National Database System (NDS-32)
Coding Standards for ORACLE Stored Procedures & Functions

Annex – 6

Objectives

A well-defined system needs to follow standards. The philosophy behind it is consistency, ease of remembrance and maintenance.

Scope

These standards apply to Oracle Stored Procedures & Functions.

In NDS-32, following nature of transactions are classified. Procedure naming convention will indicate such nature of transactions.

Nature of Transactions

Transaction Type	***Description***
MST	Master
TRN	Transaction
RPT	Report
EDI	Electronic Data Interchange

Naming convention for Procedure name
All procedures must be represented as NDS_<MODULE>_PRC_programname
For eg:
NDS_TRN_IE<name>
NDS_TRN _frmA
NDS_MST_convention
NDS_EDI_common

Function name
All functions must be represented as NDS_<MODULE>_FNC_programname

Cursor names
Cur_<cursor name>
Loop variable – lcur

Variable names
Integer - lint_variablename
Number - lnum_variablename
Long - llng_variablename
Varchar2 - lstr_variablename
Char - lstr_variablename
Date - ldte_variablename

Parameters names
Integer - pint_variablename
Number - pnum_variablename

Long - plng_variablename
Varchar2 - pstr_variablename
Char - pstr_variablename
Date - pdte_variablename

Standards for sql statements
SQL commands should be capitals for eg SELECT * FROM NDS_MST_CONVENTION

Exceptions
Name should start with e_<exception name>
The Exception 'when others' is compulsory

NOTE : Sample Code for stored procedure.

```
/*****************************HEADERSECTION*************************
**/

--PROCEDURE NAME      :      NDS_EDI_PRC_COMMON
--AUTHOR              :      DATAMATICS LTD.
--DESCRIPTION         :      THE PROCEDURE IS USED TO GENERATE
                            AN
                            EDI FILE FOR ANY MESSAGE SPECIFIED
--INPUT PARAMETERS    :      MESSSAGE ID, PROCESS ID, SENDER AND
                            RECEIVER COUNTRY CODE, CA/
                            ESTABLISHMENT CODES, EAN
                            NUMBERS, REFERENCE NUMBER AND
                            COMMENTS.

--OUTPUT PARAMETERS   :      NONE
--STEPS INVOLVED      :      1. OBTAIN THE NUMBER OF MESSAGE
                            GROUPS, TABLES FROM WHICH DATA IS
                            TO
                            BE RETRIEVED.
--                          2. OBTAIN THE COLUMNS (PRIMARY AS
                            WELL AS NON -KEY) REQUIRED FOR EDI.
                            3. DYNAMICALLY BUILD QUERY AND
                                INSERT INTO NDS_EDI_MSGTAB.
                            4. RETRIEVE THE RECORDS FROM
                                NDS_EDI_MSGTAB TABLE AND
                                INSERT INTO
                                NDS_EDI_MESSAGETABLE.
                            5. THE RECORDS ARE INSERT INTO
                                NDS_EDI_MESSAGETABLE AS PER
                                THE EDI FORMAT.


/*****************************END OF HEADER SECTION
*************** /
create or replace procedure nds_edi_prc_common(

                                    ) as


/******************EXCEPTIONS HANDLED
*****************************/
e_noscean EXCEPTION ;
e_noseean EXCEPTION ;
e_norcean EXCEPTION ;
e_noreean EXCEPTION ;
```

```
/*******************CURSORS DECLARED
*****************************/

/*** CURSOR TO OBTAIN THE NUMBER OF MESSAGE GROUPS AND THEIR
RESPECTIVE TABLES ***/

// for parameterized cursor
CURSOR name (variable varchar2) is
        SELECT NDSEDI_MSGGRPID, NDSEDI_MSGTABLE FROM
NDS_EDI_CONTROL
        WHERE NDSEDI_MSGID LIKE ltrim(rtrim(variable));

/*******************VARIABLES
USED*********************************/

lstr_errorcode varchar2(10);          -- ERROR CODE
lstr_errortext varchar2(50);          -- ERROR DESCRIPTION


lnum_nullctr    number;               -- COUNTER FOR NULL COLUMNS

Begin

                /* INITIALIZATION OF LOCAL VARIABLES */
                lstr_pkcol      := null;            -- STRING OF PRIMARY
KEYS
                lstr_selectcol  := null;            -- STRING OF COLUMNS
                lstr_selectstr  := null;            -- SELECT STRING
                lstr_finalselect := null;           -- FINAL SELECT
STATEMENT
                lstr_finalinsert := null;           -- FINAL INSERT
STATEMENT

                <MAIN BODY OF CODE>

/*******************************EXCEPTION SECTION ************ /
        EXCEPTION
                WHEN e_noscean then
                        INSERT into log_table(info) VALUES ('EAN Code not found
for Sender Competent Authority.');
                WHEN e_noseean then
                        INSERT into log_table(info) VALUES ('EAN Code not found
for Sender Establishment.');
                WHEN e_norcean then
                        INSERT into log_table(info) VALUES ('EAN Code not found
for Receiver Competent Authority');
                WHEN e_noreean then
                        INSERT into log_table(info) VALUES ('EAN Code not found
for Receiver Establishment.');
```

```
            WHEN NO_DATA_FOUND then
                  INSERT into log_table(info) VALUES ('No data found in
MASTER tables. ');
            WHEN OTHERS then
                  v_errortext :=substr(SQLERRM,1,200);
                  INSERT into log_table(info) VALUES (v_errortext);
                  IF DBMS_SQL.IS_OPEN(lint_sqlcur) THEN
                        DBMS_SQL.CLOSE_CURSOR(lnum_sqlcur);       --
for DBMS_SQL cursor
                  END IF;
                  RAISE;                              -- reraise the exception
/*******************************END OF EXCEPTION SECTION
***********/

End;

/****************************** END OF PROCEDURE
*****************/
```

--- XXX ---