Converting "Natural for DL/I" Programs

This chapter covers the following topics:

- Introduction
- Conversion of the Data Definitions and of the Data
- Modification of the Application

Introduction

The Natural for DL/I (NDL) interface provides access to data stored in an IMS/DB database for Natural applications. When the DL/I databases access by NDL are converted into Adabas files with the ADL, the NDL applications can continue to run like any other DL/I application. NDL converts the Natural generated internal Adabas call into an DL/I call which is converted by ADL into an Adabas call and forwarded to Adabas. On the way back, the response of Adabas is converted by ADL into a DL/I response and returned to NDL which converts it into an Adabas response for the Natural application.

NDL applications have special restrictions and enhancements caused by the hierarchical structure of the underlying database. Thus after the migration, the NDL application cannot run "as they are" against Adabas. This chapter describes what you have to respect when you convert a NDL program to a "normal" Natural/Adabas program.

After the conversion, the Natural program must run through the ADL Consistency as long as there is any DL/I application (Cobol, PL/1) accessing the same data. The restrictions for Natural programs running against the ADL Consistency are described in the section *Using ADL Files with Natural/Adabas* in this documentation. Although the Consistency is an overhead too, the conversion from Natural for DL/I to Natural/Consistency has several advantages:

- Better performance because read calls are routed directly to Adabas.
- The hierarchical view to the data is replaced by a flat view with foreign keys.
- Normal Natural (i.e. Natural/Adabas) syntax can be used. No special NDL syntax restrictions, error messages etc. must be obeyed.
- When all NDL applications are converted, NDL is no longer required.
- Natural with the Consistency is the path towards standard Natural/Adabas. I.e. as soon as no DL/I application accesses the data anymore, the ADL Consistency is obsolete, too.

Conversion of the Data Definitions and of the Data

The DBD definitions accessed by the NDL application must be converted with the ADL Control Block Conversion (CBC) utility. You can use one file per segment (GENSEG statement of the ADL CBC utility), or collect more than one segment in one file. The ADL Conversion Utility is described in the *ADL Conversion* documentation.

In a first step, the FDT is loaded without data into Adabas. Then load the field descriptions generated by ADL into Predict with the Predict Incorporate function.

Build new Predict masterfiles and DDMs from the ADL definitions and from the NDL definition.

Use from ADL

Field	Description
Zx	ADL internal fields.
Xx	Descriptors corresponding to secondary indices.
Sx	Adabas groups corresponding to DL/I segments. Use the DL/I segment name as long name.

Use from NDL

Field	Description
xx	All Adabas field definitions corresponding to parts of the DL/I segment. Take care that the short names are unique. These fields must be part of the group corresponding to the DL/I segment.

Special considerations for reference fields:

- Define the sequence fields as descriptors. Note that the root sequence field is already defined by ADL as descriptor. The short name of the root sequence field must be the one defined by ADL.
- For the foreign primary key fields, i.e. the PCK and VCK fields, use the ADL short name (normally Px/Qx), and the NDL long name. Define these fields as descriptors.
- Build superdescriptors with the source segment to reflect secondary indices. Use NDL long-names for those superdescriptors.
- Do not allocate fields with foreign keys of secondary indices for dependent segments. Because the secondary index fields belong to another segment, DL/I applications would not fill these fields.

After this modifications have been performed, generate ADACMP cards with Predict. Then unload the data from DL/I and load it into Adabas with ADL utilities. For the compression, use the ADACMP cards from Predict. The data conversion is described in the section ADL Conversion Utility in the *ADL Conversion* documentation.

Modification of the Application

This section describes what have to be considered in the NDL applications concerning database related commands when converting to Natural/Adabas and the ADL Consistency.

READ/FIND commands

`READ` and `FIND` commands are only affected when the application reads a dependent segment and uses in the root level a secondary index. In this case the root level must first be read with the secondary index to get the primary key, and this must be used as foreign key when reading the dependent.

Example

We have a root segment “A” with primary and secondary index and a dependent segment “B” with a sequence field. We would like to access segment B and use on the root level the secondary index.

Natural Name	Description
ROOT-A-VIEW	view of root segment A
SEG-B-VIEW	view of dependent segment B
A-PRIMKEY	root primary key
ROOT-PRIM	root primary key used as foreign key
A-SEC	root secondary index
ROOT-SEC	root secondary index used as foreign key
SEG-B-SEQ	dependent segment sequence field

NDL

```
FIND  SEG-B-VIEW WITH ROOT-SEC=SEC-VAL1    /* sec index
      AND SEG-B-SEQ=B-VAL2                 /* dependent
```

ADL

```
FIND  ROOT-A-VIEW WITH A-SEC=SEC-VAL1          /* sec index
MOVE  A-PRIMKEY to #PRIM-VAL1                  /* prim key
FIND  SEG-B-VIEW WITH ROOT-PRIM=#PRIM-VAL1     /* prim key
      AND SEG-B-SEQ=B-VAL2                     /* dependent
```

STORE

When performing a `STORE` command, the foreign keys must be filled. Before the command is executed, it must be verified that the referenced values exist. This is eventually already ensured by the application logic. In case of an inconsistence an error handling must be provided.

DELETE

DL/I and thus NDL too, offer a cascaded delete. When a segment occurrence is deleted, all dependent segment occurrences are deleted automatically. In Natural/Adabas there is no such function. If a record has dependents, i.e. records which refer the current sequence field as foreign keys, these records must be deleted explicitly.

Note:
As long as the ADL Consistency is active, the dependents must be deleted from bottom to top and finally the record itself can be deleted

UPDATE

UPDATE commands are not affected by the conversion.

Note:
Sequence fields and foreign keys can neither be updated with NDL nor with the ADL Consistency. You should carefully evaluate, before updating such a field with Natural/Adabas.

END TRANSACTION/BACKOUT TRANSACTION

DL/I makes an implicit checkpoint at each terminal I/O. Because Adabas does not make any implicit ET call, the transaction logic of the application must be checked and if required, additionally ’END TRANSACTION’ statements must be added.

ERROR HANDLING

There will be no more NDL specific error message in the application. The error handling must reflect only Natural/Adabas error situations. Additionally when the ADL Consistency is used, a Consistency response code can be obtained. This is described in details in the chapter *Using ADL Files with Natural/Adabas* in this documentation.