

Natural Parameter Modification for DB2

This section covers the following topics:

- Natural Profile Parameter Settings
- Performance Considerations for the DB2SIZE Parameter

Natural Profile Parameter Settings

▶ To set the Natural profile parameter

1. Add the following Natural profile parameter to your NATPARM module:

DB2SIZE=nn

The DB2SIZE parameter can also be specified dynamically. It indicates the size of the DB2 buffer area, which must be set to at least 6 KB.

The setting of DB2SIZE also depends on whether you use the file server or not. If the file server is **not** used, the setting can be calculated according to the following formula:

$((808 + n1 * 40 + n2 * 100) + 1023) / 1024$ KB

If the file server is used, the setting can be calculated according to the following formula:

$((904 + n1 * 40 + n2 * 140 + n3 * 8) + 1023) / 1024$ KB

The variables *n1*, *n2* and *n3* correspond to:

<i>n1</i>	the number of statements for dynamic access as specified as the second parameter in Job I055, Step 1600;
<i>n2</i>	the maximum number of nested database loops as specified with the MAXLOOP parameter in NDBPARAM;
<i>n3</i>	the maximum number of file server blocks to be allocated per user specified as the fifth parameter in Job I075, Step 1620 or the EBPMAX parameter of NDBPARAM, if you decided to use the Software AG Editor buffer pool as file server.

Important:

Ensure that you have also added the Natural parameters required for the Software AG Editor; see the relevant installation description in the section Installing the Software AG Editor, in the Natural Installation Guide for Mainframes).

As DB2SIZE applies to Natural for DB2 and Natural for SQL/DS, it must be set to the maximum value if you run more than one of these environments.

2. Add an NTDB entry specifying the list of logical database numbers that relate to DB2 tables. All Natural DDMs that refer to a DB2 table must be cataloged with a DBID from this list.

DBIDs can be any number from 1 to 254; a maximum of 254 entries can be specified. For most user environments, one entry is sufficient.

Important:

Ensure that all DB2 DDMs used when cataloging a given program have a valid DB2 DBID. Also ensure that the DBIDs selected in the NTDB macro for DB2 do not conflict with DBIDs selected for other database systems. The DBID for SQL/DS used when cataloging a Natural program does not have to be in the NTDB list of DBIDs used when executing this program. Therefore, when executing existing Natural programs, DBID 250 is not mandatory.

Two sample NTDB macros follow:

NTDB DB2,250

NTDB DB2,(200,250,251)

Performance Considerations for the DB2SIZE Parameter

During execution of an SQL statement, storage is allocated dynamically to build the SQLDA for passing the host variables to DB2.

In previous Natural for DB2 versions, this storage was always obtained from the TP monitor or operating system. For performance reasons, it is now first attempted to meet the storage requirements by free space in the Natural for DB2 buffer (DB2SIZE). Only if there is not enough space available in this buffer, the TP monitor or operating system is invoked.

To take advantage of this performance enhancement, you must specify your DB2SIZE larger than calculated according to the formula (see Natural Profile Parameter Settings above).

Depending on the SQL execution mode and on the usage of the Natural file server, the additional storage requirements (in bytes) can be calculated as follows:

- Dynamic Mode
- Static Mode
- Storage Requirements for the File Server
- Sample Calculation for Dynamic Mode without Using the File Server
- Considerations for VARCHAR Fields

Dynamic Mode

- With sending fields:

$$64 + n * 56$$

With sending fields including LOB columns:

$$64 + 2 * n * 56$$

where n is the number of sending fields in an SQL statement.

The storage is freed immediately after the execution of the SQL statement.

- With receiving fields (that is, with variables of the INTO list of a SELECT statement):

$$64 + n * 56 + 24 + n * 2$$

With receiving fields including LOB columns:

$$64 + 2 * n * 56 + 24 + n * 2$$

where n is the number of receiving fields in an SQL statement.

The storage remains allocated until the loop is terminated.

Static Mode

- With sending fields:

$$64 + n * 24$$

With sending fields including LOB columns:

$$64 + 2 * n * 56$$

where n is the number of sending fields in an SQL statement.

The storage is freed immediately after the execution of the SQL statement.

- With receiving fields (that is, with variables of the INTO list of a SELECT statement):

$$64 + n * 24 + 24 + n * 2$$

With receiving fields including LOB columns:

$$64 + 2 * n * 56 + 24 + n * 2$$

where n is the number of receiving fields in an SQL statement.

The storage remains allocated until the loop is terminated.

Storage Requirements for the File Server

When using the file server, additional storage is required for each database loop that contains Positioned UPDATE and/or DELETE statements.

For each of such loops, a buffer is allocated to save the contents of all receiving fields contained in the INTO list. Therefore, the size of this buffer corresponds to the total length of all receiving fields:

$$20 + 4 + \text{sum} (\text{length} (v1), \dots, \text{length} (vn))$$

where "v1 ... vn" refers to the variables contained in the INTO list.

The buffer remains allocated until the loop is terminated.

Sample Calculation for Dynamic Mode without Using the File Server

If you use the default value 10 for both variables ($n1$ and $n2$), the calculated DB2SIZE will be 2208 bytes. However, if you specify a DB2SIZE of 20 KB instead, the available space for dynamically allocated storage will be 18272 bytes, which means enough space for up to either 325 sending fields or 313 receiving fields.

Since space for receiving fields remains allocated until a database loop is terminated, the number of fields that can be used inside such a loop is reduced accordingly: for example, if you retrieve 200 fields, you can update about 110 fields inside the loop.

Considerations for VARCHAR Fields

When using VARCHAR fields (that is, fields with either an accompanying L@ field in the Natural view or an explicit LINDICATOR clause), additional storage is allocated dynamically if the L@ or LINDICATOR field is not specified directly in front of the corresponding base field. Therefore, always specify these fields in front of their base fields.