Natural Batch Utilities

This section covers the following topics:

- Transfer of NDBs/NSBs/UDFs from one System File to Another
- Utility NDUDFGEN for Natural Data Areas

Transfer of NDBs/NSBs/UDFs from one System File to Another

- Unloading the NDBs, NSBs and UDFs
- Loading NDBs, NSBs and UDFs
- Selecting NDBs, NSBs and UDFs from a Dataset

The transfer of NDBs, NSBs and UDFs from one `FDIC` system file to another is performed either online using the utility `SYSMAIN` (as described in the Natural *Utilities* documentation) or in two batch steps, using two Natural batch utilities provided for this purpose:

- With the `ULDDLI` unload utility, the NDBs, NSBs and UDFs are transferred from one `FDIC` system file to a sequential work file.
- With the `INPLDLI` load utility, the NDBs, NSBs and UDFs are transferred from the sequential work file to another `FDIC` system file.

Both programs, `ULDDLI` and `INPLDLI`, are contained in the library `SYSDDM`.

Unloading the NDBs, NSBs and UDFs

The utility `ULDDLI` is used to unload NDBs, NSBs and UDFs from an `FDIC` system file to a sequential work file.

`ULDDLI` requires the following input:

- the specification of the `FDIC` system file to be unloaded (either in the `NATPARM` module or dynamically) and
- one or more parameter lines containing the following:
    - **Function code** (A1); the following function codes can be specified:

A	All NSBs, NDBs and UDFs are unloaded.
D	All NDBs with valid object names and their UDFs are unloaded. If no object names are specified, all NDBs and their UDFs are unloaded.
P	All NSBs with valid object names are unloaded. If no object names are specified, all NSBs are unloaded.
U	All UDFs with valid object names are unloaded. If no object names are specified, all UDFs are unloaded.
. (period)	Terminate ULDDLI; at least one parameter card with function code "." is required.

- **Object name** (A8); 0 - 6 occurrences.

  Note:
  With UDFs, the object name must be in the form *nnn**nnn*; that is, a 3-digit database ID, followed by 2 asterisks, followed by a 3-digit file number.

Work files: CMWKF01 DD card must be provided with:

```
DCB=(RECFM=VB,LRECL=4624,BLKSIZE=4628)
```

When ULDDLI is executed, the specified NDBs, NSBs and UDFs are written from the FDIC system file to the CMWKF01 dataset.

Note:
DL/I fields of a segment are part of the NDB block and not of the UDF block, which means that you must still transfer the entire NDB block if you have modified a DL/I field in a segment.

Example 1 - Unload the NDBs TESTDB1 and TESTDB2:

```
LOGON SYSDDM
ULDDLI D TESTDB1 TESTDB2
  .
```

Example 2 - Unload all UDF Blocks:

```
LOGON SYSDDM
ULDDLI U
.
```

Example 3 - Unload UDF Blocks with DBID 10/FNR 150 and DBID 246/FNR 3:

```
LOGON SYSDDM
ULDDLI U 010**150 246**003
.
```

Loading NDBs, NSBs and UDFs

The utility INPLDLI is used to load NDBs, NSBs and UDFs - previously unloaded with ULDDLI - from the work file to an FDIC system file.

INPLDLI requires the following input:

- the specification of the FDIC system file into which the NDBs, NSBs and UDFs are to be loaded (either in the NATPARM module or dynamically);
- (optionally) the parameter DEL=Y:

  If you specify DEL=Y, all existing NDBs and UDFs found on the FDIC system file are first deleted. The ones contained on the input work file are added to the file. NSB definitions contained on the work file replace any identically named NSBs on the FDIC system file.

  If you do not specify DEL=Y, existing identically named NDBs and NSBs are not replaced. Existing UDFs which have been allocated identical DBID/FNR combinations are not replaced either. Non-existent definitions are added. If you do not specify DEL=Y, it may occur that an NDB is loaded but all or some of its segments (UDFs) are not, or that segments (UDFs) are loaded without the corresponding NDB being loaded.
- (optionally) the parameter REP=Y: If you specify REP=Y, NDBs, NSBs and UDFs contained on the work file replace any identically named NDBs, NSBs and UDFs on the FDIC system file.

DEL=Y and REP=Y are mutually exclusive. If neither DEL=Y nor REP=Y is specified, existing NDBs, NSBs and UDFs are neither deleted nor replaced.

Work files: CMWKF01 DD card must be assigned to the work file which was created by the utility program ULDDLI.

When INPLDLI is executed, the NDBs, NSBs and UDFs are loaded from the work file into the specified FDIC system file, depending on whether they already exist and on whether DEL=Y was specified.

Example:

```
LOGON SYSDDM
INPLDLI REP=Y
```

Selecting NDBs, NSBs and UDFs from a Dataset

The utility SELDLI allows you to select Natural for DL/I objects (NDBs, NSBs, UDFs) from a dataset created by the ULDDLI utility. The output of SELDLI can be used as input for INPLDLI. Since INPLDLI does not allow to select objects from a dataset created by ULDDLI, you can use SELDLI to perform this function on desired objects prior to running INPLDLI.

SELDLI can, therefore, be used for backup/recovery or transfer of selected objects from test to production.

SELDLI also supports a SCAN (command SCN) feature that will list all of the objects on the input dataset without selecting any for output.

SELDLI can be used in batch mode only.

SELDLI requires the following input:

- the specification of the output dataset CMWKF01 from ULDDLI

- up to 30 parameter lines containing the following:
  - Object type (A3); the following types can be specified:
    - NSB - Select specified NSB
    - NDB - Select specified NDB
    - NDU - Select specified NDB and related UDF
    - UDF - Select specified UDF
    - SCN - List input dataset `CMWKF01`
  - terminate `SELDLI`
- Object name (A8); 1 occurrence

Notes:

1. With NDB/NSB, a wildcard (*) can be specified at the end of the name to select a range of names.
2. With UDFs, the object name must be in the form *nnn**nnn*; that is, a 3-digit database ID, followed by 2 asterisks, followed by a 3-digit file number.

`SELDLI` provides the following output:

- Dataset containing selected objects to be used as input to `INPLDLI`. It is specified with DDNAME `CMWKF02`.

When `SELDLI` is executed, the specified NDBs, NSBs and UDFs are copied from `CMWKF01` to `CMWKF02`.

Example 1 - Select all NDBs:

```
LOGON SYSDDM
SELDLI
NDB,*
.
FIN
```

Example 2 - Select NSB "ORDPSB" and UDF for DBID 151, FNR 3:

```
LOGON SYSDDM
SELDLI
NSB,ORDPSB
UDF,151**003
.
FIN
```

Example 3 - Select NDB "CUSTDBD" and its related UDFs:

```
LOGON SYSDDM
SELDLI
NSB,ORDPSB
NDU,CUSTDBD
.
FIN
```

Example 4 - List all objects on the input dataset:

```
LOGON SYSDDM
SELDLI
SCN
FIN
```

Utility NDUDFGEN for Natural Data Areas

The batch utility NDUDFGEN can be used to generate Natural data areas.

Input is provided by a UDF definition read from a work file.

Two kinds of data areas can be generated:

- a Natural view,
- a data structure (local data area).

A view in a local data area is generated from all fields contained in the input work file. The utility normalizes the data to the requirements of a view according to the Natural syntax. The field lengths are adapted to Natural field lengths, multiple-value fields and periodic groups are generated from record data structures. Arrays are generated by NDUDFGEN with the maximum length allowed by Natural. Field definitions are collected into a redefinition and the redefined field is generated according to the length of the individual fields collected. The generated field can then be used in the segment description as UDF; this means that not all UDFs need to be defined in the segment description, but only the generated fields.

A data structure as local data area is generated of all input fields. A level increment value can be specified for the fields. No other modifications to the input file data are permitted, so that the data are generated as specified in the input file.

Input for NDUDFGEN

The input layout is similar to the one for the NDPBCUDF utility.

The first card is the definition card; it contains the definition which is valid for all of the UDF definitions.

The FLD cards contain the actual field definitions and are separated from each other by STR cards.

The END card indicates the end of the field definitions. The input is required in forms mode (IM=F) as follows:

Definition Card

Definition	Explanation
Bytes 1 - 3	The first 3 bytes are not used.
Bytes 4 - 11	These 8 bytes contain the DBD name.
Bytes 12 - 19	These 8 bytes contain the segment name.
Bytes 20 - 27	These 8 bytes contain a prefix (generated for fields).
Bytes 28 - 30	These 3 bytes contain the maximum occurrence (default is 191).
Byte 31	This byte contains either S if a data structure is to be generated or V if a view is to be generated.
Byte 32	This byte contains the level increment.

Field Card

Definition	Explanation
Bytes 1 - 3	The first 3 bytes contain FLD.
Bytes 4 - 19	These 16 bytes are not used.
Byte 20	This byte contains the field level.
Bytes 21 - 39	These 19 bytes contain the name of the field being defined. This must be an alphanumeric value.
Bytes 40 - 42	These 3 bytes are not used.
Byte 43	This byte contains the format of the field.
Bytes 44 - 47	These 4 bytes contain the byte length of the field.
Byte 48	This byte is not used.
Byte 49 - 52	This byte contains the length as required by Natural (if this length is specified, the byte length is ignored).
Byte 53 - 57	These 4 bytes contain the maximum size of the 1st dimension of an array.
Byte 58 - 62	These 4 bytes contain the maximum size of the 2nd dimension of an array.
Byte 63 - 66	These 4 bytes contain the maximum size of the 3rd dimension of an array.

Example 1 - View Generation:

```
    DBDNAME SEGMENT PREFIX  191V
FLD                 1VAR1               000A0745
STR
FLD                 1GROUP              000A0000N0000000200020000
STR
FLD                 2VAR2               000A0006N00060005
STR
FLD                 2VAR3               000A0030
STR
END
```

The above input generates the following view:

```
13:38:41               *****  E D I T  DATA  *****                    2006-05-25
Library: XYZ1      Name:          LOCAL                       DBID:  10 FNR:   5
Command:                                                                 > +
I T L Name                            F Leng  Index/Init/EM/Name/Comment
- - - -------------------------------- - ------------------------------------
    1 VAR1                            A  149 (5)
    1 GROUP                                  (4)
    2 VAR2                            A    6 (5)
    2 VAR3                            A   30
```

Example 2 - Structure Generation:

```
   DBDNAME SEGMENT PREFIX  191S
FLD               1VAR1              000A0745
STR
FLD               1GROUP             000A0000N0000000200020000
STR
FLD               2VAR2              000A0006N00060005
STR
FLD               2VAR3              000A0030
STR
END
```

The above input generates the following data structure:

```
13:41:20               *****  E D I T  DATA  *****                    2006-05-25
Library: XYZ1      Name:          LOCAL                       DBID:  10 FNR:   5
Command:                                                                 > +
I T L Name                            F Leng  Index/Init/EM/Name/Comment
- - - -------------------------------- - ------------------------------------
  V 1 DBDNAME-SEGMENT-VIEW                   DBDNAME-SEGMENT
  M 2 VAR1                            A  149 (5)
  P 2 GROUP                                  (4)
    3 PREFIX-1                        A   60 /*PREFIX-1
  R 3 PREFIX-1
    4 VAR2                            A    6 (5)
    4 VAR3                            A   30
```