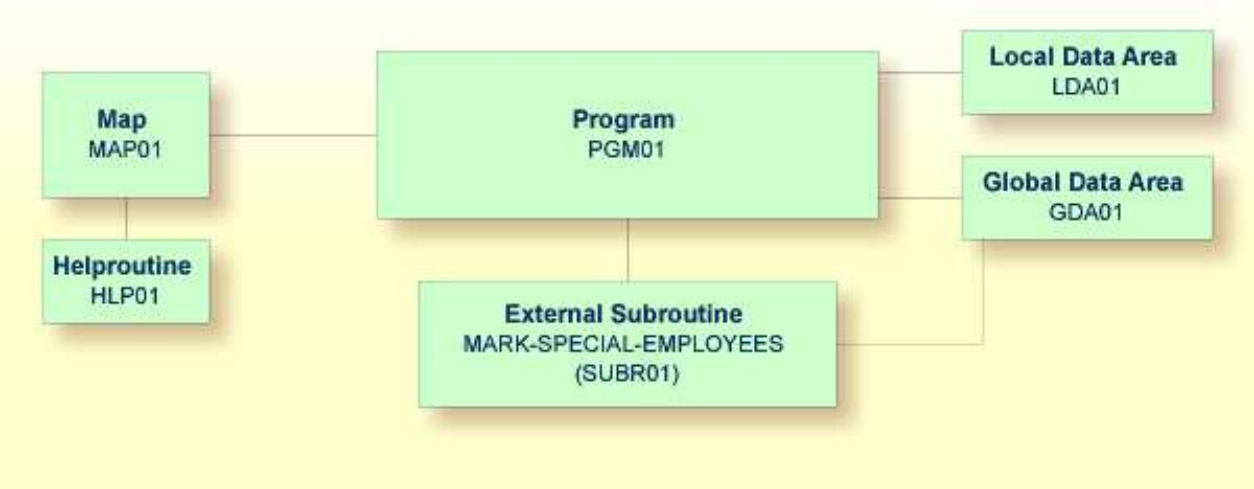
External Subroutines

Until now, the subroutine `MARK-SPECIAL-EMPLOYEES` has been defined within the program using a `DEFINE SUBROUTINE` statement. You will now define the subroutine as a separate object external to the program.

When you have completed the exercises below, your sample application will consist of the following modules:



This chapter contains the following exercises:

- Creating an External Subroutine
- Referencing the External Subroutine from Your Program

Creating an External Subroutine

You will now invoke an editor in which you will specify the code for the external subroutine.

The `DEFINE SUBROUTINE` statement of the external subroutine is coded in the same way as the inline subroutine in the program.

To create an external subroutine

1. In the library workspace, select the library which also contains your program (that is: select the **TUTORIAL** node).
2. From the context menu, choose **New Source > Subroutine**.

   An empty editor window appears.
3. Enter the following:

```
DEFINE DATA
  GLOBAL USING GDA01
  LOCAL USING LDA01
END-DEFINE
*
DEFINE SUBROUTINE MARK-SPECIAL-EMPLOYEES
  MOVE '*' TO #MARK
END-SUBROUTINE
*
END
```

4. Stow the subroutine.

   The **Stow As** dialog box appears.

5. Enter "SUBR01" as the name of the external subroutine and choose the **OK** button.

   In the library workspace, the new external subroutine is shown in the **Subroutines** node. In logical view, the name of the subroutine is shown as defined in the code: `MARK-SPECIAL-EMPLOYEES`. In all other views, the name `SUBR01` is shown.

6. Close the editor window in which you have entered the external subroutine.

Referencing the External Subroutine from Your Program

The `PERFORM` statement invokes both internal and external subroutines. When an internal subroutine is not found in the program, Natural automatically tries to perform an external subroutine with the same name. Note that Natural looks for the name that has been defined in the subroutine code (that is: the subroutine name), not for the name that you have specified when saving the subroutine (that is: the Natural object name).

Now that you have defined an external subroutine, you have to remove the inline subroutine (which has the same name as the external subroutine) from your program.

To use the external subroutine in your program

1. Return to the program editor.
2. Remove the following lines:

   ```
   DEFINE SUBROUTINE MARK-SPECIAL-EMPLOYEES
     MOVE '*' TO #MARK
   END-SUBROUTINE
   ```

   Your program should now look as follows:

   ```
   DEFINE DATA
     GLOBAL USING GDA01
     LOCAL USING LDA01
   END-DEFINE
   *
   RP1. REPEAT
   *
     INPUT USING MAP 'MAP01'
   *
     IF #NAME-START = '.' THEN
       ESCAPE BOTTOM (RP1.)
   ```

```
  END-IF
*
  IF #NAME-END = ' ' THEN
    MOVE #NAME-START TO #NAME-END
  END-IF
*
  RD1. READ EMPLOYEES-VIEW BY NAME
    STARTING FROM #NAME-START
    ENDING AT #NAME-END
*
    IF LEAVE-DUE >= 20 THEN
      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 THEN
    REINPUT 'No employees meet your criteria.'
  END-IF
*
END-REPEAT
*
END
```

3. Run the program.
4. Enter "JONES" as the starting name and press ENTER.

   The resulting list should still show an asterisk for each employee who has 20 days of leave and more.
5. Press ESC to close the output window.
6. Stow the program.

You can now proceed with the next exercises: *Subprograms*.