ADDPLOG: Dynamically Add PLOG Data Sets

The ADDPLOG function allows you to dynamically add a new protection log (PLOG) data set without terminating your current nucleus session. Using this utility function, you can specify up to eight PLOG data sets. This will reduce the chances of a wait condition in the nucleus, when the nucleus waits for an available PLOG. You might find this particularly useful during busier times of the month or year.

To add a PLOG data set dynamically, the nucleus must know about its JCL at startup time. We recommend that you set up your Adabas nucleus startup jobs to include definition statements for the maximum number of PLOG data sets as you plan to use, but limit the actual usage of the PLOGs using the ADARUN NPLOG parameter. For example, you might start a nucleus with eight PLOG definitions in the Adabas startup JCL, but limit the number of PLOGs actually used during nucleus processing to three PLOGs by setting the NPLOG parameter to "3". When the nucleus starts up, only three PLOGs will be opened and logged in the PPT, even though eight are defined in the JCL. The additional PLOG data sets can then be dynamically added using this ADADBS ADDPLOG utility or its equivalent function in the Adabas Online System (AOS).

Note:
Any PLOG data sets you add dynamically will not be retained once you recycle your Adabas nucleus. To retain these new PLOG data sets when Adabas is stopped and restarted, alter the Adabas startup JCL as well, ensuring that the number of PLOG definition statements in the JCL matches the increased number of PLOG data sets and that the NPLOG ADARUN parameter setting includes the new PLOG data sets.

Running the ADADBS ADDPLOG utility function is invalid when Adabas is running with dual PLOGs.

```
ADADBS  ADDPLOG  NUMBER = plog-ds-number
                 [NOUSERABEND]
                 [NUCID = nucid ]
                 [PLOGDEV = device-type ]
                 [TEST]
```

This chapter describes the syntax, processing, and parameters of the ADADBS ADDPLOG function.

- Essential Parameters
- Optional Parameters
- Examples

Essential Parameters

NUMBER: PLOG Data Set Number

Use the NUMBER parameter to specify the number of the nonsequential PLOG data set to be added. Valid values are integers ranging from "2" through "8" (inclusive).

Note:
Be sure that the Adabas startup JCL allows for this additional PLOG data set by including a definition statement for the data set. If a definition statement is *not* already specified for this PLOG data set in the Adabas startup JCL, you will need to add it now and recycle the nucleus. Ideally, you would already have included definition statements in the JCL for all potential PLOG data sets, even though they are not all in use when the nucleus starts up.

NUCID: Cluster Nucleus ID

This parameter is required only in cluster environments.

Use the NUCID parameter to specify the nucleus ID of the Adabas within the cluster to which the new PLOG data set should be dynamically added.

Optional Parameters

PLOGDEV

Use the optional PLOGDEV parameter to specify the device type to be used for the new PLOG data set. This parameter is required only if a different device type from the device type specified by the ADARUN DEVICE parameter is to be used. The default is to use the device type specified by the ADARUN DEVICE parameter.

NOUSERABEND: Termination without Abend

When a parameter error or a functional error occurs while this utility function is running, the utility ordinarily prints an error message and terminates with user abend 34 (with a dump) or user abend 35 (without a dump). If NOUSERABEND is specified, the utility will *not* abend after printing the error message. Instead, the message "*utility* TERMINATED DUE TO ERROR CONDITION" is displayed and the utility terminates with condition code 20.

Note:
When NOUSERABEND is specified, we recommend that it be specified as the first parameter of the utility function (before all other parameters). This is necessary to ensure that its parameter error processing occurs properly.

TEST: Test Syntax

The TEST parameter tests the operation syntax without actually performing the operation. Note that the validity of values and variables *cannot* be tested: only the syntax of the specified parameters can be tested. See Syntax Checking with the TEST Parameter for more information about using the TEST parameter in ADADBS functions.

Examples

In the following example, PLOG data set 3 is dynamically added using a 3390 device.

```
ADADBS ADDPLOG NUMBER=3,CLOGDEV=3390
```

In the following example, PLOG data set 6 is dynamically added for the Adabas nucleus 65590 in a cluster environment.

```
ADADBS ADDPLOG NUMBER=6,NUCID=65590
```