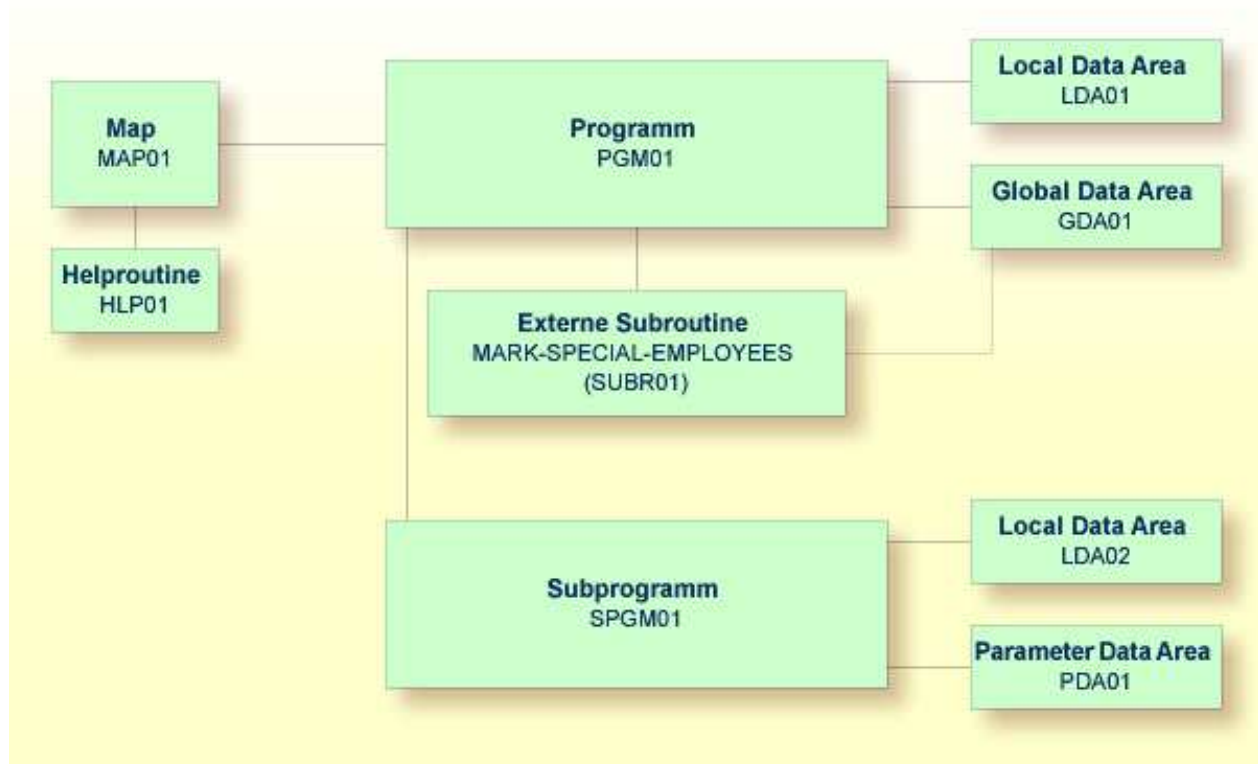


Subprogramme

Sie werden Ihr Programm jetzt durch ein CALLNAT-Statement erweitern, mit dem ein Subprogramm aufgerufen wird. In dem Subprogramm bilden die Mitarbeiter, die vom Hauptprogramm gefunden wurden, die Grundlage für eine FIND-Anfrage in der VEHICLES-Datei. Diese Datei ist ebenfalls Bestandteil der Demodatenbank. In der Ausgabe werden dann Fahrzeuginformationen aus dem Subprogramm und Mitarbeiterinformationen aus dem Hauptprogramm zu sehen sein.

Das neue Subprogramm benötigt eine weitere Local Data Area und eine Parameter Data Area.

Wenn Sie mit den Übungen in diesem Kapitel fertig sind, wird Ihre Beispielanwendung aus den folgenden Modulen bestehen:



Dieses Kapitel enthält die folgenden Übungen:

- Local Data Area ändern
- Parameter Data Area mit Hilfe einer bestehenden Local Data Area erstellen
- Eine zweite Local Data Area mit einem anderen View erstellen
- Subprogramm erstellen
- Subprogramm aus dem Programm aufrufen

Local Data Area ändern

Sie werden jetzt weitere Felder in die Local Data Area einfügen, die Sie zuvor erstellt haben. Diese Felder werden von dem Subprogramm benutzt, das Sie später erstellen werden.

▶ Weitere Felder in die Local Data Area einfügen

1. Kehren Sie zur Local Data Area zurück.
2. Markieren Sie die Zeile, die #NAME-END enthält.
3. Wählen Sie aus dem Kontextmenü den Befehl **Insert > Data Field**.

Oder:

Wählen Sie die folgende Schaltfläche in der Symbolleiste:



Das Dialogfeld **Data Field Definition** erscheint.

4. Definieren Sie die folgenden Felder:

Level	Name	Format	Länge
1	#PERS-ID	A	8
1	#MAKE	A	20
1	#MODEL	A	20

Wählen Sie die Befehlsschaltfläche **Add**, nachdem Sie ein Feld definiert haben.

5. Nachdem Sie alle Felder definiert haben, wählen Sie die Befehlsschaltfläche **Quit**.

Die Local Data Area sollte nun folgendermaßen aussehen:

T...	L...	Name	F...	L...	H...	Propert
	1	#NAME-START	A	20		
	1	#NAME-END	A	20		
	1	#PERS-ID	A	8		
	1	#MAKE	A	20		
	1	#MODEL	A	20		

6. Speichern Sie die Local Data Area mit STOW.

Parameter Data Area mit Hilfe einer bestehenden Local Data Area erstellen

Eine Parameter Data Area (PDA) wird zur Angabe der Datenparameter benutzt, die zwischen Ihrem Natural-Programm und dem Subprogramm (das Sie später noch erstellen werden) ausgetauscht werden sollen. Die Parameter Data Area wird im Subprogramm referenziert.

Mit kleinen Änderungen kann Ihre Local Data Area zur Erstellung der Parameter Data Area benutzt werden: Sie werden zwei Datenfelder in der Local Data Area löschen und die so veränderte Local Data Area als Parameter Data Area speichern. Die ursprüngliche Local Data Area wird hierdurch nicht verändert.

Anmerkung:

Es ist auch möglich, eine Parameter Data Area zu erstellen, indem Sie aus dem Menü **Object** den Befehl **New > Parameter Data Area** wählen.

▶ Parameter Data Area erstellen

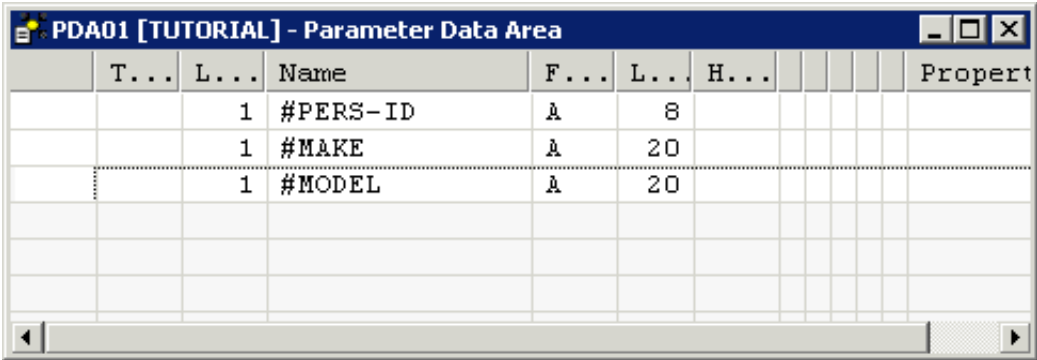
1. Löschen Sie die Felder #NAME-START und #NAME-END in der Local Data Area.
2. Wählen Sie aus dem Menü **Object** den Befehl **Save As**.

Das Dialogfeld **Save As** erscheint. Geben Sie "PDA01" als Name für die Parameter Data Area ein.

3. Achten Sie darauf, dass die Library markiert ist, die auch Ihr Programm enthält (d.h. der Knoten **TUTORIAL**).
4. Wählen Sie aus dem Dropdown-Listenfeld **Type** den Eintrag **Parameter**.
5. Wählen Sie die Befehlsschaltfläche **OK**.

Der neue Name und Typ werden nun in der Titelleiste des Editorfensters angezeigt. Im Library-Workspace wird die neue Parameter Data Area im Knoten **Parameter Data Areas** angezeigt.

Die Parameter Data Area sollte nun folgendermaßen aussehen:



T...	L...	Name	F...	L...	H...	Propert
	1	#PERS-ID	A	8		
	1	#MAKE	A	20		
	1	#MODEL	A	20		

6. Speichern Sie die Parameter Data Area mit STOW.

7. Schließen Sie das Editorfenster für die Parameter Data Area.

Eine zweite Local Data Area mit einem anderen View erstellen

Sie werden jetzt eine zweite Local Data Area erstellen und Felder aus dem DDM für die Datenbankdatei VEHICLES importieren. Dieses DDM steht auch in der System-Library SYSEXDDM zur Verfügung.

Diese Local Data Area wird dann in Ihrem Subprogramm referenziert.

▶ Local Data Area erstellen

1. Markieren Sie im Library-Workspace die Library, die auch Ihr Programm enthält (d.h. markieren Sie den Knoten **TUTORIAL**).
2. Wählen Sie aus dem Kontextmenü den Befehl **New Source > Local Data Area**.

Oder:

Wählen Sie die folgende Schaltfläche in der Symbolleiste:



Ein Editorfenster erscheint.

3. Wählen Sie aus dem Menü **Insert** den Befehl **Import**.

Das Dialogfeld **Import Data Field** erscheint.

4. Wählen Sie aus dem Dropdown-Listefeld **Library** den Eintrag **SYSEXDDM**.

Markieren Sie das Optionsfeld **DDM**.

Markieren Sie im Listefeld **Object** das Beispiel-DDM mit dem Namen **VEHICLES**.

5. Drücken Sie STRG und markieren Sie die folgenden importierbaren Felder:

```
PERSONNEL-ID  
CAR-DETAILS  
MAKE  
MODEL
```

6. Wählen Sie die Befehlsschaltfläche **Import**.

Das Dialogfeld **View Definition** erscheint.

7. Geben Sie "VEHICLES-VIEW" als Name für den View an.

8. Wählen Sie die Befehlsschaltfläche **OK**.

9. Wählen Sie die Befehlsschaltfläche **Quit**, um das Dialogfeld **Import Data Field** zu schließen.

10. Speichern Sie die neue Local Data Area mit STOW.
11. Wenn Sie dazu aufgefordert werden, einen Namen für die Local Data Area einzugeben, geben Sie "LDA02" ein und wählen Sie die Befehlsschaltfläche **OK**.

Die Local Data Area sollte nun folgendermaßen aussehen:

LDA02 [TUTORIAL] - Local Data Area						
	Type	Level	Name	Format	Length	Handl
	V	1	VEHICLES-VIEW			
		2	PERSONNEL-ID	A	8	
	G	2	CAR-DETAILS			
		3	MAKE	A	20	
		3	MODEL	A	20	

12. Schließen Sie das Editorfenster für die Local Data Area.

Subprogramm erstellen

Sie werden jetzt ein Subprogramm erstellen, das mit Hilfe einer Parameter Data Area und einer Local Data Area Informationen aus der VEHICLES-Datei abrufen. Das Programm PGM01 übergibt das Personalkennzeichen (PERSONNEL-ID) an das Subprogramm. Das Subprogramm benutzt dieses Kennzeichen für die Suche in der VEHICLES-Datei.

▶ Subprogramm erstellen

1. Markieren Sie im Library-Workspace die Library, die auch Ihr Programm enthält (d.h. markieren Sie den Knoten **TUTORIAL**).
2. Wählen Sie aus dem Kontextmenü den Befehl **New Source > Subprogram**.

Ein Editorfenster erscheint.

3. Geben Sie Folgendes ein:

```

DEFINE DATA
  PARAMETER USING PDA01
  LOCAL USING LDA02
END-DEFINE
*
FD1. FIND (1) VEHICLES-VIEW
  WITH PERSONNEL-ID = #PERS-ID
  MOVE MAKE (FD1.) TO #MAKE
  MOVE MODEL (FD1.) TO #MODEL
  ESCAPE BOTTOM
END-FIND
*
END

```

Dieses Subprogramm gibt die folgenden Informationen an ein bestimmtes Personalkennzeichen zurück: die Marke und das Modell des Firmenfahrzeuges eines Mitarbeiters.

Auf Grund des Suchkriteriums #PERS-ID wählt das FIND-Statement eine Reihe von Datensätzen (hier: einen Datensatz) aus der Datenbank aus.

Das Feld #PERS-ID des Subprogramms erhält den Wert von PERSONNEL-ID, der vom Programm PGM01 übergeben wurde. Das Subprogramm benutzt diesen Wert für die Suche in der VEHICLES-Datei.

4. Speichern Sie das Subprogramm mit STOW.
5. Wenn Sie dazu aufgefordert werden, einen Namen für das Subprogramm einzugeben, geben Sie "SPGM01" ein und wählen Sie die Befehlsschaltfläche **OK**.
6. Schließen Sie das Editorfenster für das Subprogramm.

Subprogramm aus dem Programm aufrufen

Ein Subprogramm wird mit einem CALLNAT-Statement aus dem Hauptprogramm aufgerufen. Ein Subprogramm kann nur mit einem CALLNAT-Statement aufgerufen werden; es kann selbst nicht ausgeführt werden. Ein Subprogramm kann nicht auf die Global Data Area zugreifen, die von dem aufrufenden Objekt benutzt wird.

Die Daten werden vom Hauptprogramm an das Subprogramm mit Hilfe von Parametern übergeben, die im Subprogramm mit einem DEFINE DATA PARAMETER-Statement referenziert werden.

Die Variablen, die in der Parameter Data Area des Subprogramms definiert sind, müssen nicht unbedingt dieselben Namen haben wie die Variablen im CALLNAT-Statement. Sie müssen nur in der Reihenfolge, im Format und in der Länge übereinstimmen.

Sie werden jetzt Ihr Hauptprogramm ändern, damit es das eben von Ihnen erstellte Subprogramm benutzen kann.

Subprogramm in Ihrem Hauptprogramm benutzen

1. Kehren Sie zum Programmeditor zurück.
2. Geben Sie Folgendes direkt über dem DISPLAY-Statement ein:

```
RESET #MAKE #MODEL
CALLNAT 'SPGM01' PERSONNEL-ID #MAKE #MODEL
```

Das RESET-Statement setzt die Werte für #MAKE und #MODEL auf Nullwerte.

3. Löschen Sie die Zeile, die das DISPLAY-Statement enthält, und ersetzen Sie sie mit Folgendem:

```

WRITE TITLE
  / '*** PERSONS WITH 20 OR MORE DAYS LEAVE DUE ***'
  / '*** ARE MARKED WITH AN ASTERISK ***' //
*
DISPLAY 1X '//NAME' NAME
        1X '//DEPT' DEPT
        1X '//LV/DUE' LEAVE-DUE
        ' ' #MARK
        1X '//MAKE' #MAKE
        1X '//MODEL' #MODEL

```

Der mit dem WRITE TITLE-Statement definierte Text wird bei der Ausgabe oben auf jeder Seite angezeigt. Das WRITE TITLE-Statement setzt den Standardseitentitel außer Kraft: die Informationen, die bisher oben auf jeder Seite angezeigt wurden (Seitenzahl, Datum und Uhrzeit) werden nicht mehr angezeigt. Jeder Schrägstrich (/) sorgt dafür, dass die nachfolgenden Informationen in einer neuen Zeile angezeigt werden.

Da das Subprogramm jetzt zusätzliche Fahrzeuginformationen ausgibt, muss die Größe der Spalten in der Ausgabe angepasst werden. Die Spalten erhalten kürzere Überschriften. Die Spalte, in der der Stern angezeigt wird (#MARK), bekommt überhaupt keine Überschrift. Zwischen den einzelnen Spalten wird je ein Leerzeichen eingefügt (1X). Jeder Schrägstrich in einer Spaltenüberschrift lässt die nachfolgenden Informationen in derselben Spalte in einer neuen Zeile erscheinen.

Ihr Programm sollte nun folgendermaßen aussehen:

```

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
*
  RESET #MAKE #MODEL
  CALLNAT 'SPGM01' PERSONNEL-ID #MAKE #MODEL
*
  WRITE TITLE
    / '*** PERSONS WITH 20 OR MORE DAYS LEAVE DUE ***'
    / '*** ARE MARKED WITH AN ASTERISK ***' //
*
  DISPLAY 1X '//NAME' NAME

```

```

                1X '//DEPT'   DEPT
                1X '//LV/DUE' LEAVE-DUE
                ' '          #MARK
                1X '//MAKE'   #MAKE
                1X '//MODEL'  #MODEL
*
END-READ
*
IF *COUNTER (RD1.) = 0 THEN
  REINPUT 'No employees meet your criteria.'
END-IF
*
END-REPEAT
*
END

```

4. Führen Sie das Programm mit RUN aus.
5. Geben Sie "JONES" als Startname ein und drücken Sie EINGABE.

Die daraufhin erscheinende Liste sollte der folgenden Ausgabe ähneln:

```

*** PERSONS WITH 20 OR MORE DAYS LEAVE DUE ***
*** ARE MARKED WITH AN ASTERISK ***

```

N A M E	DEPT	LV DUE	MAKE	MODEL
JONES	SALE30	25 *	CHRYSLER	IMPERIAL
JONES	MGMT10	34 *	CHRYSLER	PLYMOUTH
JONES	TECH10	11	GENERAL MOTORS	CHEVROLET
JONES	MGMT10	18	FORD	ESCORT
JONES	TECH10	21 *	GENERAL MOTORS	BUICK
JONES	SALE00	30 *	GENERAL MOTORS	PONTIAC
JONES	SALE20	14	GENERAL MOTORS	OLDSMOBILE
JONES	COMP12	26 *	DATSUN	SUNNY
JONES	TECH02	25 *	FORD	ESCORT 1.3

6. Drücken Sie ESC, um das Ausgabefenster zu schließen.
7. Speichern Sie das Programm mit STOW.

Sie haben das Tutorial jetzt erfolgreich abgeschlossen.