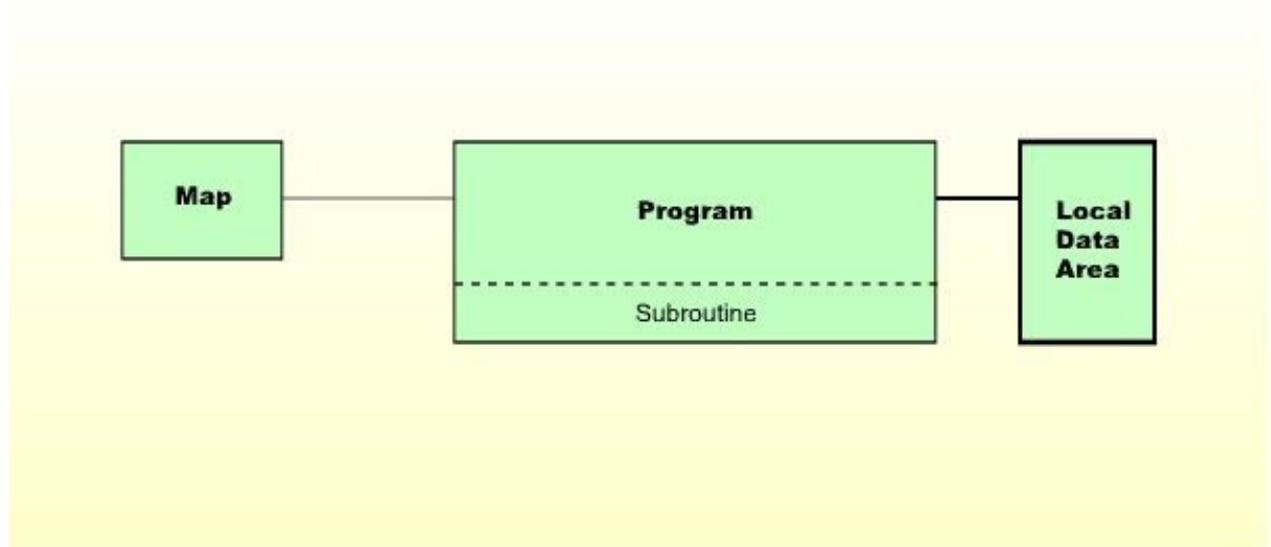


# Session 2 - Creating a Local Data Area

In Session 1, the fields used by the program were defined within the DEFINE DATA statement in the program itself. It is also possible to place the field definitions in a local data area outside the program, with the program's DEFINE DATA statement referencing that local data area by name.

In this session, the information in the DEFINE DATA statement will be relocated to a local data area outside the program. In subsequent sessions, some of this information can be used as the basis of a global data area shared by a program and an external subroutine. As you will see in Sessions 3 and 4, an important advantage of data areas is to allow a program and its external subroutine to share the same data in a single data area.

Several program editor commands will be used in this and the following sessions. For a detailed explanation, see the section Editor Commands in Program Editor in the Natural Editors documentation.



Listed below are Steps 1 to 10 of Session 2.

---

## Step 1

On the Development Functions menu, enter the code **E** and the object type **L** to create a local data area.

The data area editor will be invoked with the object type set to Local:

```

Local          Library SYSTEM          DBID  10 FNR  32
Command
I T L Name          F Leng Index/Init/EM/Name/Comment  > +
All - -----
----- S 0      L 1
    
```

In this editor, you can define the data areas to be used by Natural programs or routines. A basic outline of the fields on this screen is provided below:

Field	Explanation
I	This is an information field used by the editor to indicate the presence of a definition error (E), or to indicate other information on the variable (initial value, edit mask definition, etc.) You cannot modify information in this field.
T	This field indicates the type of the variable. For example, some fields in the data area to be created in this session will be part of a view, which will be indicated by a V in this column. Other object types include groups, data blocks, multiple-value fields, periodic groups, and constants.
L	In this field, you specify the hierarchical level of the variable (1 to 9). A variable which is not within a hierarchical structure is assigned to Level 1.
Name	In this field, you specify the name of the variable (or block or view).
F	In this field, you specify the format of the variable.
Leng	In this field, you specify the length of the variable.
Index/...	This field is used for array definition, initial value assignment, edit-mask information, originating view name (for fields derived from a view), or a comment.

## Step 2

In the Command line of the data area editor, enter the command `SPLIT P PGM01`. This puts the editor into split-screen mode with Program PGM01 appearing in the lower half of the screen and the data area (as yet blank) in the upper half.

To scroll forward in the lower half of the screen, in the Command line of the data area editor, enter the command `SPLIT +`. The command `SPLIT -` scrolls backward. Enter the command again until the `DEFINE DATA LOCAL` statement and at least three statement lines below it are displayed on the screen:

```

Local          Library SYSTEM                      DBID  10 FNR  32
Command                                             > +
I T L Name                                         F Leng Index/Init/EM/Name/Comment
All - -----
----- S 0      L 1
Program      PGM01      Library SYSTEM
0120  DEFINE DATA
0130  LOCAL
0140  01 #NAME-START          (A20)
0150  01 #NAME-END           (A20)
0160  01 #MARK                (A1)
0170  01 EMPLOYEES-VIEW VIEW OF EMPLOYEES
    
```

### Step 3

Using the program as a reference, copy the definitions of the first three variables from the DEFINE DATA statement into the data area editor screen. In the column **L**, enter the level number **1** before each variable; in the column **Name**, enter the variable names; in the column **F**, enter their formats, and in the column **Leng**, enter their lengths.

The screen should now look as follows:

```

Local          Library SYSTEM                      DBID  10 FNR  32
Command                                             > +
I T L Name                                         F Leng Index/Init/EM/Name/Comment
All - -----
      1 #NAME-START          A   20
      1 #NAME-END           A   20
      1 #MARK                A    1
----- S 3      L 1
Program      PGM01      Library SYSTEM
0120  DEFINE DATA
0130  LOCAL
0140  01 #NAME-START          (A20)
0150  01 #NAME-END           (A20)
0160  01 #MARK                (A1)
0170  01 EMPLOYEES-VIEW VIEW OF EMPLOYEES
    
```

Each of the three variable names has a hash (#) sign as the initial character. This character is used to identify user-defined variables; that is, to distinguish them from database fields.

Enter the command `CHECK` to see if everything you typed in as part of the new local data area is correct. This command ends split-screen mode and you return to the full-screen mode.

## Step 4

The local data area is not yet complete. You can read the other variables you need directly from a view into the data area editor.

In the line below the variables you have already defined, starting in the column **T**, enter the command `.V` (`EMPLOYEES`). The view `EMPLOYEES` will be displayed:

Local	Library	SYSTEM	DBID	10	FNR	32
View EMPLOYEES						
I	T	L	Name	F	Leng	Index/Init/EM/Name/Comment
-----						
	2		PERSONNEL-ID	A	8	
G	2		FULL-NAME			
	3		FIRST-NAME	A	20	
	3		MIDDLE-I	A	1	
	3		NAME	A	20	
	2		MIDDLE-NAME	A	20	
	2		MAR-STAT	A	1	
	2		SEX	A	1	
	2		BIRTH	D		
	2		NJBIRTH	I	2	
G	2		FULL-ADDRESS			
M	3		ADDRESS-LINE	A	20	(1:191) /* MU-FIELD
	3		CITY	A	20	
	3		ZIP	A	10	
	3		POST-CODE	A	10	
	3		COUNTRY	A	3	
G	2		TELEPHONE			
-----						
SYSGDA 4461: Mark fields to incorporate into data area.						

## Step 5

From this view (DDM), select the fields that you want to include in the local data area by marking them with any character in the column **I**. In this case, mark the fields `PERSONNEL-ID`, `NAME`, `DEPT` and `LEAVE-DUE`. `DEPT` and `LEAVE-DUE` are not in the first section of the view; to scroll forward within the view, keep pressing `ENTER` until the section that contains `DEPT` and `LEAVE-DUE` is displayed.

After you have marked all the fields indicated, continue pressing `ENTER` until the local data area - which now includes the fields you have selected from the `EMPLOYEES` view - is displayed again:

```

Local          Library SYSTEM          DBID 10 FNR 32
Command
I T L Name          F Leng Index/Init/EM/Name/Comment
All - -----
    1 #NAME-START          A 20
    1 #NAME-END            A 20
    1 #MARK                 A 1
    V 1 EMPLOYEES-VIEW          EMPLOYEES
      2 PERSONNEL-ID          A 8
      2 NAME                   A 20
      2 DEPT                    A 6
      2 LEAVE-DUE              N 2.0

----- S 8 L 1

SYSGDA 4462: 4 field(s) of view EMPLOYEES included.

```

## Step 6

The local data area is now complete. To check if it contains any errors, in the Command line of the editor, enter the command CHECK.

If it contains any errors, correct them and repeat the CHECK.

In the Command line, enter the command STOW LDA01. The local data area is now compiled and stored in source and object form under the name LDA01.

Note that a data area must be compiled and stored in object form before any program referencing that data area can be compiled.

## Step 7

Now Program PGM01 must be edited to reference the local data area LDA01.

In the Command line of the data area editor, enter the command EDIT PGM01. This invokes the program editor and reads Program PGM01 into the work area of the program editor.

## Step 8

Now that the variables are defined in the local data area LDA01, the DEFINE DATA statement has to be changed from actually containing the definition of variables to merely referencing the data area in which the variables are defined.

Enter the command **.D** at the beginning of each line that contains a variable definition within the DEFINE DATA statement, as shown below:

```

>
> + Program PGM01 Lib SYSTEM
....+....1....+....2....+....3....+....4....+....5....+....6....+....7..
0010 * Example Program: PGM01
0020 * Function: Demonstrate Natural
0030 * -----
0040 DEFINE DATA
0050 LOCAL
0060 .d 01 #NAME-START (A20)
0070 .d 01 #NAME-END (A20)
0080 .d 01 #MARK (A1)
0090 .d 01 EMPLOYEES-VIEW VIEW OF EMPLOYEES
0100 .d 02 PERSONNEL-ID (A8)
0110 .d 02 NAME (A20)
0120 .d 02 DEPT (A6)
0130 .d 02 LEAVE-DUE (N2)
0140 END-DEFINE

```

When you press ENTER, these lines are deleted from the program.

Then enter the following after the word LOCAL:

```
USING LDA01
```

The program should now look as follows:

### Program PGM01:

```

* Example Program 'PGM01' for Natural Tutorial
* -----
DEFINE DATA
  LOCAL USING LDA01
END-DEFINE *
REPEAT
*
INPUT USING MAP 'MAP01'
IF #NAME-START = '.'
  ESCAPE BOTTOM
END-IF
MOVE #NAME-START TO #NAME-END
*
RD1. READ EMPLOYEES-VIEW
  BY NAME
    STARTING FROM #NAME-START
    THRU #NAME-END
  IF LEAVE-DUE >= 20
    PERFORM MARK-SPECIAL-EMPLOYEES
  ELSE
    RESET #MARK
  END-IF
  DISPLAY NAME 3X DEPT 3X LEAVE-DUE 3X '>=20' #MARK
END-READ
*
  IF *COUNTER (RD1.) = 0
    REINPUT 'PLEASE TRY ANOTHER NAME'
  END-IF
*
END-REPEAT
*
DEFINE SUBROUTINE MARK-SPECIAL-EMPLOYEES

```

```

MOVE '**' TO #MARK
END-SUBROUTINE
*
END

```

## Step 9

You can enter comments into a program to document the changes you made to the program. Comments help anyone editing or maintaining a source program, and are ignored in processing.

You mark a line as a comment line by entering either an asterisk and a blank (\*) or two asterisks (\*\*) at the beginning of the line. In the rest of the line, you can then enter any comment. If you wish to append a comment to a line containing a statement, you enter the character string blank-slash-asterisk (/); anything to the right of this character string will then be considered a comment and ignored at execution.

In the second line of the program, enter the command **J(1)** to insert an empty line. In the empty line, add some comment to indicate that this program has been updated, as shown in the example below.

### Example:

```

* Example Program 'PGM01' for Natural Tutorial* PROGRAM NOW USES A LOCAL DATA AREA
* -----
DEFINE DATA
  LOCAL USING LDA01 /* This comment is for demonstration purposes only.
END-DEFINE
...

```

The subsequent sessions in this section show further examples of comments that you can add to keep track of changes you have made.

## Step 10

CHECK the program and correct any errors. RUN the program to confirm that the results are the same as when the DEFINE DATA statement did not reference a local data area. STOW the program so that it will be available for Session 3.

At the command prompt, then enter a period (.) to return to the Development Functions menu.

End of Session 2.