

View Definition

The *view-definition* option used with `DEFINE DATA LOCAL` and `DEFINE DATA OBJECT` has the following syntax:

| | | | | | | | | | | | | | | | |
|--|-----------------|---|--------------|---|---------------------|---|---|--|---|---------------|---|---|------------------------------|-----|----------------|
| <i>view-name</i> VIEW [OF] | <i>ddm-name</i> | [| <i>level</i> | { | <i>ddm-field</i> | [| { | (({ <i>format-length</i> })[/ <i>array-definition</i>]) | [| <i>emhdpm</i> |] |] |] | ... | |
| | | | | | | | | { | { | { | A | } | [/ <i>array-definition</i>] | } | DYNAMIC |
| | | | | | | | | } | } | } | | } | } | } | |
| | | | | | <i>redefinition</i> | | | | | | | | | | |

This chapter covers the following topics:

- Function
- Syntax Description

For an explanation of the symbols used in the syntax diagram, see *Syntax Symbols*.

Function

A *view-definition* is used to define a data view as derived from a data definition module (DDM).

Note:

In a parameter data area, *view-definition* is not permitted.

For further information, see the section *Accessing Data in an Adabas Database* in the *Programming Guide* and particularly the following topics:

- *Data Definition Modules - DDMs*
- *Database Arrays*
- *DEFINE DATA Views*

Syntax Description

| | |
|--|---|
| <i>view-name</i> | The name to be assigned to the view. Rules for Natural variable names apply; see <i>Naming Conventions for User-Defined Variables</i> in the <i>Using Natural</i> documentation.. |
| VIEW [OF] <i>ddm-name</i> | The name of the DDM from which the view is to be taken. |

| | |
|--------------------------------|---|
| <i>level</i> | <p>Level number is a 1- or 2-digit number in the range from 01 to 99 (the leading zero is optional) used in conjunction with field grouping. Fields assigned a level number of 02 or greater are considered to be a part of the immediately preceding group which has been assigned a lower level number.</p> <p>The definition of a group enables reference to a series of fields (may also be only one field) by using the group name. With certain statements (<i>CALL</i>, <i>CALLNAT</i>, <i>RESET</i>, <i>WRITE</i>, etc.), you may specify the group name as a shortcut to reference the fields contained in the group.</p> <p>A group may consist of other groups. When assigning the level numbers for a group, no level numbers may be skipped.</p> |
| <i>ddm-field</i> | <p>The name of a field to be taken from the DDM.</p> <p>When you define a view for a <i>HISTOGRAM</i> statement, the view must contain only the descriptor for which <i>HISTOGRAM</i> is to be executed.</p> |
| <i>redefinition</i> | <p>A <i>redefinition</i> may be used to redefine a group, a view, a DDM field or a single field/variable (that is a scalar or an array). See section <i>Redefinition</i>.</p> |
| <i>format-length</i> | <p>Format and length of the field. If omitted these are taken from the DDM.</p> <p>In structured mode, the definition of format and length (if supplied) must be the same as those in the DDM.</p> <p>In reporting mode, the definition of format and length (if supplied) must be type-compatible with those in the DDM.</p> |
| A, U or B | <p>Data type: alphanumeric (A), Unicode (U) or binary (B) for dynamic variables.</p> <p>Notes:</p> <ol style="list-style-type: none"> 1. For Adabas on mainframes, format U is available for LA fields (length <= 16381 bytes), but not for LB fields (length: <= 1 GB). 2. Format B is not available with Adabas. |
| <i>array-definition</i> | <p>Depending on the programming mode used, arrays (periodic-group fields, multiple-value fields) may have to contain information about their occurrences. See the section <i>Array Definition in a View</i> below.</p> |
| <i>emhdpm</i> | <p>With this option, additional parameters to be in effect for a field/variable may be defined. See <i>EM, HD, PM Parameters for Field/Variable</i>.</p> |
| DYNAMIC | <p>Defines a view field as <i>DYNAMIC</i>. For more information on processing dynamic variables, see the section <i>Using Dynamic and Large Variables</i>.</p> |

Array Definition in a View

Depending on the programming mode used, arrays (periodic-group fields, multiple-value fields) may have to contain information about their occurrences.

- Structured Mode
- Reporting Mode

Structured Mode

If a field is used in a view that represents an array, the following applies:

- An index value must be specified for MU/PE fields
- When no format/length specification is supplied, the values are taken from the DDM.
- When a format/length specification is supplied, it must be the same as in the DDM.

Database-Specific Considerations in Structured Mode:

| | |
|----------------|---|
| Adabas: | If MU/PE fields (defined in a DDM) are to be used inside a view, these fields must include an array index specification. For an MU field or ordinary PE field, you specify a one-dimensional index range, e.g. (1:10). For an MU field inside a PE group, you specify a two-dimensional index range, e.g. (1:10,1:5). |
|----------------|---|

Examples for Structured Mode:

```

DEFINE DATA LOCAL
1 EMP1 VIEW OF EMPLOYEES
  2 NAME(A20)
  2 ADDRESS-LINE(A20 / 1:2)

1 EMP2 VIEW OF EMPLOYEES
  2 NAME
  2 ADDRESS-LINE(1:2)

1 EMP3 VIEW OF EMPLOYEES
  2 NAME
  2 ADDRESS-LINE(2)

1 #K (I4)
1 EMP4 VIEW OF EMPLOYEES
  2 NAME
  2 ADDRESS-LINE(#K:#K+1)
END-DEFINE
END

```

Reporting Mode

In this mode, the same rules are valid as for structured mode. However, there are two exceptions:

- An index value needs not be supplied. In this case, the index range for the missing dimensions is set to (1:1).
- The format/length specification may differ from the specification in the DDM. Then the definition of format and length must be type-compatible with those in the DDM.

Examples:

```
DEFINE DATA LOCAL
1 EMP1 VIEW OF EMPLOYEES
  2 NAME(A30)
  2 ADDRESS-LINE(A35 / 5:10)

1 EMP2 VIEW OF EMPLOYEES
  2 NAME
  2 ADDRESS-LINE(A40)          /* ADDRESS LINE (1:1) IS ASSUMED

1 EMP3 VIEW OF EMPLOYEES
  2 NAME
  2 ADDRESS-LINE              /* ADDRESS LINE (1:1) IS ASSUMED

1 #K (I4)
1 EMP4 VIEW OF EMPLOYEES
  2 NAME
  2 ADDRESS-LINE(#K:#K+1)
END-DEFINE
END
```