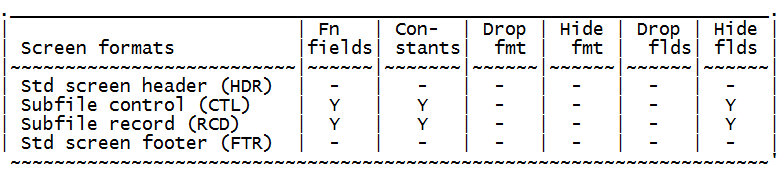
Display Transaction

Display transaction screens are associated with span access path which is more like dealing with more than one file. Once you create this function the screen looks like following image.

Display transaction screen is like edit transaction function only difference is that you can not update record in display transaction function.

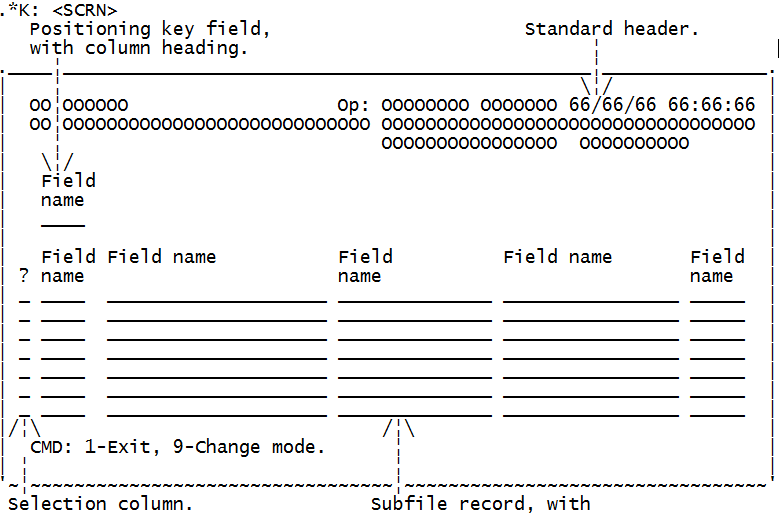
 

Following are the screen formats available for Display transaction function.

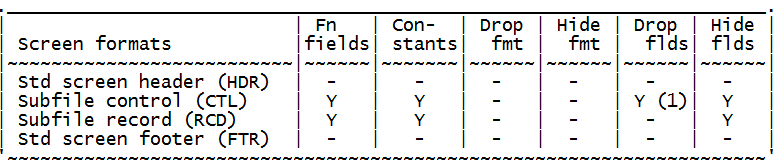


Synon Edit File

Suppose you need to provide a screen from which user can add/update/delete file records then you should use edit file. Once you create this function the screen looks like following image.

Following are the screen formats available for edit file.



**Action diagram user exit points**  
**- Initialisation**  
- USER: Initialise program  
**- Load subfile**  
- USER: Initialise subfile header  
- USER: Initialise subfile record (existing record)   
- USER: Initialise subfile record (new record)  
**- Validate subfile header**  
- CALC: Subfile control function fields   
- USER: Validate subfile header   
**- Validate subfile**  
- USER: Validate subfile record fields   
- CALC: Subfile record function fields  
- USER: Validate subfile record relations  
**- Update DBF from subfile**  
- USER: Create DBF record   
- USER: Change DBF record   
- USER: Delete DBF record   
- USER: Extra processing after DBF update   
**- Command keys**  
- USER: Process command keys   
**- Closedown**  
- USER: Exit program processing

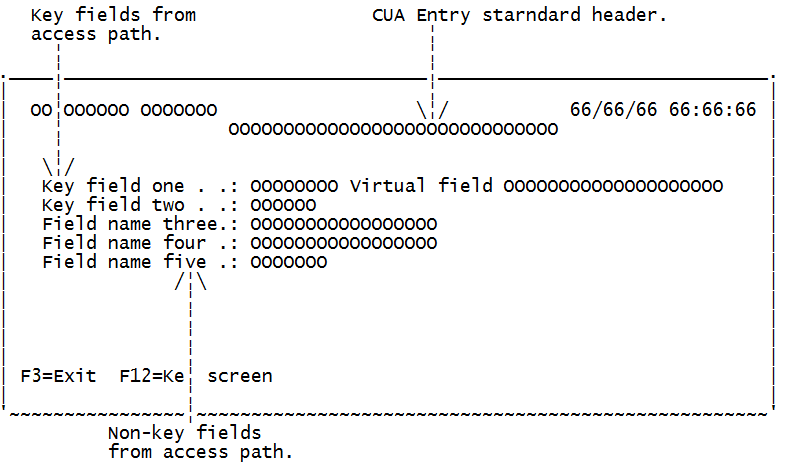
Edit Record

There are 3 different types of Edit records.

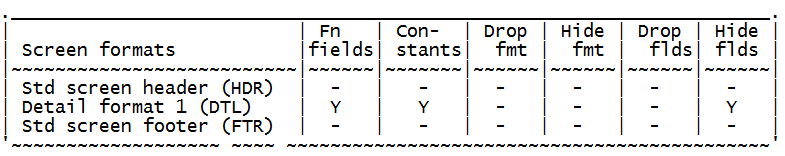
1. Edit record(1 Screen)  
2. Edit record(2 Screen)  
3. Edit record(3 Screen)

They are all same except 2,3 Screen provides extra screen so you can fit more fields. A typical Edit record screen looks like following.

Now for example there are 100 fields that you want to put on a screen but you can not fit all of them in one screen then you can choose to use Edit records 2 Screen or 3 Screen function.



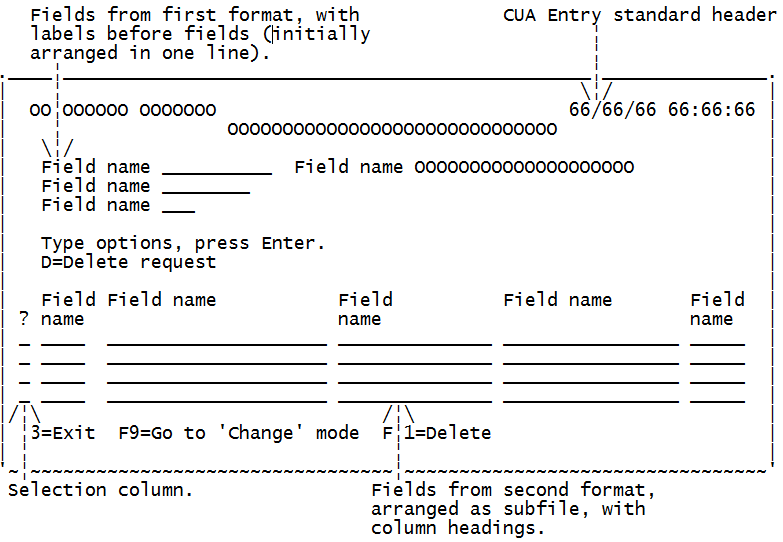
Following are the screen formats available for Edit record.



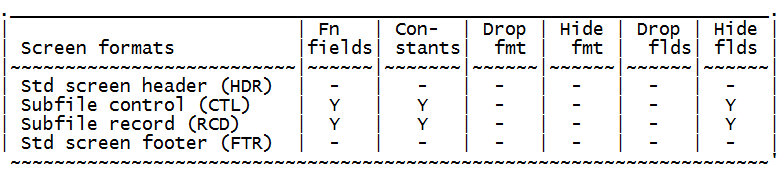
**Action diagram user exit points**   
**- Initialisation**  
- USER: Initialise program   
**- Key screen**  
- USER: Initialise key screen   
- USER: Validate key screen   
**- Detail screen**  
- USER: Initialise detail screen (new record)   
- USER: Initialise detail screen (existing record)   
- USER: Process key screen request   
- USER: Delete DBF record   
- USER: Validate detail screen fields   
- CALC: Detail screen function fields   
- USER: Validate detail screen relations   
**- Update processing**  
- USER: Create DBF record   
- USER: Change DBF record   
**- Command keys**  
- USER: Process command keys   
**- Closedown**  
- USER: Exit program processing

Edit Transaction

Edit transaction screens are associated with span access path which is more like dealing with more than one file. Once you create this function the screen looks like following image.

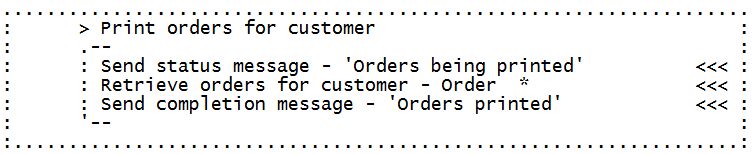
Following are the screen formats available for edit transaction function.



Execute External Function

The 'Execute external function' can be used to specify a batch process to CA 2E. For example you might define a 'Retrieve object' function to read records from a file and within that specify processing to be performed on each record read. If you wanted to implement the 'Retrieve object' function as a batch program, you could do so simply by including it within an 'Execute external function'.

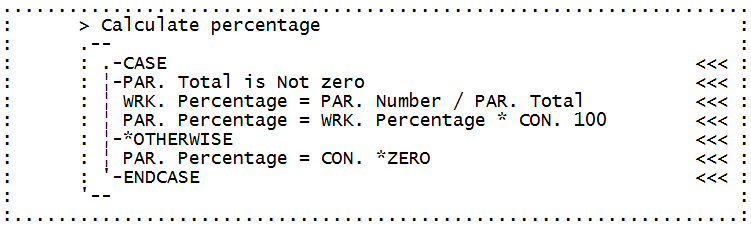
An example of the use of an 'Execute external function' might be a batch program to read and process records from a file. If you wanted to read all the records from an 'Order' file and for each record read, print the order, you could define an 'Execute external function'.



Execute Internal function

The CA 2E 'Execute internal function' allows you to specify a section of an action diagram for repeated use in other CA 2E functions. The function is implemented as in-line code within the source code of the calling function.

Internal functions do not have any object name. They are meant to be used within a external function or any other function with object name like diplay file, edit file. Internal functions are like common subroutines which are created to be shared among different programs. Following screenshot is an example of internal function which calculates %. This function can be used by any program by just including this internal function.



# Execute User Program

The CA 2E 'Execute user program' function allows you to describe the interfaces to a user written HLL program so that it may be referenced by CA 2E functions. Parameters may be specified on the call.

# Execute User Source

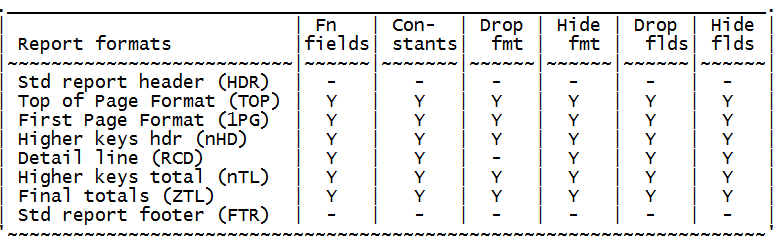
Includes HLL code to perform an arbitrary function within the source generated by CA 2E for a HLL program. The HLL code of the user source function must be of the same type as that used to implement the calling function, i.e. a function implemented in RPG III may only include RPG III user source functions and a function inplemented in COBOL may only include COBOL source functions.

Print File

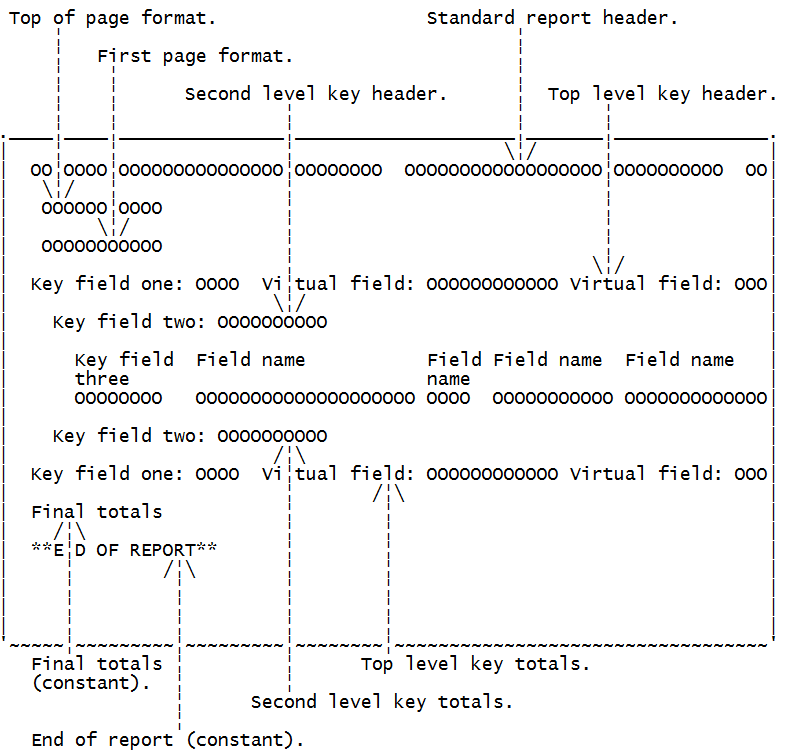
The CA 2E 'Print file' function defines a program to print the records from a specified access path. Totalling may be specified Up to 13 levels of totalling are allowed. Records from more than one access path can be printed in the same report by embedding 'Print object' functions within a 'Print file' function. The overall structure of a report is defined using the 'EDIT DEVICE STRUCTURE' display.

A 'Print file' function may be attached to a Retrieval (RTV), Resequence (RSQ) or Query (QRY) access path. This access path will determine which fields and which records are to be printed by the 'Print file' function, as well as the order of printing and the totalling levels available. Further choice of records can be made by specifying whether a record is to be selected in the appropriate point in the action diagram.

Print file prints report in spool file in the format specified by developer



A Sample print file screen looks like following

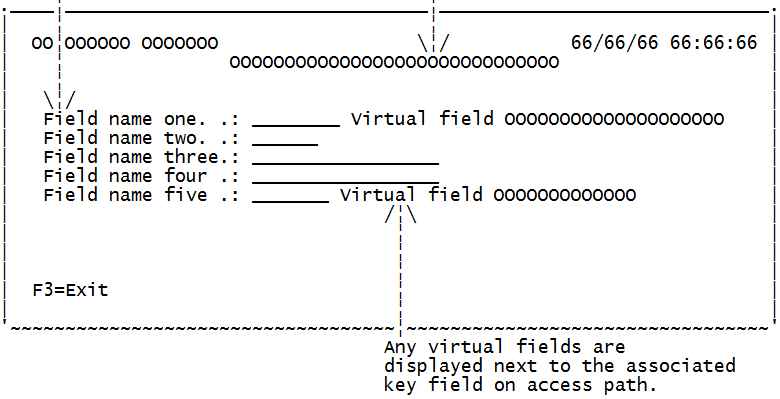


**Level breaks in Print file.**  
a. A level break is instigated whenever a major key value changes.   
b. On a level break, all fields in the associated controlling Header format and its associated Total format are reset - to blank, zero or from the DB1 context as appropriate.

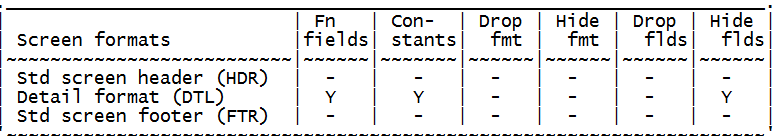
**Action diagram user exit points**  
**- Initialisation**  
- USER: Initialise program   
  
**- Read next record**  
- USER: Record selection processing   
  
**- Print top of page**  
- USER: Process top of page   
  
**- Print first page**  
- USER: Process before print of first page format  
- USER: On print of first page format   
- USER: Process after print of first page format   
  
**- Null report processing**  
- USER: Null report processing   
  
**- Process report body**  
- USER: Print required level headings   
- USER: Process before print of detail format  
- USER: On print of detail format   
- USER: Process after print of detail format  
- USER: Print required level totals   
  
**- Print final totals**  
- USER: Process before print of final totals format   
- USER: On print of final totals format   
- USER: Process after print of final totals format  
  
**- Print end of report**  
- USER: On print of End of report format

Prompt Record

Suppose you want to prompt a list of fields like following screenshot then you should use prompt and validate function. Once you create this function the screen looks like following image.

Following are the screen formats available for Prompt recrod function.



# Retrieve object

The CA 2E 'Retrieve object' function defines a routine to retrieve one or more records from a database file. Processing may be specified for each record read, by modifying the action diagram for the function. The 'Retrieve object' function may be attached to a Retrieval (RTV) or a Resequence (RSQ) access path.

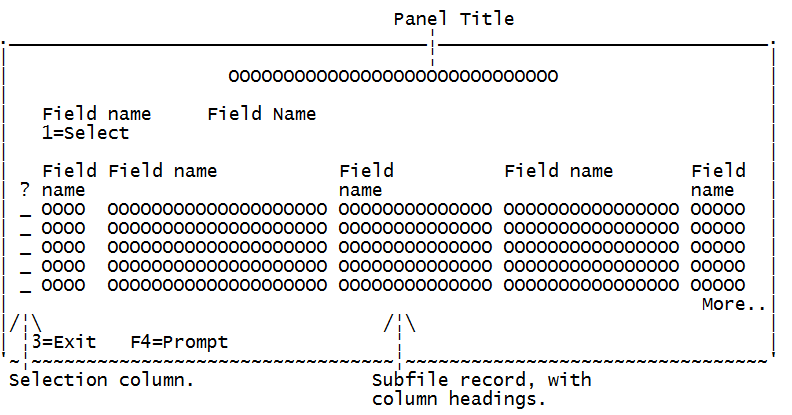
**Effect of restrictor parameters:-**  
a. If all the keys of the access path to which a 'Retrieve object' function attaches are supplied as restrictor parameters, then only the record with the given key or keys will be read.  
b. If only some of the keys (i.e. the major keys) are supplied as restrictor parameters, then all of the records with the given key will be read.  
c. If none of the keys of the access path are supplied as restrictor parameters, then all of the records in the access path will be read.

**Effect of positioner parameters:-**  
a. If all the keys of the access path to which a 'Retrieve object' function attaches are supplied as positioner parameters to the function, then only the records with a key value greater than or equal to the given key or keys will be read.  
b. If only some of the keys (i.e. the major keys) are supplied as restrictor parameters, but some or all of the remaining keys are passed as positioner values, then only those records with keys equal to the restrictor values and greater than or equal to the positioner values will be read.  
c. If none of the keys of the access path are supplied as positioner parameters, then all of the records in the access path within the specified restrictor group will be read.

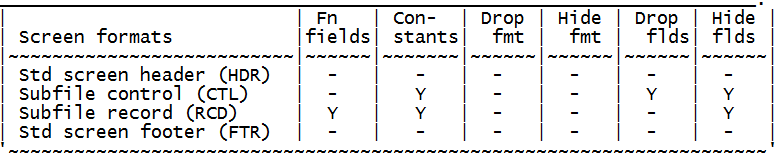
**Action diagram user exit points**  
**- Initialisation**  
- USER: Initialise routine   
**- No record found**  
- USER: Processing if DBF record not found  
**- Record found**  
- USER: Process DBF record   
**- Closedown**  
- USER: Exit processing

Select Record

The CA 2E 'Select record' function defines a program to display the records from a specified file using a subfile. The program allows the user to select one of the records. The key of the selected record will be returned to the calling program. Once you create this function the screen looks like following image.

Following are the screen formats available for Select Record file.



Service Program

The CA 2E 'Service Program' allows you to define a service program (i5/OS Object Type \*SRVPGM) within a 2E model. The function is implemented as an i5/OS service program (\*SRVPGM). Once a Service Program function has been created, you can specify that one or more model functions which were generated as \*MODULE objects should be bound into it. You can also specify one or more external \*MODULE objects (i.e. modules which were created outside the CA 2E model) to be bound into the service program.

Typically, most external functions in a CA 2E model are defined as programs (with an object type of \*PGM). These external functions are 'callable' - they can be called from a command line or from another program, as long as the correct parameters are provided. However, it is possible to define an external function as a module (with an object type of \*MODULE). These external functions are not directly callable, but must be bound into a calling program or into a service program. If a module is bound into a program, then it is 'bound-by-copy' - a copy of the \*MODULE object is bound into the calling \*PGM object. This increases the size of the \*PGM object and means that if any changes are subsequently made to the module, all program functions into which it is bound must be updated to include the new object. However, if a module is bound into a service program, then any program functions which call the module function bind to the module using functions which call the module function bind to the module using 'bind-by-reference' - the program simply contains an internal reference to the service program which exports the module. At run-time, the program calls the module which is in the service program. In this way, the size of the \*PGM object is kept to a minimum, and any subsequent changes to any of the modules bound into the service program simply require the service program to be updated, with the \*PGM object automatically calling the updated version of the module function in the service program.

Trigger Function

The CA 2E 'Trigger Function' allows you to specify an HLL program using A CA 2E action diagram.   
The function is implemented as a separate program which can be called from the CA 2E Trigger Router program YTRIGGER when a file change is made ad a trigger has been added to the file specifying that the trigger program is YTRIGGER.  
Trigger Functions are defined with a specific parameter list which cannot be changed. The parameters are as follows:   
  
1: Trigger Control Data : This parameter is defined as the \*Trigger Control Data internal file and contains information about the trigger itself (trigger time, trigger event, trigger file etc.). The fields within this RCD parameter are input only.   
  
2: Old Record Format : This parameter contains a copy of the record format of the file over which the trigger is created. The fields contain the data for the record format BEFORE the file change takes place. If the trigger event is a \*INSERT event, the fields in this RCD parameter will be blank/zero. The fields within this parameter are input only.   
  
3: New Record Format : This parameter contains a copy of the record format of the file over which the trigger is created. The fields contain the data for the record format AFTER the file change takes place. If the trigger event is a \*DELETE event, the fields in this RCD parameter will be blank/zero. If the Trigger Function is called as part of a \*BEFORE \*INSERT or \*BEFORE \*UPDATE trigger operation, changes to this parameter will be reflected in the record that is written to/updated in the database.

The 'Trigger Function' is designed to be called by the CA 2E Trigger Router program YTRIGGER when a database change (delete, insert or update) occurs and the database file has a trigger attached where the specified trigger program is YTRIGGER. Trigger Functions can contain any processing that you might find in an Execute External Function (EXCEXTFUN) except that they should not contain any calls to any database functions (CHGOBJ, CRTOBJ, DLTOBJ or RTVOBJ) for the file over which the Trigger Function is built.

Web Service

The CA 2E 'Web Service' allows you to model a web service within a 2E model. A Web Service has a one-to-one relationship with a modelled 2E Service Program function. Furthermore, a Web Service function can have zero-to-many modelled web service instances. Each web service instance may contain web service operations for each procedure within the modules that comprise the associated SRVPGM.

# Function Options

Function options can be accessed by pressing Z against a function in Synon and then press F7.

Function options screen allows you to change properties of a function like switching on/off Commitment control, Subfile selection, Confirm prompt etc.

Available Function Options   
Create, Change, Delete, Bypass key screen   
Dynamic pgm mode   
Confirm prompt, Initially Yes   
Subfile selection   
Commit control, Generation mode   
Generate Help, Help type for NPT   
Screen text constants   
Reclaim resources, Copy back messages   
Generate error routine, Send all error msg   
Closedown program  
Workstation Implementation  
Distributed File I/O Control

Synon FAQs

**1. Type of Access Path.**Ans total 6 types of Access path. PHY, RTV, RSQ, UPD,SPN,QRY.  
**2. Why we will use Span Access path.**  
Ans Span Access path will be used to create Edit and Display Transaction functions. Using this we can populate and we can update two files at a time. (For ex:Join logical file.)  
**3. Difference between Internal Function and External Function.**   
Ans Internal functions are not independent programs these are kind of subroutines. We cannot generate (compile) internal functions independently; if we want we can include this in external functions or any other functions. External Functions are stand alone functions which can be compiled independently.  
**4. How many parameters we can pass in synon program.**  
Ans: We can pass 9 parmeters at a time. If we want to pass more than then we can use arrays, or whole access paths.  
**5. How many types of files we have in synon.**  
Ans : Reference file, capture file and Structure file these are important files.  
**6. Difference between these files.**  
Ans : Whenever you create Reference file. 5 functions will be created automatically. 3 internal functions (those are Create object, change object, delete object) and 2 external functions will be created(Edit file and select file).  
**7. Types of file to file relations.**  
Ans : Refers to, Owned by, includes.  
**8. Types of File to field relations.**  
Ans : Known by, has, qualified by.  
**9. How many function options we have it for Retrieve object.**  
Ans : Only one i.e. Share Sub routine (Yes or No)  
**10. What is the use of Share Sub routine.**  
Ans : If we make it as Yes, then if we are calling this retrieve obj in one program and in many places, instead of creating different different subroutines it will create only one subroutine.  
**11. How many user points we have it in retrieve object.**  
Ans :four user points.  
**12. How we can call external programs (outside cbl or cl pgms) in synon functions.**  
Ans : Using Execute User programs.  
**13. What are function options and what is the use.**  
Ans :we have many function options. For ex: Create object, change object, delete object, confirm prompt, commitment control. In Edit file function you want to disable delete functionality you can achieve this making "Delete object" as "No" in function options.  
**14. Different between LCL and WRK variables.**  
Ans : Local variables never holds any junk values if you are not initializing also. But WRK variables always holds the values. WRK are kind of Global variables.  
**15. How you will insert print object in printer file.**  
Ans : Using Option "T" on the printer file function.  
**16. In Printer file function what all are the Context you can see**.  
Ans : CUR, NXT, WRK, LCL etc. The important are CUR and NXT.  
**17. If you have retrieve object their you have two output variables in parameter list and in you functionality you do not want second output variable. What will you do, you will create one more retrieve object or any other solution.**  
Ans: Instead of creating new retrieve object you can use the same one, the one output variable which you do not want make it as NLL.

**Some more Interview Questions on Synon.**  
  
**Questions on File Types**  
  
1. Name the different File object attributes  
  
File Name and File type:   
ie. Name:Company Type:REF; Name:Order Type:CPT; Name:Audit Type: STR  
  
REF – Database Reference File   
CPT – Database Capture File  
STR – Structure File (data structure)  
  
2. What is a STR file used for?  
  
A STR file is used for implementing a data structure of fields that are not logically included in a physical data file. An example would be audit information or job run type information (program name, user id, run time, etc.). It can also be used for passing parameters between functions.  
  
3. What is a REF file?  
  
A REF is a master file like customer or company. It contains static information that rarely changes and is typically referred to rather than appended to with additional records.  
  
4. What are the pragmatic DDS differences between REF and CPT files?  
  
CPT files will have CRTOBJ, DLTOBJ, and CHGOBJ functions automatically generated by Synon, all associasted with UPD access paths. REF files will also have SELRCD and EDTFIL functions automatically generated associated with RTV access paths. As a result, REF files will have additional DDS generated to accommodate the RTV access paths.  
  
5. What is the definition of a CPT file?  
  
A CPT (Capture file) is a transaction file like Order or Order Detail. It also contains static information but is dynamic in the sense that records are often appended to it and modified within it.   
  
6. When you create a REF type file, what are the 5 functions that are automatically created by SYNON?  
  
CHGOBJ – Change object  
CRTOBJ – Create object  
DLTOBJ – Delete object  
EDTFIL – Edit File  
SELRCD – Select Record  
  
7. What is the difference between logical and physical files?  
  
A physical file is where the data resides. A logical file provides a different view of the physical file allowing for different keys and sequencing.  
  
8. What is an assimilated file?  
  
An existing physical file that has been retrieved into a SYNON data model.  
  
9. Name 2 object types.  
  
FIL File  
FLD Field  
  
10. Name 5 relation types and their relation type level  
  
DEFINED AS – FIL Declares that the file exists (Automatically created by Synon for each file)  
EXTENDED BY – ATR Declares the file to have a one-to-one or one-to-none relationship with another file  
HAS – ATR Declares the field as a data field on the file  
INCLUDES – ATR Causes fields of the referenced file or included structures to be included as attributes in the referencing file  
KNOWN BY – KEY Declares the field to be present as a key field on the file  
OWNED BY – KEY Specifies that the keys of the owning file are to become major keys of the owned by file  
QUALIFIED BY – KEY Declares a field to be present on file as a key field; is used with continuous variables  
REFERS TO – ATR Causes the key fields of the referenced to be included as non-key fields on the referring file.  
  
11. What are the types of SYNON Primary Key relations? Write them down in the order they should be sequenced.  
  
Owned by   
Known by  
Qualified by   
Has  
Refers to   
Includes  
  
12. What are the types of foreign key SYNON relations (or file to file relations)?  
  
Owned by  
Refers to  
  
13. What is the effect on a SYNON entity that has a File to File relationship with another entity?  
  
Referential integrity between the entities is built into the functions built over the entities. Select Record functions are automatically built to provide F4 or ? lookups against the entities.  
  
14. What are the benefits proved by File to File relations in a SYNON generated application?  
  
You do not have to perform the validations in your action diagram. SYNON builds them in.  
  
15. What is the data type that is most frequently used?  
  
This will probably vary by data model but it should be the REF type so that field attributes and domains are inherited.  
  
16. What data types do we NOT create REF data types for? Why?  
  
DTE and TIM data types because the data type itself defines it is set of allowable values or domain.  
  
17. How can multiple Refers To relationships be uniquely identified on an entity?  
  
Use the FOR text or replace the field entries with correctly named fields.  
  
18. How in SYNON is the term ‘SHARING’ used?  
  
When you have relationships that have keys in common and you do not want to have additional occurrences of the key field, you may say that the 2 relationships should Share the key. Sharing \*ALL is defaulted by SYNON so different versions of the same key field are not added to the entity.  
When you do not want multiple versions of the same field as in the example ‘Employee refers to Employee for Supervisor and For Manager’, and Employee is KNOWN by Employee User ID, we want 3 of the Employee User ID field. You should specify \*NONE for sharing to make this occur and use FOR text to differentiate each of the 3 occurrences.  
  
19. In data modeling, what does SHARING \*NONE mean?  
  
The keys of the file being referred to are not shared by keys on the file has the ‘REFERS TO’ relation.  
  
20. What is the impact of designating a file as a Capture or Reference file?  
  
If you do not designate a file as a Capture or Reference file correctly, you are not describing the entity as well as you could. The only real impact is than an Edit File and Select Record function will get created automatically for a Reference File and in a Capture File, they will NOT get created automatically but you may do so manually.  
  
  
**Questions on Field Types**  
  
21. Name 8 field types.  
  
**Numeric**  
NBR – Number  
PCT – Percentage  
PRC – Price  
QTY – Quantity  
VAL – Value  
DTE – Date  
TME – Time  
SGT – Surrogate  
**Alphanumeric**  
CDE – Code  
DT# – \*ISO Date  
IGC – Ideographic  
NAR – Narrative Text  
TM# – Time  
TS# – Time Share  
TXT – Text  
VNM – Valid System Name  
**Special Fields**  
REF – Reference  
STS – Status  
  
22. Name 4 field usages.  
  
ATR – Attribute database file   
CDE – Code  
CNT – Count function field  
DRV – Derived function field  
MAX – Maximum function field   
MIN – Minimum function field  
SUM – Sum Function field  
USR – User function field  
  
23. What is a Derived field?  
  
Any field that calculated from other fields rather than physically stored in the database.  
  
  
24. How is a Derived field implemented in RPG?  
  
Using a CALC specification.  
  
25. How many output parameters can a derived function field have?  
  
One the derived field itself.  
  
26. What are ATR type fields?  
  
ATR type fields are data attribute, non–key fields associated with a file. ie. Company Address in Company file.  
  
27. What are CDE type fields?   
  
CDE type fields are key fields associated with a file defined by a ‘Known By’ relation. Ie. Company Code in a Company file.  
  
28. What are USR type fields?   
  
Fields made by a programmer for a function that are not typically classified elsewhere and hold special values.  
  
29. What are virtual fields and what are they used for?  
  
A field, which is physically present on one file but logically, used on another file. These fields are most commonly used when a relationship is going to be checked numerous times.  
  
30. Where can virtual fields be specified?  
  
Virtual fields can only be specified on the ‘Owned by’, ‘Refers To’ and ‘Extended by’ relation types. Ie. Division is Owned by Company. Company fields can be virtualized to Division but Division fields cannot be virtualized to Company.  
  
**Questions on Access Paths**  
31. Name 4 types of access paths.  
  
PHY Physical  
QRY Query  
RSQ Resequence  
RTV Retrieval  
SPN Spanned  
UPD Update  
  
32. What is the difference in the DDS between RTV, RSQ, and UPD?  
• RTV – is in the physical file key sequence, includes virtual fields, can be edited to drop some or all non–key fields, and can define select or omit logic  
• RSQ – includes all the functionality of RTV and allows for an alternative key sequence that does not need to be unique  
• UPD – is in the physical file key sequence and includes all fields from related files except virtual fields   
  
33. What OS/400 objects does the QRY access path create?  
  
A physical file, a logical file and a CL Program.  
  
34. Describe when a QRY access path can be used, and when it cannot?  
  
QRY access paths can only be used with the following function types:  
Display File  
Select Record  
Retrieve Object  
Print Object  
Print File   
Cannot be used with a Create, Change or Delete Object.  
  
35. When would an OS/400 Query be preferable to a SYNON QRY ACP?  
  
When you want to create, change or delete records based on a QRY.  
  
36. How is a SPAN access path implemented in DDS?  
  
The generated names of the fields will be prefixed with the applicable file name to uniquely distinguish the common fields.  
  
37. What would a span access path be used for?  
  
A SPN access path specifies a keyed, multi–format access path. It is used to describe to Edit and Display Transaction functions, how records are to be retrieved from a pair of related files that possess a common foreign key. The files must be related by an ‘Owned By’ or ‘Refers To’ relation, with the SPN access path created over the ‘owning’ or ‘referred to’ file.   
Note: The SPN access path must be created explicitly initially defaults to the keys defined by the key relations of the based on files but can be overridden to an alternative key sequence is associated with RTV access path that points to an associated UPD access path that is used to carry out any updates to the based on file allows virtual fields to be specified on the access path relations and defaults to the virtual fields of the based on files relations allows specific selection of multiple access path relations.  
  
38. How do the files need to be related to correctly implement a SPAN?  
  
The files must be related by an ‘Owned By’ or ‘Refers To’ relation.  
  
39. What are 2 ways to define a Select/Omit access path?  
  
Static – Only those records that satisfy the select logic are included in the access path. Records are filtered as they are added or changed, the system determines if it should be included in the access path. As data is read, no filtering is required as it has already been performed by the access path maintenance.  
Dynamic – All records that satisfy the select logic are included in the access path. Records are filtered as they are read by the program.  
  
40. Describe the difference between Static and Dynamic access paths.  
  
Static processing is where the records are selected at generation time. It requires a separate index and should be used for master files and tables. Static is the most commonly used select/omit processing option and is the fastest.  
Dynamic processing is where the records are selected at run time. It allows for the sharing of active indexes and is the only option that can be used to include virtual fields.  
  
41. What is the primary purpose of a RSQ access path?  
  
To provide an alternate sort of the entity by physical fields.  
  
42. What is a QRY type access path?  
  
It allows for the use of an Open Query file within SYNON.  
  
  
**Built–in Functions**  
\*Add \*Commit \*Compute \*Concat  
\*Cvtvar \*Date Details \*Date Increment \*Div  
\*Div with Remainder \*Duration \*Elapsed Time \*Exit Program  
\*Modulo (Mod 10/11) \*Move \*Move All \*Mult  
\*Quit \*Rollback \*RTVCND \*Set Cursor  
\*Sub \*Substring \*Time Details \*Time Increment  
  
  
**Message Functions**  
•Send Error Message (SNDERRMSG)  
•Send Informational Message (SNDINFMSG)  
•Send Complete Message (SNDCMPMSG)  
•Send Status Message (SNDSTSMSG)  
•Retrieve Message (RTVMSG)  
•Execute Message (EXCMSG)  
  
43. Describe the difference between an ‘Edit Transaction’ function and an ‘Edit File’ function.  
  
The Edit Transaction function allows the user to update 2 different file formats, which can be linked together, such as an Order Header and Order Detail file. The 2 files must be joined using a SPAN access path. The Edit File function allows the user to update multiple records with the same format simultaneously.  
  
44. What is the difference between an internal and an external function?  
  
An internal function generates code within a given function as where an external function generates a CALL to another function.  
  
  
45. Describe the difference between an ‘Edit Transaction’ function and an ‘Edit File’ function.  
  
The Edit Transaction function allows the user to update 2 different file formats, which can be linked together, such as an Order Header and Order Detail file. The 2 files must be joined using a SPAN access path. The Edit File function allows the user to update multiple records with the same format simultaneously.  
  
46. What is the difference between an internal and an external function?  
  
An internal function generates code within a given function as where an external function generates a CALL to another function.  
  
CALL to another function.  
47. Does a Prompt and Validate (PMTRCD) automatically read records from the database file that it is built over?  
  
No.  
  
48. You only wish to get the first record in a file, How?  
  
Do not pass any key parameters (a blank key) to a RTVOBJ function and, at the User Point ‘Process Data Record’, do NOT code any user logic.  
  
49. How would you design a batch function for updating records?  
  
Use an Execute External function type which has a Retrieve Object in it then, once a record is read, use a Change Object to update the record.   
(The Retrieve Object can be omitted if the Change Object is to handle reading and updating the same file.)  
  
50. How would you design a batch function to print a report for a given range of dates?  
  
Call a CL program which will call a Prompt and Validate function to fill in the record selection parameters. Then the CL program will process the SBMJOB command to submit the Print File program with the appropriate parameters.  
  
51. What are the ‘important’ function options for an EDTRCD function type?  
  
Exit after Add  
Bypass Key Screen  
  
52. List all the alternative ways for putting several variables into one field.  
  
RTV Message   
Concatenate built in function  
  
53. List the contexts available in a PRTFIL function?  
  
CUR NXT PAR  
JOB PGM CON  
CND WRK   
  
  
54. What are the different kinds of Synon message functions available and how are they used?  
  
• SNDERRMSG – an error message sent to a calling function used to provide diagnostic messages arising from user validation  
• SNDINFMSG – an informational message sent to the message queue of a calling program  
• SNDCMPMSG – a message sent to the function that called a standard function to indicate that a process has completed successfully  
• SNDSTSMSG – a message sent to a calling function to provide information about the progress of a long running process  
• RTVMSG – a function that specifies that the messages text is to be retrieved from the message file into a function  
• EXCMSG – a function that specifies that allows for a command string to be executed from with a Synon function via a request message that is to be executed by the calling function (usually a CL command like SBMJOB)   
Note: You should use SBMJOB with EXCEXTFUN or EXTUSRPGM and PRTFIL functions for batch execution from within an action diagram so that you can pass numeric parameters. Also, references to submitted functions are visible to the Synon/2E impact analysis facilities  
  
**Questions on Conditions**  
  
55. What types of Conditions are there?  
  
CMP (Compare) – used for conditions that specify a scope of values that a field can receive. The scope is defined as a fixed value and an operator.ie. X \*GE Y or A \*LT B. CMP conditions cannot be used with STS fields.  
RNG (Range) – used for conditions that specify a range of values that a field can receive. ie. ‘Order Quantity is 10 and 100’ or ‘Transaction Value is GT 25 and LE 250’. RNG conditions cannot be used with STS fields.  
VAL (Values) – used for conditions that specify a value that a field can receive and can only be used with STS fields. You need to specify 2 related values for a value condition: an internal which will be held in the database and against which the condition will be checked, and an external value which the user enters on the external function application panel. Synon generates the necessary code to interpret the values. The internal and external values can have different lengths but you will need to use value mapping facility to translate between them. To do this, you must specify Y on the Translate field on the Edit Field panel.  
LST (Values List) – Similar to the VAL condition in that it can only be used with STS fields. Provides a list of valid values that a field can receive and allows the user to prompt for the list by entering a ? in the field or pressing F4.  
  
56. What is QUIT?  
  
Quit is a built–in function that exits from an action diagram sequence construct. It has no parameters and returns control to the calling sequence construct.  
  
a. When would you use it?  
  
When you want to quit processing if a certain condition exists ie. Do not perform update processing if errors occurs on the validate processing.  
  
57. What is EXITPGM?  
  
EXIT Program is a built–in function that specifies an exit from a program that returns processing control to the calling program. A return code parameter is passed to the calling program stating the reason and circumstances under which the program is exiting.  
  
58. What happens when processing comes to a \*QUIT within an Execute Internal Function?  
  
Processing in the program will proceed to the exit point of the function that called the Execute Internal Function.  
  
  
**Questions on Parameters**  
  
59. Where may POS type parameter be used?  
  
60. How can you pass more than 9 parameter fields into a function?  
  
Specify a file name with the access path of \*NONE and context of FLD, create a STR file type specifically for the use of passing parameters, or use \*ARRAYS (rel 4.0 and greater) to define a structure for the use of passing parameters.  
  
61. Why would you want to define a parameter as \*NEITHER?  
  
To allow mapping of the field to the screen.  
  
62. What is the difference between FLD, RCD and KEY contexts when passing parameters?  
  
FLD – One parameter is passed for each field specified.  
RCD – One parameter is passed for all fields specified in the structure.  
KEY – One parameter is passed for all key fields specified in the structure.  
  
63. If the business requirements necessitate that more than just the key fields need to be passed from a Select Record function, what would have to be done to meet this requirement?  
  
Use a Display File in place of the Select Record function.  
  
  
64. Is it preferable to pass parameters as \*FIELDS or \*FILES? Why?  
  
It is usually preferable to use \*FILES because you will be able to select multiple fields from the list of fields on the file selected. If you use \*FIELDS, you will only have 9 parameters available.  
  
  
65. What does ELM mean and where is it used?  
  
ELM is the element context of an array that contains the fields defined for the last–accessed element of a specified array. It is only valid in the \*CVTVAR built–in function and may be specified for either the input or output parameter.  
  
66. Which fields in an Edit File will be positioner fields and which will be selector fields?  
  
Key fields in an Edit File will be positioners. Non–key fields will be selector fields.  
  
  
67. How do you create a new \*Standard Header/Footer to be used for a report?  
  
Display all functions over the \*Standard Header/Footer entity. Select one to copy and copy it to a new name which accurately describes the purpose you are using if for. Next, review the screen design and make the changes to the function required for your task through normal screen design techniques.  
  
68. How do you change the \*Standard Header/Footer on a screen report?  
  
From Function Options, press F5 to Select a new \*Standard Header/Footer. You cannot create a new standard header/footer when you are in the process of selecting one.  
  
  
**Journaling / Commitment Control Processing:**  
  
69. What is Journaling and Commitment Control Processing?  
  
Journaling and Commitment Control processing is a means of automatically grouping a number of database updates into a single transaction for the purpose of recovery: either all or none of the updates will take place.   
  
70. What are the commands to perform journaling within a SYNON function?  
  
\*COMMIT and \*ROLLBACK  
  
71. Where would a programmer control whether or not a program is run under commitment control?  
  
In the Edit Function Options section.  
  
72. What are the values for controlling whether or not a program is run under commitment control?  
  
M (Master ) – Program is run under commitment control, program is controlling program and contains commit points.  
S (Slave) – Program is run under commitment control; however, the commit points are in another program.  
N (None) – Commitment control does not exist in this program.  
  
  
73. What is a ‘CPF4131’?  
  
It is an IBM code for a ‘level check&rsquo condition.