Adabas Native SQL

Installation Manual for z/OS

Version 2.4.1

September 2009

Adabas Native SQL

Table of Contents

1 Installation Manual for z/OS 1
2 GENERAL INFORMATION 3
  Installation Jobs 4
  Using System Maintenance Aid 4
3 TAPE LAYOUT 5
4 INSTALLATION 9
  Copying to an z/OS Disk 10
  Loading Files from Magnetic Tape 11
  Modifying the Sample JCL Procedures 12
  Modifying the Global Parameters 12
  Relinking Adabas Native SQL 13
  Testing Adabas Native SQL 13
  DD-Names 15
5 INSTALLATION CHECKLIST 17
6 HINTS FOR USING ADABAS NATIVE SQL 19
  Periodic Groups and Multiple Fields 20
  Group Structure of Periodic Groups 20
  Dynamic Command-IDs 20
  Locating Errors 20
  Preprocessor Copy and Generate Facilities 20
  Restriction for MU Fields Within PE 21
  Hyphens and Break Characters in PL/I Programs 21
  READ ISN Statement 21
  PL/I - Margin Settings 21
  PL/I - Structure Variables in Superdescriptors 22
  Last Statement Restriction in COBOL/II 22
  Adabas Native SQL Preprocessor Condition Codes 22
  SQL0217 Error Message When Running the Precompiler 22
  Hints for Improving Adabas Native SQL Efficiency 23

1 Installation Manual for z/OS

This document describes how to install Adabas Native SQL on z/OS.

This information is structured into the following sections:

●	*General Information*	This section provides general information about the installation procedure.
●	*Tape Layout*	This section describes the contents of the distribution medium.
●	*Installation*	This section describes how to perform the installation.
●	*Installation Checklist*	This section describes how to verify that the installation has been successful.
●	*Hints for Using Adabas Native SQL*	This section contains a list of useful hints about using Adabas Native SQL.

2 GENERAL INFORMATION

- Installation Jobs 4
- Using System Maintenance Aid 4

This chapter covers the following topics:

Installation Jobs

The installation of Software AG products is performed by installation jobs. These jobs are either adapted "manually" or generated by System Maintenance Aid (SMA).

For each step of the installation procedure described below, the job number of a job performing the respective task is indicated. This job number refers to an installation job generated by SMA.

Using System Maintenance Aid

If you are using Software AG's System Maintenance Aid (SMA) for the installation process, refer to the System Maintenance Aid manual.

3 TAPE LAYOUT

The tape contains the following three data sets. The LOAD and SRCE data sets were written by the utility program IEBCOPY; the ERRN data set was written by the utility program IEBGENER.

The notation *nnn* represents the version, release and SM level. Space requirements apply to a 3390-type DASD.

Dataset	Contents	Format	Space requirements

SQL*nnn*.LOAD	Preprocessor load modules	RECFM =U
		BLKSIZE =6447
		DSORG =PO
			approx. 4 cylinders

SQL*nnn*.SRCE	Preprocessor source modules and sample programs	RECFM =FB
		LRECL =80
		BLKSIZE =4000
		DSORG =PO
			approx. 1 cylinder

SQL*nnn*.ERRN	Preprocessor error messages	RECFM =VB
		LRECL =4624
		BLKSIZE =4628
		DSORG =PS
			approx. 4 tracks

The data set SQL*nnn*.LOAD contains the following members:

Name	Description
ADASQLA	The executable Adabas Native SQL preprocessor (load module) that generates Ada code.
ADASQLC	The executable Adabas Native SQL preprocessor (load module) that generates COBOL code.
ADASQLF	The executable Adabas Native SQL preprocessor (load module) that generates FORTRAN code.
ADASQLP	The executable Adabas Native SQL preprocessor (load module) that generates PL/I code.
FCID	Module that dynamically generates command-IDs during execution of the application program. Used in FORTRAN programs.

Name	Description
FINDCID	Module that dynamically generates command-IDs during execution of the application program. Used in Ada, COBOL and PL/I programs.
PRPABEND	This module is used to abend an application program if an error occurred.
PRTFLOW	Module that prints a flow-trace of all executed Adabas Native SQL statements during the execution of the application program if MODE FLOW is set (object module, COBOL).
PRTRACE	Module that traces the record buffers and format buffers during the execution of the application program if MODE TRACE is set (object module, COBOL).
RESPINT	The response code interpretation routine (object module, COBOL).
SQTODATE	Routine which converts numeric date to format D number.
SQFRDATE	Routine which converts format D number to numeric date.
SQTOTIME	Routine which converts numeric date and numeric time to format T number.
SQFRTIME	Routine which converts format T number to numeric date and numeric time.

The dataset SQL*nnn*.SRCE contains the following members:

Name	Description
ADAGO	A sample JCL procedure to preprocess, compile, link and execute an Ada program.
AEX1...AEX3	Ada examples using various Adabas Native SQL statements.
ADAPARM	Adabas runtime parameters.
CEX1...CEX3	COBOL examples using various Adabas Native SQL statements.
CEXC	A COBOL example using Adabas Native SQL under CICS.
COBOLGO	A sample JCL procedure to preprocess, compile, link and execute a COBOL program.
FEX1...FEX3	FORTRAN examples using various Adabas Native SQL statements.
FORTGO	A sample JCL procedure to preprocess, compile, link and execute a FORTRAN program.
GO	A sample JCL procedure to preprocess an Ada, COBOL, FORTRAN or PL/I program. The sub-procedure ASQL must be called with parameter SQL = "ADASQLA" (Ada), "ADASQLC" (COBOL), "ADASQLF" (FORTRAN) or "ADASQLP" (PL/I) as appropriate. Also, the global parameter `LANG` must be set to the correct value.
LINKSQL	A sample JCL procedure to link Adabas Native SQL.
PEX1...PEX3	PL/I examples using various Adabas Native SQL statements.
PLIGO	A sample JCL procedure to preprocess, compile, link and execute a PL/I program.
PRTFLO	The source code of the routine that prints a flow-trace of all executed Adabas Native SQL statements during the execution of the program if MODE FLOW is set (FORTRAN).
PRTFLOP	The source code of the routine that prints a flow-trace of all executed Adabas Native SQL statements during the execution of the program if MODE FLOW is set (PL/I).
PRTFLOW	The source code of the routine that prints a flow-trace of all executed Adabas Native SQL statements during the execution of the program if MODE FLOW is set (COBOL).
PRTRAC	The source code of the routine that traces the record buffers and format buffers during the execution of the application program if MODE TRACE is set (FORTRAN).

Name	Description
PRTRACE	The source code of the routine that traces the record buffers and format buffers during the execution of the application program if MODE TRACE is set (COBOL).
PRTRACP	The source code of the routine that traces the record buffers and format buffers during the execution of the application program if MODE TRACE is set (PL/I).
PRTRCICS	The source code of the routine that traces the record buffers and format buffers during the execution of the application program if MODE TRACE is set (COBOL; for use in CICS programs).
RESPCICS	The source code of the response code interpretation routine (COBOL; for use in CICS programs).
RESPF	The source code of the response code interpretation routine (FORTRAN).
RESPINT	The source code of the response code interpretation routine (COBOL).
RESPPL1	The source code of the response code interpretation routine (PL/I).
SQFRDATE	Routine which converts format D number to numeric date.
SQFRTIME	Routine which converts format T number to numeric date and numeric time.
SQTODATE	Routine which converts numeric date to format D number.
SQTOTIME	Routine which converts numeric date and numeric time to format T number.
TYPESADA	Data definitions for use in Ada programs.

If necessary, the source module RESPCICS (COBOL/CICS), RESPF (FORTRAN), RESPINT (COBOL) or RESPPL1 (PL/I) may be edited to accommodate installation-specific requirements. The original source modules should be retained for future reference.

4 Installation

- Copying to an z/OS Disk 10
- Loading Files from Magnetic Tape 11
- Modifying the Sample JCL Procedures 12
- Modifying the Global Parameters 12
- Relinking Adabas Native SQL 13
- Testing Adabas Native SQL 13
- DD-Names 15

Predict Version 3.4 or above is a prerequisite for the installation of Adabas Native SQL. Please check which version of Predict is installed.

The Adabas files and fields that will be used by Adabas Native SQL application programs must be defined in the data dictionary.

This chapter covers the following topics:

Copying to an z/OS Disk

If you are using System Maintenance Aid (SMA), refer to the SMA documentation (included on the current edition of the Natural documentation CD).

If you are not using SMA, follow the instructions below.

This section explains how to:

- Copy data set COPY.JOB from tape to disk.
- Modify this data set to conform with your local naming conventions.

The JCL in this data set is then used to copy all data sets from tape to disk.

If the datasets for more than one product are delivered on the tape, the dataset COPY.JOB contains the JCL to unload the datasets for all delivered products from the tape to your disk.

After that, you will have to perform the individual install procedure for each component.

Step 1: Copy data set COPY.JOB from tape to disk

The data set COPY.JOB (label 2) contains the JCL to unload all other existing data sets from tape to disk. To unload COPY.JOB, use the following sample JCL:

```
//SAGTAPE JOB SAG,CLASS=1,MSGCLASS=X
//* ---------------------------------
//COPY EXEC PGM=IEBGENER
//SYSUT1 DD DSN=COPY.JOB,
// DISP=(OLD,PASS),
// UNIT=(CASS,,DEFER),
// VOL=(,RETAIN,SER=<Tnnnnn>),
// LABEL=(2,SL)
//SYSUT2 DD DSN=<hilev>.COPY.JOB,
// DISP=(NEW,CATLG,DELETE),
// UNIT=3390,VOL=SER=<vvvvvv>,
// SPACE=(TRK,(1,1),RLSE),
// DCB=*.SYSUT1
//SYSPRINT DD SYSOUT=*
```

```
//SYSIN DD DUMMY
//
```

Where:

<hilev>	is a valid high level qualifier.
<Tnnnnn>	is the tape number.
<vvvvvv>	is the desired volser.

Step 2: Modify COPY.JOB to conform with your local naming conventions

There are three parameters you have to set before you can submit this job:

- Set HILEV to a valid high level qualifier.
- Set LOCATION to a storage location.
- Set EXPDT to a valid expiration date.

Step 3: Submit COPY.JOB

Submit COPY.JOB to unload all other data sets from the tape to your disk.

Loading Files from Magnetic Tape

Use a batch Natural job similar to the following to load the error messages into the system file:

```
//NATB     EXEC PGM=batch-natural,REGION=2400K,TIME=1400,
//     PARM='IM=D,MADIO=0,MAXCL=0,MT=0,AUTO=OFF'
//STEPLIB   DD DSN=batch-natural-load-library,DISP=SHR
//          DD DSN=adabas-load-library,DISP=SHR
//DDPRINT   DD SYSOUT=*
//DDDRUCK   DD SYSOUT=*
//CMPRINT   DD SYSOUT=*
//MPMDUMP   DD DUMMY
//DDKARTE   DD DUMMY
//DDCARD    DD *
ADARUN DA=dbid,DE=3390,SVC=249,MODE=MULTI
//CMWKF02   DD DSN=SAGLIB.SQLnnn.ERRN,DISP=SHR
//CMSYSIN   DD *
LOGON SYSERR
ERRLODUS
FIN
/*
//*
//
```

This job is supported by SMA, Job I061, Step 3602. Input file SQL*nnn*.ERRN is used from tape.

Modifying the Sample JCL Procedures

Modify (edit) the supplied JCL procedures ADAGO, COBOLGO, FORTGO, PLIGO and GO as required and then execute the example programs. The following specifications in the procedures should be adapted to suit the system environment. The standard values of the procedure parameters are given in parentheses.

- The Adabas load library index (ADABAS.V*mmm*)
- The Adabas Native SQL load library index (SQL*nnn*)
- The name of the Adabas Native SQL preprocessor module (e.g. *ADASQLC* in job COBOLGO)
- Language-specific libraries (SYS1.COBLIB, SYS2.LINKLIB for COBOL; SYS1.VFORTLIB for FORTRAN; SYS1.PLIBASE, SYS2.PLI.LINKLIB for PL/I)
- The SYSOUT classes of the output datasets (X)
- The membername of the program to be preprocessed (AEX1, CEX1, FEX1 or PEX1)
- The name of the library that contains the member to be preprocessed (SQL*nnn*.SRCE).

Modifying the Global Parameters

The global parameters for Adabas Native SQL must also be modified.

Enter the number of the Predict data dictionary file in the SYSFILE FDIC parameter.

The error message texts for decoding preprocessor-time errors are typically stored in the Natural system file, specified by the global SYSFILE FNAT parameter.

The error messages for decoding run-time errors are typically stored in the data dictionary file. If this is not the case, use the global parameter `ABORT FILE` to specify the location (number) of this file.

See chapter *Global Parameters* in the *Adabas Native SQL Reference Manual* for further information.

Typical global parameter definitions:

```
SYSFILE FDIC=(1,11) FNAT=(1,8).
LANG COBOL.
XREF ON.
ABORT FILE=8.
```

Relinking Adabas Native SQL

If a different Adabas interface module is to be used instead of the standard module ADAUSER, Adabas Native SQL must be relinked. The procedure LINKSQL can be used to link the module ADASQLA, ADASQLC, ADASQLF or ADASQLP.

The module ADAUSER uses the DD-name `DDCARD` to refer to the dataset containing ADARUN parameters. If an alternative module is used, this dataset may not be needed.

Testing Adabas Native SQL

Before testing the examples, check that the files EMPLOYEES and VEHICLES are loaded, and that they are documented in the Predict data dictionary. If the file definitions are not yet present in the data dictionary, they can be loaded from the file PRD*nnn*.DEMO, which can be found on the Predict installation tape. The notation *nnn* represents the Version, Release and SM level of Predict at your site. If you wish to use Adabas Native SQL to preprocess FORTRAN programs, ensure that the necessary language synonyms are defined in the data dictionary. See for example Appendix B in the *Adabas Native SQL Reference Manual*.

The procedure ADAGO, COBOLGO, FORTGO or PLIGO executes the following steps:

1. List the source program;
2. Preprocess the program with Adabas Native SQL;
3. Compile the preprocessed program (Ada, COBOL, FORTRAN or PL/I);
4. Link-edit the program. The modules listed below may be called by the user program at runtime:
5. Execute the program.

If the program is to be passed through other preprocessors in addition to Adabas Native SQL, the corresponding job step should be included in the procedure following the Adabas Native SQL step and preceding compilation.

COBOL	COBOL/CICS	PL/I	Description
RESPINT (C)	RESPCICS (C)	RESPINT (C,P)	Error handling routine.
PRTRACE (C)	PRTRCICS (C)	PRTRACE (C,P)	Runtime trace routine. Activated by the error handling routine if `MODE TRACE` is set.
PRTFLOW (C)	--	PRTFLOW (C,P)	Runtime Adabas Native SQL statement printing. Activated if `MODE FLOW` is set.
FINDCID (A)	FINDCID (A)	FINDCID (A)	Generates dynamic Adabas command IDs. Activated if `OPTION DYNAMCID` is set.
PRPABEND (A)	CICSABEND	PRPABEND (A)	Terminates program execution. Called by the error handling routine.
SQFRDATE (C)	SQFRDATE (C)	SQFRDATE (C)	Routine which converts format D number to numeric date.
SQFRTIME (C)	SQFRTIME (C)	SQFRTIME (C)	Routine which converts format T number to numeric date and numeric time.
SQTODATE (C)	SQTODATE (C)	SQTODATE (C)	Routine which converts numeric date to format D number.
SQTOTIME (C)	SQTOTIME (C)	SQTOTIME (C)	Routine which converts numeric date and numeric time to format T number.

Ada	FORTRAN	Description
RESPF (F)	RESPF(F)	Error handling routine.
PRTRAC (F)	PRTRAC (F)	Runtime trace routine. Activated by the error handling routine if `MODE TRACE` is set.
PRTFLO (F)	PRTFLO (F)	Runtime Adabas Native SQL statement printing. Activated if `MODE FLOW` is set.
FCID (A)	FCID (A)	Generates dynamic Adabas command IDs. Activated if `OPTION DYNAMCID` is set.
PRPABEND (A)	--	Terminates program execution. Called by the error handling routine.
SQFRDATE (C)	SQFRDATE (C)	Routine which converts format D number to numeric date.
SQFRTIME (C)	SQFRTIME (C)	Routine which converts format T number to numeric date and numeric time.
SQTODATE (C)	SQTODATE (C)	Routine which converts numeric date to format D number.
SQTOTIME (C)	SQTOTIME (C)	Routine which converts numeric date and numeric time to format T number.

Notes:

1. The FORTRAN error handling routine RESPF always returns condition code 8 when the job step terminates. The other error handling routines return the Adabas response code.
2. The characters in parentheses indicate the language in which the routines are written (Assembler, COBOL, FORTRAN or PL/I). Only the Assembler and COBOL modules are supplied in object form; if you wish to use FORTRAN or PL/I modules, you must compile the source modules.
3. The COBOL object modules of the routines RESPINT, PRTRACE and PRTFLOW supplied by Software AG can be linked with PL/I object modules to produce an executable load module. If you wish to use the PL/I versions of these routines, you must rename them: RESPPL1 - RESPINT, PRTRACP - PRTRACE, PRTFLOP - PRTFLOW, and then compile them, replacing the Software AG-supplied COBOL object modules by the PL/I object modules. The global parameter `ABORT PLI` should be coded. The procedure as supplied links the standard Adabas interface module ADAUSER to the application program. If a different module is used, the procedure must be modified accordingly.

DD-Names

Adabas Native SQL refers to datasets using the following DD-names:

DDNAME	Description
ADAIN	The source program to be preprocessed. LRECL=80,RECFM=FB
ADAOUT	The output of the preprocessor. Embedded Adabas Native SQL statements in the source program appear in this dataset as comments and are followed by the generated COBOL or PL/I code. LRECL=80,BLKSIZE=3120,RECFM=FB
ADAGLOB	The global parameter definitions. LRECL=80,RECFM=FB
ADAMES	Adabas Native SQL writes messages at preprocessing time to this dataset.
SYSUT1	A temporary work file. LRECL=80,BLKSIZE=3120,RECFM=FB
DDCARD	The Adabas parameters that are used by Adabas Native SQL at preprocessor time.
SYSDBOUT	Error messages are written to this dataset.
SYSOUT	Adabas Native SQL writes messages to this dataset if it abends.

At run time, the application program generated by Adabas Native SQL refers to datasets using the following DD-names (in addition to any other user-defined datasets):

DDNAME	Description
DDCARD	The Adabas parameters that are used by the ADARUN module at run time.
DDPRINT	Adabas writes messages to this dataset at run time.
SYSOUT	If the FLOW facility is used, the flow-tracing output is written to this dataset.
TSTDMP	If the TRACE facility is used, the trace output is written to this dataset. LRECL=80,BLKSIZE=3680,RECFM=FB

5 INSTALLATION CHECKLIST

- Verify that Predict Version 3.4 or later is installed.
- Verify that the files and fields that are to be used by Adabas Native SQL applications are documented in the data dictionary. Adabas Native SQL recognizes only the PREDICT file types *Adabas file* and *Adabas userview*. Other file types are ignored by the Adabas Native SQL preprocessor.
- Load the libraries from tape to disk.
- Verify that the error messages have been loaded (ERRLODUS).
- If an Adabas interface module other that ADAUSER is to be used, relink Adabas Native SQL.
- Edit the procedures ADAGO, COBOLGO, FORTGO, PLIGO and GO as required.
- Ensure that the standard test files EMPLOYEES and VEHICLES are available and that they are correctly documented in the data dictionary. In particular, ensure that the file numbers and the database ID are correct. Ensure that the relationships for soft coupling are defined in the data dictionary. If you wish to use FORTRAN, ensure that appropriate language synonyms are defined in the data dictionary (see Appendix B in the *Adabas Native SQL Reference Manual*).
- Run the procedures ADAGO, COBOLGO, FORTGO, PLIGO and GO to execute the test programs. Verify the output.

6 HINTS FOR USING ADABAS NATIVE SQL

- Periodic Groups and Multiple Fields 20
- Group Structure of Periodic Groups 20
- Dynamic Command-IDs 20
- Locating Errors 20
- Preprocessor Copy and Generate Facilities 20
- Restriction for MU Fields Within PE 21
- Hyphens and Break Characters in PL/I Programs 21
- READ ISN Statement 21
- PL/I - Margin Settings 21
- PL/I - Structure Variables in Superdescriptors 22
- Last Statement Restriction in COBOL/II 22
- Adabas Native SQL Preprocessor Condition Codes 22
- SQL0217 Error Message When Running the Precompiler 22
- Hints for Improving Adabas Native SQL Efficiency 23

This chapter covers the following topics:

Periodic Groups and Multiple Fields

If the maximum number of occurrences of periodic groups or multiple fields is known, specify this number in Predict. Otherwise Adabas Native SQL will allocate buffers using the default maximum values, resulting in wasted storage.

Group Structure of Periodic Groups

Correct use of the `GROUP STRUCT` attribute in Predict can save significant amounts of space in the format buffers that are used when accessing records containing periodic groups. See *Defining More Attributes of Fields* in section *Field* of chapter *Predefined Object Types* in the *Predict Reference Manual*.

Dynamic Command-IDs

If the database is accessed from many modules within one linked program, the global parameter `OPTIONS DYNAMCID.` should be specified. This ensures that a unique command-ID will be generated for each Adabas command if the same cursor-name is used in more than one module. Be aware that this can degrade run-time performance. See ***Improving Adabas Native SQL Efficiency*** and the section on using command-IDs in the *Adabas Command Reference Manual*.

Locating Errors

The TRACE and FLOW facilities, which are switched on by means of global parameters, can be used to find runtime errors in the application program.

Preprocessor Copy and Generate Facilities

The Adabas Native SQL preprocessor supports the COPY and GENERATE statements. These are compatible with the implementation of the COPY and GENERATE statements in the Predict preprocessor, so it should not normally be necessary to use both Adabas Native SQL and Predict preprocessors.

Restriction for MU Fields Within PE

The data dictionary definition of a multiple-value field within a periodic group should not specify a counter field.

Hyphens and Break Characters in PL/I Programs

Field names as defined in the data dictionary may include hyphens ("-") and/or break characters ("_"). With Adabas Native SQL, each reference to a field name must match the definition in the data dictionary exactly. However, when generating PL/I output, Adabas Native SQL will change all hyphens found in data dictionary definitions to break characters, since hyphens are not valid in PL/I identifiers.

With previous versions of Adabas Native SQL, it was not necessary for references to field names to match the definitions in the data dictionary exactly: all break characters found in the source program were converted to hyphens before being compared with the data dictionary definitions; then, when generating PL/I output, Adabas Native SQL changed all hyphens to break characters.

The global parameter `UNDERSCORE NO.` can be used to ensure that Adabas Native SQL processes hyphens and break characters in a manner compatible with earlier versions.

READ ISN Statement

After issuing the READ ISN statement with `OPTIONS SEQUENCE`, the program should check for end-of-file (ADACODE = 3). In some applications, it may be necessary to compare the ISN of the record that was read with the ISN that was specified in the `WHERE` clause of the statement.

PL/I - Margin Settings

When using the PL/I compiler, the margins should be set to (2,72) (these are the default values).

PL/I - Structure Variables in Superdescriptors

In PL/I, it is not possible to assign a structure variable to a superdescriptor in the WHERE clause. A string variable should be used in place of the structure variable.

Last Statement Restriction in COBOL/II

The last statement in a COBOL/II program should not be an Adabas Native SQL statement. If necessary, the statement `EXIT.` can be coded at the end of the program.

Adabas Native SQL Preprocessor Condition Codes

The Adabas Native SQL preprocessor can set the following condition codes:

Code	Meaning
0	No error was detected
4	One or more warning messages issued during processing.
8	One or more errors found during processing.
12	Adabas Native SQL abended.

Programs generated by the Adabas Native SQL preprocessor can cause various errors at runtime. Adabas errors are detected by the generated code. In COBOL and PL/I programs, the Adabas response code as described in the *Adabas Messages and Codes Manual* is returned to the calling procedure. In FORTRAN programs, response code 8 is returned to the calling procedure.

SQL0217 Error Message When Running the Precompiler

This message is:

```
SQL0217 FDIC DBID OR FDIC FNR DOES NOT MATCH THE CONTROL RECORD
```

This message occurs if a DDM from another system is loaded into the Natural FDIC file and thereafter the DBID and FNR of the DDM are changed online to the current FDIC's DBID and FNR.

To rectify this, start the Natural containing the Predict:

```
LOGON SYSDIC
MENU
D       for defaults
A       for Adabas Native SQL
<Enter>
.       to exit
```

This will reset the Adabas Native SQL control record.

Hints for Improving Adabas Native SQL Efficiency

Reducing Database Accesses

Using the HISTOGRAM and READ statements as much as possible in preference to FIND can help to reduce the number of accesses made to the database.

Command IDs

An explicit Adabas command ID is assigned whenever a cursor is declared in an Adabas Native SQL statement. Adabas Native SQL statements that are performed repeatedly should have a cursor, since the command ID derived from the cursor name is used by Adabas to indicate that the format buffer need not be translated repeatedly.

Variable Indices

Periodic groups and multiple fields should only be referenced using variable indices when this is unavoidable, since variable indices cause additional format translations and also additional RC commands to be executed.

Record Buffers

The number of record buffers generated by Adabas Native SQL can be reduced by coding UPDATE and INSERT statements with the `WHERE CURRENT OF` *`cursor-name`* clause but without the `SET` clause. This only applies if the UPDATE or INSERT statement and the statement referenced by the cursor-name contain the same field structure.

SELECT Clause

Program efficiency is improved if only those fields that are needed are coded in the SELECT clause. You may be tempted to write "SELECT *" instead of coding the name of each field, but this can result in excessively large record buffers and slower-running programs.

Global Format ID

Application programs, in particular online application programs, should use the the global options parameter OPTIONS GFORMAT. This can improve efficiency by reducing the number of format buffer translations that are performed at runtime.