



**Flexible Subfile Functions in 2E  
- presentation for the 5th int'l  
CA Plex/2E user conference  
(Session 9C)**

Magnus Gullö (Sweden)  
Nordea Liv & Pension (Stockholm)  
May 2011



## Background ("Why do this?")

- **Are the options 2E Access Paths offers always enough for building good subfile functions?**
  - I don't think so.
  - When using surrogate keys, you tend to need "complex" access ways to your files in order to build useful screens.
- **My experience from using Query Access Paths is unfortunately not a success story.**
  - Have you also struggled with them?
- **Thanks to Peter Verstegen (Group Solution) who taught me this approach, showed me some of the tweaks and gave me the user source I'm presenting!**

## Ambition with this presentation

- To demonstrate the IDEA for an alternate approach.
- (Inarguably; a lot of what I show can be accomplished in other ways.
  - Including details of how to code.)
- To show EXACTLY how to write some common user source that's needed for the concept to work.
- To show how building blocks of a certain type can be written.
  - You'll need them repeatedly.
  - They are EASY to copy between files!
- To demonstrate ONE example how the technique can be used.

# Consider the following tiny model

```

FIL REGION ..... Known by      10 FLD Region Id
FIL REGION ..... Has           20 FLD Region Name
    
```

File name. . . . : REGION

Field	Type	Ocr	Et	Ksq	GEN name	Length
Region Id	NBR		K	1	AQNB	5.0
Region Name	TXT		A		AQTX	35

```

FIL CUSTOMER ..... Known by      10 FLD Customer Id
FIL CUSTOMER ..... Has           20 FLD Customer Name
FIL CUSTOMER ..... Refers to     30 FIL REGION .....
    
```

File name. . . . : CUSTOMER

Field	Type	Ocr	Et	Ksq	GEN name	Length
Customer Id	NBR		K	1	ASNB	5.0
Customer Name	TXT		A		ATTX	35
Region Id	NBR		A		AQNB	5.0

# Access Paths for REGION

Typ	Access path	Source mbr	Key	Index options
PHY	Physical file	EXARCPP	NONE	
UPD	Update index	EXARCPL0	UNIQUE	IMMED
RTV	Retrieval index	EXARCPL1	UNIQUE	IMMED
RSQ	By Name	EXARCPL2	FIFO	IMMED

```
le name . . . . . : REGION
cess path name. . . . . : By Name
rmat text . . . . . : REGION
sed on. . . . . : REGION
```

Field		GEN	Name	Type	Key no.	Dsc
Region Id	NBR		AQNB	K	<u>2</u>	-
Region Name	TXT		AQTX	K	<u>1</u>	-

# Access Paths for CUSTOMER

Typ	Access path	Source mbr	Key	Index options
PHY	Physical file	EXAUCPP	NONE	
UPD	Update index	EXAUCPL0	UNIQUE	IMMED
RTV	Retrieval index	EXAUCPL1	UNIQUE	IMMED
RSQ	By Region Id/Name	EXAUCPL3	FIFO	IMMED

```
File name . . . . . : CUSTOMER          Attribut
Access path name. . . . . : By Region Id/Name Type. .
Format text . . . . . : CUSTOMER
Based on. . . . . : CUSTOMER          Format N
```

? Field		GEN	Name	Type	Key no.	Dsc
_ Customer Id	NBR		ASNB	K	<u>3</u>	_
_ Customer Name	TXT		ATTX	K	<u>2</u>	_
_ Region Id	NBR		AQNB	K	<u>1</u>	_

# We have some REGIONS and CUSTOMERS...

```
Type options, press Enter.
4=Delete

Opt  Region  Region Name
     Id
--    1      Götaland
--    2      Svealand
--    3      Södra Norrland
--    4      Norra Norrland
```

```
Type options, press Enter.
4=Delete

Opt  Customer  Customer Name                Region
     Id       Id
--    1      Allers                    1
--    2      MLT                      2
--    3      Dalälvens el              3
--    4      Smurfit Kappa            4
--    5      Ratos                    1
--    6      SSAB                     4
--    7      Dannes plåt i Älvdalen    3
```

## Now let's say the system owner wants this:

```

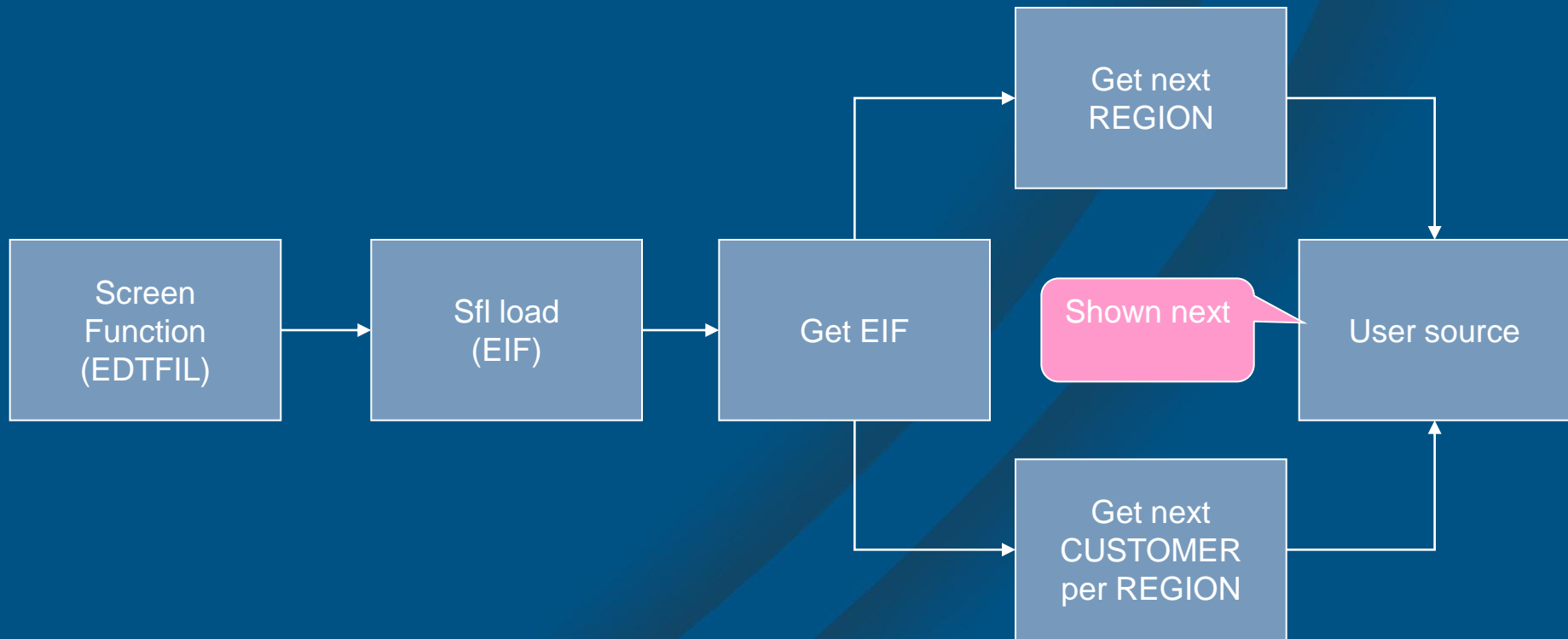
Work with Customers
Sort 1
? Region Name
_ Götaland
_ Norra Norrland
_ Södra Norrland
_ Svealand
Sort 2
Customer Name
Allers
Ratos
Smurfit Kappa
SSAB
Dalälvens el
Dannes plåt i Älvdalen
MLT
    
```

- How would you build it?
- I'm sure this simple subfile could be built using QRY-ACP.
  - If you feel like dealing with QRY ACP:s.
  - But what if it gets a bit more complex?

## Approach

- **Bottom up:**
- **First, I'll describe some common building blocks.**
  - RPGLE user source. Could probably be translated to COBOL as well.
- **Then, I'll describe how to build two effective Get Next functions that we'll need.**
- **Internal Function to get records using the Get Next functions**
- **Subfile load internal function**
- **Screen function**

# Function structure



# Common Building Blocks

# Common fields needed

Field name	Type	REF	Length	Field name	Field usage
f					
First/Next	STS	REF	1	AMST	USR
First/Next basefield	STS		1	AIST	USR
First/Next 1	STS	REF	1	ALST	USR
First/Next 2	STS	REF	1	AJST	USR

Condition	Type	Op	File/From value
*ALL values	LST	**	
End-of-file	VAL		E
First	VAL		F
Next	VAL		N

# User source functions needed

```

EDIT FUNCTIONS                                     @Magnus diverse
file name. . . : User source                        **
-----
Function                Function type            Access
Goto tag 1              Execute user source    *NONE
Goto tag 2              Execute user source    *NONE
Put tag 1               Execute user source    *NONE
Put tag 2               Execute user source    *NONE
Stop load edtfil       Execute user source    *NONE
    
```

- No parameters

# Tag functions

```
C* START OF USERSOURCE  
C   TAG1           TAG  
C* END OF USERSOURCE
```

```
C* START OF USERSOURCE  
C   TAG2           TAG  
C* END OF USERSOURCE
```

```
C* START OF USERSOURCE  
C           GOTO   TAG1  
C* END OF USERSOURCE
```

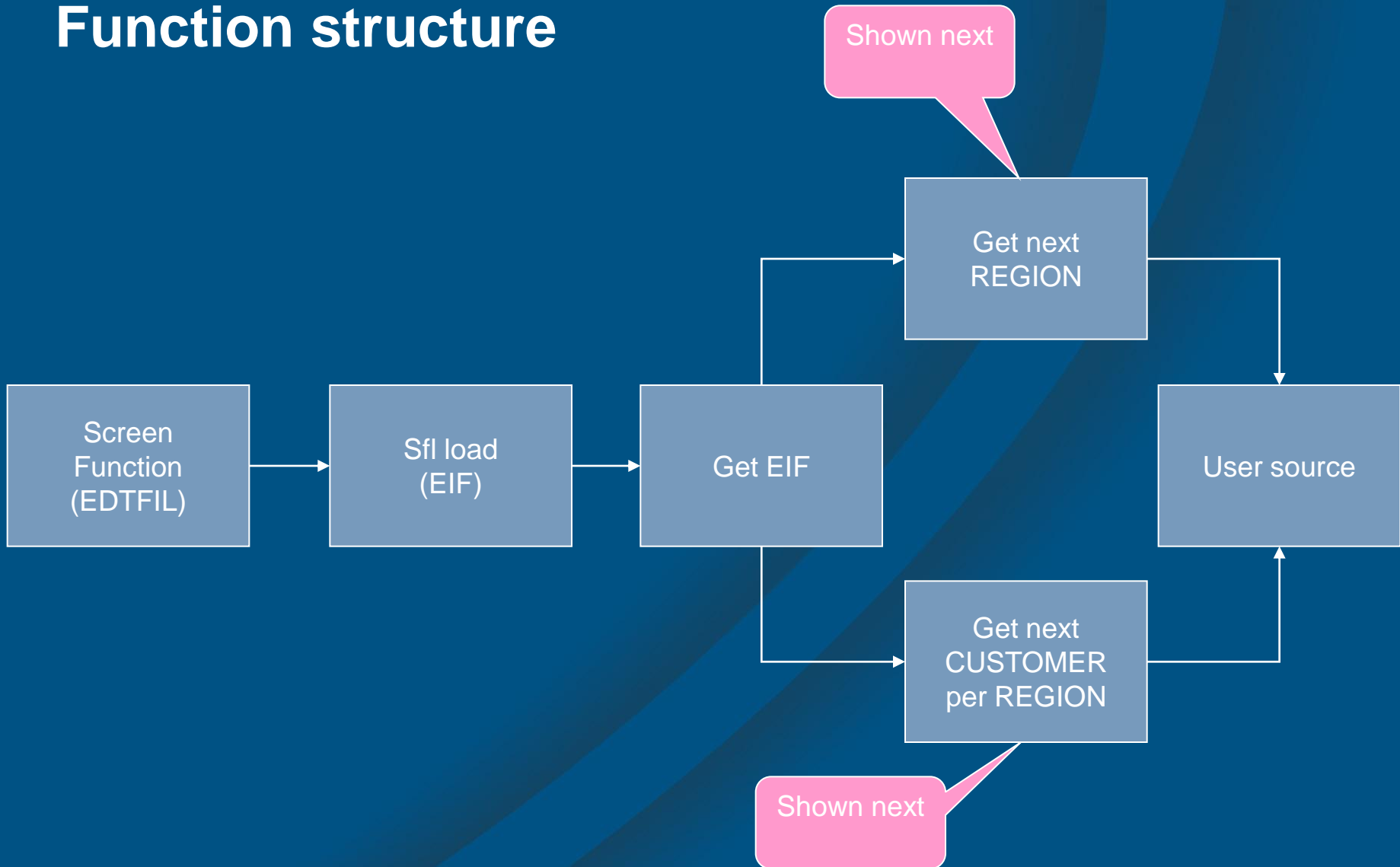
```
C* START OF USERSOURCE  
C           GOTO   TAG2  
C* END OF USERSOURCE
```

# Stop load edtfil

```

C* START OF USERSOURCE
C          SUB          1          ÅÅRR
C          MOVE        '1'        *IN82
C          * SAVE HIGHEST SFL RECORD LOAD CAN CONTINUE AT END POINT
C          ÅÅRR          IFGT        ÅÅRRMX
C          * CALCULATE TOP LINE
C          ÅÅRROK        DIV          ÅÅSFPG          ÅÅSPG
C          ÅÅSLIN        MVR          ÅÅSLIN
C          ÅÅSLIN        IFGT        0
C          ÅÅRR          SUB          ÅÅSLIN          ZZSFRC
C          ÅÅRR          ELSE
C          ÅÅRR          SUB          ÅÅSFPG          ZZSFRC
C          ÅÅRR          END
C          ÅÅRR          ADD          1          ZZSFRC
C          ÅÅRR          Z-ADD        ÅÅRR          ÅÅRRMX
C          ÅÅRR          END
C* END OF USERSOURCE
    
```

# Function structure



# Get Next functions

# Get Next for REGION – By Name

```
File name. . . : REGION                **      1ST LEV
```

Function	Function type	Access path
Change REGION	Change object	Update index
Create REGION	Create object	Update index
Delete REGION	Delete object	Update index
Edtfil	Edit file	Retrieval index
Get one	Retrieve object	Retrieval index
<b>Next by Name</b>	<b>Retrieve object</b>	<b>By Name</b>
T-Next by Name	Execute external function	*NONE

File/*FIELD	Access path/Field/Array	Seq
REGION	By Name	KEY 10
REGION	By Name	RCD 20
*FIELD	First/Next	FLD 30

Field	Usage	Role
Region Name	B	POS
Region Id		

Field	Usage	Role
Region Id	0	MAP
Region Name		

Field	Usage	Role
First/Next	B	MAP

# Get Next for REGION – By Name

```
> USER: Initialize routine
.--
.-CASE
.ö-PAR.First/Next is Next
.ö Goto tag 1 - User source *
.ö
.ö-*OTHERWISE
.ö PAR.First/Next = CND.Next
.'-ENDCASE
.--
```

```
> USER: Processing if Data record not found
.--
. PAR = CON By name
. PGM.*Return code = CND.*Record does not exist
.--
```

```
> USER: Process Data record
.--
. PAR = DB1 By name
.<-- *QUIT
. Put tag 1 - User source *
.--
```

```
> USER: Exit processing
.--
. PAR = CON By name
. PGM.*Return code = CND.*Record does not exist
.--
```

# Get Next for REGION – By Name

```

=====
* Next by Name - REGION *
=====
C          EVAL      WORTN = ' '
* USER: Initialize routine
* CASE: PAR.First/Next/Eof is Next
* Goto tag 1 - User source *
C          IF        WL0003 = 'N'
C* START OF USERSOURCE
C          GOTO      TAG1
C* END OF USERSOURCE
C          ELSE
* CASE: *OTHERWISE
C          MOVEVL    'N'          WL0003
C          END
    
```

```

* USER: Process Data record
* PAR = DB1 By name
C          DOW       NOT *IN90
C          MOVEVL    WAAQTX          WL0001
C          Z-ADD     WAAQNB          WL0002
C          GOTO      SAEXIT
* Put tag 1 - User source *
C* START OF USERSOURCE
C          TAG1      TAG
C* END OF USERSOURCE
C          READ      FARCPB2
C          ENDDO
* USER: Exit processing
* PAR = CON By name
C          EVAL      WL0001 = ' '
C          Z-ADD     *ZERO          WL0002
C          EVAL      WL0003 = ' '
C          MOVEVL    'Y2U0005'     WORTN
    
```

# Get Next for CUSTOMER – By Name (RST by REGION)

```

le name. . . : CUSTOMER                                **    1ST LEVEL
n
-----
Function                Function type            Access path
Next by Region Id/Name  Retrieve object          By Region Id/Name
  
```

- Copy the Get Next on REGION
- Select the right ACP

# Get Next for CUSTOMER – By Name (RST by REGION)

- Change the parameters (page 1)

```

tion name. . : Next by Region Id/Name      Type : Retrieve object
ived by file : CUSTOMER                    Acpth: By Region Id/Name
                                           Passed      P
File/*FIELD  Access path/Field/Array      Seq      C
CUSTOMER     By Region Id/Name            KEY      10
CUSTOMER     By Region Id/Name            RCD      20
*FIELD       First/Next                   FLD      30
    
```

```

on name. . : Next by Region Id/Name      Type : Retrieve object
ed by file : CUSTOMER                    Acpth: By Region Id/Name
ter (file) : CUSTOMER                    Passed as: KEY

Field      Usage  Role
Region Id  I      RST
Customer Name  B      POS
Customer Id
    
```

# Get Next for CUSTOMER – By Name (RST by REGION)

- Change the parameters (page 2)

```

on name. . : Next by Region Id/Name      Type : Retrieve object
ed by file : CUSTOMER                   Acpth: By Region Id/Name
ter (file) : CUSTOMER                   Passed as: RCD

Field                Usage  Role
Customer Id         0      MAP
Customer Name
Region Id
    
```

```

on name. . : Next by Region Id/Name      Type : Retrieve object
ed by file : CUSTOMER                   Acpth: First/Next
ter (file) : *FIELD                     Passed as: FLD

Field                Usage  Role
First/Next          B      MAP
    
```

## Get Next for CUSTOMER – By Name (RST by REGION)

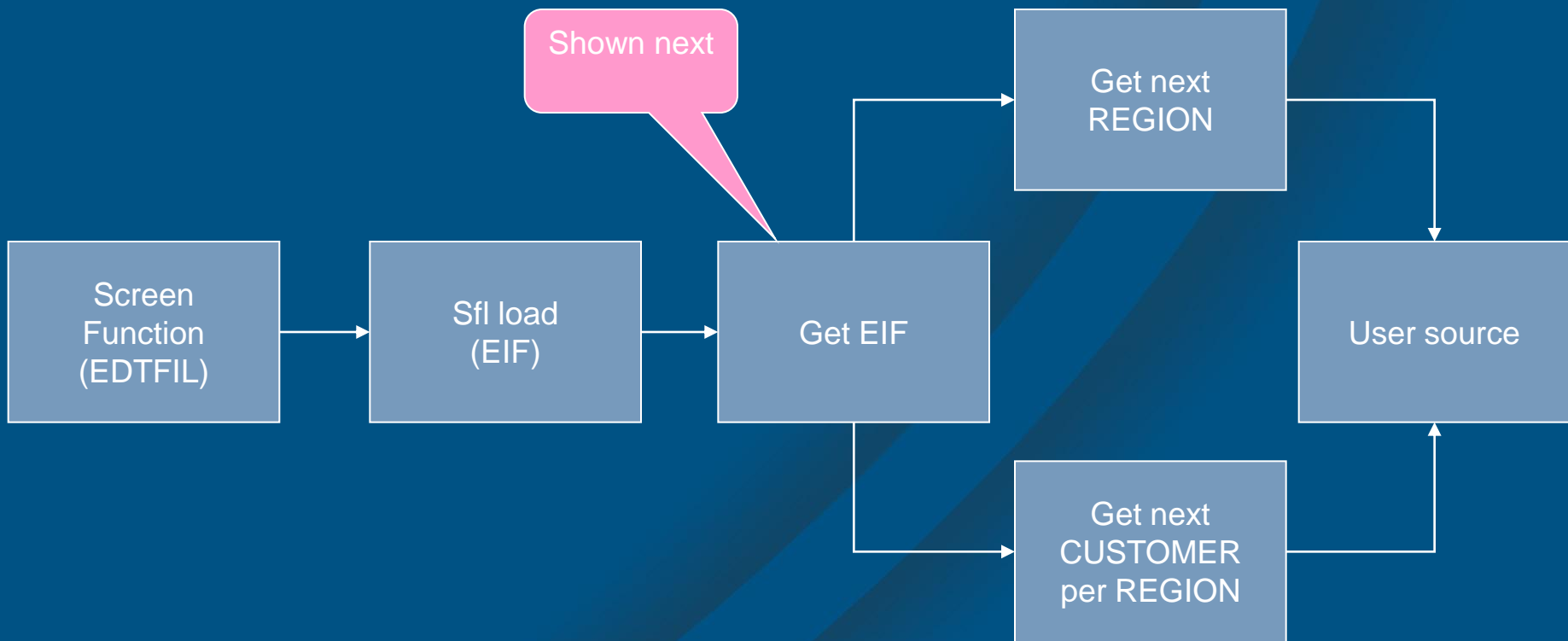
- Change to use unique TAG names if this is the second Get Next to be included in the same program.

```
> USER: Initialize routine
.--
.-CASE
ö-PAR.First/Next is Next
ö Goto tag 2 - User source *
ö
ö
ö-*OTHERWISE
ö PAR.First/Next = CND.Next
'-ENDCASE
.--
```

```
> USER: Process Data record
.--
PAR = DB1 By name
<-- *QUIT
ö Put tag 2 - User source *
.--
```

- Could be handled by encapsulating in EEF:s instead.

# Function structure



## Internal Function to get records

# Create an EXCINTFUN function and set parameters

```
EDIT FUNCTIONS                                @magnus diverse
file name. . . : CUSTOMER                      **      1ST LEVEL
n
Function      Function type      Access path
Next by Region Id/Name  Retrieve object      By Region Id/Name
Next by RegName/CustName Execute internal function *NONE
T-Next by Region Id/Name Execute external function *NONE
T-Next FIF      Execute external function *NONE
```

File/*FIELD	Access path/Field/Array	Seq
REGION	By Name	KEY 10
REGION	Retrieval index	RCD 20
CUSTOMER	By Region Id/Name	KEY 30
CUSTOMER	Retrieval index	RCD 40
*FIELD	First/Next	FLD 50

**BOTH**

# Parameter details

```

ved by file : CUSTOMER
eter (file) : REGION
Acpth: By Name
Passed as: KEY
    
```

Field	Usage	Role
Region Name	B	MAP
Region Id		

```

ved by file : CUSTOMER
eter (file) : REGION
Acpth: Retrieval index
Passed as: RCD
    
```

Field	Usage	Role
Region Id	0	MAP
Region Name		

```

ved by file : CUSTOMER
eter (file) : CUSTOMER
Acpth: By Region Id/Name
Passed as: KEY
    
```

Field	Usage	Role
Region Id		
Customer Name	B	MAP
Customer Id		

```

ved by file : CUSTOMER
eter (file) : CUSTOMER
Acpth: Retrieval index
Passed as: RCD
    
```

Field	Usage	Role
Customer Id	0	MAP
Customer Name		
Region Id		

# Action diagram

```

> Next by RegName/CustName
.--
.-CASE
  -PAR.First/Next is Next
  -Next by Region Id/Name - CUSTOMER *
  ... ** Check return code and act correspondingly
  -*OTHERWISE
  ** Initialize for REGION:
  -LCL.First/Next 1 = CND.First
  -Next by Name - REGION *
  ... ** Check return code and act correspondingly
  -ENDCASE
.--
  
```



Next page

B	Region Name	POS	FLD	<u>P</u> AR	<u>R</u> egion Name
0	Region Id	MAP	FLD	<u>L</u> CL	<u>R</u> egion Id
B	First/Next	MAP	FLD	<u>L</u> CL	<u>F</u> irst/ <u>N</u> ext 1

# First REGION found?

```

>  ** Check return code and act correspondingly
.-CASE
ö-PGM.*Return code is *Normal
ö  PAR.Region Id = LCL.Region Id
ö  ** Tell the calling function to keep reading:
ö  PGM.*Return code = CND.*Record does not exist
ö  ** Initialize for CUSTOMER:
ö  LCL.First/Next 2 = CND.First
ö  PAR.First/Next = CND.Next
ö
ö-PGM.*Return code is *Record does not exist
ö  ** No more REGIONS? Signal this fact to calling function:
ö  PGM.*Return code = CND.*Normal
ö  PAR = CON By name
ö  PAR.First/Next = CND.End-of-file
ö
ö- *OTHERWISE
ö  ** Unexpected! :(
ö  <-- *QUIT
'-ENDCASE

```

# Action diagram – unless First

```

> Next by RegName/CustName
--
.-CASE
ö-PAR.First/Next is Next
ö Next by Region Id/Name - CUSTOMER *
ö ... ** Check return code and act correspondingly
ö
ö-*OTHERWISE
ö ** Initialize for REGION:
ö LCL.First/Next 1 = CND.First
ö Next by Name - REGION *
ö ... ** Check return code and act correspondingly
'-ENDCASE
--
    
```

**FF**

Next page

I	Region Id	RST	FLD	<u>LCL</u>	Region Id
B	Customer Name	POS	FLD	<u>PAR</u>	Customer Name
0	Customer Id	MAP	FLD	<u>PAR</u>	Customer Id
B	First/Next	MAP	FLD	<u>LCL</u>	First/Next 2

# CUSTOMER found for previously found REGION?

```

> ** Check return code and act correspondingly
.-CASE
ö-PGM.*Return code is *Normal
ö PAR.First/Next = CND.Next
ö
ö-PGM.*Return code is *Record does not exist
ö ** Try to find the next REGION:
ö Next by Name - REGION *
ö ... ** Check return code and act correspondingly
ö
ö-*OTHERWISE
ö ** Unexpected! :(
ö <-- *QUIT
'-ENDCASE
    
```



Next page

B	Region Name	POS	FLD	<u>P</u> AR	Region Name
0	Region Id	MAP	FLD	<u>L</u> CL	Region Id
B	First/Next	MAP	FLD	<u>L</u> CL	First/Next 1

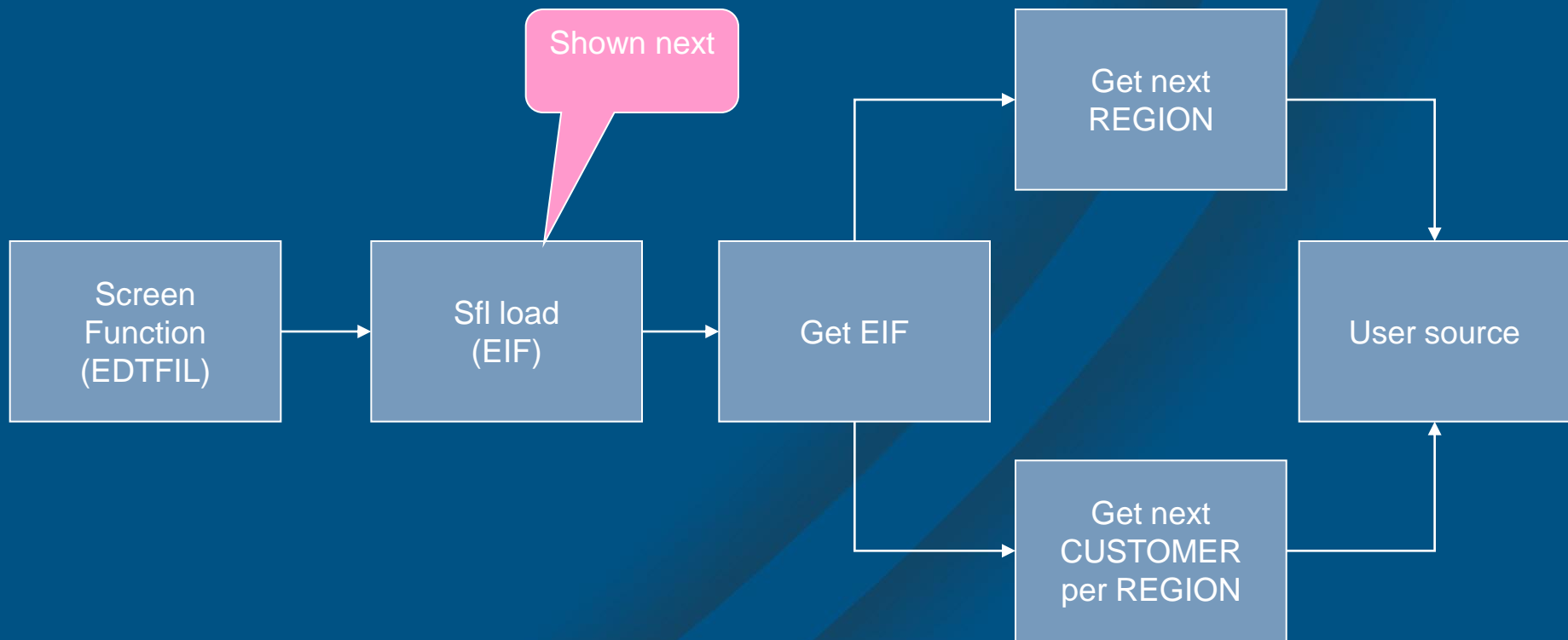
## Next REGION found?

```

>  ** Check return code and act correspondingly
.-CASE
ö-PGM.*Return code is *Normal
ö  ** Tell the calling function to keep reading:
ö PGM.*Return code = CND.*Record does not exist
ö LCL.First/Next 2 = CND.First
ö PAR.First/Next = CND.Next
ö
ö-PGM.*Return code is *Record does not exist
ö  ** No more REGIONS? Signal this fact to calling function:
ö PGM.*Return code = CND.*Normal
ö PAR = CON By name
ö PAR.First/Next = CND.End-of-file
ö
ö-*OTHERWISE
ö  ** Unexpected! :(
ö <-- *QUIT
'-ENDCASE

```

# Function structure



# Subfile load function

# Create an EXCINTFUN function and set parameters

Function	Function type	Access path
Load sfl by RegN/CustName	Execute internal function	*NONE

*FIELD		*Record selected	FLD	10
REGION	BOTH	By Name	KEY	20
REGION		Retrieval index	RCD	30
CUSTOMER		By Region Id/Name	KEY	40
CUSTOMER	BOTH	Retrieval index	RCD	50
*FIELD		First/Next	FLD	60

# Parameter details

```

ved by file : CUSTOMER
eter (file) : REGION
Acpth: By Name
Passed as: KEY
    
```

Field	Usage	Role
Region Name	B	MAP
Region Id		

```

ved by file : CUSTOMER
eter (file) : REGION
Acpth: Retrieval index
Passed as: RCD
    
```

Field	Usage	Role
Region Id	0	MAP
Region Name		

```

ved by file : CUSTOMER
eter (file) : CUSTOMER
Acpth: By Region Id/Name
Passed as: KEY
    
```

Field	Usage	Role
Region Id		
Customer Name	B	MAP
Customer Id		

```

ved by file : CUSTOMER
eter (file) : CUSTOMER
Acpth: Retrieval index
Passed as: RCD
    
```

Field	Usage	Role
Customer Id	0	MAP
Customer Name		
Region Id		

# Action diagram

```

> Load sfl by RegN/CustName
--
PAR.*Record selected = CND.*NO
.=REPEAT WHILE
!-PAR.*Record selected is *NO
! Next by RegName/CustName - CUSTOMER *
! .-CASE
!   ö-PGM.*Return code is *Normal
!   ö... ** EOF or OK?
!   ö
!   ö-PGM.*Return code is *Record does not exist
!   ö ** Keep reading
!   ö
!   ö-*OTHERWISE
!   ö<-- *QUIT
!   '-ENDCASE
! -ENDWHILE
--

```

FF

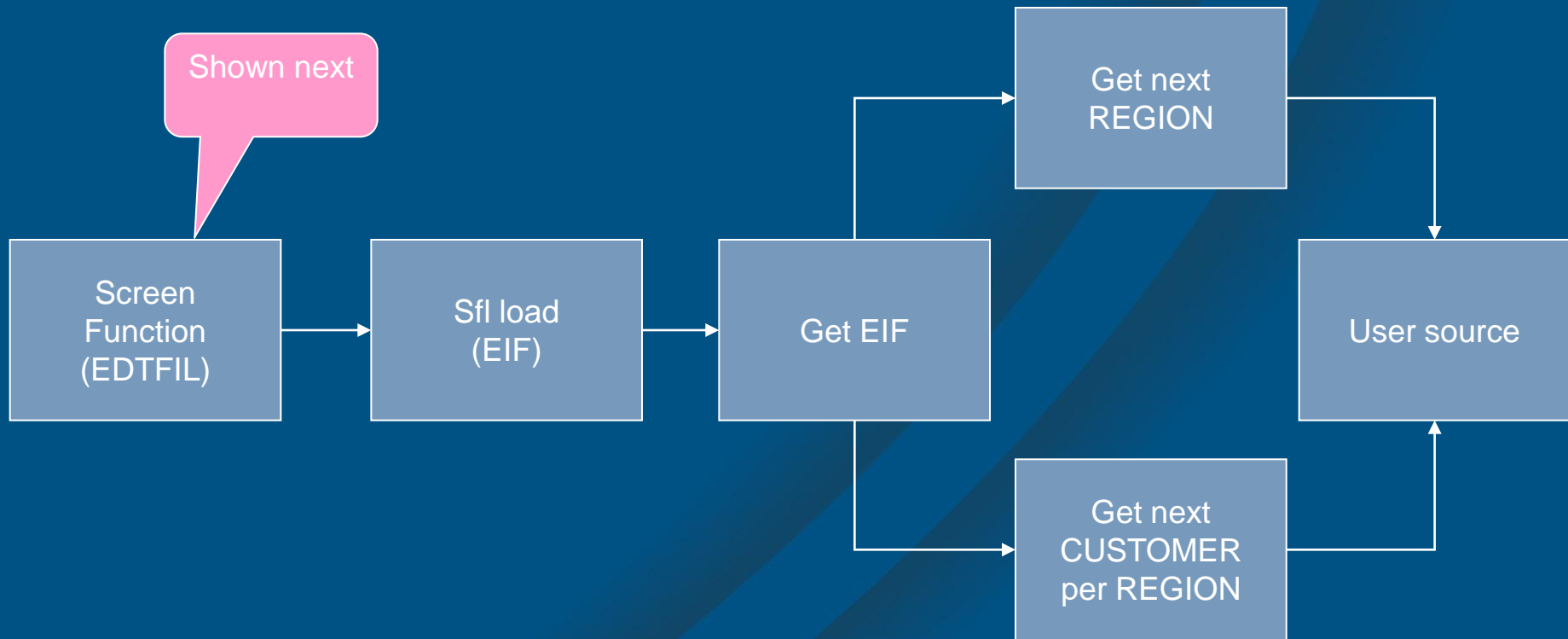
Next page

B	Region Name	MAP	FLD	PAR	Region Name
O	Region Id	MAP	FLD	PAR	Region Id
B	Customer Name	MAP	FLD	PAR	Customer Name
O	Customer Id	MAP	FLD	PAR	Customer Id
B	First/Next	MAP	FLD	PAR	First/Next

## EOF or OK?

```
> ** EOF or OK?
.-CASE
ö-PAR.First/Next is End-of-file
ö  <-- *QUIT
ö
ö
ö-*OTHERWISE
ö  PAR.*Record selected = CND.*YES
'-ENDCASE
```

# Function structure



# Screen Function

# Create an EDTFIL function and set options

File name . . .	CUSTOMER	**	1ST LEV
W			
Function	Function type	Access path	
W/w by RegName/CustName	Edit file	Retrieval index	

Create record . . . . .	<u>Y</u>	( <u>Y-Yes</u> , <u>N-No</u> )
Change record . . . . .	<u>N</u>	( <u>Y-Yes</u> , <u>N-No</u> )
Delete record . . . . .	<u>N</u>	( <u>Y-Yes</u> , <u>N-No</u> )
Dynamic program mode . . . . .	<u>N</u>	( <u>Y-Yes</u> , <u>N-No</u> )
Subfile selection . . . . .	<u>N</u>	( <u>Y-Yes</u> , <u>N-No</u> )
Confirm prompt . . . . .	<u>N</u>	( <u>Y-Yes</u> , <u>N-No</u> )



# Set function relations to user check

```

EDIT SCREEN FORMAT RELATIONS          @Magnus diverse
File name . . . . . : CUSTOMER          Attribute . . . : CPT
Access path name. . . . . : Retrieval index  Type. . . . . : RTV
Format text . . . . . : CUSTOMER
Based on. . . . . : CUSTOMER          Format No . . . : 1

? Verb      File/for      Access path/Function      Check
  Known by  Customer Id
  Has       Customer Name      (USER)
  Refers to REGION      Retrieval index          (USER)
    
```

# Edit the "create" user point to do nothing

```

: USER EXIT POINTS  Opt: X,Z=Select  V=Summary  W=Wrapper  :
:  USER: Initialize program  :
: - USER: Initialize subfile header  <<<  :
: - USER: Initialize subfile record (new record)  <<<  :
: - CALC: Subfile control function fields  :
: - USER: Validate subfile control  :
: - USER: Validate subfile record fields  :
: - CALC: Subfile record function fields  :
: - USER: Validate subfile record relations  :
: - USER: Create DBF record  + <<<  :

```

```

F  <-- *QUIT

```

EDIT ACTION - FUNCTION NAME	
Function file :	<input type="checkbox"/>
Function. . . :	*QUIT
Comment . . . :	

# Edit the header initialization

```

: USER EXIT POINTS  Opt: X,Z=Select  V=Summary  W=Wrapper  :
| USER: Initialize program                               :
: - USER: Initialize subfile header                     <<< :
: - USER: Initialize subfile record (new record)       <<< :
: - CALC: Subfile control function fields              :
: - USER: Validate subfile control                     :
: - USER: Validate subfile record fields               :
: - CALC: Subfile record function fields               :
: - USER: Validate subfile record relations            :
: - USER: Create DBF record                             + <<< :

```

```

> USER: Initialize subfile header
.--
.  ** Set start value for initial read or re-reads (F5):
.  LCL.First/Next = CND.First
.--

```

# Edit the subfile load

```

: USER EXIT POINTS  Opt: X,Z=Select  V=Summary  W=Wrapper
: _ USER: Initialize program
: _ USER: Initialize subfile header <<<
: X USER: Initialize subfile record (new record) <<<
: █ CALC: Subfile control function fields
: _ USER: Validate subfile control
: _ USER: Validate subfile record fields
: _ CALC: Subfile record function fields
: _ USER: Validate subfile record relations
: _ USER: Create DBF record + <<<
    
```

```

> USER: Initialize subfile record (new record)
--
.  ** Call to EIF that returns the SF
█ Load sfl by RegN/CustName - CUSTOMER
█ ... ** Exit if unexpected return code
█ ... ** Stop loading after EOF
--
    
```

Next page

Next page

B	*Record selected	MAP	FLD	PGM	*Record selected
B	Region Name	MAP	FLD	RCD	Region Name
0	Region Id	MAP	FLD	RCD	Region Id
B	Customer Name	MAP	FLD	RCD	Customer Name
0	Customer Id	MAP	FLD	RCD	Customer Id
B	First/Next	MAP	FLD	LCL	First/Next

## Exit if unexpected return code & Stop loading after EOF

```

>  ** Exit if unexpected return code
.-CASE
ö-PGM.*Return code is *Normal
ö  ** OK
ö
ö- *OTHERWISE
ö  Exit program - return code PGM.*Return code
'-ENDCASE
  
```

```

>  ** Stop loading after EOF
.-CASE
ö-LCL.First/Next is End-of-file
ö  Stop load edtfil - User source *
ö  <-- *QUIT
'-ENDCASE
  
```

## Voilà!

```
Work with Customers
```

Sort 1	Sort 2
? Region Name	Customer Name
_ Götaland	Allers
_ Norra Norrland	Ratos
_ Södra Norrland	Smurfit Kappa
_ Svealand	SSAB
	Dalälvens el
	Dannes plåt i Älvdalen
	MLT

## Up- and Downsides

- A bit more coding than if you could have the model generate subfile functions based on requirements like this.
- A lot more flexible than the standard way of doing reads to load subfiles and record selection processing.
- Other?
- Discussion?

**Thank you.**

Magnus Gullö  
Nordes Liv & Pension

[magnus.gullo@nordea.se](mailto:magnus.gullo@nordea.se)