

WRDC-TR-90-8007  
Volume V  
Part 6  
Section 5 of 6

AD-A251 999



1

INTEGRATED INFORMATION SUPPORT SYSTEM (IISS)  
Volume V - Common Data Model Subsystem  
Part 6 - Neutral Data Definition Language (NDDL) Product  
Specification  
Section 5 of 6

J. Althoff, M. Apicella, S. Singh

Control Data Corporation  
Integration Technology Services  
2970 Presidential Drive  
Fairborn, OH 45324-6209

DTIC  
ELECTE  
JUN 19 1992  
S A D

September 1990

Final Report for Period 1 April 1987 - 31 December 1990

Approved for Public Release; Distribution is Unlimited

MANUFACTURING TECHNOLOGY DIRECTORATE  
WRIGHT RESEARCH AND DEVELOPMENT CENTER  
AIR FORCE SYSTEMS COMMAND  
WRIGHT-PATTERSON AIR FORCE BASE, OHIO 45433-6533

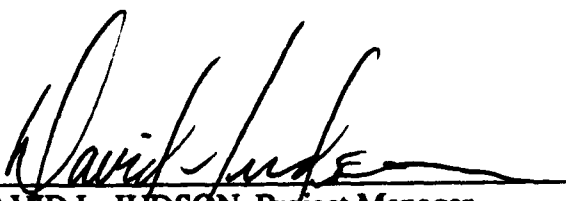
92-16011

## NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever, regardless whether or not the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data. It should not, therefore, be construed or implied by any person, persons, or organization that the Government is licensing or conveying any rights or permission to manufacture, use, or market any patented invention that may in any way be related thereto.

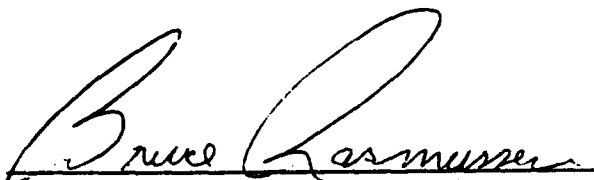
This technical report has been reviewed and is approved for publication.

This report is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations

  
DAVID L. JUDSON, Project Manager  
WRDC/MTI  
Wright-Patterson AFB, OH 45433-6533

25 July 91  
DATE

FOR THE COMMANDER:

  
BRUCE A. RASMUSSEN, Chief  
WRDC/MTI  
Wright-Patterson AFB, OH 45433-6533

25 July 91  
DATE

If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify WRDC/MTI, Wright-Patterson Air Force Base, OH 45433-6533 to help us maintain a current mailing list.

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

# REPORT DOCUMENTATION PAGE

FORM APPROVED  
OMB NO. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

|                                                                                                                                                                                                                                                                     |                                           |                                          |                                                                                                                             |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------|------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------|
| 1. AGENCY USE ONLY (Leave Blank)                                                                                                                                                                                                                                    |                                           | 2. REPORT DATE<br>September 1990         | 3. REPORT TYPE AND DATES COVERED<br>Final Technical Report<br>1Apr87 - 31Dec90                                              |
| 4. TITLE AND SUBTITLE<br>INTEGRATED INFORMATION SUPPORT SYSTEM (IIS)<br>Volume V - Common Data Model Subsystem<br>Part 6 - Neutral Data Definition Language (NDDL) Product Specification<br>Section 5 of 6                                                          |                                           |                                          | 5. FUNDING NUMBERS<br>Contract No.: F33600-87-C-0464<br>PR: 78011F<br>Proj. No.: 595600<br>Task No.: P95600<br>WU: 20950607 |
| 6. AUTHOR(S)<br>J. Althoff, M. Apicella, S. Singh                                                                                                                                                                                                                   |                                           |                                          | 8. PERFORMING ORGANIZATION<br>REPORT NUMBER<br>PS 620341100                                                                 |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>Controld Data Corporation<br>Integration Technology Services<br>2970 Presidential Drive<br>Fairborn, OH 45324-6209                                                                                            |                                           |                                          | 10. SPONSORING/MONITORING<br>AGENCY REP NUMBER<br>WRDC-TR-90-8007, Vol. V, Part 6<br>Section 5 of 6                         |
| 9. SPONSORING MONITORING AGENCY NAME(S) AND ADDRESS(ES)<br>Manufacturing Technology Directorate (WRDC/MTI)<br>Wright-Patterson AFB, OH 45433-6533                                                                                                                   |                                           |                                          |                                                                                                                             |
| 11. SUPPLEMENTARY NOTES<br>WRDC/MTI Project Priority 6203                                                                                                                                                                                                           |                                           |                                          |                                                                                                                             |
| 12a. DISTRIBUTION/AVAILABILITY STATEMENT<br>Approved for Public Release; Distribution is Unlimited.                                                                                                                                                                 |                                           |                                          | 12b. DISTRIBUTION CODE                                                                                                      |
| 13. ABSTRACT<br><br>This specification establishes the development, test and qualification requirements of a computer program identified as the "Neutral Data Definition Language Processor" (NDDL Processor).<br><br>This report is divided into six (6) sections. |                                           |                                          |                                                                                                                             |
| 14. SUBJECT TERMS                                                                                                                                                                                                                                                   |                                           |                                          | 15. NUMBER OF PAGES<br>1881                                                                                                 |
|                                                                                                                                                                                                                                                                     |                                           |                                          | 16. PRICE CODE                                                                                                              |
| 17. SECURITY CLASSIFICATION<br>OF REPORT<br>SAR                                                                                                                                                                                                                     | 18. SECURITY CLASS<br>OF THIS PAGE<br>SAR | 19. SECURITY CLASS<br>OF ABSTRACT<br>SAR | 20. LIMITATION ABSTRACT<br>SAR                                                                                              |

Standard Form 298 (Rev 2-89)  
Prescribed by ANSI Std Z39-18  
298-102

DOCGROUP PS41100 Module Documentation

NAME: DELDSL1  
PURPOSE: DELETE A RECORD FROM DF\_SET\_LINKAGE ENITY  
LANGUAGE: C  
SOURCE FILE: DELDSL1  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

SYNOPSIS

C -- DELDSL1(&DB\_ID,&STATUS)  
COBOL -- CALL "DELDL1" USING  
DB-ID,  
STATUS.  
FORTRAN -- CALL DELDSL1(DBID,STATUS)

INPUT:

INT \*DB\_ID ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
DF\_SET\_LINKAGE. IF SUCCESSFULLY DELETED THE RECORD , THE  
STATUS  
AND RETURN VALUE BOTH WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

DB\_ID  
STATUS

INT \*  
INT \*

ROUTINES CALLED:

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

|                    |                                            |
|--------------------|--------------------------------------------|
| Accession For      |                                            |
| NTIS CRA&I         | <input checked="checked" type="checkbox"/> |
| DTIC TAB           | <input type="checkbox"/>                   |
| Unannounced        | <input type="checkbox"/>                   |
| Justification      |                                            |
| By                 |                                            |
| Distribution /     |                                            |
| Availability Codes |                                            |
| Dist               | Avail and/or Special                       |
| A-1                |                                            |

DOCGROUP PS41100 Module Documentation

NAME: DELDSL2  
PURPOSE: DELETE A RECORD FROM DF\_SET\_LINKAGE ENITY  
LANGUAGE: C  
SOURCE FILE: DELDSL2  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

|          |    |                                                    |
|----------|----|----------------------------------------------------|
| C        | -- | DELDL2(&DB_ID, &RT_ID, &STATUS)                    |
| COBOL    | -- | CALL "DELDL2" USING<br>DB-ID,<br>RT-ID,<br>STATUS. |
| FORTTRAN | -- | CALL DELDSL2(DBID, RTID, STATUS)                   |

INPUT:

INT \*DB\_ID ;  
CHAR \*RT\_ID ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
DF\_SET\_LINKAGE  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

|        |        |
|--------|--------|
| DB_ID  | INT *  |
| RT_ID  | CHAR * |
| STATUS | INT *  |

ROUTINES CALLED:

-----

STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELDSL3  
PURPOSE: DELETE A RECORD FROM DF\_SET\_LINKAGE ENITY  
LANGUAGE: C  
SOURCE FILE: DELDSL3  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C           --    DELSL3(&DB\_ID, &RT\_ID, &DF\_ID, &STATUS)  
  
COBOL       --    CALL "DELSL3" USING  
                    DB-ID,  
                    RT-ID,  
                    DF-ID,  
                    STATUS.  
  
FORTRAN     --    CALL DELDSL3(DBID, RTID, DFID, STATUS)

INPUT:

INT   \*DB\_ID ;  
CHAR  \*RT\_ID ;  
CHAR  \*DF\_ID ;

OUTPUT:

INT  \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
DF\_SET\_LINKAGE

IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

|        |        |
|--------|--------|
| DB_ID  | INT *  |
| RT_ID  | CHAR * |
| DF_ID  | CHAR * |
| STATUS | INT *  |

ROUTINES CALLED:

-----

STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELEC  
PURPOSE: DELETE THE RECORD FROM ENTITY\_CLASS  
LANGUAGE: C  
SOURCE FILE: DELEC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
WHERE EC\_NO =: "INPUT PARAMETER 1 "

SYNOPSIS

C -- DELEC(&EC\_NO, &STATUS)  
COBOL -- CALL "DELEC" USING  
                                EC-NO,  
                                STATUS.  
FORTRAN -- CALL DELEC(ECNO,STATUS)

INPUT:

INT \*EC\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
ENTITY CLASS

WHERE EC\_NO EQUALS TO "INPUT PARAMETER 1".

IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

EC\_NO                                  INT \*  
STATUS                                 INT \*

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELECAL  
PURPOSE: DELETE RECORD CLASS FROM ENTITY\_NAME  
LANGUAGE: C  
SOURCE FILE: DELECAL  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
WHERE EC\_NAME =: "INPUT PARAMETER 1 " AND  
EC\_NO := "INPUT PARAMETER 2" ;  
SYNOPSIS  
C -- DELECAL(EC\_NAME, &EC\_NO, &STATUS)  
COBOL -- CALL "DELECAL" USING  
EC-NAME,  
EC-NO,  
STATUS.  
FORTRAN -- CALL DELECAL(ECNAME, ECNO, STATUS)  
INPUT: CHAR \*EC\_NAME ;  
INT \*EC\_NO ;  
OUTPUT: INT \*STATUS ;  
DESCRIPTION:  
THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
ENTITY\_NAME  
WHERE EC NAME EQUALS TO "INPUT PARAMETER 1" AND EC\_NO  
EQUALS TO "INPUT PARAMETER 2".  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
EC\_NAME CHAR \*  
EC\_NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: DELECKW  
PURPOSE: DELETE A RECORD FROM EC\_KEYWORD  
LANGUAGE: C  
SOURCE FILE: DELECKW  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

WHERE EC\_NO =: "INPUT PARAMETER 1" ;

SYNOPSIS

C -- DELECKW(&EC\_NO,&STATUS)  
COBOL -- CALL "DELECKW" USING  
EC-NO,  
STATUS.  
FORTRAN -- CALL DELECKW(ECNO,STATUS)

INPUT:

INT \*EC\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
EC\_KEYWORD

WHERE EC\_NO EQUALS TO "INPUT PARAMETER 1".

IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

EC\_NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: DELFLDA  
PURPOSE: DELETE A RECORD FROM DATA\_FIELD ENITY  
LANGUAGE: C  
SOURCE FILE: DELFLDA  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

SYNOPSIS

```
      C          --  DELFLDA(&DB_ID, &RT_ID,  &STATUS)

      COBOL       --  CALL "DELFLDA" USING
                        DB-ID,
                        RT-ID,
                        STATUS.

      FORTRAN     --  CALL DELFLDA(DBID, RTID,  STATUS)
```

INPUT:

```
      INT  *DB_ID ;
      CHAR *RT_ID ;
```

OUTPUT:

```
      INT *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
DATA\_FIELD  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

|        |        |
|--------|--------|
| DB_ID  | INT *  |
| RT_ID  | CHAR * |
| STATUS | INT *  |

ROUTINES CALLED:

-----

STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELIAUC  
PURPOSE: DELETE A REORD FROM INHERITED\_ATT\_USE  
LANGUAGE: C  
SOURCE FILE: DELIAUC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
WHERE TAG\_NO = : 1 ;  
SYNOPSIS  
C -- DELIAUC(&TAG\_NO,&STATUS)  
COBOL -- CALL "DELIAUC" USING  
TAG-NO,  
STATUS.  
FORTRAN -- CALL DELIAUC(TAGNO,STATUS)  
INPUT:  
INT \*TAG\_NO ;  
OUTPUT:  
INT \*STATUS ;  
DESCRIPTION:  
THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
INHERITED\_ATT\_USE  
WHERE TAG\_NO EQUALS TO "INPUT PAPAMETER 1".  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
TAG\_NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTTL  
SQLOPN  
SQLCSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELIAUK  
PURPOSE: DELETE A RECORD FROM INHERITED\_ATT\_USE ENITY  
LANGUAGE: C  
SOURCE FILE: DELIAUK  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

          C                  --  DELIAUK(&KC\_NO, &STATUS)  
  
          COBOL              --  CALL "DELIAUK" USING  
                                  KC-NO,  
                                  STATUS.  
  
          FORTRAN          --  CALL DELIAUK(KCNO, STATUS)

INPUT:

      INT  \*KC\_NO ;

OUTPUT:

      INT  \*STATUS ;

DESCRIPTION:

      THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
      INHERITED\_ATT\_USE. IF SUCCESSFULLY DELETED THE RECORD ,  
THE STATUS

      AND RETURN VALUE BOTH WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
KC\_NO                          INT \*  
STATUS                         INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELIRCS  
PURPOSE: DELETE RECORD CLASS FROM RC\_BASED\_REC\_SET  
LANGUAGE: C  
SOURCE FILE: DELIRCS  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
WHERE DB\_ID =: "INPUT PARAMETER 1 " AND  
SET\_ID := "INPUT PARAMETER 2" AND  
RT\_ID := "INPUT PARAMETER 3 " AND  
RC\_NO := "INPUT PARAMETER 4" ;

SYNOPSIS

C -- DELIRCS(&DB\_ID, SET\_ID, RT\_ID, &RC\_NO  
&STATUS)  
  
COBOL -- CALL "DELIRCS" USING  
DB-ID,  
SET-ID,  
RT-ID,  
RC-NO,  
STATUS.

FOSETRAN -- CALL DELIRCS(DBID,SETID,RTID,RCNO,STATUS)  
INPUT:  
CHAR \*SET\_ID ;  
CHAR \*RT\_ID ;  
INT \*DB\_ID ;  
INT \*RC\_NO ;

OUTPUT:  
INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
RC\_BASED\_REC\_SET  
WHERE DB\_ID EQUALS TO "INPUT PARAMETER 1" AND RT\_ID  
EQUALS  
TO "INPUT PARAMETER 3" AND SET\_ID EQUALS INPUT PARAMETER 2  
AND RC NO EQUALS INPUT PARAMETER 4 ".  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
DB\_ID INT \*  
SET\_ID CHAR \*  
RT\_ID CHAR \*  
RC\_NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELKC  
PURPOSE: DELETE A RECORD FROM KEY\_CLASS  
LANGUAGE: C  
SOURCE FILE: DELKC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
WHERE KC\_NO =: 1 ;

SYNOPSIS

C -- DELKC(&KC\_NO,&STATUS)  
COBOL -- CALL "DELKC" USING  
KC-NO,  
STATUS.  
FORTRAN -- CALL DELKC(KCNO,STATUS)

INPUT:

INT \*KC\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
KY\_CLASS

WHERE KC\_NO = "INPUT PARAMETER 1".

IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
KC\_NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELKCM  
PURPOSE: DELETE A RECORD FROM KEY\_CLASS\_MEMBER  
LANGUAGE: C  
SOURCE FILE: DELKCM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

WHERE KC\_NO =: 1 ;

SYNOPSIS

    C                  --  DELKCM(&KC\_NO,&STATUS)

    COBOL              --  CALL DELKCM"  USING

                                  KC\_NO,

                                  STATUS.

    FORTRAN          --  CALL DELKCM(KCNO,STATUS)

INPUT:  INT  \*KC\_NO ;

OUTPUT: INT  \*STATUS ;

DESCRIPTION:

    THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM

    KEY CLASS MEMBER WHERE KC\_NO = "INPUT PARAMETER 1".

    IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND

RETURN VALUE BOTH

    WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI

INTERFACE

ARGUMENTS:

-----

KC\_NO                          INT \*

STATUS                         INT \*

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2



SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELKCMT  
PURPOSE: DELETE A RECORD FROM KEY\_CLASS\_MEMBER  
LANGUAGE: C  
SOURCE FILE: DELKCMT  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
WHERE TAG\_NO =: 1 ;  
SYNOPSIS  
C -- DELTAGMT(&TAG NO,&STATUS)  
COBOL -- CALL "DELKCMT" USING  
TAG-NO,  
STATUS.  
FORTRAN -- CALL DELKCMT(TAGNO,STATUS)  
INPUT: INT \*TAG NO ;  
OUTPUT: INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
KEY CLASS MEMBER WHERE TAG NO = "INPUT PARAMETER 1".  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
TAG\_NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELKW  
PURPOSE: DELETE A RECORD FROM AC\_KEYWORD  
LANGUAGE: C  
SOURCE FILE: DELKW  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
WHERE KW\_NO =: 1 ;  
SYNOPSIS  
C -- DELKWAC(&KW\_NO,&STATUS)  
COBOL -- CALL "DELKW" USING  
KW-NO,  
STATUS.  
FORTRAN -- CALL DELKW(KWNO,STATUS)  
INPUT: INT \*KW\_NO ;  
OUTPUT: INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
AC\_KEYWORD

WHERE KW\_NO = "INPUT PARAMETER 1".

IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
KW\_NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELKWAC  
PURPOSE: DELETE A RECORD FROM AC\_KEYWORD  
LANGUAGE: C  
SOURCE FILE: DELKWAC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
WHERE KW\_NO =: 1 ;

SYNOPSIS

C -- DELKWAC(&KW\_NO,&STATUS)  
COBOL -- CALL "DELKWAC" USING  
KW\_NO,  
STATUS.  
FORTRAN -- CALL DELKWAC(KWNO,STATUS)

INPUT:

INT \*KW\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
AC\_KEYWORD

WHERE KW\_NO = "INPUT PARAMETER 1".

IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
KW\_NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELKWEC  
PURPOSE: DELETE A RECORD FROM EC\_KEYWORD  
LANGUAGE: C  
SOURCE FILE: DELKWEC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
WHERE KW\_NO =: 1 ;  
SYNOPSIS  
C -- DELKWEC(&KW\_NO,&STATUS)  
COBOL -- CALL "DELKWEC" USING  
KW-NO,  
STATUS.  
FORTRAN -- CALL DELKWEC(KWNO,STATUS)

INPUT: INT \*KW\_NO ;  
OUTPUT: INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
EC\_KEYWORD

WHERE KW\_NO = "INPUT PARAMETER 1".

IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
KW\_NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELKWRC  
PURPOSE: DELETE A RECORD FROM RC\_KEYWORD  
LANGUAGE: C  
SOURCE FILE: DELKWRC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

WHERE KW\_NO =: 1 ;

SYNOPSIS

C -- DELKWRC(&KW\_NO,&STATUS)  
COBOL -- CALL "DELKWRC" USING  
KW-NO,  
STATUS.  
FORTRAN -- CALL DELKWRC(KWNO,STATUS)

INPUT:

INT \*KW\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
RC\_KEYWORD

WHERE KW\_NO = "INPUT PARAMETER 1".

IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

KW\_NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELLR  
PURPOSE: DELETE A RECORD FROM LINK\_RELATION  
LANGUAGE: C  
SOURCE FILE: DELLR  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

WHERE RC\_NO =: "INPUT PARAMETER 1" ;

SYNOPSIS

C -- DELLR(&RC\_NO,&STATUS)  
COBOL -- CALL "DELLR" USING  
RC-NO,  
STATUS.  
FORTRAN -- CALL DELLR(RCNO,STATUS)

INPUT:

INT \*RC\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
LINK\_RELATION

WHERE RC\_NO = "INPUT PARAMETER 1".

IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

RC\_NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



SQLCLS  
DELIAUC  
SPRINTF  
UERROR  
ADDRNUM  
FRETAGLST  
ERRRPT  
DELKCMT  
DELAUCL  
SQLAD2  
SQLFCH  
INSTAGLST  
SQLTOC  
SQLOSQ  
SQLAB2  
SQLEXE

DOCGROUP PS41100 Module Documentation

NAME: INSTAGLST  
PURPOSE:  
LANGUAGE: C  
SOURCE FILE: DELMIGK  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

INCLUDE FILES:

-----  
STDIO  
CSTDYTP  
UNIQENO  
EOD

ROUTINES CALLED:

-----  
MALLOC



DOCGROUP PS41100 Module Documentation

NAME: FRETAGLST  
PURPOSE:  
LANGUAGE: C  
SOURCE FILE: DELMIGK  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

ARGUMENTS:

-----  
DEL\_NODE

TAGUSED \*

INCLUDE FILES:

-----  
STDIO  
CSTDYTP  
UNIQENO  
EOD

ROUTINES CALLED:

-----  
FRETAGLST  
FREE

DOCGROUP PS41100 Module Documentation

NAME: DELMOD  
PURPOSE: DELETE A RECORD FROM MODEL\_CLASS ENITY  
LANGUAGE: C  
SOURCE FILE: DELMOD  
SOURCE FILE TYPE: DAT  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: TEST

DESCRIPTION:

-----  
SYNOPSIS

          C                  --      DELMOD(&MODEL\_NO, &STATUS)  
  
          COBOL              --      CALL "DELMOD" USING  
                                  MODEL-NO,  
                                  STATUS.  
  
          FORTRAN          --      CALL DELMOD(MODELNO, STATUS)

INPUT:

          INT \*MODEL\_NO ;

OUTPUT:

          INT \*STATUS ;

DESCRIPTION:

          THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
          MODEL\_CLASS. IF SUCCESSFULLY DELETED THE RECORD , THE  
STATUS

          AND RETURN VALUE BOTH WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
MODEL NO                          INT \*  
STATUS                            INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELMTKC  
PURPOSE: DELETE EMPTY KEY CLASSES GIVEN THE MODEL NUMBER  
LANGUAGE: C  
SOURCE FILE: DELMTKC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C               -- DELMTCK(&MODEL\_NO)  
  
COBOL           -- CALL "DELMTKC"            USING  
                                          MODEL-NO.  
FORTRAN       -- CALL DELMTKC (MODELNO)

INPUT:

INT \*MODEL\_NO

OUTPUT:

DESCRIPTION

THIS ROUTINE USES AN ORACLE SELECT STATEMENT TO OBTAIN ALL  
KEY CLASS NUMBERS WHICH HAVE NO MEMBERS. FOR EACH KEY

CLASS

FOUND, THE RECORD IS DELETED FROM THE COMPLETE RELATION

CLASS,

THE KEY CLASS ITSELF, AND THE KEY CLASS COPY CROSS

REFERENCE.

THIS ROUTINE WILL ALSO DELETE TEXTUAL DESCRIPTIONS AND ADD  
THE REUSABLE KEY CLASS NUMBER BACK TO THE CDM

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
MODEL\_NO

INT \*

INCLUDE FILES:

-----  
CSTDTP  
UNIQENO  
EOD

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLCLS  
ERRRPT  
SQLAD2  
SQLFCH  
DELCMPR

SPRINTF  
UERROR  
ADDRNUM  
DELKC  
SQLTOC  
SQLOSQ  
SQLAB2  
SQLEXE

DOCGROUP PS41100 Module Documentation

NAME: DELOACE  
PURPOSE: DELETE A RECORD FROM OWNED\_ATTRIBUTE ENITY  
LANGUAGE: C  
SOURCE FILE: DELOACE  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

SYNOPSIS

|          |    |                            |
|----------|----|----------------------------|
| C        | -- | DELOACE(&EC_NO, &STATUS)   |
| COBOL    | -- | CALL "DELOACE" USING       |
|          |    | EC-NO,                     |
|          |    | STATUS.                    |
| FORTTRAN | -- | CALL DELOACE(ECNO, STATUS) |

INPUT:

INT \*EC\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
OWNED\_ATTRIBUTE. IF SUCCESSFULLY DELETED THE RECORD , THE  
STATUS

AND RETURN VALUE BOTH WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

|        |       |
|--------|-------|
| EC_NO  | INT * |
| STATUS | INT * |

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELOWAC  
PURPOSE: DELETE A RECORD FROM OWNED\_ATTRIBUTE  
LANGUAGE: C  
SOURCE FILE: DELOWAC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

WHERE AC\_NO =: "INPUT PARAMETER 1" ;

SYNOPSIS

C -- DELOWAC(&AC\_NO,&STATUS)  
COBOL -- CALL "DELOWAC" USING  
AC-NO,  
STATUS.  
FORTRAN -- CALL DELOWAC(ACNO,STATUS)

INPUT:

INT \*AC\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
OWNED\_ATTRIBUTE

WHERE AC\_NO =: "INPUT PARAMETER 1".

IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

AC\_NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELPCB  
PURPOSE: DELETE A RECORD FROM PSB\_PCB ENITY  
LANGUAGE: C  
SOURCE FILE: DELPCB  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C                   --    DELPCEB(&DB\_ID,&STATUS)  
  
COBOL               --    CALL "DELPCEB" USING  
                                DB-ID,  
                                STATUS.  
  
FORTRAN            --    CALL DELPCB(DBID,STATUS)

INPUT:

INT   \*DB\_ID ;

OUTPUT:

INT   \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
PSB\_PCB. IF SUCCESSFULLY DELETED THE RECORD , THE STATUS  
AND RETURN VALUE BOTH WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
DB\_ID                                   INT \*  
STATUS                                  INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELPSWD  
PURPOSE: DELETE A RECORD FROM DB\_PASSWORD ENTITY  
LANGUAGE: C  
SOURCE FILE: DELPSWD  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

          C                  --      DELPSWD(&DB\_ID,&STATUS)  
  
          COBOL              --      CALL "DELPSWD" USING  
                                  DB-ID,  
                                  STATUS.  
  
          FORTRAN          --      CALL DELPSWD(DBID,STATUS)

INPUT:

      INT \*DB\_ID ;

OUTPUT:

      INT \*STATUS ;

DESCRIPTION:

      THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
DB\_PASSWORD  
      IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
      WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
DB\_ID                          INT \*  
STATUS                         INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELRBRI  
PURPOSE: DELETE A RECORD FROM RC\_BASED\_REC\_SET ENITY  
LANGUAGE: C  
SOURCE FILE: DELRBRI  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

          C                  --  DELRBRI(&DB\_ID,&STATUS)  
  
          COBOL              --  CALL "DELRBRI" USING  
                                  DB-ID,  
                                  STATUS.  
  
          FORTRAN          --  CALL DELRBRI(DBID,STATUS)

INPUT:

          INT  \*DB\_ID ;

OUTPUT:

          INT  \*STATUS ;

DESCRIPTION:

          THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
          RC\_BASED\_REC\_SET. IF SUCCESSFULLY DELETED THE RECORD ,  
THE STATUS

          AND RETURN VALUE BOTH WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
DB\_ID                          INT \*  
STATUS                         INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: DELRBR2  
PURPOSE: DELETE A RECORD FROM RC\_BASED\_REC\_SET ENITY  
LANGUAGE: C  
SOURCE FILE: DELRBR2  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

SYNOPSIS

```
C          --  DELRBR2(&DB_ID, &SET_ID, &STATUS)

COBOL      --  CALL "DELRBR2" USING
                  DB-ID,
                  SET-ID,
                  STATUS.

FORTRAN    --  CALL DELRBR2(DBID, SETID, STATUS)
```

INPUT:

```
INT  *DB_ID ;
CHAR *SET_ID ;
```

OUTPUT:

```
INT *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
RC\_BASED\_REC\_SET  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

|        |        |
|--------|--------|
| DB_ID  | INT *  |
| SET_ID | CHAR * |
| STATUS | INT *  |

ROUTINES CALLED:

STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELRC  
PURPOSE: DELETE A RECORD FROM RELATION\_CLASS  
LANGUAGE: C  
SOURCE FILE: DELRC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
WHERE RC\_NO =: "INPUT PARAMETER 1" ;  
SYNOPSIS  
C -- DELRC(&RC\_NO,&STATUS)  
COBOL -- CALL "DELRC" USING  
RC-NO,  
STATUS.  
FORTRAN -- CALL DELRC(RCNO,STATUS)  
INPUT:  
INT \*RC\_NO ;  
OUTPUT:  
INT \*STATUS ;  
DESCRIPTION:  
THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
RELATION\_CLASS  
WHERE RC\_NO = "INPUT PARAMETER 1".  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
RC NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELRCKW  
PURPOSE: DELETE A RECORD FROM RC\_KEYWORD  
LANGUAGE: C  
SOURCE FILE: DELRCKW  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

WHERE RC\_NO =: 1 ;

SYNOPSIS

```
C          -- DELRCKW(&RC_NO,&STATUS)
COBOL      -- CALL "DELRCKW" USING
              RC-NO,
              STATUS.
FORTRAN    -- CALL DELRCKW(RCNO,STATUS)
```

INPUT:

INT \*RC\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
RC\_KEYWORD

WHERE RC\_NO = "INPUT PARAMETER 1".

IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

```
RC_NO          INT *
STATUS         INT *
```

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELRCST  
PURPOSE: DELETE RECORD CLASS FROM RC\_BASED\_REC\_SET  
LANGUAGE: C  
SOURCE FILE: DELRCST  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

WHERE RC\_NO =: "INPUT PARAMETER 1" ;  
SYNOPSIS  
C -- DELRCST( &AC\_NO, &STATUS)  
COBOL -- CALL "DELRCST" USING  
RC-NO,  
STATUS.  
FORTRAN -- CALL DELRCST(RCNO,STATUS)  
INPUT:  
INT \*RC\_NO ;  
OUTPUT:  
INT \*STATUS ;  
DESCRIPTION:  
THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
RC\_BASED\_REC\_SET  
WHERE RC NO EQUALS "INPUT PARAMETER 1" .  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

|        |       |
|--------|-------|
| RC_NO  | INT * |
| STATUS | INT * |

ROUTINES CALLED:  
-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELRST2  
PURPOSE: DELETE A RECORD FROM RECORD\_SET ENTITY  
LANGUAGE: C  
SOURCE FILE: DELRST2  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

          C                  --      DELRST2(&DB\_ID, SET\_ID, &STATUS)  
  
          COBOL              --      CALL "DELRST2" USING  
                                  DB-ID,  
                                  SET-ID,  
                                  STATUS.  
  
          FRTRAN          --      CALL DELRST2(DBID, SETID, STATUS)

INPUT:

          INT  \*DB\_ID ;  
          CHAR \*SET\_ID ;

OUTPUT:

          INT  \*STATUS ;

DESCRIPTION:

          THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
RECORD SET  
          IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
          WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

|        |        |
|--------|--------|
| DB_ID  | INT *  |
| SET_ID | CHAR * |
| STATUS | INT *  |

ROUTINES CALLED:

-----

STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELRST3  
PURPOSE: DELETE A RECORD FROM RECORD\_SET ENITY  
LANGUAGE: C  
SOURCE FILE: DELRST3  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

          C                  --  DELRST3(&DB\_ID, &SET\_ID, &RT\_ID\_OF\_OWNER,  
&STATUS)  
          COBOL              --  CALL "DELRST3" USING  
                              DB-ID,  
                              SET-ID,  
                              DF-ID,  
                              STATUS.

          FORTRAN          --  CALL DELRST3(DBID, SETID, DFID, STATUS)

INPUT:

          INT  \*DB\_ID ;  
          CHAR \*SET\_ID ;  
          CHAR \*RT\_ID\_OF\_OWNER ;

OUTPUT:

          INT \*STATUS ;

DESCRIPTION:

          THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
RECORD SET

          IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH

          WILL BE 0 , OTHERWISE -1.

          REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

|                |        |
|----------------|--------|
| DB_ID          | INT *  |
| SET_ID         | CHAR * |
| RT_ID_OF_OWNER | CHAR * |
| STATUS         | INT *  |

ROUTINES CALLED:

-----

STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELRTY2  
PURPOSE: DELETE A RECORD FROM RECORD\_TYPE ENITY  
LANGUAGE: C  
SOURCE FILE: DELRTY2  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

```
      C          --  DELRTY2(&DB_ID, &RT_ID, &STATUS)

      COBOL       --  CALL "DELRTY" USING
                        DB-ID,
                        RT-ID,
                        STATUS.

      FORTRAN     --  CALL DELRTY(DBID, RTID, STATUS)
```

INPUT:

```
      INT  *DB_ID ;
      CHAR *RT_ID ;
```

OUTPUT:

```
      INT *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
RECORD TYPE  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

```
-----
DB_ID          INT *
RT_ID          CHAR *
STATUS         INT *
```

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELSDF1  
PURPOSE: DELETE A RECORD FROM SEGMENT\_DATA\_FIELD ENITY  
LANGUAGE: C  
SOURCE FILE: DELSDF1  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C               --    DELSDF1(&DB\_ID,&STATUS)  
  
COBOL           --    CALL "DELSDF1" USING  
                          DB-ID,  
                          STATUS.  
  
FORTRAN       --    CALL DELSDF1(DBID,STATUS)

INPUT:

INT   \*DB\_ID ;

OUTPUT:

INT   \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
SEGMENT\_DATA\_FIELD. IF SUCCESSFULLY DELETED THE RECORD ,  
THE STATUS

AND RETURN VALUE BOTH WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

DB\_ID                           INT \*  
STATUS                         INT \*

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: DELSN1  
PURPOSE: DELETE A RECORD FROM SCHEMA\_NAMES ENITY  
LANGUAGE: C  
SOURCE FILE: DELSN1  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

SYNOPSIS

C -- DELSN1(&DB\_ID,&STATUS)  
  
COBOL -- CAL "DELSN1" USING  
DB-ID,  
STATUS.  
  
FORTRAN -- CALL DELSN1(DBID,STATUS)

INPUT:

INT \*DB\_ID ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
SCHEMA\_NAMES  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

|        |       |
|--------|-------|
| DB_ID  | INT * |
| STATUS | INT * |

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELSTM1  
PURPOSE: DELETE A RECORD FROM SET\_TYPE\_MEMBER ENITY  
LANGUAGE: C  
SOURCE FILE: DELSTM1  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

          C                  --  DELSTM1(&DB\_ID,&STATUS)  
  
          COBOL              --  CALL "DELSTM1" USING  
                                  DB-ID,  
                                  STATUS.  
  
          FORTRAN          --  CALL DELSTM1(DBID,STATUS)

INPUT:

      INT  \*DB\_ID ;

OUTPUT:

      INT  \*STATUS ;

DESCRIPTION:

      THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
      SET\_TYPE\_MEMBER. IF SUCCESSFULLY DELETED THE RECORD , THE  
STATUS  
      AND RETURN VALUE BOTH WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

DB\_ID                          INT \*  
STATUS                         INT \*

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELSTM2  
PURPOSE: DELETE A RECORD FROM SET\_TYPE\_MEMBER ENTITY  
LANGUAGE: C  
SOURCE FILE: DELSTM2  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

SYNOPSIS

C -- DELSTM2(&DB\_ID, SET\_ID, &STATUS)  
  
COBOL -- CALL "DELSTM2" USING  
                                DB-ID,  
                                SET-ID,  
                                STATUS.  
  
FRTRAN -- CALL DELSTM2(DBID, SETID, STATUS)

INPUT:

INT \*DB\_ID ;  
CHAR \*SET\_ID ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
SET\_TYPE\_MEMBER  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

|        |        |
|--------|--------|
| DB_ID  | INT *  |
| SET_ID | CHAR * |
| STATUS | INT *  |

ROUTINES CALLED:

-----

STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELSTM3  
PURPOSE: DELETE A RECORD FROM SET\_TYPE\_MEMBER ENITY  
LANGUAGE: C  
SOURCE FILE: DELSTM3  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C -- DELSTM3(&DB\_ID, &SET\_ID, &RT\_ID\_OF\_MEMBER,  
&STATUS)

COBOL -- CALL "DELSTM3" USING  
DB-ID,  
SET-ID,  
DF-ID,  
STATUS.

FORTRAN -- CALL DELSTM3(DBID, SETID, DFID, STATUS)

INPUT:

INT \*DB\_ID ;  
CHAR \*SET\_ID ;  
CHAR \*RT\_ID\_OF\_MEMBER ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
SET\_TYPE\_MEMBER

IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
DB\_ID INT \*  
SET\_ID CHAR \*  
RT\_ID\_OF\_MEMBER CHAR \*  
STATUS INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELTEXT  
PURPOSE: DELETE A RECORD FROM DESC\_TEXT ENITY  
LANGUAGE: C  
SOURCE FILE: DELTEXT  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

```
      C      --      DELTEXT(&OBJECT_TYPE, &OBJECT_NO, &STATUS)

      COBOL   --      CALL "DELTEXT" USING
                        OBJECT-TYPE,
                        OBJECT-NO,
                        STATUS.

      FORTRAN --      CALL DELTEXT(OBJECT-TYPE,OBJECT-NO,STATUS)
```

INPUT:

```
      CHAR *OBJECT_TYPE ;
      INT  *OBJECT_NO  ;
```

OUTPUT:

```
      INT *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
DESC\_TEXT  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

```
-----
OBJECT_TYPE      CHAR *
OBJECT_NO        INT  *
STATUS           INT  *
```

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DELTXT

PURPOSE: DELETE DESCRIPTION TEXT GIVEN THE OBJECT TYPE,

LANGUAGE: C

SOURCE FILE: DELTXT

SOURCE FILE TYPE: PC

HOST:

SUBSYSTEM: CDM

SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
DESCRIPTION TYPE AND THE OBJECT NUMBER.

SYNOPSIS

```
C      --  DELTXT (OBJ_TYPE, DESC_TYPE, &OBJ_NO, &STATUS)
COBOL  --  CALL  "DELTXT"  USING
                                OBJ-TYPE
                                DESC-TYPE
                                OBJ-NO
                                STATUS
```

```
FORTRAN  --  CALL  DELTXT
(OBJTYPE, DESC_TYP, OBJNO, STATUS)
```

INPUT: CHAR OBJ\_TYPE[]

CHAR DESC\_TYPE[]

INT \*OBJ\_NO

OUTPUT: INT \*STATUS

DESCRIPTION DELTXT DELETES ALL DESCRIPTION TEXT CORRESPONDING TO A PARTICULAR OBJECT TYPE, OBJECT NUMBER AND DESCRIPTION TYPE USING A SQL DELETE COMMAND. IF THE DELETE WAS SUCCESSFUL, STATUS IS ZERO AND THE VALUE OF THE FUNCTION IS ZERO. IF UNSUCCESSFUL, STATUS AND THE FUNCTION VALUE ARE SET TO -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI INTERFACE

ARGUMENTS:

```
-----
OBJ_TYPE          CHAR []
DESC_TYPE         CHAR []
OBJ_NO            INT *
STATUS            INT *
```

INCLUDE FILES:

-----  
CSTDTP

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DEPEND  
PURPOSE: SELECT ALL THE DEPENDANT ENTITY CLASSES  
LANGUAGE: C  
SOURCE FILE: DEPEND  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
IN THE STRUCTURE OF THE FROM-MODEL.

SYNOPSIS  
C -- DEPEND(&EC\_NO, &MODEL\_NAME, &KEYWORD\_FLAG,  
&ALIAS\_FLAG, &DESC\_FLAG,  
&ATTR\_FLAG, &DEOEBT\_FLAG, &NKEY\_FLAG,  
&MODEL\_NO, &FROM\_MODEL\_NO, &DEP\_EC\_LIST, EC\_NO)

; COBOL -- CALL "DEPEND" USING  
MODEL-NO,  
MODEL-NAME,  
KEYWORD-FLAG,  
ALIAS-FLAG,  
DESC-FLAG,  
ATTR-FLAG,  
DEPEND-FLAG,  
NKEY-FLAG,  
FROM-MODEL-NO,  
DEP-EC-LIST,  
EC-NO.

FORTRAN -- CALL DEPEND(MODELNO, MODELNAME,  
KEYWORDFLAG, ALIASFLAG,

DESCFLAG, ATTRFLAG, DEPENDFLAG, NKEYFLAG,

MODELNO, FROMMODELNO, DEPECLIST, ECNO)

INPUT:

INT \*EC\_NO ;  
CHAR \*MODEL\_NAME ;  
CHAR \*KEYWORD\_FLAG ;  
CHAR \*ALIAS\_FLAG ;  
CHAR \*DESC\_FLAG ;  
CHAR \*ATTR\_FLAG ;  
CHAR \*DEPEND\_FLAG ;  
CHAR \*NKEY\_FLAG ;  
INT \*MODEL\_NO ;  
INT \*FROM\_MODEL\_NO ;  
INT \*DEP\_EC\_LIST ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO RETRIEVE RECORDS FROM  
ATTRIBUTE\_CLASS, ATTRIBUTE\_NAME, DOMAIN\_CLASS, THEN USE  
THE INFORMATION TO GENERATE NDDL COMMAND STATEMENT.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

3/22/89: REWRITTEN TO TAKE OUT 'CONNECT BY PRIOR' TO MAKE  
STANDARD SQL.

THIS WILL EMULATE THE ORACLE SQL COMMAND:

```
SELECT B.EC_NAME, B.EC_NO
FROM ENTITY_NAME B
WHERE B.EC_NAME_TYPE = 'PRIMARY' AND
B.EC_NO IN
( SELECT UNIQUE DEP_EC_NO FROM LINK_RELATION
  CONNECT BY PRIOR DEP_EC_NO = IND_EC_NO
  START WITH IND_EC_NO = :EC_NO_SQL )
ORDER BY LEVEL, EC_NO;
```

REWRITTEN 8/1/89: TO HANDLE BOTH CATEGORY AND LINK RELATIONS,  
THE SELECT THE WENT THRU THE RELATION WAS REMOVED AND  
NOW THE EC-NO IS BEING SUPPLIED BY A COBOL DRIVER THAT  
GOES THRU A COBOL LIST THAT WAS CREATED WHEN TRAVERSING THRU  
BOTH CATEGORY AND LINK STRUCTURES.

ARGUMENTS:

-----

|               |        |
|---------------|--------|
| EC_NO         | INT *  |
| MODEL_NAME    | CHAR * |
| KEYWORD_FLAG  | CHAR * |
| ALIAS_FLAG    | CHAR * |
| DESC_FLAG     | CHAR * |
| ATTR_FLAG     | CHAR * |
| DEPENT_FLAG   | CHAR * |
| NKEY_FLAG     | CHAR * |
| MODEL_NO      | INT *  |
| FROM_MODEL_NO | INT *  |
| DEP_EC_LIST   | INT *  |

INCLUDE FILES:

-----

STDIO  
EOD

ROUTINES CALLED:

-----

STRNCPY  
ERRRPT  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLAD2  
SQLFCH  
FNDEC  
GENENT2





DOCGROUP PS41100 Module Documentation

NAME: DLDITAG  
PURPOSE: DELETE FROM AUC\_IS\_MAPPING A TAG THAT IS MAPPED  
LANGUAGE: C  
SOURCE FILE: DLDITAG  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
THRU A COMPLEX MAPPING ALGORITHM, FOR THE CASE,  
WHERE THE TAG MAPS TO A DATAITEM IN  
COMPLEX\_MAPPING\_PARM.

SYNOPSIS

          C                  -- DLDITAG(PREF-NO, TAG\_NO, STATUS)  
          COBOL              -- CALL "DLDITAG"          USING  
                                  PREF-NO  
                                  TAG\_NO  
                                  RET=STATUS.  
      FORTRAN          -- CALL DLDITAG (PREF-NO,TAG\_NO,STATUS)

INPUT:

      PREF-NO  
      TAG\_NO

OUTPUT:

      STATUS

DESCRIPTION

      THIS ROUTINE USES AN ORACLE SEQUEL CALL TO DELETE  
      FROM AUC\_IS\_MAPPING  
      IF SUCCESSFUL, THE STATUS WILL BE 0.  
      REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
      INTERFACE

ARGUMENTS:

-----  
PREF NO                  INT \*  
TAG NO                  INT \*  
STATUS                  INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLSQS  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DLDSL2  
PURPOSE: DELETE A RECORD FROM DF\_SET\_LINKAGE ENTITY  
LANGUAGE: C  
SOURCE FILE: DLDSL2  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C           --    DLDSL2(&DB\_ID, SET\_ID, &STATUS)  
  
COBOL       --    CALL "DLDSL2" USING  
                    DB-ID,  
                    SET-ID,  
                    STATUS.  
  
FRTRAN      --    CALL DLDSL2(DBID, SETID, STATUS)

INPUT:  
    INT  \*DB\_ID ;  
    CHAR \*SET\_ID ;

OUTPUT:  
    INT  \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
DF\_SET LINKAGE  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
DB\_ID                   INT \*  
SET\_ID                  CHAR \*  
STATUS                  INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DLMIGRC  
PURPOSE: DELETE MIGRATING KEY CLASS  
LANGUAGE: C  
SOURCE FILE: DLMIGRC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
ONLY THRU A SINGLE RELATION CLASS

SYNOPSIS

```
      C          -- DLMIGRC(&TAG_NO,&RC_NO)
      COBOL       -- CALL "DLMIGRC"          USING
                                          TAG_NO
                                          RC_NO
                                          STATUS.
```

```
      FORTRAN    -- CALL DLMIGRC (TAGNO,RCNO,STATUS)
```

INPUT:

```
      INT *TAG_NO
      INT *RC_NO
      INT *RTNSTA
```

OUTPUT:

DESCRIPTION

THIS ROUTINE USES AN ORACLE SELECT STATEMENT TO RETRIEVE ALL MIGRATING KEY CLASSES GIVEN A TAG NUMBER. ROUTINES ARE CALLED WHICH, FOR EACH NEW TAG NUMBER SELECTED, DELETE THE INHERITED ATTRIBUTE CLASS, ATTRIBUTE USE CLASS, COPY CROSS REFERENCE AND KEY CLASS MEMBER ENTITIES WHOSE TAG NUMBERS MATCH. THE QUERY IS PHRASED SUCH THAT ONLY THE ATTRIBUTES INHERITED THRU A SINGLE RELATION CLASS ARE DELETED, NOT EVERYWHERE THAT ATTRIBUTE IS MIGRATED TO. REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI INTERFACE  
3/23/89: REWRITTEN TO REPLACE 'CONNECT BY PRIOR' TO MAKE STANDARD SQL.

THIS REPLACED THE ORACLE SQL STATEMENT:

```
      SELECT UNIQUE TAG_NO
      FROM INHERITED_ATT_USE
      CONNECT BY PRIOR TAG_NO = KCM_TAG_NO
      START WITH KCM_TAG_NO = :TAG_NO_SQL AND RC_NO =
```

:RC\_NO\_SQL;

ARGUMENTS:

```
-----
TAG_NO          INT *
RC_NO           INT *
RTNSTA          INT *
```

INCLUDE FILES:

-----  
STDIO  
CSTDYTP  
UNIQENO  
EOD  
LLIST

ROUTINES CALLED:

-----  
SQLAB2  
SQLAD2  
SQLBS2  
SQLCLS  
DELIAUC  
SPRINTF  
UERROR  
ADDRNUM  
FRELST2  
ERRRPT  
DELCMT  
DELAUCL  
SQLEXE  
SQLFCH  
INSNODE  
SQLOSQ  
SQLSCA  
SQLSCC  
SQLSCH  
SQLTOC

DOCGROUP PS41100 Module Documentation

NAME: DLMTMAP  
PURPOSE: DELETE FROM AUC\_IS\_MAPPING A TAG THAT IS MAPPED  
LANGUAGE: C  
SOURCE FILE: DLMTMAP  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
THRU A COMPLEX MAPPING ALGORITHM, FOR THE CASE,  
WHERE THE TAG NO LONGER EXISTS IN  
COMPLEX\_MAPPING\_PARM.

SYNOPSIS

C -- DLMTMAP(PREF-NO, STATUS)  
COBOL -- CALL "DLMTMAP" USING  
PREF-NO  
RET-STATUS.  
FORTRAN -- CALL DLMTMAP (PREF-NO,STATUS)

INPUT:

PREF-NO

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO DELETE  
FROM AUC\_IS\_MAPPING  
IF SUCCESSFUL, THE STATUS WILL BE 0.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

PREF\_NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLSQS  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DLTPRFA  
PURPOSE: DELETE FROM AUC\_IS\_MAPPING A TAG THAT WAS MAPPED  
LANGUAGE: C  
SOURCE FILE: DLTPRFA  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
THRU A COMPLEX MAPPING ALGORITHM, AND THAT TAG  
WAS THE FIRST PREFERENCE MAPPING.  
IN ACCORDANCE WITH THE CURRENT PHILOSOPHY,  
WHEN THE FIRST PREFERENCE IS DELETED, ALL  
PREFERENCE MAPPINGS OF THE TAG MUST ALSO BE  
DELETED. THIS ROUTINE SEARCHES FOR AND DELETES SECONDARY  
MAPPINGS WHEN IT LOCATES A TAG WITHOUT ITS  
FIRST PREFERENCE MAPPING.

SYNOPSIS

C -- DLTPRFA( TAG\_NO, STATUS)  
COBOL -- CALL "DLTPRFA" USING  
TAG\_NO  
RET-STATUS.  
FORTRAN -- CALL DLTPRFA (TAG\_NO,STATUS)  
INPUT: TAG\_NO  
OUTPUT: STATUS  
DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO DELETE  
FROM AUC\_IS\_MAPPING  
IF SUCCESSFUL, THE STATUS WILL BE 0.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
TAG\_NO INT \*  
STATUS INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: DMIGKKC  
PURPOSE: DELETE MIGRATING KEY CLASS,  
LANGUAGE: C  
SOURCE FILE: DMIGKKC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
QUALIFYING ON KC\_NO AND TAG\_NO

SYNOPSIS

C -- DMIGKKC(&TAG\_NO,KC\_NO)  
COBOL -- CALL "DMIGKKC" USING  
TAG-NO  
KC-NO.  
FORTRAN -- CALL DMIGKKC (TAGNO, KC-NO)

INPUT:

INT \*TAG\_NO  
INT \*KC\_NO

OUTPUT:

INT \*STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SELECT STATEMENT TO RETRIEVE  
ALL

MIGRATING KEY CLASSES GIVEN A TAG NUMBER, AND KEY CLASS  
NUMBER, AS THE STARTING POINT. ROUTINES ARE CALLED  
WHICH, FOR EACH NEW TAG NUMBER SELECTED, DELETE THE

INHERITED

ATTRIBUTE CLASS, ATTRIBUTE USE CLASS AND  
KEY CLASS MEMBER ENTITIES WHOSE TAG NUMBERS MATCH.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

3/23/89: REWRITTEN TO TAKE OUT 'CONNECT BY PRIOR' TO MAKE  
STANDARD SQL.

THE ORACLE SQL STATEMENT REPLACED WAS:

SELECT UNIQUE TAG\_NO  
FROM INHERITED\_ATT USE  
CONNECT BY TAG\_NO = KCM\_TAG\_NO  
START WITH KCM\_TAG\_NO = :TAG\_NO\_SQL AND KC\_NO =  
:KC\_NO\_SQL;

ARGUMENTS:

-----  
TAG\_NO INT \*  
KC\_NO INT \*  
RTNSTA INT \*

INCLUDE FILES:

-----  
STDIO  
CSTDYTP  
UNIQUENO  
EOD  
LLIST



ROUTINES CALLED:

-----  
SQLAB2  
SQLAD2  
SQLBS2  
SQLCLS  
DELIAUC  
SPRINTF  
UERROR  
ADDRNUM  
FRELST2  
ERRRPT  
DELKCMT  
DELAUCL  
SQLEXE  
SQLFCH  
INSNODE  
SQLOSQ  
SQLSCA  
SQLSCC  
SQLSCH  
SQLTOC

DOCGROUP PS41100 Module Documentation

NAME: DMIGKRC  
PURPOSE: DELETE MIGRATING KEY CLASS,  
LANGUAGE: C  
SOURCE FILE: DMIGKRC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
QUALIFYING ON RC\_NO AND TAG\_NO

SYNOPSIS

|          |    |                                          |
|----------|----|------------------------------------------|
| C        | -- | DMIGKRC(&TAG_NO,RC_NO)                   |
| COBOL    | -- | CALL "DMIGKRC" USING<br>TAG-NO<br>RC-NO. |
| FORTTRAN | -- | CALL DMIGKRC (TAGNO, RC-NO)              |

INPUT:

INT \*TAG NO  
INT \*RC\_NO

OUTPUT:

INT \*STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SELECT STATEMENT TO RETRIEVE  
ALL  
MIGRATING KEY CLASSES GIVEN A TAG NUMBER, AND RELATION  
CLASS  
NUMBER, AS THE STARTING POINT. ROUTINES ARE CALLED

WHICH, FOR EACH NEW TAG NUMBER SELECTED, DELETE THE  
INHERITED

ATTRIBUTE CLASS, ATTRIBUTE USE CLASS AND  
KEY CLASS MEMBER ENTITIES WHOSE TAG NUMBERS MATCH.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

3/29/89: REWRITTEN TO REMOVE 'CONNECT BY SQL' TO MAKE  
STANDARD SQL.

THE ORACLE SQL STATEMENT REPLACED WAS:

```
'SELECT UNIQUE TAG NO
FROM INHERITED ATT USE
CONNECT BY PRIOR TAG NO = KCM TAG NO
START WITH KCM TAG NO = :TAG NO_SQL AND
RC_NO = :RC_NO_SQL'
```

ARGUMENTS:

```
-----
TAG NO          INT *
RC NO           INT *
RTNSTA         INT *
```

INCLUDE FILES:

```
-----
STDIO
CSTDYTP
UNIQENO
EOD
LLIST
```

ROUTINES CALLED:

```
-----
SQLAB2
SQLAD2
SQLBS2
SQLCLS
DELIAUC
SPRINTF
UERROR
ADDRNUM
FRELST2
ERRRPT
DELKCMT
DELAUCL
SQLEXE
SQLFCH
INSNODE
SQLOSQ
SQLSCA
SQLSCC
SQLSCH
SQLTOC
```

DOCGROUP PS41100 Module Documentation

NAME: DPKCLST  
PURPOSE: CREATE AN KEY\_CLASS\_LIST TABLE CONTAINING ALL THE  
ENTITY  
LANGUAGE: C  
SOURCE FILE: DPKCLST  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
CLASS MEMBERS IN THE STRUCTURE, THEIR  
KEY CLASS NOS AND NAMES AND TAG  
NUMBERS AND NAMES.

SYNOPSIS

```
C          -- DPKCLST(&MODEL_NO, &EC_NO, DEP_EC_LIST,  
DEPENT_FLAG) ;  
COBOL     -- CALL "DPKCLST"    USING  
                                         MODEL-NO  
                                         EC-NO  
                                         DEP-EC-LIST  
                                         DEPENT-FLAG.
```

```
FORTRAN   -- CALL  
DPKCLST(MODELNO, ECNO, DEPECLIST, DEPENTFLAG)  
INPUT:
```

```
INT  *MODEL_NO ;  
INT  *FROM_EC_NO ;  
INT  *DEP_EC_LIST ;  
CHAR *DEPENT_FLAG ;
```

DESCRIPTION:

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

3/16/89: REWRITTEN TO TAKE OUT 'CONNECT BY PRIOR' TO MAKE  
STANDARD SQL.

THIS WILL EMULATE THE ORACLE SQL COMMAND:

```
SELECT K.EC_NO,  
       K.KC_NO,  
       K.KC_NAME,  
       A.TAG_NO,  
       A.TAG_NAME  
FROM   KEY_CLASS K,  
       KEY_CLASS_MEMBER KCM,  
       ATTRIBUTE_USE CLASS A  
WHERE  K.KC_NO = KCM.KC_NO AND  
       KCM.TAG_NO = A.TAG_NO AND  
       K.EC_NO IN  
( SELECT UNIQUE DEP_EC_NO  
  FROM LINK_RELATION  
    CONNECT BY PRIOR DEP_EC_NO = IND_EC_NO  
    START WITH IND_EC_NO = :FROM_EC_NO SQL )
```

REWRITTEN 8/1/89: TO HANDLE CATEGORY AND LINK RELATIONS,  
REMOVED THE FIRST SELECT BEACUSE THIS IS NOW

CALLED  
A COBOL  
TRAVERSES DOWN

BY A COBOL PROG THAT SELECTS THE EC-NO FROM  
LIST THAT WAS BUILT PRIOR TO THIS THAT  
BOTH CATEGORY AND LINK STRUCTURES

ARGUMENTS:

-----  
MODEL NO  
FROM EC NO  
DEP EC LIST  
DEPENT\_FLAG

INT \*  
INT \*  
INT \*  
CHAR \*

INCLUDE FILES:

-----  
STDIO  
EOD

ROUTINES CALLED:

-----  
SQLAB2  
SQLAD2  
SQLBS2  
SQLCLS  
ERRRPT  
SQLEXE  
SQLFCH  
FNDEC  
AKCROW  
STRNCPY  
SQLOSQ  
SQLSCA  
SQLSCC  
SQLSCH  
SQLSQS  
SQLTOC

DOCGROUP PS41100 Module Documentation

NAME: DRPDF  
PURPOSE: DELETE A RECORD FROM DATA\_FIELD ENITY  
LANGUAGE: C  
SOURCE FILE: DRPDF  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

          C                  --      DRPDF(DF\_NO, &STATUS)  
  
          COBOL              --      CALL "DRPDF" USING  
                                  DF-NO,  
                                  STATUS.  
  
          FORTRAN          --      CALL DRPDF(DFNO, STATUS)

INPUT:

          INT \*DF\_NO;

OUTPUT:

          INT \*STATUS ;

DESCRIPTION:

          THIS ROUTINE USES ORACLE CALLS TO DELETE A RECORD FROM  
DATA\_FIELD

          IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH

          WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
DF\_NO                          INT \*  
STATUS                         INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: ECSTRUC  
PURPOSE: SELECT ALL DEPENDENT ENTITIES IN THE TREE STRUCTURE  
LANGUAGE: C  
SOURCE FILE: ECSTRUC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

EC-KC-KCM                   FOR A GIVEN ENTITY AND INSERT IT INTO THE  
                              UNBOUNDED C STRUCTURE BY CALLING BLECLST.

SYNOPSIS

```
      C           -- ECSTRUC(EC_NO,DEP_EC-LIST,DEPENT_FLAG)
      COBOL       -- CALL "ECSTRUC"            USING
                                              EC-NO
                                              DEP-EC-LIST
                                              DEPENT-FLAG
      FORTRAN     -- CALL ECSTRUC (ECNO,DEPECLIST,DEPENTFLAG)
```

INPUT:

```
          *EC_NO
          *DEP_EC_LIST
      CHAR *DEPENT_FLAG
```

OUTPUT:

DESCRIPTION

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

3/17/89: REWRITTEN TO TAKE OUT 'CONNECT BY PRIOR' TO MAKE  
STANDARD SQL

THIS ROUTINE WILL EMULATE THE ORACLE SQL STMT:

```
SELECT A.EC_NAME, A.EC_NO
FROM ENTITY_NAME A
WHERE A.EC_NAME_TYPE = 'PRIMARY' AND
A.EC_NO IN
```

```
( SELECT UNIQUE DEP_EC_NO FROM LINK_RELATION
CONNECT BY PRIOR DEP_EC_NO = IND_EC_NO
START WITH IND_EC_NO = :EC_NO_SQL)
ORDER BY A.EC_NO DESC;
```

REWRITTEN 8/1/89: TO HANDLE BOTH LINK AND CATEGORY RELATIONS,  
THIS NOW IS CALLED BY A COBOL DRIVER THAT  
SELECTS EC-NO  
THRU BOTH  
FROM A COBOL LIST BUILT AFTER TRAVERSING  
CATEGORY AND LINK STRUCTURES

ARGUMENTS:

-----

|             |        |
|-------------|--------|
| EC_NO       | INT *  |
| DEP_EC_LIST | INT *  |
| DEPENT_FLAG | CHAR * |

INCLUDE FILES:

-----

STDIO  
EOD

ROUTINES CALLED:

-----

SQLAB2  
SQLAD2  
SQLBS2  
SQLEXE  
SQLFCH  
FNDEC  
BLECLST  
STRNCPY  
ERRRPT  
SQLOPN  
SQLOSQ  
SQLSCA  
SQLSCC  
SQLSCH  
SQLTFL

DOCGROUP PS41100 Module Documentation

NAME: UWARN  
PURPOSE: ISSUE A WARNING MESSAGE TO THE USER  
LANGUAGE: C  
SOURCE FILE: ERRORS  
SOURCE FILE TYPE: INP  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: SFORMS

DESCRIPTION:

-----  
SYNOPSIS

|          |    |                                  |
|----------|----|----------------------------------|
| C        | -- | UWARN(MESG)                      |
| COBOL    | -- | CALL "UWARN" USING<br>MESG-DESC. |
| FORTTRAN | -- | CALL UWARN (MESG)                |

INPUT:           MESG    A CHARACTER STRING CONTAINING THE ERROR  
MESSAGE FOR THE USER. IF NOT NULL  
TERMINATED, THE 60TH CHARACTER WILL BE  
MADE A NULL. ( TRUNCATION)

OUTPUT:

DESCRIPTION

TRUNCATE THE MESSAGE TO 60 CHARACTERS AND DEPENDING ON  
THE MODE, ISSUE THE PROPER CALL TO HANDLE THE MESSAGE.  
FOR BATCH, USE THE PRINTF WHICH WILL GO TO STANDARD  
OUTPUT.  
ALSO, INCREMENT THE GLOBAL WARNING ERROR COUNT.

ARGUMENTS:

-----  
MESG                                   CHAR \*



PS 620341100  
30 September 1990

INCLUDE FILES:

-----  
STDIO  
NDDLEX  
CMDIDEX

ROUTINES CALLED:

-----  
CRTERR  
PMSGLS  
PRINTF

DOCGROUP PS41100 Module Documentation

NAME: UERROR  
PURPOSE: ISSUE A FATAL ERROR MSG, COMMAND IS ROLLED BACK  
LANGUAGE: C  
SOURCE FILE: ERRORS  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: SHARE

DESCRIPTION:

-----

SYNOPSIS

|          |    |                                   |
|----------|----|-----------------------------------|
| C        | -- | UERROR(MESG)                      |
| COBOL    | -- | CALL "UERROR" USING<br>MESG-DESC. |
| FORTTRAN | -- | CALL UERROR (MESG)                |

INPUT:

|      |                                                                                                                                                      |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| MESG | A CHARACTER STRING CONTAINING THE ERROR<br>MESSAGE FOR THE USER. IF NOT NULL<br>TERMINATED, THE 60TH CHARACTER WILL BE<br>MADE A NULL. ( TRUNCATION) |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------|

OUTPUT:

DESCRIPTION

TRUNCATE THE MESSAGE TO 60 CHARACTERS AND DEPENDING ON  
THE MODE, ISSUE THE PROPER CALL TO HANDLE THE MESSAGE.  
FOR BATCH, USE THE PRINTF WHICH WILL GO TO STANDARD  
OUTPUT.

ALSO, INCREMENT THE GLOBAL WARNING ERROR COUNT.

ISSUE A MESSAGE TO THE USER, CONSIDERED A  
ERROR BY THE CALLER. NOTE THESE TYPES  
OF ERRORS ARE ENOUGH TO ROLL BACK THE  
EFFECTS OF THE COMMAND.

ARGUMENTS:

-----

MESG

CHAR \*

INCLUDE FILES:

-----

STDIO

NDDLEX

CMDIDEX

ROUTINES CALLED:

-----

CRERR

PMSGSL

PRINTF

DOCGROUP PS41100 Module Documentation

NAME: ERRRPT  
PURPOSE: HANDLE ANY ERROR CODE FROM ORACLE  
LANGUAGE: C  
SOURCE FILE: ERRORS  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: SHARE

DESCRIPTION:

-----

SYNOPSIS

|          |    |                                                                       |
|----------|----|-----------------------------------------------------------------------|
| C        | -- | ERRRPT( RETCODE, MESSG, MOD_NAME)                                     |
| COBOL    | -- | CALL "ERRRPT" USING<br>RET-CODE,<br>MESSG-DESC,<br>MODULE-NAME.       |
| FORTTRAN | -- | CALL ERRRPT(RETCOD, MESSG, MODNAM)                                    |
| INPUT:   |    |                                                                       |
| RETCODE  | -- | THE ORACLE ERROR CODE.                                                |
| MESSG    |    | A CHARACTER STRING CONTAINING THE ERROR<br>MESSAGE FOR THE ERROR LOG. |
| MOD_NAME | -- | THE NAME OF THE MODULE EXPERIENCING<br>THE ORACLE ERROR.              |

OUTPUT:

DESCRIPTION

COMBINE THE RETURN CODE, THE MESSAGE, AND THE  
MODULE NAME INTO A SINGLE MESSAGE AND  
USE "ERRPRO" TO PUT IT INTO THE ERROR LOG.  
THEN CALL THE ORACLE ROUTINE OERMSG TO  
TO GET THE EXPANDED ERROR MESSAGE.  
AND PLACE THE ORACLE ERROR MESSAGE ON THE LOG.

ARGUMENTS:

-----

RETCODE  
MMSG  
MOD\_NAME

INT  
CHAR \*  
CHAR \*

INCLUDE FILES:

-----

STDIO  
NDDLEX  
CMDIDEX

ROUTINES CALLED:

-----

ERRPRO  
SPRINTF

DOCGROUP PS41100 Module Documentation

NAME: FREELST  
PURPOSE:  
LANGUAGE: C  
SOURCE FILE: FREELST  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

ARGUMENTS:

-----

DEL\_NODE

ECUSED \*

INCLUDE FILES:

-----

STDIO  
LLIST

ROUTINES CALLED:

-----

FREELST  
FREE



DOCGROUP PS41100 Module Documentation

NAME: FRELST2  
PURPOSE: FREE MEMORY USED BY LINKED LIST.  
LANGUAGE: C  
SOURCE FILE: FRELST2  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

INPUT:

LLIST2 \*DEL\_NODE;

CALLED BY:

DLMIGRC, DMIGKRC, DMIGKRC, DPKCLST, DPKCLS1

DESCRIPTION:

THIS ROUTINE WILL FREE UP MEMORY USED BY LISTS CONTAINING  
INFORMATION RETREIVED FROM THE DATABASE.

ARGUMENTS:

-----

DEL\_NODE

LLIST2 \*

INCLUDE FILES:

-----

LLIST  
STDIO

ROUTINES CALLED:

-----



PS 620341100  
30 September 1990

FRELST2  
FREE

DOCGROUP PS41100 Module Documentation

NAME: GETECNM  
PURPOSE: THIS ROUTINE SEARCHES THE UNBOUNDED EC\_LIST DATA  
LANGUAGE: C  
SOURCE FILE: GETECNM  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
AND RETURNS                   STRUCTURE FOR A MATCHING ROW GIVEN EC\_NO  
                                  EC\_NAME.

SYNOPSIS

      C                   --    GETECNM(&EC\_NO, &EC\_NAME, &STATUS)  
      COBOL               --    CALL "GETECNM"    USING  
                                  EC-NO,  
                                  EC-NAME,  
                                  STATUS.  
      FORTRAN           --    CALL GETECNM(ECNO, ECNAME, STATUS)

INPUT:

      INT   \*EC\_NO ;  
      CHAR   \*EC\_NAME ;  
      INT   \*STATUS ;

OUTPUT:

DESCRIPTION:

ARGUMENTS:

-----

|         |        |
|---------|--------|
| EC_NO   | INT *  |
| EC_NAME | CHAR * |
| STATUS  | INT *  |

PS 620341100  
30 September 1990

INCLUDE FILES:

-----  
KEYLIST

ROUTINES CALLED:

-----  
STRNCPY

DOCGROUP PS41100 Module Documentation

NAME: GETGLOB  
PURPOSE: WILL PROVIDE GLOBAL VARIABLES  
LANGUAGE: C  
SOURCE FILE: GETGLOB  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

```
      C      --  GETGLOB(DB_ID, DB_NAME, HOST_NAME, DBMS_NAME,
STATUS)
      COBOL   --  CALL "GETGLOB"      USING
                                DB_ID,
                                DB_NAME,
                                HOST_NAME,
                                DBMS_NAME,
                                STATUS.

      FORTRAN --  CALL
GETGLOB(DB_ID,DB_NAME,HOST_NAME,DBMS_NAME,STATUS)
INPUT:
OUTPUT:
      DB_ID,
      DB_NAME,
      HOST_NAME
      DBMS_NAME,
      STATUS.
```

DESCRIPTION

THIS ROUTINE WILL GET THE FOLLOWING 4 GLOBAL VARIABLE

1. CUR\_DBID (CURRENT DATABASE ID)
2. CUR\_DBNAME (CURRENT DATABASE NAME)
3. CUR\_HOST (CURRENT HOST NAME )
4. CUR\_DBMS (CURRENT DATABASE MANAGEMENT)

PS 620341100  
30 September 1990

IF ALL 4 ABOVE GLOBAL VARIABLES ARE DEFINED, THEN  
THE RETURN STATUS IS 0, OTHERWISE -1

ARGUMENTS:

|           |        |
|-----------|--------|
| DB_ID     | INT *  |
| DB_NAME   | CHAR * |
| HOST_NAME | CHAR * |
| DBMS_NAME | CHAR * |
| STATUS    | INT *  |

ROUTINES CALLED:

STRNCPY

DOCGROUP PS41100 Module Documentation

NAME: GETNCHR  
PURPOSE: GET THE NEXT CHARACTER FROM EITHER STANDARD INPUT  
LANGUAGE: C  
SOURCE FILE: GETNCHR  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

OR FORMS COMMAND BUFFER DEPENDING ON MODE

SYNOPSIS

|          |    |                         |                       |
|----------|----|-------------------------|-----------------------|
| C        | -- | GETNCHR(&CHR, &FLAG)    |                       |
| COBOL    | -- | CALL "GETNCHR"          | USING<br>CHR<br>FLAG. |
| FORTTRAN | -- | CALL GETNCHR(CHR, FLAG) |                       |

INPUT:

OUTPUT:

CHAR \*CHR  
INT \*FLAG

DESCRIPTION

GETNCHR GETS THE NEXT CHARACTER FROM THE INPUT SOURCE  
DEPENDING  
ON WHETHER IN BATCH MODE (STANDARD INPUT) OR INTERACTIVE  
OR UIMS  
MODES (FORMS COMMAND BUFFER). IF OK, FLAG = 0, IF NEWLINE  
CHARACTER, FLAG = 1, IF TAB CHARACTER, FLAG = 2 AND IF EOF,  
FLAG = 3. ALSO, IF EOF ENCOUNTERED, THE GLOBAL SKIP\_FLAG  
IS SET  
TO EOF.  
MODIFICATION - 02/09/89 - CHANGED CHR TO BE CHAR INSTEAD OF  
INT.

PS 620341100  
30 September 1990

ARGUMENTS:

-----  
CHR  
FLAG

CHAR \*  
INT \*

INCLUDE FILES:

-----  
CSTDYTP  
STDIO  
NDDLEX  
FORMSEX

ROUTINES CALLED:

-----  
GETCHAR

DOCGROUP PS41100 Module Documentation

NAME: GETNNUM  
PURPOSE:  
LANGUAGE: C  
SOURCE FILE: GETNNUM  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C                   --     GETNNUM (POOL\_NO, NEXT\_NO)  
  
COBOL               --     CALL "GETNNUM" USING  
                              POOL-NO,  
                              NEXT\_NO.  
FORTRAN            --     CALL GETNNUM (POOLNO, NEXT\_NO)

INPUT:

INT     \*POOL\_NO ;

OUTPUT:

INT     \*NEXT\_NO ;

NEXT\_NO WILL BE A ZERO IF ANY TROUBLES WERE FOUND

DESCRIPTION

THIS ROUTINE RETURNS A UNIQUE NUMBER FOR A POOL NUMBER

ARGUMENTS:

-----  
POOL\_NO                               INT \*  
NEXT\_NO                               INT \*



PS 620341100  
30 September 1990

ROUTINES CALLED:

-----  
GETNXNO  
ERRPRO  
SPRINTF  
UERROR

DOCGROUP PS41100 Module Documentation

NAME: GETNXNO  
PURPOSE: GET A NUMBER THE REUSABLE POOL  
LANGUAGE: C  
SOURCE FILE: GETNXNO  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

```
      C          --  GETNXNO (POOL_NO, NEXT_NO, STATUS)

      COBOL       --  CALL "GETNXNO"          USING
                                POOL-NO,
                                NEXT_NO,
                                STATUS.

      FORTRAN     --  CALL GETNXNO (POOLNO,NEXT_NO,STATUS)
```

INPUT:

INT \*POOL\_NO ;

OUTPUT:

INT \*NEXT\_NO ;

INT \*STATUS ;

DESCRIPTION

THIS ROUTINE GET A UNIQUE NUMBER OF A POOL NUMBER FROM  
NEXT\_NUMBER  
RELATION

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
POOL\_NO  
NEXT\_NO

INT \*  
INT \*

PS 620341100  
30 September 1990

STATUS

INT \*

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLCLS  
UPDNXNO  
ERRRPT  
SQLAD2  
SQLFCH  
SQLTOC  
SQLOSQ  
SQLAB2  
SQLEXE

DOCGROUP PS41100 Module Documentation

NAME: HALT  
PURPOSE: HALT WITH 'COMMIT' OR 'ROLLBACK'.  
LANGUAGE: C  
SOURCE FILE: HALT  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

SYNOPSIS

```
      C          --  HALT(&STATUS)  ;  
  
      COBOL      --  CALL "HALT"    USING  
                      STATUS.  
  
      FORTRAN    --  CALL HALT (STATUS)
```

INPUT:

NONE

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE PROCESSES THE 'HALT' COMMAND WITH 'COMMIT'  
OR 'ROLLBACK'  
IF SUCCESSFULLY DELETED THE RECORD , THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.

ARGUMENTS:

-----

STATUS INT \*

INCLUDE FILES:

-----

PS 6203411~  
30 September 19

LISTID  
NDDLEX

ROUTINES CALLED:

-----  
CPFONE  
ROLBACK  
COMMIT

DOCGROUP PS41100 Module Documentation

NAME: INITSES  
PURPOSE: PERFORM ANY SESSION INITIALIZATION NECESSARY  
LANGUAGE: C  
SOURCE FILE: INITSES  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

SYNOPSIS

C        --        INITSES()

COBOL    --        CALL "INITSES".

FORTRAN --        CALL INITSES

INPUT:

OUTPUT:

DESCRIPTION

CURRENTLY DEFINED MODE IS BATCH:

DO :

    THE ORACLE LOGON (VIA VERTYP),  
    SET AUTOCOMMIT OFF.

INCLUDE FILES:

-----

FPPARM

FPCODE

OK

NDDLEX

PS 620341100  
30 September 1990

ROUTINES CALLED:

-----  
HOOK FORM  
STRNC̄PY  
VERTYP  
STRNCMP  
SPRINTF  
UERROR  
ERRPRO

DOCGROUP PS41100 Module Documentation

NAME: HOOK\_FORM  
PURPOSE:  
LANGUAGE: C  
SOURCE FILE: INITSES  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

INCLUDE FILES:

-----  
FPPARM  
FPCODE  
OK  
NDDLEX

ROUTINES CALLED:

-----  
INITFP  
OPNFRM  
STRNCMP  
SPRINTF  
UERROR  
ERRPRO  
ADDFRM



DOCGROUP PS41100 Module Documentation

NAME: INSAC  
PURPOSE: INSERT A RECORD INTO ATTRIBUTE\_CLASS  
LANGUAGE: C  
SOURCE FILE: INSAC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
VALUES (AC\_NO,MODEL\_NO,DOMAIN\_NO) ;

SYNOPSIS  
C -- INSAC(&AC\_NO, &MODEL\_NO, &DOMAIN\_NO,  
&STATUS)  
COBOL -- CALL "INSAC" USING  
AC-NO,  
MODEL-NO,  
DOMAIN-NO,  
STATUS.  
FORTRAN -- CALL INSAC(ACNO, MODELNO, DOMAINNO, STATUS)

INPUT:  
INT \*AC\_NO ;  
INT \*MODEL\_NO ;  
INT \*DOMAIN\_NO ;

OUTPUT:  
INT \*STATUS ;

DESCRIPTION:  
THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
ATTRIBUTE NAME  
THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.  
IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

|           |       |
|-----------|-------|
| AC NO     | INT * |
| MODEL NO  | INT * |
| DOMAIN NO | INT * |
| STATUS    | INT * |

ROUTINES CALLED:

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSACNM  
PURPOSE: INSERT A RECORD INTO ATTRIBUTE\_NAME  
LANGUAGE: C  
SOURCE FILE: INSACNM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
VALUES (AC\_NO,AC\_NO,AC\_NAME\_TYPE) ;

SYNOPSIS  
C -- INSACNM(&AC\_NAME, &AC\_NO, &AC\_NAME\_TYPE,  
&STATUS)  
COBOL -- CALL "INSACNM" USING  
AC-NAME,  
AC-NO,  
AC-NAME-TYPE,  
STATUS.  
FORTRAN -- CALL INSACNM(ACNAME,ACNO,ACNAMETYPE,  
STATUS)  
INPUT:  
INT \*AC\_NAME ;  
INT \*AC\_NO ;  
INT \*AC\_NAME\_TYPE ;  
OUTPUT:  
INT \*STATUS ;

DESCRIPTION:  
THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
ATTRIBUTE\_NAME  
THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.  
IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
AC\_NAME  
AC\_NO  
AC\_NAME\_TYPE  
STATUS

CHAR \*  
INT \*  
CHAR \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSAUC  
PURPOSE: INSERT A RECORD INTO ATTRIBUTE\_USE\_CL  
LANGUAGE: C  
SOURCE FILE: INSAUC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
VALUES (TAG\_NO,TAG\_NAME,EC\_NO,AC\_NO)

SYNOPSIS

C           --    INSAUC(&TAG\_NO, TAG\_NAME, &EC\_NO, &AC\_NO,  
&STATUS)

COBOL       --    CALL "INSAUC"    USING  
                                  TAG-NO,  
                                  TAG-NAME,  
                                  EC-NO,  
                                  AC-NO,  
                                  STATUS.

FORTTRAN    --    CALL INSAUC(TAGNO, TAGNAME, ECNO, ACNO,  
STATUS)

INPUT:

INT   \*TAG\_NO ;  
CHAR \*TAG\_NAME ;  
INT   \*EC\_NO ;  
INT   \*AC\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
ATTRIBUTE\_USE\_CL, THE DATA ARE TAKEN FROM THE INPUT  
PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

|          |        |
|----------|--------|
| TAG_NO   | INT *  |
| TAG_NAME | CHAR * |
| EC_NO    | INT *  |
| AC_NO    | INT *  |
| STATUS   | INT *  |

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSCATM  
PURPOSE: INSERT A RECORD INTO CATEGORY\_MEMBER  
LANGUAGE: C  
SOURCE FILE: INSCATM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
VALUES  
(RC\_NO, CAT\_EC\_NO, DOM\_NO, SPECIFIC\_VALUE)

SYNOPSIS

```
C          -- INSCATM(&RC_NO, &CAT_EC_NO, &DOM_NO,  
              SPECIFIC_VALUE, &STATUS)  
COEOL      -- CALL "INSCATM" USING  
              RC-NO,  
              CAT-EC-NO,  
              DOM-NO,  
              SPECIFIC_VALUE,  
              STATUS.
```

```
FORTTRAN  -- CALL  
INSCATM(RCNO, CATECNO, DOMNO, SPECIFICVALUE, STATUS)
```

INPUT:

```
INT  *RC_NO ;  
INT  *CAT_EC_NO ;  
INT  *DOM_NO ;  
INT  *SPECIFIC_VALUE ;
```

OUTPUT:

```
INT  *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
CATEGORY\_MEMBER.

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

|                |        |
|----------------|--------|
| RC_NO          | INT *  |
| CAT_EC_NO      | INT *  |
| DOM_NO         | INT *  |
| SPECIFIC_VALUE | CHAR * |
| STATUS         | INT *  |

ROUTINES CALLED:

-----

STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: INSCATR  
PURPOSE: INSERT A RECORD INTO CATEGORY\_RELATION  
LANGUAGE: C  
SOURCE FILE: INSCATR  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

VALUES  
(RC\_NO, RC\_NAME, GEN\_EC\_NO, CAT\_TYPE\_CODE, TAG\_NO)

SYNOPSIS

C -- INSCATR(&RC\_NO, RC\_NAME, &GEN\_EC\_NO,  
CAT\_TYPE\_CODE,

COBOL -- CALL "INSCATR" USING  
RC-NO,  
RC-NAME  
GEN-EC-NO,  
CAT-TYPE-CODE,  
TAG-NO,  
STATUS.

FORTRAN -- CALL  
INSCATR(RCNO, RCNAME, GENEENO, CATTYPECODE,  
TAGNO, STATUS)

INPUT:

INT \*RC\_NO ;  
INT \*RC\_NAME ;  
INT \*GEN\_EC\_NO ;  
INT \*CAT\_TYPE\_CODE ;  
INT \*DESC\_TAG\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
CATEGORY\_RELATION.  
THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.  
IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

|               |        |
|---------------|--------|
| RC_NO         | INT *  |
| RC_NAME       | CHAR * |
| GEN_EC_NO     | INT *  |
| CAT_TYPE_CODE | CHAR * |
| DESC_TAG_NO   | INT *  |
| STATUS        | INT *  |

ROUTINES CALLED:

-----  
STRLEN  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSCRC  
PURPOSE: INSERT A RECORD INTO COMPLETE\_RELATION  
LANGUAGE: C  
SOURCE FILE: INSCRC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

VALUES (RC\_NO,KC\_NO) ;

SYNOPSIS

```
C          -- INSCRC(&RC_NO, &KC_NO, &STATUS)
COBOL      -- CALL "INSCRC" USING
              RC-NO,
              KC-NO,
              STATUS.
FORTRAN    -- CALL INSCRC(RCNO, KCNO, STATUS)
```

INPUT:

```
INT  *RC_NO ;
INT  *KC_NO ;
```

OUTPUT:

```
INT  *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
COMPLETE\_RELATION. THE DATA ARE TAKEN FROM THE INPUT  
PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

|        |       |
|--------|-------|
| RC_NO  | INT * |
| KC_NO  | INT * |
| STATUS | INT * |

ROUTINES CALLED:

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSDSL  
PURPOSE: INSERT A RECORD INTO THE DF\_SET\_LINKAGE ENTITY. IF  
LANGUAGE: C  
SOURCE FILE: INSDSL  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
RETURN CODE TO SUCCESSFUL, THEN SET THE STATUS AND THE  
TO 0, OTHERWISE TO -1.

SYNOPSIS

C --  
INSDSL(&DB\_ID,&SET\_ID,&RT\_ID,&DF\_ID,&LINKAGE\_TYPE,&STATU  
COBOL -- CALL "INDSLET" USING  
DB-ID,  
SET-ID,  
RT-ID,  
LINKAGE-ID,  
STATUS.

FORTRAN -- CALL INSDSL(DBID, SETID, RTID,  
DFID, LINKAGETYPE, STATUS)

INPUT:

INT \*DB\_ID ;  
CHAR \*SET\_ID ;  
CHAR \*RT\_ID ;  
CHAR \*DF\_ID ;  
CHAR \*LINKAGE\_TYPE ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
DF\_SET\_LINKAGE  
ENTITY. IF SUCCESSFULLY INSERT THE RECORD , THE STATUS  
AND RETURN

VALUE BOTH WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

|              |        |
|--------------|--------|
| DB_ID        | INT *  |
| SET_ID       | CHAR * |
| RT_ID        | CHAR * |
| DF_ID        | CHAR * |
| LINKAGE_TYPE | CHAR * |
| STATUS       | INT *  |

ROUTINES CALLED:

-----

STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSEC  
PURPOSE: INSERT A RECORD INTO ENTITY\_CLASS  
LANGUAGE: C  
SOURCE FILE: INSEC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
VALUES (EC\_NO,MODEL\_NO) ;  
  
SYNOPSIS  
C -- INSEC(&EC\_NO, &MODEL\_NO, &STATUS)  
COBOL -- CALL "INSEC" USING  
EC-NO,  
MODEL-NO,  
STATUS.  
FORTRAN -- CALL INSEC(ECNO, MODELNO, STATUS)

INPUT:

INT \*EC\_NO ;  
INT \*MODEL\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
ENTITY\_CLASS

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

EC\_NO

INT \*

MODEL\_NO

INT \*

STATUS

INT \*

ROUTINES CALLED:

-----

SQLSCA

SQLBS2

SQLSCH

SQLSCC

SQLTFL

SQLOPN

SQLSQ

SQLAB2

SQLXEXE

ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: INSECNM  
PURPOSE: INSERT A RECORD INTO ENTIYT\_NAME  
LANGUAGE: C  
SOURCE FILE: INSECNM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

VALUES (EC\_NAME,EC\_NO,EC\_NAME\_TYPE)

SYNOPSIS

```
      C          -- INSECNM(EC_NAME, &EC_NO, EC_NAME_TYPE,
&STATUS)
      COBOL      -- CALL "INSECNM" USING
                  EC-NAME,
                  EC-NO,
                  EC-NAME-TYPE,
                  STATUS.
```

```
      FORTRAN   -- CALL
NSECNM(EC-NAME,EC-NO,EC-NAME-TYPE,STATUS)
```

INPUT:

```
      CHAR *EC_NAME ;
      INT  *EC_NO ;
      INT  *EC_NAME_TYPE ;
```

OUTPUT:

```
      INT *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
ENTITY NAME

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
EC\_NAME  
EC\_NO  
EC\_NAME\_TYPE  
STATUS

CHAR \*  
INT \*  
CHAR \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSIAUC  
PURPOSE: INSERT A RECORD INTO INHERITED\_ATT\_USE  
LANGUAGE: C  
SOURCE FILE: INSIAUC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

VALUES (TAG\_NO, KC\_NO, KCM\_TAG\_NO, RC\_NO)

SYNOPSIS

```
      C      --  
INSIAUC(&TAG_NO, &KC_NO, &KCM_TAG_NO, &RC_NO, &STATUS);  
      COBOL      --      CALL "INSIAUC" USING  
                                TAG-NO,  
                                KC-NO,  
                                KCM-TAG-NO,  
                                RC-NO,  
                                STATUS.
```

```
      FORTRAN      --      CALL  
INSIAUC(TAGNO, KCNO, KCMTAGNO, RCNO, STATUS)  
INPUT:
```

```
      INT      *TAG_NO ;  
      INT      *KC_NO ;  
      INT      *KCM_TAG_NO ;  
      INT      *RC_NO ;
```

OUTPUT:

```
      INT      *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
INHERITED ATT USE.  
THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.  
IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

|            |       |
|------------|-------|
| TAG_NO     | INT * |
| KC_NO      | INT * |
| KCM_TAG_NO | INT * |
| RC_NO      | INT * |
| STATUS     | INT * |

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSKCM  
PURPOSE: INSERT A RECORD INTO KEY\_CLASS\_MEMBER  
LANGUAGE: C  
SOURCE FILE: INSKCM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
VALUES (KC\_NO, TAG\_NO)

SYNOPSIS

|          |    |                                                     |
|----------|----|-----------------------------------------------------|
| C        | -- | INSKCM(&KC_NO, &TAG_NO, &STATUS)                    |
| COBOL    | -- | CALL "INSKCM" USING<br>KC-NO,<br>TAG-NO,<br>STATUS. |
| FORTTRAN | -- | CALL INSKCM(KCNO, TAGNO, STATUS)                    |

INPUT:

INT \*KC\_NO ;  
INT \*TAG\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
KEY\_CLASS\_MEMBER.  
THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.  
IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
KC NO  
TAG NO  
STATUS

INT \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSKW  
PURPOSE: INSERT A RECORD INTO KEYWORD  
LANGUAGE: C  
SOURCE FILE: INSKW  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

VALUES (KW\_NO,KEYWORD) ;

SYNOPSIS

```
      C          -- INSKW(&KW_NO, &KEYWORD, &STATUS)
      COBOL       -- CALL "INSKW"      USING
                                KW-NO,
                                KEYWORD,
                                STATUS.
      FORTRAN     -- CALL INSKW(KWNO, KEYWORD, STATUS)
```

INPUT:

```
      INT *KW_NO ;
      CHAR *KEYWORD ;
```

OUTPUT:

```
      INT *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
KEYWORD

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
KW\_NO  
KEYWORD  
STATUS

INT \*  
CHAR \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: INSKWAC  
PURPOSE: INSERT A RECORD INTO AC\_KEYWORD  
LANGUAGE: C  
SOURCE FILE: INSKWAC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

VALUES (KW\_NO, AC\_NO) ;

SYNOPSIS

```
C          -- INSKWAC(&KW_NO, &AC_NO, &STATUS)
COBOL      -- CALL "INSKWAC" USING
              KW-NO,
              AC-NO,
              STATUS.
```

```
FORTRAN    -- CALL INSKWAC(KWNO, ACNO, STATUS)
```

INPUT:

```
INT  *KW_NO ;
INT  *AC_NO ;
```

OUTPUT:

```
INT  *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
AC\_KEYWORD

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
KW\_NO  
AC\_NO  
STATUS

INT \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSKWEC  
PURPOSE: INSERT A RECORD INTO EC\_KEYWORD  
LANGUAGE: C  
SOURCE FILE: INSKWEC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

VALUES (KW\_NO, EC\_NO) ;

SYNOPSIS

```
C          -- INSKWEC(&KW_NO, &EC_NO, &STATUS)
COBOL      -- CALL "INSKWEC" USING
                        KW-NO,
                        AC-NO,
                        STATUS.
FORTRAN    -- CALL INSKWEC(KWNO, ACNO, STATUS)
```

INPUT:

```
INT  *KW_NO ;
INT  *EC_NO ;
```

OUTPUT:

```
INT  *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
EC\_KEYWORD

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
KW\_NO  
EC\_NO  
STATUS

INT \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSKWRC  
PURPOSE: INSERT A RECORD INTO RC\_KEYWORD  
LANGUAGE: C  
SOURCE FILE: INSKWRC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

VALUES (KW\_NO, RC\_NO) ;

SYNOPSIS

```
C          -- INSKWRC(&KW_NO, &RC_NO, &STATUS)
COBOL      -- CALL "INSKWRC" USING
              KW-NO,
              RC-NO,
              STATUS.
FORTRAN    -- CALL INSKWRC(KWNO, RCNO, STATUS)
```

INPUT:

```
INT  *KW_NO ;
INT  *RC_NO ;
```

OUTPUT:

```
INT  *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
RC\_KEYWORD

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.  
IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
KW\_NO  
RC\_NO  
STATUS

INT \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSLNKR  
PURPOSE: INSERT A RECORD INTO LINK\_RELATION  
LANGUAGE: C  
SOURCE FILE: INSLNKR  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
VALUES  
(RC\_NO, RC\_NAME, IND\_EC\_NO, DEP\_EC\_NO,  
NO\_INTD\_ENT, MIN\_NO\_DEP\_ENT, MAX\_NO\_DEP\_ENT)

SYNOPSIS

```
C          -- INSLNKR(&RC_NO, RC_NAME, &IND_EC_NO,
&DEP_EC_NO,          &NO_IND_ENT, &MIN_NO_DEP_ENT,
                      &MAX_NO_DEP_ENT, &STATUS)
COBOL      -- CALL "INSLNKR" USING
                      RC-NO,
                      RC-NAME,
                      IND-EC-NO,
                      DEP-EC-NO,
                      NO-IND-ENT,
                      MIN-NO-DEP-ENT,
                      MAX-NO-DEP-ENT,
                      STATUS.

FORTRAN    -- CALL
INSLNKR(RCNO, RCNAME, INDECNO, DEPECNO, NOINDENT,
                      MINDEPENT, MAXDEPENT, STATUS)
```

INPUT:

```
INT  *RC_NO ;
INT  *RC_NAME ;
INT  *IND_EC_NO ;
INT  *DEP_EC_NO ;
```

```
      INT  *NO_IND_ENT ;
      INT  *MIN_NO_DEP_ENT ;
      INT  *MAX_NO_DEP_ENT ;
OUTPUT:
      INT  *STATUS ;
DESCRIPTION:
      THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO
LINK_RELATION
      THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.
      IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND
RETURN VALUE BOTH
      WILL BE 0 , OTHERWISE -1.
REWRITTEN TO PRO*C DUE TO PROBLEM WITH SQL*FORMS AND HLI
INTERFACE
```

ARGUMENTS:

```
-----
RC_NO              INT *
RC_NAME            CHAR *
IND_EC_NO          INT *
DEP_EC_NO          INT *
NO_IND_ENT         INT *
MIN_NO_DEP_ENT     INT *
MAX_NO_DEP_ENT     INT *
STATUS             INT *
```

ROUTINES CALLED:

```
-----
STRNCPY
SQLSCA
SQLBS2
SQLSCH
SQLSCC
SQLTFL
SQLOPN
SQLOSQ
SQLAB2
SQLEXE
ERRRPT
```



# DOCGROUP PS41100 Module Documentation

NAME: INSMOD  
PURPOSE: INSERT A RECORD INTO MODEL\_CLASS  
LANGUAGE: C  
SOURCE FILE: INSMOD  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

**DESCRIPTION:**

```
VALUES
(MODEL_NO,SYSDATE,SYSDATE,MODEL_NAME,
'UNCHECKED')
```

## SYNOPSIS

```

C          --      INSMOD(&MODEL_NO, MODEL_NAME, &STATUS) ;
COBOL      --      CALL "INSMOD" USING
                                SYSDATE,
                                MODEL-NAME,
                                STATUS.

```

```
FORTRAN  --  CALL INSMOD(MODELNO, MODELNAME, STATUS)
```

**INPUT:**

```
INT  *MODEL_NO ;
CHAR *MODEL_NAME ;
```

**OUTPUT:**

```
INT *STATUS ;
```

**DESCRIPTION:**

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
MODEL CLASS

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PROC DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
MODEL\_NO  
MODEL\_NAME  
STATUS

INT \*  
CHAR \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
CDSDATE  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSNODE  
PURPOSE: INSERT A NEW NODE ONTO A LINKED LIST STRUCTURE  
LANGUAGE: C  
SOURCE FILE: INSNODE  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

INPUT:

LLIST2 \*(\*CURRLIST);  
INT NEWREFNO;

CALLED BY:

DLMIGRC, DMIGKKC, DMIGKRC, DPKCLST, DPKCLS1

DESCRIPTION:

THIS ROUTINE WILL ADD A NODE ONTO A LINKED LIST. THE TABLE  
HOLDS DATA RETREIVED FROM THE DATABASE IN A TREE STRUCTURE  
FASHION.

ARGUMENTS:

-----

CURRLIST  
NEWREFNO

LLIST2 \*\*  
INT

INCLUDE FILES:

-----

STDIO  
LLIST

PS 620341100  
30 September 1990

ROUTINES CALLED:  
-----  
MALLOC

DOCGROUP PS41100 Module Documentation

NAME: INSOAC  
PURPOSE: INSERT A RECORD INTO OWNED\_ATTRIBUTE  
LANGUAGE: C  
SOURCE FILE: INSOAC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

VALUES (AC\_NO,EC\_NO) ;

SYNOPSIS

```
      C          -- INSOAC(&AC_NO, &EC_NO, &STATUS)
      COBOL       -- CALL "INSOAC" USING
                        AC-NO,
                        EC-NO,
                        STATUS.
      FORTRAN     -- CALL INSOAC(ACNO, ECNO, STATUS)
```

INPUT:

```
      INT  *AC_NO ;
      INT  *EC_NO ;
```

OUTPUT:

```
      INT  *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
OWNED\_ATTRIBUTE

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

AC\_NO  
EC\_NO  
STATUS

INT \*  
INT \*  
INT \*

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSORDR  
PURPOSE: INSERT NEW NODE INTO A LINKED LIST SO IT CAN BE  
LANGUAGE: C  
SOURCE FILE: INSORDR  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

TRAVERSED IN ORDER.

INPUT:

ECUSED \*ECNOLIST, \*TAILNODE, \*ECORDHEAD;  
INT NEWEC;

CALLED BY:

ECSTRUC, ECSTRC1

DESCRIPTION:

THIS WILL INSERT A NEW UNIQUE EC\_NO FOUND INTO A LINKED  
LIST

WITH POINTERS SET UP SO IT CAN BE TRAVERSED IN  
DESCENDING ORDER.

ARGUMENTS:

-----

ECNOLIST  
TAILNODE  
ECORDHEAD  
NEWEC

ECUSED \*  
ECUSED \*  
ECUSED \*  
INT

INCLUDE FILES:

-----

LLIST

STDIO

ROUTINES CALLED:

-----

MALLOC



DOCGROUP PS41100 Module Documentation

NAME: INSRC  
PURPOSE: INSERT A RECORD INTO RELATION\_CLASS  
LANGUAGE: C  
SOURCE FILE: INSRC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

VALUES (RC\_NO,REL\_TYPE)

SYNOPSIS

```
      C      --  INSRC(&RC_NO, REL_TYPE, &STATUS)
      COBOL   --  CALL "INSRC"      USING
                                RC-NO,
                                REL-TYPE,
                                STATUS.
      FORTRAN --  CALL INSRC(RCNO,RELTYPE,STATUS)
```

INPUT:

```
      INT  *RC_NO ;
      INT  *REL_TYPE ;
```

OUTPUT:

```
      INT  *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
RELATION\_CLASS.

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
RC NO  
REL TYPE  
STATUS

INT \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRLEN  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSRCRS  
PURPOSE: INSERT A RECORD INTO RC\_BASED\_REC\_SET  
LANGUAGE: C  
SOURCE FILE: INSRCRS  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

VALUES (DB\_ID,SET\_ID,RT\_ID,RC\_NO)

SYNOPSIS

```
C          -- INSRCRS(&DB_ID,&SET_ID,&RT_ID,&RC_NO,&STATUS)
COBOL      -- CALL "INSRCRS" USING
                                DB-ID,
                                SET_ID,
                                RT-ID,
                                RC-NO,
                                STATUS.
```

```
FORTTRAN  -- CALL INSRCRS(DBID,SETID,RTID,RCNO,STATUS)
```

INPUT:

```
INT  *RC_NO ;
INT  *DB_ID ;
CHAR *SET_ID ;
CHAR *RT_ID ;
```

OUTPUT:

```
INT  *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
RC\_BASED\_REC\_SET, THE DATA ARE TAKEN FROM THE INPUT

PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

|        |        |
|--------|--------|
| DB_ID  | INT *  |
| SET_ID | CHAR * |
| RT_ID  | CHAR * |
| RC_NO  | INT *  |
| STATUS | INT *  |

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSRSET  
PURPOSE: INSERT A RECORD INTO THE RECORD\_SET ENTITY. IF  
LANGUAGE: C  
SOURCE FILE: INSRSET  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
RETURN CODE TO SUCCESSFUL, THEN SET THE STATUS AND THE  
TO 0, OTHERWISE TO -1.

SYNOPSIS

C --  
INSRSET(&DB\_ID,&SET\_ID,&OWNER\_ID,&MEMBER\_NO,&SET\_NO,  
&STATUS)  
COBOL -- CALL "INSRSET" USING  
DB-ID,  
SET-ID,  
OWNER-ID,  
MEMBER-NO,  
SET-NO,  
STATUS.  
FORTRAN -- CALL INSRSET(DBID, SETID, OWNERID,  
MEMBERNO, SETNO,  
STATUS)

INPUT:

INT \*DB\_ID ;  
CHAR \*SET\_ID ;  
CHAR \*OWNER\_ID ;  
INT \*MEMBER\_NO ;  
INT \*SET\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:  
THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
RECORD SET  
ENTITY. IF SUCCESSFULLY INSERT THE RECORD , THE STATUS  
AND RETURN  
VALUE BOTH WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C 5/13/88

ARGUMENTS:

|           |        |
|-----------|--------|
| DB_ID     | INT *  |
| SET_ID    | CHAR * |
| OWNER_ID  | CHAR * |
| MEMBER_NO | INT *  |
| SET_NO    | INT *  |
| STATUS    | INT *  |

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: INSSTM  
PURPOSE: INSERT A RECORD INTO THE SET\_TYPE\_MEMBER ENTITY. IF  
LANGUAGE: C  
SOURCE FILE: INSSTM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
RETURN CODE TO SUCCESSFUL, THEN SET THE STATUS AND THE  
TO 0, OTHERWISE TO -1.

SYNOPSIS

C --  
INSSTM(&DB\_ID,&SET\_ID,&MEMBER\_ID,&REQ\_MEM\_IND, &STATUS)  
COBOL -- CALL "INSSTM" USING  
DB-ID,  
SET-ID,  
MEMBER-ID,  
REQ-MEM-IND,  
STATUS.

FORTRAN -- CALL INSSTM(DBID, SETID, MEMBERID,  
REQMEMIND, STATUS)

INPUT:

INT \*DB\_ID ;  
CHAR \*SET\_ID ;  
CHAR \*MEMBER\_ID ;  
CHAR \*REQ\_MEM\_IND ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO INSERT A RECORD INTO  
SET\_TYPE\_MEMBER

ENTITY. IF SUCCESSFULLY INSERT THE RECORD , THE STATUS  
AND RETURN

VALUE BOTH WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C 5/13/88

ARGUMENTS:

-----  
DB\_ID  
SET\_ID  
MEMBER\_ID  
REQ\_MEM\_IND  
STATUS

INT \*  
CHAR \*  
CHAR \*  
CHAR \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: INSTFLD  
PURPOSE: INSERT A ROW INTO CDM TABLE DATA\_FIELD  
LANGUAGE: C  
SOURCE FILE: INSTFLD  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

```
      C      --  INSTFLD(DB_ID,RT_ID,DF_ID,DF_NO
                  REC_SEQ_NO,REC_KEY_CODE,OCCURS,DBMS_ACCESS
                  INDEX_IND,REDEF_DF,COMP_DF,OCC_DEP_DF,
                  INDEX_DF,DATA_TYPE_NAME,FILLER_SIZE,
                  STATUS)

      COBOL   --  CALL INSTFLD      USING
                  DB_ID,
                  RT_ID,
                  DF_NO,
                  STATUS.

      FORTRAN --  CALL  INSTFLD (DB_ID,RT_ID,DF_ID,DF_NO
INPUT:
ALL FIELDS OF DATA_FIELD
OUTPUT:
STATUS
DESCRIPTION
A NEW ROW IS INSERTED INTO DATA FIELD.
IF THE COLUMN CONTAINS ZERO A NULL IS INSERTED.
IF SUCCESSFUL, A STATUS OF ZERO IS RETURNED.
REWRITTEN TO PRO*C, 5/13/88
```

ARGUMENTS:

|                |        |
|----------------|--------|
| DB_ID          | INT *  |
| RT_ID          | CHAR * |
| DF_ID          | CHAR * |
| DF_NO          | INT *  |
| REC_SEQ_NO     | INT *  |
| REC_KEY_CODE   | CHAR * |
| OCCURS         | INT *  |
| DBMS_ACCESS    | CHAR * |
| INDEX_IND      | CHAR * |
| REDEF_DF_NO    | INT *  |
| COMP_DF_NO     | INT *  |
| OCC_DEP_DF_NO  | INT *  |
| INDEX_BY_DF_NO | INT *  |
| DATA_TYPE_NAME | CHAR * |
| FILLER_SIZE    | INT *  |
| STATUS         | INT *  |

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLSQS  
SQLOSQ  
SQLAB2  
SQLEXE  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: LOOPDEP  
PURPOSE: CHECK A RELATION TREE FOR LOOP DEPENDENCIES  
LANGUAGE: C  
SOURCE FILE: LOOPDEP  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
INPUT:

LLIST1 \*\*CURRENT\_NODE;  
INT NEWREFNO;

CALLED BY:

BLOOPCK.PC, TLOOPCK.PC

DESCRIPTION:

THROUGH USE OF A LINKED LIST STRUCTURE THIS ROUTINE WILL  
CHECK TO MAKE SURE A DEPENDENCY LOOP DOES NOT EXIST IN A  
MODEL.

ARGUMENTS:

-----  
CURRENT\_NODE  
NEWREFNO

LLIST1 \*\*  
INT

INCLUDE FILES:

-----  
STDIO  
LLIST

DOCGROUP PS41100 Module Documentation

NAME: LOWUPP  
PURPOSE: CONVERT A STRING TO UPPER CASE CHARACTERS  
LANGUAGE: C  
SOURCE FILE: LOWUPP  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: SHARE

DESCRIPTION:

-----

SYNOPSIS

|          |    |                             |                                 |
|----------|----|-----------------------------|---------------------------------|
| C        | -- | LOWUPP (STRING,&STRING_LEN) |                                 |
| COBOL    | -- | CALL "LOWUPP"               | USING<br>STRING,<br>STRING-LEN, |
| FORTTRAN | -- | CALL LOWUPP (STRING,STRLEN) |                                 |

INPUT:

CHAR STRING []  
INT\* STRING\_LEN

OUTPUT:

DESCRIPTION

LOWUPP WILL CONVERT THE CONTENTS OF A CHARACTER STRING  
TO UPPER CASE CHARACTERS. THE LENGTH OF THE STRING IS  
PASSED AS A PARAMETER.

ARGUMENTS:

-----

|            |         |
|------------|---------|
| STRING     | CHAR [] |
| STRING_LEN | INT *   |

INCLUDE FILES:

-----  
CTYPE

ROUTINES CALLED:

-----  
ISLOWER  
TOUPPER

DOCGROUP PS41100 Module Documentation

NAME: LVLCHK  
PURPOSE: RETRIEVE THE MAX LEVEL IN THE TREE STRUCTURE SEARCH  
LANGUAGE: C  
SOURCE FILE: LVLCHK  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

THIS ROUTINE IS CALLED BY ALLREL.C

SYNOPSIS

```
      C      --  
LVLCHK(RC_NO,MODEL_NO,MAX_LEVEL_FOUND,STATUS)  
      COBOL      --  CALL "LVLCHK"      USING  
                                   RC_NO,MODEL-NO,  
                                   MAX-LEVEL-FOUND  
                                   STATUS.  
      FORTRAN      --  CALL LVLCHK(RC-NO,MODEL_NO,MAX-LEVEL-FOUND  
                                   STATUS)
```

INPUT:

```
      INT  *MODEL_NO ;  
      INT  *RC_NO ;  
      INT  *MAX_LEVEL ;  
      INT  *STATUS ;
```

DESCRIPTION:

ARGUMENTS:

-----

|                 |       |
|-----------------|-------|
| RC_NO           | INT * |
| MODEL_NO        | INT * |
| MAX_LEVEL_FOUND | INT * |

STATUS

INT \*

ROUTINES CALLED:

-----  
COOPEN  
ERRRPT  
COCLOSE  
COFETCH  
COEXEC  
COBINDN  
CODFINN  
COSQL3

DOCGROUP PS41100 Module Documentation

NAME: LVLCHK1  
PURPOSE: RETRIEVE THE MAX LEVEL IN THE TREE STRUCTURE SEARCH  
LANGUAGE: C  
SOURCE FILE: LVLCHK1  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

THIS ROUTINE IS CALLED BY DEPREL.C

SYNOPSIS

```
      C          -- LVLCHK1(RC_NO,EC_NO,MAX_LEVEL_FOUND,STATUS)
      COBOL       -- CALL "LVLCHK1" USING
                        RC_NO,EC-NO,
                        MAX-LEVEL-FOUND
                        STATUS.
      FORTRAN     -- CALL LVLCHK1(RC-NO,EC_NO,MAX-LEVEL-FOUND
                                STATUS)
```

INPUT:

```
      INT  *MODEL_NO ;
      INT  *RC_NO ;
      INT  *MAX_LEVEL ;
      INT  *STATUS ;
```

DESCRIPTION:

ARGUMENTS:

-----

|                 |       |
|-----------------|-------|
| RC_NO           | INT * |
| EC_NO           | INT * |
| MAX_LEVEL_FOUND | INT * |



STATUS

INT \*

ROUTINES CALLED:

-----  
COOPEN  
ERRRPT  
COCLOSE  
COFETCH  
COEXEC  
COBINDN  
CODFINN  
COSQL3

DOCGROUP PS41100 Module Documentation

NAME: MAPRC  
PURPOSE: MAP A RELATION CLASS TO A SET  
LANGUAGE: C  
SOURCE FILE: MAPRC  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

SYNOPSIS

```
      C          --      MAPRC (&RC_NO, &DB_NO, SET_NAME,
RECORD_NAME)
      COBOL       --      CALL  "MAPRC"          USING
                                      RC-NO
                                      DB-NO,
                                      SET-NAME,
                                      RECORD-NAME.

      FORTRAN     --      CALL  MAPRC (RCNO, DBNO, SETNAME, RENAME)
INPUT:
      INT *RC_NO
      INT *DB_NO
      CHAR SET_NAME[]
      CHAR RECORD_NAME[]
OUTPUT:
DESCRIPTION
      MAPRC IS RESPONSIBLE FOR CALLING ROUTINES WHICH ENFORCE
RC TO SET
      MAPPING RULES.  IF NO RULES ARE BROKEN, A ROUTINE IS
CALLED WHICH
      PERFORMS THE MAPPING.
```

ARGUMENTS:

-----

RC\_NO  
DB\_NO  
SET\_NAME  
RECORD\_NAME

INT \*  
INT \*  
CHAR []  
CHAR []

INCLUDE FILES:

-----  
CSTDTP

ROUTINES CALLED:

-----  
STRNCPY  
FND1MEM  
VERRCMP  
SPRINTF  
UERROR  
INSRCRS  
FNDASA

DOCGROUP PS41100 Module Documentation

NAME: MODACKW  
PURPOSE: UPDATE THE KW\_NO IN THE AC\_KEYWORD CDM TABLE  
LANGUAGE: C  
SOURCE FILE: MODACKW  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C -- MODACKW(OLD\_OBJ\_NO, NEW\_OBJ\_NO, STATUS)  
  
COBOL -- CALL "MODACKW" USING OLD-OBJ-NO  
NEW-OBJ-NO  
STATUS.  
  
FORTRAN -- CALL MODACKW (OLD-OBJ-NO,NEW-OBJ-NO,STATUS)

INPUT:

OLD\_OBJ\_NO  
NEW\_OBJ\_NO

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE  
THE KEYWORD NUMBER IN THE AC KEYWORD CDM TABLE.  
IF SUCCESSFUL, THE STATUS WILL BE 0.  
REWRITTEN TO PRO\*C, 5/13/88

ARGUMENTS:

-----  
OLD\_OBJ\_NO  
NEW\_OBJ\_NO

INT \*  
INT \*

STATUS

INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: MODECKW  
PURPOSE: UPDATE THE KW\_NO IN THE EC\_KEYWORD CDM TABLE  
LANGUAGE: C  
SOURCE FILE: MODECKW  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

      C          -- MODECKW(OLD\_OBJ\_NO, NEW\_OBJ\_NO, STATUS)  
  
      COBOL      -- CALL "MODECKW" USING OLD-OBJ-NO  
                                        NEW-OBJ-NO  
                                        STATUS.  
  
      FORTRAN   -- CALL MODECKW (OLD-OBJ-NO,NEW-OBJ-NO,STATUS)

INPUT:

      OLD\_OBJ\_NO  
      NEW\_OBJ\_NO

OUTPUT:

      STATUS

DESCRIPTION

      THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE  
      THE KEYWORD NUMBER IN THE EC\_KEYWORD CDM TABLE.  
      IF SUCCESSFUL, THE STATUS WILL BE 0.  
REWRITTEN TO PRO\*C, 5/13/88

ARGUMENTS:

-----  
OLD\_OBJ\_NO  
NEW\_OBJ\_NO

INT \*  
INT \*

STATUS

INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: MODRCKW  
PURPOSE: UPDATE THE KW\_NO IN THE RC\_KEYWORD CDM TABLE  
LANGUAGE: C  
SOURCE FILE: MODRCKW  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C -- MODRCKW(OLD\_OBJ\_NO, NEW\_OBJ\_NO, STATUS)

COBOL -- CALL "MODRCKW" USING OLD-OBJ-NO  
NEW-OBJ-NO  
STATUS.

FORTRAN -- CALL MODRCKW (OLD-OBJ-NO,NEW-OBJ-NO,STATUS)

INPUT:

OLD\_OBJ\_NO  
NEW\_OBJ\_NO

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE  
THE KEYWORD NUMBER IN THE RC\_KEYWORD CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C, 5/13/88

ARGUMENTS:

-----  
OLD\_OBJ\_NO  
NEW\_OBJ\_NO

INT \*  
INT \*



STATUS

INT \*

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: MODYFLD  
PURPOSE: UPDATE FIELDS IN DATA\_FIELD GIVEN THE FIELD NAME  
LANGUAGE: C  
SOURCE FILE: MODYFLD  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

SYNOPSIS

C       --  
MODYFLD(DB\_ID,RT\_ID,DF\_ID,SEQ\_NO,KEY\_CODE,OCCURS,KNOWN,  
          INDEX\_IND,REDEF\_DF\_NO,COMP\_DF,OCC\_DEP\_DF,  
          INDEX\_DF,DATA\_TYPE,FILLER\_SIZE,STATUS)

COBOL    -- CALL "MODYFLD" USING DB-ID  
                                  RT-ID  
                                  DF-ID  
                                  SEQ-NO  
                                  KEY-CODE  
                                  OCCURS  
                                  KNOWN  
                                  INDEX-IND  
                                  REDEF-DF  
                                  COMP-DF  
                                  OCC-DEP  
                                  INDEX-DF  
                                  DATA-TYPE-NAME  
                                  FILLER-SIZE  
                                  STATUS.

FORTTRAN -- CALL  
MODYFLD(DB-ID,RT-ID,DF-ID,SEQ-NO,KEY-CODE,OCCURS,KNOWN  
INDEX-IND,REDEF-DF,COMP-DF,OCC-DEP,INDEXED-DF,  
                                  DATA-TYPE,FILLER-SIZE,STATUS) .

INPUT:

DB\_ID  
RT\_ID  
DF\_ID

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE  
ALL THE FIELDS IN THE DATA\_FIELD GIVEN THE DB\_ID,  
RT\_ID, AND DF\_ID.

IF THESE FIELDS ARE ZERO , A NULL IS INSERTED  
IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C, 5/13/88

ARGUMENTS:

-----

|             |        |
|-------------|--------|
| DB_ID       | INT *  |
| RT_ID       | CHAR * |
| DF_ID       | CHAR * |
| DF_NO       | INT *  |
| SEQ_NO      | INT *  |
| KEY_CODE    | CHAR * |
| OCCURS      | INT *  |
| KNOWN       | CHAR * |
| INDEX_IND   | CHAR * |
| REDEF_DF_NO | INT *  |
| COMP_DF     | INT *  |
| OCC_DEP_DF  | INT *  |
| INDEXED_DF  | INT *  |
| DT_NAME     | CHAR * |
| FILLER_SIZE | INT *  |
| STATUS      | INT *  |

ROUTINES CALLED:

-----

STRNCPY  
SQLSCA  
SQLBS2

PS 620341100  
30 September 1990

SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLSQS  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: MRGNODE  
PURPOSE: SELECT ALL THE TOP NODE (LEVEL 1) INDEPENDENT  
          ENTITY'S  
LANGUAGE: C  
SOURCE FILE: MRGNODE  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
                          IN MODEL-2 AND CONTROL THE LOGIC FOR  
GENERATING NDDL          STATEMENTS TO EITHER CREATE THE MODEL-2  
ENTITY OR                  COMBINE THE ENTITY WITH A MODEL-1 ENTITY.

SYNOPSIS

```
      C          -- MRGNODE(&MOD_NO2, MOD_NAME2, &KEYWORD_FLAG,  
                     &ALIAS_FLAG, &DESC_FLAG, &RC);  
      COBOL      -- CALL "MRGNODE" USING  
                     MOD_NO2,  
                     MOD_NAME2,  
                     KEYWORD_FLAG,  
                     ALIAS_FLAG,  
                     DESC_FLAG,  
                     RC.  
      FORTRAN    -- CALL MRGNODE(MODNO, MODNAME, KEYWORD,  
      ALIAS_FLAG, DESC_FLAG, RC)
```

INPUT:

```
      INT  *MOD_NO2 ;  
      CHAR *MOD_NAME2 ;  
      INT  *KEYWORD_FLAG ;  
      INT  *ALIAS_FLAG ;  
      INT  *DESC_FLAG ;
```

INT \*RC ;  
DESCRIPTION:  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

|              |        |
|--------------|--------|
| MOD_NO1      | INT *  |
| MOD_NO2      | INT *  |
| MOD_NAME2    | CHAR * |
| KEYWORD FLAG | CHAR * |
| ALIAS FLAG   | CHAR * |
| DESC FLAG    | CHAR * |
| STATUS       | INT *  |

INCLUDE FILES:

EOD

ROUTINES CALLED:

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLCLS  
ERRRPT  
SQLAD2  
SQLFCH  
SELECNM  
VERENT  
GENENT1  
GENALTE  
SPRINTF  
UERROR  
SQLTOC  
SQLSQS  
SQLOSQ

PS 620341100  
30 September 1990

SQLAB2  
SQLEXE

DOCGROUP PS41100 Module Documentation

NAME: NDDL/MAIN  
PURPOSE: MAIN PROGRAM FOR THE NDDL COMMAND PROCESSOR  
LANGUAGE: C  
SOURCE FILE: NDDL  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

SYNOPSIS

FROM THE OPERATING SYSTEM:

(GET IN UNIX)

%NDDL [-I] [<FILE]

WHERE:

-I IS OPTIONAL AND STANDS FOR INTERACTIVE, IN WHICH CASE  
THE PRPGRAM WILL USE THE FORMS PROCESSOR

OTHERWISE THE PROGRAM WILL ACCEPT INPUT FROM THE  
STANDARD INPUT, WHICH CAN BE RE-DIRECTED WITH <FILE  
GIVING A BATCH OPERATING MODE.

\*\*\*\*\* BATCH VERSION \*\*\*\*\*

INPUT:

THE USER ENTERED, OPTIONAL -I PARAMETER

OUTPUT:

DESCRIPTION

THIS IS THE MAIN FOR THE NDDL COMMAND PROCESSOR.

ARGUMENTS:

-----

ARGC  
ARGV

INT  
CHAR \* []



INCLUDE FILES:

-----

SIGNAL  
STDIO  
CMDID  
LISTS  
NDDLEX

ROUTINES CALLED:

-----

ERRTRAP  
STRNCPY  
PRINTF  
SCANF  
STRCPY  
STRCAT  
SIGNAL  
INITSES  
PROCMD  
TERMSES  
TRMDDL  
EXIT

DOCGROUP PS41100 Module Documentation

NAME: ERRTRAP  
PURPOSE:  
LANGUAGE: C  
SOURCE FILE: UINDDL  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

ARGUMENTS:

-----

XXX

INT

INCLUDE FILES:

-----

SIGNAL  
STDIO  
CMDID  
LISTS  
NDDLEX

ROUTINES CALLED:

-----

TRMDDL  
EXIT

DOCGROUP PS41100 Module Documentation

NAME: YYWRAP  
PURPOSE:  
LANGUAGE: C  
SOURCE FILE: NDDLYTB  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: SHARE

INCLUDE FILES:

-----

NDDLYAC.INP"  
STDIO  
CMDID  
ATTRID  
LISTID  
LISTSEX  
NDDLEX  
FORMS  
NDMLSTB  
NDDLEXY

DOCGROUP PS41100 Module Documentation

NAME: YYERROR  
PURPOSE:  
LANGUAGE: C  
SOURCE FILE: NDDLTYTB  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: SHARE

ARGUMENTS:

-----

S

CHAR \*

INCLUDE FILES:

-----

NDDLTYAC.INP"  
STDIO  
CMDID  
ATTRID  
LISTID  
LISTSEX  
NDDLEX  
FORMS  
NDMLSTB  
NDDLEXY

ROUTINES CALLED:

-----

SPRINTF  
UERROR  
FREE\_SYMTAB

DOCGROUP PS41100 Module Documentation

NAME: YYNUL  
PURPOSE:  
LANGUAGE: C  
SOURCE FILE: NDDLTYB  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: SHARE

INCLUDE FILES:

-----  
NDDLTYAC.INP"  
STDIO  
CMDID  
ATTRID  
LISTID  
LISTSEX  
NDDLEX  
FORMS  
NDMLSTB  
NDDLEXY

DOCGROUP PS41100 Module Documentation

NAME: YYPARSE  
PURPOSE:  
LANGUAGE: C  
SOURCE FILE: NDDLVTB  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: SHARE

INCLUDE FILES:

-----  
NDDLVTB.INP"  
STDIO  
CMDID  
ATTRID  
LISTID  
LISTSEX  
NDDLEX  
FORMS  
NDMLSTB  
NDDLEXY

ROUTINES CALLED:

-----  
PRINTF  
ADD\_TO\_LST  
ADD\_CORR  
PUT\_SYMB  
CHAIN\_SYMB  
ADD\_TO\_CNT  
YYERROR  
MEMCPY  
NEW\_SCOPE

PS 620341100  
30 September 1990

YYNUL  
EXIT SCOPE  
YYLEX

DOCGROUP PS41100 Module Documentation

NAME: NEXTKC  
PURPOSE: THIS ROUTINE RETURNS A KC\_NAME FOR A GIVEN EC\_NO FROM  
LANGUAGE: C  
SOURCE FILE: NEXTKC  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
THE UNBOUNDED KEYLIST DATA STRUCTURE.

1. IF THE FIRST\_FLAG IS 0, THE ROUTINE  
RETURNS THE FIRST KC\_NAME.  
IF THE CHAIN IS EMPTY, THEN RETURNS  
ERROR CODE.

2. IF THE FIRST\_FLAG IS 1, THE ROUTINE  
IF THERE IS NO CURRENT KC\_NO,  
RETURNS THE NEXT THE FIRST KC\_NAME WILL BE RETURNED AS  
KC\_NAME, THEN IF IT EXISTS.  
NEXT KC\_NAME IF THE CURRENT KC\_NO IS THE LAST ONE,  
THEN ERROR CODE IS RETURNED.

3. IF THE FIRST\_FLAG IS -1, THE ROUTINE  
RETURNS KC\_NAME FOR THE GIVEN EC\_NO AND KC\_NO.  
IF THE DESIRED KC\_NAME IS NOT FOUND,  
THEN RETURNS THE ERROR CODE.  
IF THE OPERATION IS SUCCESSFUL, SET THE  
STATUS AND RETURN CODE TO 0.

SYNOPSIS  
C -- NEXTKC(&EC\_NO, &FLAG, &KC\_NO, &KC\_NAME,  
&STATUS) -- CALL "NEXTKC" USING  
COBOL -- EC-NO,  
FLAG,  
KC-NO,



```

                                KC-NAME,
                                STATUS.
      FORTRAN  --  CALL MEXTKC(ECNO, FLAG, KCNO, KCNAE,
STATUS)
INPUT:
      INT  *EC_NO  ;
      INT  *FLAG  ;
      INT  *KC_NO  ;
      CHAR *KC_NAME ;
      INT  *STATUS ;
OUTPUT:
DESCRIPTION:
```

ARGUMENTS:

```

-----
EC_NO          INT *
FLAG           INT *
KC_NO          INT *
KC_NAME        CHAR *
STATUS         INT *
```

INCLUDE FILES:

```

-----
KEYLIST
```

ROUTINES CALLED:

```

-----
STRNCPY
```

DOCGROUP PS41100 Module Documentation

NAME: NEXTKCM  
PURPOSE: THIS ROUTINE RETURNS A KG\_NO,KG\_NAME FOR A GIVEN  
          EC\_NO  
LANGUAGE: C  
SOURCE FILE: NEXTKCM  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
AND KC\_NO FROM THE UNBOUNDED KEYLIST DATA  
STRUCTURE.  
1. IF THE FIRST\_FLAG IS 0, THE ROUTINE  
RETURNS THE FIRST  
          KG\_NAME.  
          IF THE CHAIN IS EMPTY, THEN RETURNS  
ERROR CODE.  
2. IF THE FIRST\_FLAG IS 1, THE ROUTINE  
RETURNS THE NEXT  
          IF THERE IS NO CURRENT KC\_NO,  
          THE FIRST KG\_NAME WILL BE RETURNED AS  
KG\_NAME, THEN  
          IF IT EXISTS.  
NEXT KG\_NAME  
          IF THE CURRENT KG\_NO IS THE LAST ONE,  
THEN ERROR  
          CODE IS RETURNED.  
3. IF THE FIRST\_FLAG IS -1, THE ROUTINE  
RETURNS KG\_NAME  
          FOR THE GIVEN EC\_NO AND KG\_NO.  
          IF THE DESIRED KG\_NAME IS NOT FOUND,  
THEN RETURNS  
          THE ERROR CODE.  
STATUS AND  
          IF THE OPERATION IS SUCCESSFUL, SET THE  
SYNOPSIS  
          RETURN CODE TO 0.  
          C           -- NEXTKCM(&EC\_NO, &KC\_NO, &FLAG, &KG\_NO,  
&KG\_NAME,            &STATUS)  
          COBOL       -- CALL "NEXTKCM"    USING  
                          EC-NO,

KC-NO,  
FLAG,  
KG-NO,  
KG-NAME,  
STATUS.

FORTTRAN -- CALL NEXTKCM(ECNO, KCNO, FLAG, KGNO,  
KGNAME, STATUS)

INPUT:

INT \*EC\_NO ;  
INT \*KC\_NO ;  
INT \*FLAG ;  
INT \*KG\_NO ;  
CHAR \*KG\_NAME ;  
INT \*STATUS ;

OUTPUT:

DESCRIPTION:

ARGUMENTS:

-----

EC\_NO

FLAG

KC\_NO

KG\_NO

KG\_NAME

STATUS

INT \*

INT \*

INT \*

INT \*

CHAR \*

INT \*

INCLUDE FILES:

-----

KEYLIST

ROUTINES CALLED:

-----

STRNCPY

DOCGROUP PS41100 Module Documentation

NAME: NFLCLOS  
PURPOSE: THIS ROUTINE CLOSSES AN OUTPUT FILE. THE FILE WILL  
LANGUAGE: C  
SOURCE FILE: NFLCLOS  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

CONTAINS THE USER'S NDDL.

SYNOPSIS

|          |    |                                             |
|----------|----|---------------------------------------------|
| C        | -- | NFLCLOS(&FILE NAME, &READ_WRITE, &STATUS) ; |
| COBOL    | -- | CALL "NFLCLOS" USING                        |
|          |    | FILE-NAME,                                  |
| FORTTRAN | -- | CALL NFLCLOS(FILENAME, READWRITE, STATUS)   |

INPUT:

OUTPUT:

DESCRIPTION:

INCLUDE FILES:

-----

NDDLEX  
STDIO

ROUTINES CALLED:

-----

FCLOSE

DOCGROUP PS41100 Module Documentation

NAME: NFLOPEN  
PURPOSE: THIS ROUTINE OPENS AN OUTPUT FILE.  
LANGUAGE: C  
SOURCE FILE: NFLOPEN  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
MODIFIED 04/01/86 TO NULL TERMINATE THE FILENAME ARGUMENT

SYNOPSIS

C           -- NFLOPEN(&FILE NAME) ;  
COBOL       -- CALL "NFLOPEN" USING  
                                  FILE-NAME.  
FORTRAN     -- CALL NFLOPEN(FILENAME)

INPUT:

CHAR \*FILE\_NAME ;

OUTPUT:

DESCRIPTION:

THE FILE WILL CONTAIN THE USER'S GENERATED NDDL.

FILE OPN APP FLAG MAY CONTAIN 3 VALUES :

0 : OUTPUT TO SCREEN       :SCREENIO  
1 : OPEN A NEW FILE       :FILEOPN  
2 : APPEND TO EXISTING FILE :FILEAPP

ARGUMENTS:

-----  
STATUS

INT \*

INCLUDE FILES:

-----  
STDIO  
NDDLEX

ROUTINES CALLED:

-----  
FOPEN  
ESCPY  
SPRINTF  
UERROR

DOCGROUP PS41100 Module Documentation

NAME: NRGET  
PURPOSE: GET AN UNIQUE NUMBER FROM TH REUSABLE POOL  
LANGUAGE: C  
SOURCE FILE: NRGET  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

SYNOPSIS

```
C          -- NRGET (POOL_NO, NEXT_NO, STATUS)

COBOL      -- CALL "NRGET"    USING
                        POOL-NO,
                        NEXT_NO,
                        STATUS.

FORTRAN    -- CALL NRGET (POOLNO,NEXT_NO,STATUS)
```

INPUT:

INT \*POOL\_NO ;

OUTPUT:

INT \*NEXT\_NO ;  
INT \*STATUS ;

DESCRIPTION

THIS ROUTINE RETURNS A UNIQUE NUMBER FOR A POOL NUMBER  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

|         |       |
|---------|-------|
| POOL_NO | INT * |
| NEXT_NO | INT * |
| STATUS  | INT * |

INCLUDE FILES:

-----

EOD

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLCLS  
ERRRPT  
SQLAD2  
SQLFCH  
GETNXNO  
DELREUS  
SQLTOC  
SQLOSQ  
SQLAB2  
SQLEXE



DOCGROUP PS41100 Module Documentation

NAME: NRSTORE  
PURPOSE: STORE A AVAILABLE NO OF A POOL NO BACK TO NO LINKED  
LANGUAGE: C  
SOURCE FILE: NRSTORE  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

LIST DATA STRUCTURE.

SYNOPSIS

|          |    |                                               |
|----------|----|-----------------------------------------------|
| C        | -- | NRSTORE(PPOOL_NO, AVAIL_NO)                   |
| COBOL    | -- | CALL "NRSTORE" USING<br>POOL-NO,<br>AVAIL-NO. |
| FORTTRAN | -- | CALL NRSTORE(PPOOLNO, AVAILNO)                |

INPUT:

INT \*POOL\_NO ;  
INT \*AVAIL\_NO ;

OUTPUT:

NONE

DESCRIPTION

THIS ROUTINE PUT THE AVAILABLE NO OF THE POOL NO BACK TO  
THE NO  
LINKED LIST. THE ROUTINE CALLS 'ALLOC' FUNCTION TO  
ALLOCATE SPACE  
FOR STORING THE AVAIL NO AND POOL NO.

ARGUMENTS:

-----

|         |       |
|---------|-------|
| AC_NO   | INT * |
| NEXT_NO | INT * |

PS 620341100  
30 September 1990

STATUS

INT \*

ROUTINES CALLED:

-----  
MALLOC

DOCGROUP PS41100 Module Documentation

NAME: PROCMD  
PURPOSE: THIS ROUTINE IS CALLED TO HANDLE ALL  
LANGUAGE: C  
SOURCE FILE: PROCMD  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

PROCESSING FOR A SINGLE NDDL COMMAND.

SYNOPSIS

|         |    |                |
|---------|----|----------------|
| C       | -- | PROCMD()       |
| COBOL   | -- | CALL "PROCMD". |
| FORTRAN | -- | CALL PROCMD    |

INPUT:

OUTPUT:

DESCRIPTION

CALL THE YACC GENERATED PARSER TO ACCEPT AND PARSE  
A SINGLE COMMAND. IF A SYNTAX ERROR, LET USER  
KNOW COMMAND WAS NOT PROCESSED AND RETURN  
THAT THERE IS MORE INPUT IN THE GLOBAL END STATUS.  
CHECK TO SEE IF ALL OTHER COMMANDS SHOULD BE  
SKIPPED. CHECK TO SEE IF AN END OF FILE CONDITION  
WAS RAISED. PROCESS THE COMMAND BY A CALL TO BRANCHR.  
RESET THE COMMAND ERROR COUNTERS.

MODIFIED: OCT 25,88 TO HANDLE EOF WHEN COMMIT MODE IS MANUAL.

INCLUDE FILES:

-----

CMDID

NDDLEX  
OK  
FPPARM  
FPCODE  
FNMNIO

ROUTINES CALLED:

-----  
RESETFN  
COMMITMN  
COMMITFN  
RESETMN  
INITCMD  
PDATA  
STRNCMP  
SPRINTF  
PMSGLS  
UERROR  
ROLBACK  
BRANCHR  
RMVPAG  
ERRPRO  
GETNCHR  
PUTCHAR  
UWARN  
COMMIT  
STRCPY  
YYPARSE  
OISCR  
STRNCPY  
GDATA  
RPLFRM

DOCGROUP PS41100 Module Documentation

NAME: RCLASA  
PURPOSE: FOR MODEL NAME1 SELECT ALL THE RELATION CLASSES  
LANGUAGE: C  
SOURCE FILE: RCLASA  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
BY NAME AND LOOK FOR A MATCHING RELATION  
CLASS NAME IN MODEL\_NAME2

SYNOPSIS

```
      C      -- RCLASA(&MODEL_NAME1, &MODEL_NAME2,
&STATUS) ;
      COBOL   -- CALL "RCLASA"    USING
                                MODEL-NAME1,
                                MODEL-NAME2,
                                STATUS.
      FORTRAN -- CALL RCLASA(MODELNAME1, MODELNAME2, STATUS)
```

INPUT:

```
      CHAR *MODEL_NAME1 ;
      CHAR *MODEL_NAME2 ;
      INT  *STATUS ;
```

DESCRIPTION:

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
MODEL\_NAME1  
MODEL\_NAME2  
STATUS

CHAR \*  
CHAR \*  
INT \*

INCLUDE FILES:

-----  
EOD

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLCLS  
ERRRPT  
SQLTFL  
SQLOPN  
SQLAD2  
SQLFCH  
CPARM  
SQLTOC  
SQLSQS  
SQLOSQ  
SQLAB2  
SQLEXE

DOCGROUP PS41100 Module Documentation

NAME: RCLASS  
PURPOSE: FOR MODEL NAME1 SELECT THE PRIMARY RELATION CLASS  
LANGUAGE: C  
SOURCE FILE: RCLASS  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
RELATION CLASS      BY NAME AND LOOK FOR A MATCHING PRIMARY  
                      NAME IN MODEL\_NAME2

SYNOPSIS

      C            -- RCLASS(&MODEL\_NAME1, &MODEL\_NAME2,  
&STATUS) ;  
      COBOL        -- CALL "RCLASS"      USING  
                                      MODEL-NAME1,  
                                      MODEL-NAME2,  
                                      STATUS.  
      FORTRAN     -- CALL RCLASS(MODELNAME1, MODELNAME2, STATUS)

INPUT:

      CHAR \*MODEL\_NAME1 ;  
      CHAR \*MODEL\_NAME2 ;  
      INT \*STATUS ;

DESCRIPTION:

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
MODEL\_NAME1            CHAR \*  
MODEL\_NAME2            CHAR \*  
STATUS                 INT \*

INCLUDE FILES:

-----  
EOD

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLCLS  
ERRRPT  
SQLTFL  
SQLOPN  
SQLAD2  
SQLFCH  
CPARM  
SQLTOC  
SQLSQS  
SQLOSQ  
SQLAB2  
SQLEXE



DOCGROUP PS41100 Module Documentation

NAME: RDDESC  
PURPOSE: STORE DESCRIPTION ON THE CDM  
LANGUAGE: C  
SOURCE FILE: RDDESC  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

```
      C      --  RDDESC (OBJ_TYPE, &OBJ_NO, DESC_TYPE,
FILE_NAME,      &TXT_TYPE, &LINE_COUNT)
      COBOL   --  CALL  "RDDESC"      USING
                                OBJ-TYPE
                                OBJ-NO
                                DESC-TYPE
                                FILE-NAME
                                TXT-TYPE
                                LINE-COUNT.
      FORTRAN --  CALL RDDESC (OBJTYPE,OBJNO,DESC TYP, FILENAM,
                                +TXTTYP,LINECNT)
```

INPUT:

```
CHAR OBJ_TYPE[]
INT *OBJ_NO
CHAR DESC_TYPE[]
CHAR FILE_NAME[]
INT *TXT_TYPE
```

OUTPUT:

```
INT *LINE_COUNT
```

DESCRIPTION

RDDESC STORES DESCRIPTIONS OF TXT\_TYPE 2 (FILE INPUT) AND  
TXT\_TYPE

3 (COMMAND TEXT INPUT) INTO THE CDM AFTER DELETING THE  
PREVIOUS  
DESCRIPTION, IF ANY.

ARGUMENTS:

|            |         |
|------------|---------|
| OBJ_TYPE   | CHAR [] |
| OBJ_NO     | INT *   |
| DESC_TYPE  | CHAR [] |
| FILE_NAME  | CHAR [] |
| TXT_TYPE   | INT *   |
| LINE_COUNT | INT *   |

INCLUDE FILES:

-----  
CSTDTP

ROUTINES CALLED:

-----  
COOPEN  
ERRRPT  
FILEINS  
STRINS  
SPRINTF  
UERROR  
COCLOSE  
COBINDN  
COSQL3

DOCGROUP PS41100 Module Documentation

NAME: ROLBACK  
PURPOSE: ROLBACK THE TRANSACTIONS. NDML AND ORACLE ROLLBACK  
LANGUAGE: C  
SOURCE FILE: ROLBACK  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

```
C          -- ROLBACK(&STATUS)  ;  
  
COBOL      -- CALL "ROLBACK"  USING  
              STATUS.  
FORTRAN    -- CALL ROLBACK (STATUS)
```

INPUT:

NONE

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

MODIFIED: 1 NOV , 1986      REMOVE CALL TO REUSABLE\_NUMBER  
MODIFIED: 23 SEP,1988      CHANGE COROL TO OROL

ARGUMENTS:  
-----

STATUS

INT \*

ROUTINES CALLED:  
-----

NROLBAK

PS 620341100  
30 September 1990

SPRINTF  
UERROR  
ERRPRO

DOCGROUP PS41100 Module Documentation

NAME: SELIAUC  
PURPOSE: SELECTS ALL THE INHERITED TAG NAMES  
LANGUAGE: C  
SOURCE FILE: SELIAUC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

AND OWNER TAG NAMES ASSOCIATED WITH  
THE DEPENDENT ENTITY IN A GIVEN RELATION,  
THEN CALLS A ROUTINE TO FORMAT A LINE FOR  
FOR THE SET CLAUSE.

INHERITED\_ATT\_USE      NOTE: THE KC\_NO CARRIED IN  
IS ASSOCIATED WITH THE DEPENDENT ENTITY,  
NOT THE INDEPENDENT.

SYNOPSIS

```
C          -- SELIAUC(MODEL_NO,EC_NO,KC_NO,RC_NO) ;
COBOL      -- CALL "SELIAUC"  USING
                                MODEL-NO
                                EC-NO
                                KC_NO
                                RC-NO.
FORTRAN    -- CALL SELIAUC(MODELNO,ECNO,KCNO,RCNO)
```

INPUT:

```
INT  *MODEL_NO ;
INT  *EC_NO ;
INT  *KC_NO ;
INT  *RC_NO ;
```

DESCRIPTION:

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

MODEL\_NO  
EC\_NO  
KC\_NO  
RC\_NO

INT \*  
INT \*  
INT \*  
INT \*

INCLUDE FILES:

-----

EOD

ROUTINES CALLED:

-----

SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLCLS  
ERRRPT  
SQLAD2  
SQLFCH  
FMTIAUC  
STRNCPY  
SQLTOC  
SQLSQS  
SQLOSQ  
SQLAB2  
SQLEXE

DOCGROUP PS41100 Module Documentation

NAME: SELSET  
PURPOSE: SELECT A RELATIONS IN THE CDM FOR A GIVEN  
LANGUAGE: C  
SOURCE FILE: SELSET  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
DB-NAME AND SET-ID AND FOR EACH  
ONE SELECTED CALL GENMREL TO START GENERATING  
CREATE MAP FOR SET COMMANDS.

SYNOPSIS

C -- SELSET(&DB-NAME, &SET-ID)  
COBOL -- CALL "SELSET" USING DB-NAME  
SET-ID.  
FORTRAN -- CALL SELSET (DBNAME, SETID)

INPUT:

CHAR \*DB\_NAME;  
CHAR \*SET\_ID;

OUTPUT:

DESCRIPTION

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
DB\_NAME CHAR \*  
SET\_ID CHAR \*

INCLUDE FILES:

-----

EOD

ROUTINES CALLED:

-----

STRNCPY  
ADDPARM  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLCLS  
ERRRPT  
SQLAD2  
SQLFCH  
GENMREL  
SQLTOC  
SQLSQS  
SQLOSQ  
SQLAB2  
SQLEXE



DOCGROUP PS41100 Module Documentation

NAME: SELSETC  
PURPOSE: SELECT A RELATIONS IN THE CDM FOR A GIVEN  
LANGUAGE: C  
SOURCE FILE: SELSETC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
DB-NAME AND SET-ID AND FOR EACH  
ONE SELECTED CALL GENMREL TO START GENERATING  
CREATE MAP FOR SET COMMANDS.

SYNOPSIS

C -- SELSETC(&DB-NAME, &SET-ID)  
COBOL -- CALL "SELSETC" USING DB-NAME  
SET-ID.  
FORTRAN -- CALL SELSETC (DBNAME, SETID)

INPUT:

CHAR \*DB\_NAME;  
CHAR \*SET\_ID;

OUTPUT:

DESCRIPTION  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
DB\_NAME CHAR \*  
SET\_ID CHAR \*

INCLUDE FILES:

-----  
EOD

ROUTINES CALLED:

-----  
STRNCPY  
ADDPARM  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLCLS  
ERRRPT  
SQLAD2  
SQLFCH  
GENMREL  
SQLTOC  
SQLSQS  
SQLOSQ  
SQLAB2  
SQLEXE

DOCGROUP PS41100 Module Documentation

NAME: SETCOMM  
PURPOSE: SET COMMIT OPTION TO MANUAL OR AUTOMATIC  
LANGUAGE: C  
SOURCE FILE: SETCOMM  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

```
C          --  SETCOMM()  ;  
  
COBOL      --  CALL "SETCOMM"  
FORTRAN    --  CALL SETCOMM ()
```

INPUT:

NONE

OUTPUT:

DESCRIPTION:

RETRIEVE FROM THE PARSER LIST WHETHER THE COMMIT/ROLLBACK  
IS AUTOMATIC OR MANUAL. THE DEFAULT OPTION IS AUTOMATIC.  
UPDATE GLOBAL VARIABLES TO REFLECT THE CURRENT OPTION.

INCLUDE FILES:  
-----

LISTID  
NDDLEX

ROUTINES CALLED:  
-----

CPFONE

PS 620341100  
30 September 1990

STRNCMP

DOCGROUP PS41100 Module Documentation

NAME: SETGLOB  
PURPOSE: THIS ROUTINE ESTABLISHES GLOBAL OUTPUT  
LANGUAGE: C  
SOURCE FILE: SETGLOB  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
VARIABLES FOR THE SET OUTPUT COMMAND.

SYNOPSIS

```
      C          -- SETGLOB(FILE_FLAG,FILENAME) ;  
      COBOL       -- CALL "SETGLOB" USING  
                      FILE_FLAG,FILENAME.  
      FORTRAN    -- CALL SETGLOB(FILE_FLAG,FILENAME)
```

INPUT:

```
      INT *FILE_FLAG ;  
      CHAR *FILENAME ;
```

OUTPUT:

DESCRIPTION:

FILE OPN APP FLAG MAY HAVE 3 VALUES

- 0 : OUTPUT GENERATED COMMANDS TO THE SCREEN
- 1 : OUTPUT GENERATED COMMANDS TO A NEW USER DEFINED FILE
- 2 : APPEND GENERATED COMMANDS TO A USER DEFINED FILE

ARGUMENTS:

-----  
FILE\_FLAG                   INT \*  
FILENAME                    CHAR \*

INCLUDE FILES:

-----  
STDIO  
NDDLEX

ROUTINES CALLED:

-----  
NFOPEN  
NFLCLOS  
SPRINTF  
UERROR

DOCGROUP PS41100 Module Documentation

NAME: STRTLST  
PURPOSE: CREATE A NEW LINKED LIST STRUCTURE.  
LANGUAGE: C  
SOURCE FILE: STRTLST  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

INPUT:

LLIST1 \*(\*NEWLIST);  
INT NEWREF;

CALLED BY:

BLOOPCK, TLOOPCK

DESCRIPTION:

THIS PROGRAM WILL ALLOCATE MEMORY FOR THE HEAD OF A NEW  
LINKED  
LIST STRUCTURE, AND SET UP A POINTER TO THIS.

ARGUMENTS:  
-----

NEWLIST  
NEWREF

LLIST1 \*\*  
INT

INCLUDE FILES:  
-----

STDIO  
LLIST

PS 620341100  
30 September 1990

ROUTINES CALLED:

-----  
MALLOC



DOCGROUP PS41100 Module Documentation

NAME: TERMSES  
PURPOSE: ROUTINE TO TERMINATE AN NDDL  
LANGUAGE: C  
SOURCE FILE: TERMSES  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

SESSION.

SYNOPSIS

|          |    |                 |
|----------|----|-----------------|
| C        | -- | TERMSES();      |
| COBOL    | -- | CALL "TERMSES". |
| FORTTRAN | -- | CALL TERMSES    |

INPUT:

OUTPUT:

DESCRIPTION

IF INTERACTIVE OR UIMS THE FORMS PROCESSOR SHOULD BE  
TERMINATED.

CURRENTLY, THESE TWO OPTIONS ARE ONLY AVAILABLE ON THE  
VAX.

AND LOGOFF FROM THE CDM.

MODIFIED: 23 SEP, 1988 CHANGE COLOGOF TO OLOGOF AND  
ADDED IFDEF

INCLUDE FILES:

-----

NDDLEX

ROUTINES CALLED:

-----

TERMFP

DOCGROUP PS41100 Module Documentation

NAME: TLOOPCK  
PURPOSE: CHECK FOR LOOPS FROM THE ENTITY GIVEN DOWN THE  
HIERARCHY  
LANGUAGE: C  
SOURCE FILE: TLOOPCK  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

|          |    |                     |        |
|----------|----|---------------------|--------|
| C        | -- | TLOOPCK(EC_NO)      |        |
| COBOL    | -- | CALL "TLOOPCK"      | USING  |
|          |    |                     | EC-NO. |
| FORTTRAN | -- | CALL TLOOPCK (ECNO) |        |

INPUT: INT EC\_NO THE ENTITY CLASS FOR WHICH A DOWNWARD  
HIERARCHY WILL BE SEARCHED

OUTPUT:  
WARNING MESSAGES WILL BE ISSUED  
STATUS SET TO -1 IF A LOOP IS FOUND  
LEFT ALONE OTHERWISE

\* DESCRIPTION  
USE A SQL TREE SEARCH FOR ALL RELATIONS BELOW THIS  
ENTITY, IF A LOOP  
IN THE USER DATA IS ENCOUNTERED, REPORT AN ERROR TO  
THE USER.  
REWRITTEN TO PROCC DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE  
3/14/89: REWRITTEN TO TAKE OUT 'CONNECT BY PRIOR' TO MAKE IT  
ALL  
STANDARD SQL.  
8/30/89: MODIFIED TO HANDLE CATEGORY RELATIONS AS WELL AS  
LINK RELATIONS

ARGUMENTS:

-----  
EC NO  
STATUS

INT \*  
INT \*

INCLUDE FILES:

-----  
STDIO  
EOD  
LLIST

ROUTINES CALLED:

-----  
SQLAB2  
SQLAD2  
SQLBS2  
SQLCLS  
FRELST1  
STRCPY  
ADDPARM  
ERRRPT  
SQLEXE  
SQLFCH  
LOOPDEP  
ADDNODE  
STRTLST  
SQLOSQ  
SQLSCA  
SQLSCC  
SQLSCH  
SQLTOC

DOCGROUP PS41100 Module Documentation

NAME: TOPNODE  
PURPOSE: SELECT ALL THE LEVEL 1 TOP NODE INDEPENDENT ENTITY  
LANGUAGE: C  
SOURCE FILE: TOPNODE  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

ENTITY                    CLASSES IN THE FROM\_MODEL AND GENERATE ALTER  
                          STATEMENTS TO ADD THEIR KEY CLASSES.

SYNOPSIS

      C                    -- TOPNODE(&MODEL NO,&RC) ;  
      COBOL                -- CALL "TOPNODE"        USING  
                                                      MODEL-NO,  
                                                      RC.  
      FORTRAN             -- CALL TOPNODE(MODELNO, RC)

INPUT:

      INT \*MODEL NO ;  
      INT \*STATUS ;

DESCRIPTION:

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

MODEL NO                                    INT \*  
STATUS                                      INT \*

INCLUDE FILES:

-----  
EOD

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLCLS  
ERRRPT  
SQLAD2  
SQLFCH  
GETECNM  
GENALTE  
SQLTOC  
SQLSQS  
SQLOSQ  
SQLAB2  
SQLEXE



INCLUDE FILES:

-----  
CSTDYTP  
LISTID

ROUTINES CALLED:

-----  
CPFONE

DOCGROUP PS41100 Module Documentation

NAME: UINDDL/MAIN  
PURPOSE: MAIN PROGRAM FOR THE NDDL COMMAND PROCESSOR  
LANGUAGE: C  
SOURCE FILE: UINDDL  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

THE PROGRAM WILL USE THE FORMS PROCESSOR

\*\*\*\*\* UIMS VERSION \*\*\*\*\*  
INPUT:  
OUTPUT:  
DESCRIPTION  
THIS IS THE MAIN FOR THE NDDL COMMAND PROCESSOR.

ARGUMENTS:  
-----

|      |           |
|------|-----------|
| ARGC | INT       |
| ARGV | CHAR * [] |

INCLUDE FILES:  
-----

SIGNAL  
CMDID  
LISTS  
NDDLEX  
STDIO



ROUTINES CALLED:

-----  
ERRTRAP  
STRNCPY  
SIGNAL  
INITAL  
STRNCMP  
ERRPRO  
EXIT  
INITSES  
PROCMD  
TERMSSES  
TRMDML

DOCGROUP PS41100 Module Documentation

NAME: ERRTRAP  
PURPOSE:  
LANGUAGE: C  
SOURCE FILE: UINDDL  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

ARGUMENTS:

-----

XXX

INT

INCLUDE FILES:

-----

SIGNAL  
CMDID  
LISTS  
NDDLEX  
STDIO

ROUTINES CALLED:

-----

TRMDML  
EXIT

DOCGROUP PS41100 Module Documentation

NAME: UPDAC  
PURPOSE: UPDATE ATTRIBUTE\_CLASS SET DOMAIN\_NO = : 1  
LANGUAGE: C  
SOURCE FILE: UPDAC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

WHERE AC\_NO = :2 ;

SYNOPSIS

C -- UPDAC(&DOMAIN\_NO, &AC\_NO, &STATUS)  
COBOL -- CALL "UPDAC" USING  
DOMAIN-NO,  
AC-NO,  
STATUS.  
FORTRAN -- CALL UPDAC(DOMAINNO, ACNO, STATUS)

INPUT:

INT \*DOMAIN\_NO ;  
INT \*AC\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
ATTRIBUTE\_NAME. THE DATA ARE TAKEN FROM THE INPUT  
PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
DOMAIN\_NO  
AC\_NO  
STATUS

INT \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDACAL  
PURPOSE: UPDATE MODEL\_CLASS SET AC\_NAME\_TYPE = : 1  
LANGUAGE: C  
SOURCE FILE: UPDACAL  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

WHERE AC\_NO = : 2 AND  
OBJ\_ID = :3 ;

SYNOPSIS

C -- UPDACAL(AC\_NAME\_TYPE, &AC\_NO, OBJ\_ID,  
&STATUS) ;  
COBOL -- CALL "UPDACAL" USING  
EC-NAME-TYPE,  
EC-NO,  
OBJ-ID,  
STATUS.

FORTRAN -- CALL UPDACAL(ECNAMETYPE, ECNO, OBJID,  
STATUS)

INPUT:

CHAR \*OBJ\_ID ;  
INT \*AC\_NO ;  
CHAR \*AC\_NAME\_TYPE ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
ENTITY\_NAME

TABLE. THE DATA ARE TAKEN FROM INPUT PARAMETERS.

IF THE RECORD IS SUCCESSFULLY INSERTED, THE STATUS AND  
RETURN VALUE BOT

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
AC\_NAME\_TYPE  
AC\_NO  
OBJ\_ID  
STATUS

CHAR \*  
INT \*  
CHAR \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDACNM  
PURPOSE: UPDATE ATTRIBUTE\_NAME SET AC\_NAME = : 1  
LANGUAGE: C  
SOURCE FILE: UPDACNM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

WHERE AC\_NAME = : 2 AND  
AC\_NO = :3;

SYNOPSIS

C -- UPDACNM(AC\_NAME, AC\_NAME, &STATUS) ;  
COBOL -- CALL "UPDACNM" USING  
AC-NAME,  
AC-NO,  
STATUS.

FORTTRAN -- CALL UPDACNM(ACNAME, ACNAME, ACNO, STATUS)

INPUT:

CHAR \*AC\_NAME ;  
INT \*AC\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
ATTRIBUTE NAME

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
NEW AC NAME  
AC\_NAME  
AC\_NO  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: UPDAPP  
PURPOSE: RENAME THE HOST-ID IN APPLICATION CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDAPP  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDAPP(NEW\_ID, OLD\_ID, STATUS)

COBOL -- CALL "UPDAPP" USING NEW-ID  
                                  OLD-ID  
                                  STATUS.

FORTRAN -- CALL UPDAPP (NEW-ID,OLD-ID,STATUS)

INPUT:

NEW\_ID

OLD\_ID

OUTPUT:

STATUS

DESCRIPTION

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE HOST-ID IN THE APPLICATION CDM TABLE.  
IF SUCCESSFUL, THE STATUS WILL BE 0.

ARGUMENTS:  
-----

NEW\_ID

CHAR \*

OLD\_ID

CHAR \*

STATUS

INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDAUC  
PURPOSE: RENAME THE SET\_ID IN AUC\_ST\_MAPPING TABLE  
LANGUAGE: C  
SOURCE FILE: UPDAUC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDAUC(NEW\_ID, OLD\_ID, DB\_NO, STATUS)

COBOL -- CALL "UPDAUC" USING NEW-ID  
OLD-ID  
DB-ID  
STATUS.

FORTRAN -- CALL UPDAUC (NEW-ID,OLD-ID,DB-ID,STATUS)

INPUT:

NEW\_ID  
OLD\_ID  
DB\_ID

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE SET-ID IN THE AUC\_ST\_MAPPING CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

PS 620341100  
30 September 1990

NEW\_ID  
OLD\_ID  
DB\_NO  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDAUCE  
PURPOSE: MODIFY THE AUC OCCURENCE TO A DIFFERENT ENTITY  
LANGUAGE: C  
SOURCE FILE: UPDAUCE  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

SYNOPSIS

C -- UPDAUCE(TAG\_NO\_OLD, EC\_NO\_NEW, STATUS)

COBOL -- CALL "UPDAUCE" USING TAG-NO-OLD  
EC-NO-NEW  
STATUS.

FORTRAN -- CALL UPDAUCE (TAG-NO-OLD,EC-NO-NEW,STATUS)

INPUT:

TAG NO OLD

EC NO NEW

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE MODIFY  
THE ATTRIBUTE USE OCCURENCE TO A DIFFERENT ENTITY IN THE  
CDM TABLE ATTRIBUTE USE CL.

(CALLED BY ALTER ATTRIBUTE OWNERSHIP..)

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

PS 620341100  
30 September 1990

TAG NO OLD  
EC NO NEW  
STATUS

INT \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDCRNM  
PURPOSE: UPDATE RELATION\_CLASS SET RC\_NAME\_PTR = : 1  
LANGUAGE: C  
SOURCE FILE: UPDCRNM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
WHERE RC\_NO\_PTR = : 2 ;

SYNOPSIS

C -- UPDCRNM(RC\_NAME\_PTR, &RC\_NO\_PTR, &STATUS) ;  
COBOL -- CALL "UPDCRNM" USING  
RC-NAME,  
RC-NO,  
STATUS.

FORTRAN -- CALL UPDCRNM(RCNAME, RCNO, STATUS)

INPUT:

CHAR \*RC\_NAME\_PTR ;  
INT \*RC\_NO\_PTR ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
CATEGORY\_RELATION.

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND

RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
RC\_NAME PTR  
RC\_NO PTR  
STATUS

CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: UPDDAA  
PURPOSE: RENAME THE RT\_ID IN DB\_AREA\_ASSIGNMENT CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDDAA  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C -- UPDDAA(NEW\_ID, OLD\_ID, DB\_NO, STATUS)  
  
COBOL -- CALL "UPDDAA" USING NEW-ID  
                                  OLD-ID  
                                  DB-NO  
                                  STATUS.  
  
FORTRAN -- CALL UPDDAA (NEW-ID,OLD-ID,DB-NO,STATUS)

INPUT:

NEW\_ID  
OLD\_ID  
DB NO

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE RT-ID IN THE DB AREA ASSIGNMENT CDM TABLE.  
IF SUCCESSFUL, THE STATUS WILL BE 0.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

PS 620341100  
30 September 1990

NEW\_ID  
OLD\_ID  
DB\_NO  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDDB  
PURPOSE: RENAME THE HOST-ID IN DATA\_BASE CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDDB  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

SYNOPSIS

C -- UPDDB(NEW\_ID, OLD\_ID, STATUS)

COBOL -- CALL "UPDDB" USING NEW-ID  
                                  OLD-ID  
                                  STATUS.

FORTRAN -- CALL UPDDB (NEW-ID,OLD-ID,STATUS)

INPUT:

NEW\_ID  
OLD\_ID

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE HOST-ID IN THE DATA\_BASE CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

NEW\_ID  
OLD\_ID

CHAR \*  
CHAR \*

STATUS

INT \*

ROUTINES CALLED:

-----

STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDDBH  
PURPOSE: RENAME THE HOST-ID IN DBMS\_ON\_HOST CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDDBH  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

      C          -- UPDDBH(NEW\_ID, OLD\_ID, STATUS)  
  
      COBOL      -- CALL "UPDDBH" USING NEW-ID  
                                  OLD-ID  
                                  STATUS.  
      FORTRAN -- CALL UPDDBH (NEW-ID,OLD-ID,STATUS)

INPUT:

      NEW\_ID  
      OLD\_ID

OUTPUT:

      STATUS

DESCRIPTION

      THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
      THE HOST-ID IN THE DBMS\_ON\_HOST CDM TABLE.  
      IF SUCCESSFUL, THE STATUS WILL BE 0.  
      REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

NEW\_ID                          CHAR \*  
OLD\_ID                          CHAR \*

PS 620341100  
30 September 1990

STATUS

INT \*

ROUTINES CALLED:

-----

STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDDBN  
PURPOSE: RENAME THE DB\_NAME IN DATA\_BASE CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDDBN  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDDBN(NEW\_ID, OLD\_ID, STATUS)

COBOL -- CALL "UPDDBN" USING NEW-ID  
OLD-ID  
STATUS.

FORTRAN -- CALL UPDDBN (NEW-ID,OLD-ID,STATUS)

INPUT:

NEW\_ID  
OLD\_ID

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE DB\_NAME IN THE DATA\_BASE CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

NEW\_ID  
OLD\_ID

CHAR \*  
CHAR \*

STATUS

INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: UPDDDI  
PURPOSE: RENAME THE DATA\_TYPE\_NAME IN DATA\_ITEM CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDDDI  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDDDI(NEW\_ID, OLD\_ID, STATUS)

COBOL -- CALL "UPDDDI" USING NEW-ID  
OLD-ID  
STATUS.

FORTRAN -- CALL UPDDDI (NEW-ID,OLD-ID,STATUS)

INPUT:

NEW\_ID  
OLD\_ID

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE DATA\_TYPE\_NAME IN THE DATA\_ITEM CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

|        |        |
|--------|--------|
| NEW_ID | CHAR * |
| OLD_ID | CHAR * |

STATUS

INT \*

ROUTINES CALLED:

-----

STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDDF  
PURPOSE: RENAME THE RT-ID IN DATA\_FIELD CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDDF  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDDF(NEW\_ID, OLD\_ID, DB\_NO, STATUS)

COBOL -- CALL "UPDDF" USING NEW-ID  
                                OLD-ID  
                                DB-NO  
                                STATUS.

FORTRAN -- CALL UPDDF (NEW-ID,OLD-ID,DB-NO,STATUS)

INPUT:

NEW\_ID  
OLD\_ID  
DB\_NO

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE RT\_ID IN THE DATA\_FIELD CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

PS 620341100  
30 September 1990

NEW\_ID  
OLD\_ID  
DB\_NO  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDDFU  
PURPOSE: RENAME THE SET\_ID IN DF\_USED\_AS\_SET\_LINKAGE TABLE  
LANGUAGE: C  
SOURCE FILE: UPDDFU  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C -- UPDDFU(NEW\_ID, OLD\_ID, DB\_NO, STATUS)

COBOL -- CALL "UPDDFU" USING NEW-ID  
OLD-ID  
DB-NO  
STATUS.

FORTRAN -- CALL UPDDFU (NEW-ID,OLD-ID,DB-NO,STATUS)

INPUT:

NEW\_ID  
OLD\_ID  
DB\_NO

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE SET-ID IN THE DF USED AS SET LINKAGE CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

NEW\_ID  
OLD\_ID  
DB\_NO  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
• STRNCPY  
SQLSCA  
SQLBS2  
• SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

# DOCGROUP PS41100 Module Documentation

NAME: UPDDI  
PURPOSE: RENAME THE DI-ID IN DATA\_ITEM CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDDI  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

**DESCRIPTION:**

## SYNOPSIS

```
C      -- UPDDI (NEW_ID, OLD_ID, STATUS)
```

```
COBOL      -- CALL "UPDDI" USING NEW-ID
                                OLD-ID
                                STATUS.
```

**FORTRAN -- CALL UPDDI (NEW-ID,OLD-ID,STATUS)**

**INPUT:**

```
NEW_ID
OLD_ID
VIEW NO
```

**OUTPUT:**

**STATUS****DESCRIPTION**

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE DI\_ID IN THE DATA\_ITEM CDM TABLE.  
IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

**ARGUMENTS:**

NEW ID CHAR \*

PS 620341100  
30 September 1990

OLD ID  
VIEW NO  
STATUS

CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: UPDECAL  
PURPOSE: UPDATE MODEL\_CLASS SET EC\_NAME\_TYPE = : 1  
LANGUAGE: C  
SOURCE FILE: UPDECAL  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
WHERE EC\_NO = : 2 AND  
OBJ\_ID = :3 ;

SYNOPSIS

C -- UPDECAL(EC\_NAME\_TYPE, &EC\_NO, OBJ\_ID,  
&STATUS) ;  
COBOL -- CALL "UPDECAL" USING  
EC-NAME-TYPE,  
EC-NO,  
OBJ-ID,  
STATUS.

FORTRAN -- CALL UPDECAL(ECNAMETYPE, ECNO, OBJID,  
STATUS)

INPUT:

CHAR \*OBJ\_ID ;  
INT \*EC\_NO ;  
CHAR \*EC\_NAME\_TYPE ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
ENTITY NAME

TABLE. THE DATA ARE TAKEN FROM INPUT PARAMETERS.

IF THE RECORD IS SUCCESSFULLY INSERTED, THE STATUS AND  
RETURN VALUE BOT

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
EC\_NAME\_TYPE  
EC\_NO  
OBJ\_ID  
STATUS

CHAR \*  
INT \*  
CHAR \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDECNM  
PURPOSE: UPDATE MODEL\_CLASS SET ENTITY\_NAME = : 1  
LANGUAGE: C  
SOURCE FILE: UPDECNM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
WHERE EC\_NAME = : 2 AND  
EC\_NO = :3 ;

SYNOPSIS

C  
UPDECNM(NEW\_EC\_NAME, EC\_NAME, &EC\_NO, &STATUS) ;  
COBOL -- CALL "UPDECNM" USING  
NEW-EC-NAME,  
EC-NAME,  
EC-NO,  
STATUS.  
FORTRAN -- CALL UPDECNM(NEWECNAME, ECNAME, ECNO, STATUS)

INPUT:

CHAR \*NEW\_EC\_NAME ;  
CHAR \*EC\_NAME ;  
INT \*EC\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
ENTITY NAME

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
NEW EC NAME  
EC\_NAME  
EC\_NO  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDEDF  
PURPOSE: RENAME THE DATA\_TYPE\_NAME IN ELEMENTARY\_DATA\_FIELD  
TABLE  
LANGUAGE: C  
SOURCE FILE: UPDEDF  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDEDF(NEW\_ID, OLD\_ID, STATUS)

COBOL -- CALL "UPDEDF" USING NEW-ID  
OLD-ID  
STATUS.

FORTRAN -- CALL UPDEDF (NEW-ID,OLD-ID,STATUS)

INPUT:

NEW\_ID  
OLD\_ID

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE DATA\_TYPE\_NAME IN THE (ELEMENTARY)DATA\_FIELD CDM TABLE.  
IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

NEW\_ID CHAR \*

PS 620341100  
30 September 1990

OLD ID  
STATUS

CHAR \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDFDF  
PURPOSE: RENAME THE DF-ID IN DATA\_FIELD CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDFDF  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C -- UPDFDF(NEW\_ID, OLD\_ID, RT-ID, DB\_NO, STATUS)

COBOL -- CALL "UPDFDF" USING NEW-ID  
OLD-ID  
RT-ID  
DB-NO  
STATUS.

FORTRAN -- CALL UPDFDF (NEW-ID,OLD-ID,RT-ID,DB-NO,STATUS)

INPUT:

NEW\_ID  
OLD\_ID  
RT\_ID  
DB\_NO

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE DF\_ID IN THE DATA\_FIELD CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

|        |        |
|--------|--------|
| NEW_ID | CHAR * |
| OLD_ID | CHAR * |
| RT_ID  | CHAR * |
| DB_NO  | INT *  |
| STATUS | INT *  |

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: UPDFDFU  
PURPOSE: RENAME THE DF\_ID IN DF\_USED\_AS\_SET\_LINKAGE TABLE  
LANGUAGE: C  
SOURCE FILE: UPDFDFU  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C -- UPDFDFU(NEW\_ID, OLD\_ID, RT\_ID, DB\_NO, STATUS)

COBOL -- CALL "UPDFDFU" USING NEW-ID  
                                  OLD-ID  
                                  RT-ID  
                                  DB-NO  
                                  STATUS.

FORTRAN -- CALL UPDFDFU (NEW-ID,OLD-ID,RT-ID,DB-NO,STATUS)

INPUT:

NEW\_ID  
OLD\_ID  
RT-ID  
DB\_NO

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE DF-ID IN THE DF\_USED\_AS\_SET LINKAGE CDM TABLE.  
IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

|        |        |
|--------|--------|
| NEW_ID | CHAR * |
| OLD_ID | CHAR * |
| RT_ID  | CHAR * |
| DB_NO  | INT *  |
| STATUS | INT *  |

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDFPDF  
PURPOSE: RENAME THE DF\_ID IN PROJECT\_DATA\_FIELD CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDFPDF  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDFPDF(NEW\_ID, OLD\_ID, RT\_ID, DB\_NO, STATUS)

COBOL -- CALL "UPDFPDF" USING NEW-ID  
                                  OLD-ID  
                                  RT-ID  
                                  DB-NO  
                                  STATUS.

FORTTRAN -- CALL UPDFPDF (NEW-ID,OLD-ID,RT-ID,DB-NO,STATUS)

INPUT:

NEW\_ID  
OLD\_ID  
RT\_ID  
DB\_NO

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE DF-ID IN THE PROJECT DATA FIELD CDM TABLE.  
IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
NEW\_ID  
OLD\_ID  
RT\_ID  
DB\_NO  
STATUS

CHAR \*  
CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDHST  
PURPOSE: RENAME THE HOST-ID IN HOST CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDHST  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

      C          -- UPDHST(NEW\_ID, OLD\_ID, STATUS)  
  
      COBOL      -- CALL "UPDHST" USING NEW-ID  
                                OLD-ID  
                                STATUS.  
      FORTRAN   -- CALL UPDHST (NEW-ID,OLD-ID,STATUS)

INPUT:

      NEW\_ID  
      OLD\_ID

OUTPUT:

      STATUS

DESCRIPTION

      THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
      THE HOST-ID IN THE HOST CDM TABLE.  
      IF SUCCESSFUL, THE STATUS WILL BE 0.  
      REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

NEW\_ID                                CHAR \*  
OLD\_ID                                CHAR \*

STATUS

INT \*

ROUTINES CALLED:

-----

STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDKCT  
PURPOSE: UPDATE KEY\_CLASS.KEY\_TYPE  
LANGUAGE: C  
SOURCE FILE: UPDKCT  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

```
C          --  UPDKCT(&KC_NO, KEY_TYPE, &STATUS) ;

COBOL      --  CALL "UPDKCT"  USING
                    KC-NO,
                    KC-TYPE,
                    STATUS.

FORTRAN    --  CALL UPDKCT(KCNO, KCTYPE, STATUS)
```

INPUT:

```
INT  *KC_NO ;
CHAR *KC_TYPE ;
```

OUTPUT:

```
INT  *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
KEY\_CLASS

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

PS 620341100  
30 September 1990

KC\_NO  
KC\_TYPE  
STATUS

INT \*  
CHAR \*  
INT \*

ROUTINES CALLED:

-----

STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: UPDLRNM  
PURPOSE: UPDATE RELATION\_CLASS SET RC\_NAME\_PTR = : 1  
LANGUAGE: C  
SOURCE FILE: UPDLRNM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

WHERE RC\_NO\_PTR = : 2 ;

SYNOPSIS

```
C          -- UPDLRNM(RC_NAME_PTR, &RC_NO_PTR, &STATUS) ;
COBOL      -- CALL "UPDLRNM" USING
              RC-NAME,
              RC-NO,
              STATUS.
FORTRAN    -- CALL UPDLRNM(RCNAME, RCNO, STATUS)
```

INPUT:

```
CHAR *RC_NAME_PTR ;
INT  *RC_NO_PTR ;
```

OUTPUT:

```
INT *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
LINK\_RELATION

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.  
IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
RC\_NAME\_PTR  
RC\_NO\_PTR  
STATUS

CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDMNAM  
PURPOSE: UPDATE MODEL\_CLASS SET MODEL\_NAME = : 1  
LANGUAGE: C  
SOURCE FILE: UPDMNAM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

WHERE MODEL\_NO = : 2 ;

SYNOPSIS

```
C          -- UPDMNAM(MODEL_NAME, &MODEL_NO, &STATUS) ;
COBOL      -- CALL "UPDMNAM" USING
                MODEL-NAME,
                MODEL-NO,
                STATUS.
FORTRAN    -- CALL UPDMNAM(MODELNAME, MODELNO, STATUS)
```

INPUT:

```
CHAR *MODEL_NAME ;
INT  *MODEL_NO ;
```

OUTPUT:

```
INT *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
MODEL CLASS

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
MODEL\_NAME  
MODEL\_NO  
STATUS

CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
CDSDATE  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDMOD  
PURPOSE: UPDATE MODEL\_CLASS SET MODEL\_STATUS = : 1  
LANGUAGE: C  
SOURCE FILE: UPDMOD  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
DATA MODIFIED = SYSDATE  
WHERE MODEL\_NO = : 2 ;

SYNOPSIS

C -- UPDMOD(MODEL\_STATUS, &MODEL\_NO, &STATUS) ;  
COBOL -- CALL "UPDMOD" USING  
MODEL-STATUS,  
MODEL-NO,  
STATUS.  
FORTRAN -- CALL UPDMOD(MODELSTATUS, MODELNO, STATUS)

INPUT:

CHAR \*MODEL\_STATUS ;  
INT \*MODEL\_NO ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
MODEL CLASS

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C 5/6/88

ARGUMENTS:

-----  
MODEL\_STATUS  
MODEL\_NO  
STATUS

CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
CDSDATE  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDMP  
PURPOSE: RENAME THE DATA\_TYPE\_NAME IN MODULE\_PARAMETER CDM  
TABLE  
LANGUAGE: C  
SOURCE FILE: UPDMP  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDMP(NEW\_ID, OLD\_ID, STATUS)

COBOL -- CALL "UPDMP" USING NEW-ID  
OLD-ID  
STATUS.

FORTRAN -- CALL UPDMP (NEW-ID,OLD-ID,STATUS)

INPUT:

NEW\_ID  
OLD\_ID

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE DATA\_TYPE\_NAME IN THE MODULE\_PARAMETER CDM TABLE.  
IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

NEW\_ID

CHAR \*

PS 620341100  
30 September 1990

OLD ID  
STATUS

CHAR \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: UPDNXNO  
PURPOSE: UPDATE THE POOL TO RETURN NEXT AVAILABLE NO  
LANGUAGE: C  
SOURCE FILE: UPDNXNO  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C -- UPDNXNO (POOL\_NO\_PTR, NEXT\_NO\_PTR, STATUS)  
  
COBOL -- CALL "UPDNXNO" USING  
                                POOL-NO,  
                                NEXT\_NO,  
                                STATUS.  
  
FORTRAN -- CALL UPDNXNO (POOLNO,NEXT\_NO,STATUS)

INPUT:

INT \*POOL\_NO\_PTR ;  
INT \*NEXT\_NO\_PTR ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION

THIS ROUTINE GET A UNIQUE NUMBER OF A POOL NUMBER  
FROM NEXT NUMBER RELATION  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

POOL\_NO\_PTR  
NEXT\_NO\_PTR

INT \*  
INT \*

STATUS

INT \*

ROUTINES CALLED:

-----

ERRRPT  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR

DOCGROUP PS41100 Module Documentation

NAME: UPDPDF  
PURPOSE: RENAME THE RT\_ID IN PROJECT\_DATA\_FIELD CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDPDF  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDPDF(NEW\_ID\_PTR, OLD\_ID\_PTR, DB\_NO\_PTR,  
STATUS)

COBOL -- CALL "UPDPDF" USING NEW-ID  
OLD-ID  
DB-NO  
STATUS.

FORTAN -- CALL UPDPDF (NEW-ID,OLD-ID,DB-NO,STATUS)  
INPUT:

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE RT-ID IN THE PROJECT DATA FIELD CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

PS 620341100  
30 September 1990

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDPSB  
PURPOSE: RENAME THE HOST-ID IN PSB CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDPSB  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDPSB(NEW\_ID\_PTR, OLD\_ID\_PTR, STATUS)

COBOL -- CALL "UPDPSB" USING NEW-ID  
                                  OLD-ID  
                                  STATUS.

FORTRAN -- CALL UPDPSB (NEW-ID,OLD-ID,STATUS)

INPUT:

NEW\_ID\_PTR

OLD\_ID\_PTR

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE HOST-ID IN THE PSB CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

NEW\_ID\_PTR

CHAR \*

OLD\_ID\_PTR

CHAR \*

STATUS

INT \*

ROUTINES CALLED:

-----

STRNCPY

SQLSCA

SQLBS2

SQLSCH

SQLSCC

SQLTFL

SQLOPN

SQLSQ

SQLAB2

SQLQX

SQLWNR

ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDRAUC  
PURPOSE: RENAME THE RT\_ID IN AUC\_IS\_MAPPING TABLE  
LANGUAGE: C  
SOURCE FILE: UPDRAUC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDRAUC(NEW\_ID\_PTR, OLD\_ID\_PTR, DB\_NO\_PTR,  
STATUS)

COBOL -- CALL "UPDRAUC" USING NEW-ID  
                                  OLD-ID  
                                  DB-NO  
                                  STATUS.

FORTRAN -- CALL UPDRAUC (NEW-ID,OLD-ID,DB-NO,STATUS)

INPUT:

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE RT-ID IN THE AUC\_IS\_MAPPING CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

PS 620341100  
30 September 1990

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: UPDRCB  
PURPOSE: RENAME THE SET\_ID IN RC\_BASED\_REC\_SET TABLE  
LANGUAGE: C  
SOURCE FILE: UPDRCB  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDRCB(NEW\_ID\_PTR, OLD\_ID\_PTR, DB\_NO\_PTR,  
STATUS)

COBOL -- CALL "UPDRCB" USING NEW-ID  
OLD-ID  
DB-NO  
STATUS.

FORTTRAN -- CALL UPDRCB (NEW-ID,OLD-ID,DB-NO,STATUS)

INPUT:

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE SET ID IN THE RC BASED REC SET CDM TABLE.  
IF SUCCESSFUL, THE STATUS WILL BE 0.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

PS 620341100  
30 September 1990

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDRCNM  
PURPOSE: UPDATE RELATION\_CLASS SET RC\_NAME = : 1  
LANGUAGE: C  
SOURCE FILE: UPDRCNM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

WHERE RC\_NO = : 2 ;

SYNOPSIS

```
C          -- UPDRCNM(RC_NAME, &RC_NO, &STATUS) ;
COBOL      -- CALL "UPDRCNM" USING
                        RC-NAME,
                        RC-NO,
                        STATUS.
FORTRAN    -- CALL UPDRCNM(RCNAME, RCNO, STATUS)
```

INPUT:

```
CHAR *RC_NAME ;
INT  *RC_NO  ;
```

OUTPUT:

```
INT *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES CALLS TO DETERMINE THE RELATION TYPE  
AND THEN UPDATES

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

PS 620341100  
30 September 1990

ARGUMENTS:

-----  
RC\_NAME  
RC\_NO  
STATUS

CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
VERRCT  
STRNCMP  
UPDCRNM  
UPDLRNM

DOCGROUP PS41100 Module Documentation

NAME: UPDRDFU  
PURPOSE: RENAME THE RT\_ID IN DF\_USED\_AS\_SET\_LINKAGE TABLE  
LANGUAGE: C  
SOURCE FILE: UPDRDFU  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDRDFU(NEW\_ID\_PTR, OLD\_ID\_PTR, DB\_NO\_PTR,  
STATUS)

COBOL -- CALL "UPDRDFU" USING NEW-ID  
                                  OLD-ID  
                                  DB-NO  
                                  STATUS.

FORTRAN -- CALL UPDRDFU (NEW-ID,OLD-ID,DB-NO,STATUS)

INPUT:

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE RT-ID IN THE DF USED AS SET LINKAGE CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

PS 620341100  
30 September 1990

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDRRCB  
PURPOSE: RENAME THE RT\_ID IN RC\_BASED\_REC\_SET TABLE  
LANGUAGE: C  
SOURCE FILE: UPDRRCB  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDRRCB(NEW\_ID\_PTR, OLD\_ID\_PTR, DB\_NO\_PTR,  
STATUS)

COBOL -- CALL "UPDRRCB" USING NEW-ID  
OLD-ID  
DB-NO  
STATUS.

FORTTRAN -- CALL UPDRRCB (NEW-ID,OLD-ID,DB-NO,STATUS)

INPUT:

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE RT-ID IN THE RC\_BASED\_REC\_SET CDM TABLE.  
IF SUCCESSFUL, THE STATUS WILL BE 0.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

PS 620341100  
30 September 1990

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: UPDRRS  
PURPOSE: RENAME THE RT\_ID\_OF\_OWNER IN RECORD\_SET CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDRRS  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDRRS (NEW\_ID\_PTR, OLD\_ID\_PTR, DB\_NO\_PTR,  
STATUS)

COBOL -- CALL "UPDRRS" USING NEW-ID  
OLD-ID  
DB-NO  
STATUS.

FORTTRAN -- CALL UPDRRS (NEW-ID,OLD-ID,DB-NO,STATUS)

INPUT:

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE RT-ID OF OWNER IN THE RECORD SET CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

PS 620341100  
30 September 1990

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDRSS  
PURPOSE: RENAME THE SET\_ID IN RECORD\_SET CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDRSS  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDRSS (NEW\_ID\_PTR, OLD\_ID\_PTR, DB\_NO\_PTR,  
STATUS)

COBOL -- CALL "UPDRSS" USING NEW-ID  
OLD-ID  
DB-NO  
STATUS.

FORTTRAN -- CALL UPDRSS (NEW-ID, OLD-ID, DB-NO, STATUS)

INPUT:

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE SET\_ID IN THE RECORD\_SET CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

PS 620341100  
30 September 1990

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDRSTM  
PURPOSE: RENAME THE RT\_ID\_OF\_MEMBER IN SET\_TYPE\_MEMBER TABLE  
LANGUAGE: C  
SOURCE FILE: UPDRSTM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDRSTM(NEW\_ID\_PTR, OLD\_ID\_PTR, DB\_NO\_PTR,  
STATUS)

COBOL -- CALL "UPDRSTM" USING NEW-ID  
                                  OLD-ID  
                                  DB-NO  
                                  STATUS.

FORTTRAN -- CALL UPDRSTM (NEW-ID,OLD-ID,DB-NO,STATUS)

INPUT:

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE RT-ID OF MEMBER IN THE SET\_TYPE\_MEMBER CDM TABLE.  
IF SUCCESSFUL, THE STATUS WILL BE 0.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

PS 620341100  
30 September 1990

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDRT  
PURPOSE: RENAME THE RT-ID IN RECORD\_TYPE CDM TABLE  
LANGUAGE: C  
SOURCE FILE: UPDRT  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDRT(NEW\_ID\_PTR, OLD\_ID\_PTR, DB\_NO\_PTR, STATUS)

COBOL -- CALL "UPDRT" USING NEW-ID  
                                  OLD-ID  
                                  DB-NO  
                                  STATUS.

FORTRAN -- CALL UPDRT (NEW-ID,OLD-ID,DB-NO,STATUS)

INPUT:

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE RT\_ID IN THE RECORD\_TYPE CDM TABLE.

IF SUCCESSFUL, THE STATUS WILL BE 0.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: UPDSTM  
PURPOSE: RENAME THE SET\_ID IN SET\_TYPE\_MEMBER TABLE  
LANGUAGE: C  
SOURCE FILE: UPDSTM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C           -- UPDSTM(NEW\_ID\_PTR, OLD\_ID\_PTR, DB\_NO\_PTR,  
STATUS)

COBOL    -- CALL "UPDSTM" USING NEW-ID  
                                OLD-ID  
                                DB-NO  
                                STATUS.

FORTTRAN -- CALL UPDSTM (NEW-ID,OLD-ID,DB-NO,STATUS)

INPUT:

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE SET\_ID IN THE SET\_TYPE\_MEMBER CDM TABLE.  
IF SUCCESSFUL, THE STATUS WILL BE 0.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:  
-----

NEW\_ID\_PTR  
OLD\_ID\_PTR  
DB\_NO\_PTR  
STATUS

CHAR \*  
CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDTCR  
PURPOSE: UPDATE CATEGORY\_RELATION TABLE  
LANGUAGE: C  
SOURCE FILE: UPDTCR  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

SYNOPSIS

C -- UPDTCR(&RC\_NO, &GEN\_EC\_NO, CAT\_TYPE\_CODE,  
&STATUS) ;

COBOL -- CALL "UPDTCR" USING  
RC-NO  
GEN-EC-NO  
CAT-TYPE-CODE  
STATUS.

FORTRAN -- CALL  
UPDTCR(RCNO,GENECNO,CATTYPECODE,STATUS)  
INPUT:

INT \*RC\_NO ;  
INT \*GEN\_EC\_NO ;  
INT \*CAT\_TYPE\_CODE ;

OUTPUT:  
INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
CATEGORY\_RELATION

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----

RC\_NO  
GEN\_EC\_NO  
CAT\_TYPE\_CODE  
STATUS

INT \*  
INT \*  
CHAR \*  
INT \*

ROUTINES CALLED:

-----

STRLEN  
STRNCPY  
SOLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDTDOM  
PURPOSE: UPDATE DOMAIN SET DOMAIN\_NAME = : 1  
LANGUAGE: C  
SOURCE FILE: UPDTDOM  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
WHERE DOMAIN\_NO = : 2 ;

SYNOPSIS

C -- UPDTDOM(DOMAIN\_NAME\_PTR, &DOMAIN\_NO\_PTR,  
&STATUS) ;  
COBOL -- CALL "UPDTDOM" USING  
DOMAIN-NAME,  
DOMAIN-NO,  
STATUS.  
FORTRAN -- CALL UPDTDOM(DOMAINNAME, DOMAINNO, STATUS)

INPUT:

CHAR \*DOMAIN\_NAME\_PTR ;  
INT \*DOMAIN\_NO\_PTR ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
DOMAIN

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
DOMAIN\_NAME\_PTR  
DOMAIN\_NO\_PTR  
STATUS

CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDTKW  
PURPOSE: UPDATE KEYWORD SET KEYWORD = : 1  
LANGUAGE: C  
SOURCE FILE: UPDTKW  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:  
-----

WHERE KW\_NO = : 2 ;

SYNOPSIS

```
C          -- UPDTKW(KEYWORD_PTR, &KW_NO_PTR, &STATUS) ;
COBOL      -- CALL "UPDTKW" USING
                KEYWORD,
                KW_NO,
                STATUS.
FORTRAN    -- CALL UPDTKW(KEYWORD, KWNO, STATUS)
```

INPUT:

```
CHAR *KEYWORD_PTR ;
INT  *KW_NO_PTR ;
```

OUTPUT:

```
INT *STATUS ;
```

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
KEYWORD

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND

RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
KEYWORD PTR  
KW NO PTR  
STATUS

CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT



DOCGROUP PS41100 Module Documentation

NAME: UPDTRC  
PURPOSE: UPDATE LINK\_RELATION TABLE  
LANGUAGE: C  
SOURCE FILE: UPDTRC  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C           --    UPDTRC(&IND\_PTR, &MIN\_DEP\_PTR,  
&MAX\_DEP\_PTR,                            &RC\_NO\_PTR, &STATUS) ;

COBOL       --    CALL "UPDTRC"    USING  
                                  NO-IND-ENT,  
                                  MIN-NO-DEP-ENT,  
                                  MAX-NO-DEP-ENT,  
                                  RC-NO,  
                                  STATUS.

FORTRAN     --    CALL  
UPDTRC(NOINDENT,MINNODEPENT,MAXNODEPENT,RCNO,  
                                  STATUS)

INPUT:

INT   \*IND\_PTR ;  
INT   \*MIN\_DEP\_PTR ;  
INT   \*MAX\_DEP\_PTR ;  
INT   \*RC\_NO\_PTR ;

OUTPUT:

INT   \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
LINK\_RELATION

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH

WILL BE 0 , OTHERWISE -1.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
IND\_PTR INT \*  
MIN\_DEP\_PTR INT \*  
MAX\_DEP\_PTR INT \*  
RC\_NO\_PTR INT \*  
STATUS INT \*

ROUTINES CALLED:

-----  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDUDT  
PURPOSE: RENAME THE DATA\_TYPE\_NAME IN USER\_DEF\_DATA\_TYPE CDM  
TABLE  
LANGUAGE: C  
SOURCE FILE: UPDUDT  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

C -- UPDUDT(NEW\_ID\_PTR, OLD\_ID\_PTR, STATUS)  
  
COBOL -- CALL "UPDUDT" USING NEW-ID  
OLD-ID  
STATUS.  
  
FORTRAN -- CALL UPDUDT (NEW-ID,OLD-ID,STATUS)

INPUT:

NEW\_ID\_PTR  
OLD\_ID\_PTR

OUTPUT:

STATUS

DESCRIPTION

THIS ROUTINE USES AN ORACLE SEQUEL CALL TO UPDATE (RENAME)  
THE DATA\_TYPE\_NAME IN THE USER\_DEF\_DATA\_TYPE CDM TABLE.  
IF SUCCESSFUL, THE STATUS WILL BE 0.  
REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
NEW\_ID\_PTR

CHAR \*

PS 620341100  
30 September 1990

OLD\_ID\_PTR  
STATUS

CHAR \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: UPDVIEW  
PURPOSE: UPDATE VIEW SET SEC\_ID = : 1  
LANGUAGE: C  
SOURCE FILE: UPDVIEW  
SOURCE FILE TYPE: PC  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----

WHERE VIEW\_NO = : 2 ;

SYNOPSIS

C -- UPDVIEW(SEC\_ID\_PTR, &VIEW\_NO\_PTR, &STATUS)  
;  
COBOL -- CALL "UPDVIEW" USING  
SEC-ID,  
VIEW-NO,  
STATUS.  
FORTRAN -- CALL UPDVIEW(SECID, VIEWNO, STATUS)

INPUT:

CHAR \*SEC\_ID\_PTR ;  
INT \*VIEW\_NO\_PTR ;

OUTPUT:

INT \*STATUS ;

DESCRIPTION:

THIS ROUTINE USES ORACLE CALLS TO UPDATE A RECORD OF THE  
VIEW

THE DATA ARE TAKEN FROM THE INPUT PARAMETERS.

IF SUCCESSFULLY INSERTED THE RECORD, THE STATUS AND  
RETURN VALUE BOTH  
WILL BE 0 , OTHERWISE -1.

REWRITTEN TO PRO\*C DUE TO PROBLEM WITH SQL\*FORMS AND HLI  
INTERFACE

ARGUMENTS:

-----  
SEC\_ID\_PTR  
VIEW\_NO\_PTR  
STATUS

CHAR \*  
INT \*  
INT \*

ROUTINES CALLED:

-----  
STRNCPY  
SQLSCA  
SQLBS2  
SQLSCH  
SQLSCC  
SQLTFL  
SQLOPN  
SQLOSQ  
SQLAB2  
SQLEXE  
SQLWNR  
ERRRPT

DOCGROUP PS41100 Module Documentation

NAME: VEROBJ  
PURPOSE: VERIFY THAT THE OBJECT EXISTS.  
LANGUAGE: C  
SOURCE FILE: VEROBJ  
SOURCE FILE TYPE: C  
HOST:  
SUBSYSTEM: CDM  
SUBDIRECTORY: NDDL

DESCRIPTION:

-----  
SYNOPSIS

```
      C          --  VEROBJ (OBJ_TYPE, &MODEL_NO, &OBJ_NO)
      COBOL       --  CALL "VEROBJ"          USING
                                           OBJ-TYPE
                                           MODEL-NO
                                           OBJ-NO.
      FORTRAN     --  CALL VEROBJ (OBJTYPE,MODELNO,OBJNO)
```

INPUT:

CHAR\*OBJ\_TYPE[]  
INT \*MODEL\_NO

OUTPUT:

INT \*OBJ\_NO

DESCRIPTION

VEROBJ'S FUNCTION IS TO VERIFY THAT THE OBJECT TYPE  
ENTERED IN THE  
DESCRIBE COMMAND IS VALID. IF THE OBJECT TYPE IS NOT VALID,  
OBJ NO WILL CONTAIN ZERO. IF THE OBJECT TYPE IS VALID,  
OBJ\_NO WILL  
CONTAIN THE OBJECT NUMBER OF THE OBJECT IN QUESTION.

ARGUMENTS:

-----  
OBJ\_TYPE

CHAR []