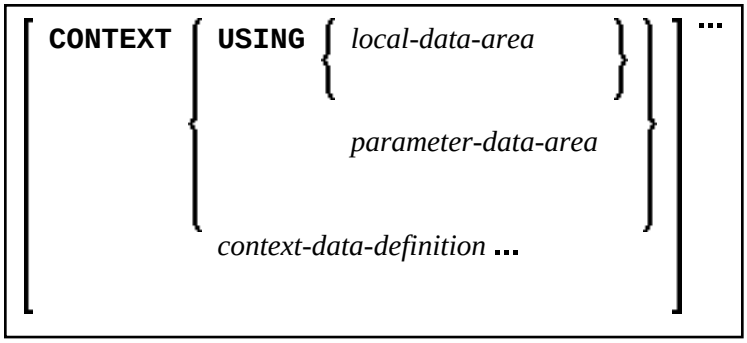
Defining Context Variables for Natural RPC

General syntax of `DEFINE DATA CONTEXT`:



This chapter covers the following topics:

- Function
- Restrictions
- Syntax Description

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

Belongs to Function Group: *Natural Remote Procedure Call*

Function

The `DEFINE DATA CONTEXT` statement is used in conjunction with the Natural Remote Procedure Call (RPC). It is used to define variables known as context variables, which are meant to be available to multiple remote subprograms within one conversation, without having to explicitly pass the variables as parameters with the corresponding `CALLNAT` statements.

A context variable is referenced by its name, and its content is shared by all programming objects executed in one conversation that refer to that name. The variable is allocated by the first executed programming object that contains the definition of the variable and is deallocated when the conversation ends.

Context variables can also be used in a non-conversational `CALLNAT`. In this case, the context variables only exist during a single invocation of this `CALLNAT` but the variables can be shared with all its callees.

A context variable is not shared with subprograms that are called within the conversation. If such a subprogram or one of its callees references a context variable, a separate storage area is allocated for this variable.

The optional `INIT` clause is evaluated in each executed programming object that contains this clause (not only in the programming object that allocates the variable). This is different to the way the `INIT` works for global variables.

For further information, see *Defining a Conversation Context* in the *Natural Remote Procedure Call (RPC)* documentation.

Restrictions

A context variable must be defined at Level 01. Other levels are only used in a redefinition.

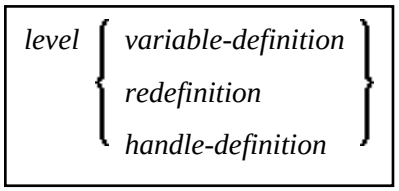
Syntax Description

USING *local-data-area*	A local data area (LDA) contains data elements which are to be used in a single Natural module. You may reference more than one data area; in that case you have to repeat the reserved words `CONTEXT` and `USING`, for example:
	
	`DEFINE DATA`
	`  CONTEXT USING DATX_L`
	`  CONTEXT USING DATX_P`
	
	`  ...`
	`END-DEFINE ;`
	
	For further information, see also *Defining Fields in a Separate Data Area* in the *Programming Guide*.

USING *parameter-data-area*	A parameter data area contains data elements which are used as parameters in a subprogram, external subroutine or dialog.
context-data-definition	Context data can be defined directly within a program or routine. For direct data definition, the syntax shown below applies.
END-DEFINE	The Natural reserved word `END-DEFINE` must be used to end the `DEFINE DATA` statement.

Context Data Definition

Context data can be defined directly within a program or routine. For direct data definition, the following syntax applies:



For further information, see *Defining Fields within a DEFINE DATA Statement* in the *Programming Guide*.

level	An application-independent variable must be defined at Level 01. Other levels are only used in a redefinition.
variable-definition	A *variable-definition* is used to define a single field/variable that may be single-valued (scalar) or multi-valued (array). See *Variable Definition*. **Note:** The `CONSTANT` clause must not be used in this context
redefinition	A *redefinition* 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 *Redefinition*.
handle-definition	A handle identifies a dialog element in code and is stored in handle variables. See *Handle Definition*.

Note:
The fields resulting from the redefinition are not considered a context variable. These fields are treated as local variables.