CA Date Logic Generator

Installation and Reference Manual

r6

This documentation (the “Documentation”) and related computer software program (the “Software”) (hereinafter collectively referred to as the “Product”) is for the end user’s informational purposes only and is subject to change or withdrawal by CA at any time.

This Product may not be copied, transferred, reproduced, disclosed, modified or duplicated, in whole or in part, without the prior written consent of CA. This Product is confidential and proprietary information of CA and protected by the copyright laws of the United States and international treaties.

Notwithstanding the foregoing, licensed users may print a reasonable number of copies of the Documentation for their own internal use, and may make one copy of the Software as reasonably required for back-up and disaster recovery purposes, provided that all CA copyright notices and legends are affixed to each reproduced copy. Only authorized employees, consultants, or agents of the user who are bound by the provisions of the license for the Software are permitted to have access to such copies.

The right to print copies of the Documentation and to make a copy of the Software is limited to the period during which the license for the Product remains in full force and effect. Should the license terminate for any reason, it shall be the user’s responsibility to certify in writing to CA that all copies and partial copies of the Product have been returned to CA or destroyed.

EXCEPT AS OTHERWISE STATED IN THE APPLICABLE LICENSE AGREEMENT, TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS PRODUCT “AS IS” WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IN NO EVENT WILL CA BE LIABLE TO THE END USER OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS PRODUCT, INCLUDING WITHOUT LIMITATION, LOST PROFITS, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF CA IS EXPRESSLY ADVISED OF SUCH LOSS OR DAMAGE.

The use of this Product and any product referenced in the Documentation is governed by the end user’s applicable license agreement.

The manufacturer of this Product is CA.

This Product is provided with “Restricted Rights.” Use, duplication or disclosure by the United States Government is subject to the restrictions set forth in FAR Sections 12.212, 52.227-14, and 52.227-19(c)(1) - (2) and DFARS Section 252.227-7013(c)(1)(ii), as applicable, or their successors.

All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

Copyright © 2008 CA. All rights reserved.

Contents

Chapter 1: Installing the CA Date Logic Generator

Chapter 2: Using the CA Date Logic Generator

Appendix A: CA Date Logic Generator ISPF Table Files

Appendix B: COBOL CA Invocation Architecture

Index

Chapter 1: Installing the CA Date Logic Generator

This chapter describes the installation process for the CA Date Logic Generator on the mainframe environment.

This chapter includes the following sections:

- Introduction to CA Date Logic Generator
- Installation overview

Introduction to CA Date Logic Generator

Purpose

The CA Date Logic Generator helps programmers create business date logic by:

- Stepping them through the business rules required to create that logic.
- Testing the business rules.
- Generating an appropriately formatted CALL statement to the underlying CA Calendar Routines package or, for COBOL application programs, performing CA processing functions:

Code	Function
C101	Days Between
C102	Date +/- N Days
C201	Get Calendar Next Day
C202	Get Calendar Previous Day
U001	Get Current Date/Time
U002	Validate Date
U003	Determine Leap Year
U006	Get Day of Week
U007	Date-->Absolute Days

Code	Function
U008	Absolute Days-->Date
U009	Reformat Date
U011	Compare Two Dates Same Format
U012	Concatenate Century
U013	Truncate Century

The product can be used in development to create business date logic, and/or during maintenance to re-create business date logic.

How It Works

The product is an ISPF EDIT macro and runs under z/OS/ISPF. It is invoked by a PFKEY, whereupon the programmer is lead through a series of screens to identify the type of date calculation he/she would like to perform, to provide details for the selected calculation, and to test the logic with actual live values.

After the programmer is satisfied with the results, the CA Date Logic Generator creates a CALL or PERFORM statement in the original program source code so that during execution, the base Calendar Routines product can be invoked with appropriately set parameters.

Application COBOL source PERFORM statements can be executed for the following CA calendar Processing functions:

Code	Function
C101	Days Between
C102	Date +/- N Days
C201	Get Calendar Next Day
C202	Get Calendar Previous Day
U001	Get Current Date/Time
U002	Validate Date
U003	Determine Leap Year
U006	Get Day of Week
U007	Date-->Absolute Days
U008	Absolute Days-->Date
U009	Reformat Date

Code	Function
U011	Compare Two Dates (Same Format)
U012	Concatenate Century
U013	Truncate Century

The above functions can be included into an application program by including a Working Storage Copy (or include) and a Procedure Division Copy which will insert CA Calendar Processing storage elements and procedure statements.

See"Using CA Date Logic Generator", for instructions when specifying CALL or PERFORM of CA Calendar Processing functions.

Benefits

With CA Date Technology, companies can build stronger business applications by giving programmers the ability to create and test business date logic in just one step. This translates to savings in time and money—while actually improving the quality of the application.

Installation Overview

Important! Before you install the CA Date Logic Generator, you must first install the CA Calendar Routines.

The installation procedures for the CA Date Logic Generator are found in this chapter. You will perform the following steps:

- Step 1—Create and Load EXEC, ISPTLIB, and ISPPLIB Libraries
- Step 2—Specify CA Date Logic Generator Processing Profile Defaults
- Step 3—Concatenate EXEC, ISPTLIB, and ISPPLIB Libraries to Session
- Step 4—Setup PFKEYS in User TSO ISPF EDIT Profile

Installation

This section describes how to install the CA Date Logic Generator onto a mainframe computer system.

Step 1. Create and Load EXEC, ISPTLIB, and ISPPLIB Libraries

The instructions in this section are only applicable if the product is delivered to you on a single tape cartridge. If the product is delivered to you on any other media, contact CA Technical Support for installation instructions and assistance.

1. Copy the following JCL into your library.

```
//LOADTRC  JOB (accnt ###),'CA',MSGCLASS=X,MSGLEVEL=(1,1),
//  NOTIFY=userid,CLASS=A
//*******************************************************************/
//**  PRIOR EXECUTION MODIFY: ACCNT ###, USERID, TAPEVOL, USERNODE ****/
//**                          AND DISKVOL.                          ****/
//*******************************************************************/
//*
//*******************************************************************/
//*** INSTALL EXEC MEMBERS *****************************************/
//*******************************************************************/
//S01 EXEC PGM=IEBCOPY
//SYSPRINT DD SYSOUT=*
//IN1      DD DISP=OLD,DSN=TRD.TAPE6000.EXEC,
//         UNIT=TAPE,LABEL=(2,SL),VOL=SER=tapevol <= CHANGE
//OUT1     DD DSN=usernode.TCDS.EXEC, <= CHANGE
//         DISP=(NEW,CATLG,DELETE),
//         SPACE=(TRK,(15,5,15)),UNIT=SYSDA, <= CHANGE
//         DCB=(RECFM=FB,LRECL=80,BLKSIZE=6400)
//SYSIN    DD *
           COPY OUTDD=OUT1,INDD=((IN1,R))
/*
//*******************************************************************/
//*** INSTALL PANEL MEMBERS ****************************************/
//*******************************************************************/
//S02 EXEC PGM=IEBCOPY
//SYSPRINT DD SYSOUT=*
//IN2      DD DISP=OLD,DSN=TRD.TAPE6000.ISPPLIB,
//         UNIT=TAPE,LABEL=(3,SL),VOL=SER=tapevol <= CHANGE
//OUT2     DD DSN=usernode.TCDS.ISPPLIB,   <= CHANGE
//         DISP=(NEW,CATLG,DELETE),
//         SPACE=(TRK,(15,5,20)),UNIT=DISK, <= CHANGE
//         DCB=(RECFM=FB,LRECL=80,BLKSIZE=6400)
//SYSIN    DD *
  COPY     OUTDD=OUT2,INDD=((IN2,R))
//*******************************************************************/
//*** INSTALL TABLE MEMBERS ****************************************/
//*******************************************************************/
//S03      EXEC PGM=IEBCOPY
//SYSPRINT DD SYSOUT=*
//IN3      DD DISP=OLD,DSN=TRD.TAPE6000 ISPTLIB,
//         UNIT=TAPE,LABEL=(4,SL),VOL=SER=tapevol <= CHANGE
//OUT3     DD DSN=usernode.TCDS.ISPTLIB,   <= CHANGE
//         DISP=(NEW,CATLG,DELETE),
//         SPACE=(TRK,(15,5,40)),UNIT=DISK, <= CHANGE
//         DCB=(RECFM=FB,LRECL=80,BLKSIZE=6400)
//SYSIN    DD *
           COPY OUTDD=OUT3,INDD=((IN3,R))
/*
```

2. Modify the lowercase character variables and dataset size allocation parameters to meet your data center standards and procedures.
3. Submit LOADTRC and verify return codes.
4. If all return codes are zeros (0), then continue with Step 2. Specify CA Date Logic Generator Processing Profile Defaults.

Step 2. Specify CA Date Logic Generator Processing Profile Defaults

Edit the following member: 'nodeout.TCDS.EXEC(TCDS$USR)'.

```
/* REXX                                                               */
/*--------------------------------------------------------------------*/
/* Establish CA Date Logic Generator default processing options       */
/*                                                                    */
/* (C) COPYRIGHT CA DATA SYSTEMS 2007 ALL RIGHTS RESERVED             */
/*                 ------------------------------                     */
/*                       version 6.00                                 */
/*                                                                    */
/* TCDS$USR. 12/26/98 BRUCE GODDARD                                   */
/*                                                                    */
/*--------------------------------------------------------------------*/
         /* <==== DATE LOGIC GENERATOR VERSION     */
TCDSVERS = "6.00"
TCDSCUST = "ABABABABAB"
        /* <=== Special FISCAL Calendar processing */
TCDSFSCL = "Y" /* "N" This installation DOES NOT process special
                    FISCAL Calendars (i.e. 4-4-4 4-5-4 4-5-5)
              "Y" This installation uses FISCAL Calendars */
        /* <==== DATE LOGIC GENERATOR VERSION      */
TCDSQUOT = "N" /* "N" This instalaltion uses apostrophe(') as the
                    delimeter for COBOL literals.
              "Y" This installation uses QUOTATION(") */
        /* <==== BATCH PGM FOR T.D.S SMALL VERSION UXXX, C101-2, C201-2 */
TCDSMALL = "TRCSMALL"
        /* <==== BATCH PGM FOR T.D.S ENGINE    */
TCDSBMBR = "WS-TRCENGIN"
        /* <==== PL/1 PGM AND EXTERNAL DEFINITION */
TCDSPMBR = "TRCENGN "
TCDSPDCL = "DCL TRCENGN EXT ENTRY OPTIONS(ASM);"
        /* <==== CICS PGM FOR T.D.S.ENGINE     */
TCDSCPGM = "TRCENGNC"
/*--------------------------------------------------------------------*/
/*                                                                    */
/* The following dialog variables "MUST" be set to Load               */
/* Library executable module member names, IF your                    */
/* installation elects to DYNAMICALLY "CALL" the Calendar             */
/* Routines Engine program.                                           */
/*                                                                    */
/* NOTE: TCDSBMBR and/or TCDSCPGM must be set to data                 */
/* element names (e.g. "WS-TRCENGIN-PGM")                             */
/*                                                                    */
/*--------------------------------------------------------------------*/
TCDSDCBL = "TRCENGIN" /* BATCH COBOL */
TCDSDCIS = "TRCENGNC" /* CICS COBOL */
        /* <==== ISPF TEST TRANSACTION FILE NAME */
TCDSTDSN = "PDGBO.WRK6000.CNTL256(DLGTEST)"
        /* <==== ISPF TABLE LIBRARY DSN */
TCDSTLIB = "PDGBO.WRK6000.ISPTLIB"
        /* <==== FOREGROUND LOADLIB AND ENGINE MEMBER NAME */
TCDSEDSN = "PDGBO.WRK6000.LOADLIB"
TCDSEMBR = "TRCENGIN"
        /* <==== VSAM HOLIDAY TABLE FILE NAME */
TCDSHLVS = "Y2K.REL5000.HOLIDAY.KSDS"
        /* <==== COPYBOOK MBRS & SOURCE INCLUSION STATEMENT(VERB) */
TCDSCINC = "COPY"
TCDSACPY = "TRCCONVA"
```

```
TCDSCCPY = "TRCCONVR"
TCDSPCPY = "TRCCONVP"
TCDSU1XP = "TRCU1XPD"
TCDSU1XW = "TRCU1XWS"
TCDSMLPD = "TRCSMLPD"
TCDSMLWS = "TRCSMLWS"
TCDSLGPD = "TRCCONPD"
TCDSLGWS = "TRCCONWS"
TCDSCMNP = "TRCCOMNP"
TCDSCMNW = "TRCCOMNW"
TCDSU1PP = "TRPU1XPD"
TCDSU1PW = "TRPU1XWS"
TCDSMLPP = "TRPSMLPD"
TCDSMLPW = "TRPSMLWS"
TCDSLPPD = "TRPCONPD"
TCDSLPWS = "TRPCONWS"
TCDSCPNP = "TRPCOMNP"
TCDSCPNW = "TRPCOMNW"
        /* <==== DEFAULT DATA ELEMENT SEARCH FILE */
        /* PROGRAM ID START POSITION */
        /* DATA ELEMENT START POSITION */
TCDSDTES = "PDGBO.WRK6000.DATA.ELEMENT(IDNL)"
TCDSDEPG = 11
TCDSDELM = 31
        /* <==== DEFAULT COPYLIB FILE NAME(S) */
TCDSMCPU = "Y" /* "Y" allow User to change Copylib file names, "N" stop */
TCDSMCP1 = "PDGBO.PDS.COPYLIB"
TCDSMCP2 = "PDGBO.WRK6000.COPYLIB"
TCDSMCP3 = "PDGBO.WRK6000.SAMPLE.COPYLIB"
TCDSMCP4 = ""
TCDSMCP5 = ""
TCDSMCP6 = ""
TCDSMCP7 = ""
TCDSMCP8 = ""
TCDSMCP9 = ""
        /* <==== COPYBOOK PASS AREA DATA ELEMENT NAMES */
TCDSA01L = "TRCCONVA"
TCDSC01L = "TRC-CONVR-CONVERSATIONAL"
TCDSP01L = "TRC_CONVR_CONVERSATIONAL"
        /* SET ISPF DLG PROCESSING PARAMETERS */
TCDSBUSP = "Y" /* "Y" allow updates to Business Parms, "N" no changes */
TCDSCALL = "P" /* "C" for call, "P" for perform, "" allow selection */
TCDSTEST = "Y" /* "Y" force User testing (F5) before code generation */
TCDSAUTO = "" /* "Y" Always Auto Remediate COBOL program code; */
        /* "N" Never Auto Remediate; -or- */
        /* "" User Specifies, TCDSBUSP must equal "Y" */
TCDSIDNL = "N" /* "Y" When the Data Elements file does not contain a */
        /* valid TCDS Date MASK for the data elements of the */
        /* application program being processed, build a */
        /* default TCDS Date MASK based upon the data element */
        /* definition length and types stored in the file */
        /* "N" DO NOT build default TCDS Date MASKS for the */
        /* application program begin processed based upon */
        /* the Data Element file contents */
TCDSRKRS = "P" /* "Y" Reset data element key values - COBOL data element */
        /* names set by PF keys for FROM, TO dates, etc.. */
        /* "N" The key setting values will be set SIMILAR to */
        /* ISPFEDIT PF5(find) or PF6(replace) keys */
        /* "P" Force 'Pop-up' display of data element, update */
        /* TCDS mask definition with User supplied data */
        /* SET DEFAULT CALENDAR BUSINESS PARAMETER VALUES */
TCDSBHO = '00' /* Holiday Table ID */
TCDSBFY = '01' /* Fiscal Year Begin Month */
TCDSBFM = '01' /* Fiscal Month Begin Day */
TCDSBEN = 'T' /* From/To Date End Points Definition */
TCDSBDW = '1234567' /* Numerical Day-of-Week Setting */
TCDSBPD = 'NEEEEEN' /* Process Day-of-Week Setting */
TCDSBCB = 'SLIDE-20' /* Century 'Window' Assignment */
TCDSIFRM = 'YYMMDD--' /* Input 'FROM' Date Mask */
TCDSITOM = 'YYMMDD--' /* Input 'TO' Date Mask */
TCDSIO1M = 'CCYYMMDD' /* 1st Output Date Mask */
TCDSIO2M = '--------' /* 2nd Output Date Mask */
TCDSIO3M = '--------' /* 3rd Output Date Mask */
TCDSBCK = 'Y' /* Test for null input MASK Values */
```

```
TCDSBCS = 'M' /* Character Output CASE of Literals U=UPPER, M=Mixed */
        /* SET DEFAULT GENERATION PARAMETER VALUES */
TCDSUSTD = 'N' /* "Y" DLG will generate code that will (COBOL example)
MOVE TRC-CONVR-STDOUT-DATE-X
TO ?-USER-STANDARD-DATE-OUT */
/*-------------------------------------------------------------------------*/
        /* Following applies to statements generated after a PERFORM of the */
        /* Calendar Routines has been made, EXAMPLE SHOWN is COBOL: */
        /* IF TRC-CONVR-RETURN-GOOD */
        /* . . . . . (DLG generated procedure statements) */
        /* ELSE */
        /* PERFORM ?-USER-ERROR-ROUTINE */
TCDSEROR = 'Y' /* "Y" DLG will generate the above ELSE and */
        /* PERFORM TRC-ERROR-ROUTINE (see TCDSERPC) -OR- */
        /* PERFORM ?-USER-ERROR-ROUTINE */
        /* "N" No ELSE or PERFORM statements are generated */
TCDSERPC = 'TRC-ERROR-ROUTINE' /* COBOL Error Paragraph */
TCDSERPP = 'TRC_ERROR_ROUTINE' /* PL/1 Error Paragraph */
TCDSERPA = 'TRAERROR' /* Assembler BRANCH-TO Error Routine label */
TCDSERTC = 'TRCERROR' /* COPYBOOK containing COBOL Error Routine */
TCDSERTP = 'TRPERROR' /* COPYBOOK CONTAINING PL/1 ERROR ROUTINE */
TCDSEDIF = 'Y' /* Y(yes) or N(no), To determine if an END-IF is */
        /* generated for COBOL-2 to terminate the above */
        /* "IF TRC-CONVR-RETURN-GOOD" statement */
/*-------------------------------------------------------------------------*/
        /* VPUT names must be all alpha-num, no special chars */
'ISPEXEC VPUT ( TCDSVERS, TCDSCUST, TCDSTLIB, TCDSTDSN ) SHARED'
'ISPEXEC VPUT ( TCDSDTES, TCDSDEPG, TCDSELM ) SHARED'
'ISPEXEC VPUT ( TCDSBMBR, TCDSCPGM, TCDSDCBL, TCDSDCIS, TCDSEDSN ) SHARED'
'ISPEXEC VPUT ( TCDSEMBR, TCDSCINC, TCDSPMBR, TCDSPDCL, TCDSHLVS ) SHARED'
'ISPEXEC VPUT ( TCDSACPY, TCDSCCPY, TCDSPCPY, TCDSMCPU ) SHARED'
'ISPEXEC VPUT ( TCDSMLPD, TCDSMLWS, TCDSU1XP, TCDSU1XW, TCDSMALL ) SHARED'
'ISPEXEC VPUT ( TCDSMCP1, TCDSMCP2, TCDSMCP3, TCDSMCP4 ) SHARED'
'ISPEXEC VPUT ( TCDSMCP5, TCDSMCP6, TCDSMCP7, TCDSMCP8 ) SHARED'
'ISPEXEC VPUT ( TCDSMCP9, TCDSCMNP, TCDSCMNW, TCDSLGPD, TCDSLGWS ) SHARED'
'ISPEXEC VPUT ( TCDSEROR, TCDSERPC, TCDSERTC, TCDSEDIF ) SHARED'
'ISPEXEC VPUT ( TCDSERPA, TCDSERPP, TCDSERTP, TCDSBCS) SHARED'
'ISPEXEC VPUT ( TCDSU1PP, TCDSU1PW, TCDSMLPP, TCDSMLPW ) SHARED'
'ISPEXEC VPUT ( TCDSLPPD, TCDSLPWS, TCDSCPNP, TCDSCPNW ) SHARED'
'ISPEXEC VPUT ( TCDSA01L, TCDSC01L, TCDSP01L, TCDSQUOT, TCDSIDNL ) SHARED'
'ISPEXEC VPUT ( TCDSBUSP, TCDSCALL, TCDSTEST, TCDSAUTO, TCDSRKRS ) SHARED'
'ISPEXEC VPUT ( TCDSBHO, TCDSBFY, TCDSBFM, TCDSBEN ) SHARED'
'ISPEXEC VPUT ( TCDSBDW, TCDSBPD, TCDSBCB, TCDSBCK, TCDSUSTD ) SHARED'
'ISPEXEC VPUT ( TCDSIFRM, TCDSITOM, TCDSIO1M, TCDSIO2M, TCDSIO3M ) SHARED'
RETURN 0
/*-------------------------------------------------------------------------*/
/* (c) Copyright CA Data Systems 2007 ALL RIGHTS RESERVED */
/*-------------------------------------------------------------------------*/
```

TCDSFSCL

Set this variable to 'Y' if your installation uses special Fiscal Calendars (for example 4-4-5, 4-5-4, 4-5-5). These are four/five week month calendars. Set this variable to 'N' (default) if your organization does not use special Fiscal Calendars.

TCDSQUOT

Set this variable to 'Y' if your installation uses the quotation (“) to delimit literals in COBOL programs. Set this variable to 'N' if apostrophes are used.

TCDSMALL

This is the name of the BATCH limited function CA Calendar Routines Engine. It is normally set to the Program source member TRCSMALL.

TCDSBMBR

This is the name of the BATCH full function CA Calendar Routines Engine. It is normally set to the Program source member TRCENGIN. When this variable is set to an eight (or less) character name, a static call to the full function Calendar Routines is generated: CALL TRCENGIN.

If this variable is set to a value other than a PDS member name, a dynamic call to the full function Calendar Routines is generated: CALL WS-PROGRAM-NAME. If the application program is COBOL, a scan of the program source is made to locate the program variable (for example, WS-PROGRAM-NAME). CA Date Logic Generator will prompt the user to insert the variable (or not) should the variable not exist in the COBOL program source.

TCDSPMBR

Some installations which invoke the full function Calendar Routines from PL/1 application programs will execute a special compiled version of program source member TRCENGIN. This is due to the COBOL-PL/1 run time library requirements.

TCDSPDCL

If the application program is PL/1, a scan of the program source is made to locate a program variable defining the PL/1 Calendar Routines execution program name defined by TCDSPMBR. CA Date Logic Generator will insert the variable definition stored in TCDSPDCL, if the PL/1 Calendar Routines execution program name is not located.

TCDSCPGM

This is the name of the CICS full function version of the CA Calendar Routines Engine. It is normally set to the Program source member TRCENGNC. When this variable is set to an eight-character name, a CICS LINK to the full function Calendar Routines is generated:

```
EXEC CICS LINK
PROGRAM('TRCENGNC')
COMMAREA(TRC-CONVR-CONVERSATIONAL)
LENGTH(1000)
END-EXEC.
```

If this variable is set to a value other than a CICS program name, the following a CICS LINK to the full function Calendar Routines is generated:

```
EXEC CICS LINK
PROGRAM(WS-PROGRAM-NAME)
COMMAREA(TRC-CONVR-CONVERSATIONAL)
LENGTH(1000)
END-EXEC.
```

TCDSDCBL

This variable contains the name of the CA Calendar Routines (batch) executable module which will be placed in the COBOL source program statement defining the program variable should the application program dynamically call the Calendar Routines.

If TCDSBMBR is the name of a program variable, CA Date Logic Generator can (optionally) insert the following statement in a batch COBOL application program:

Example:

```
TCDSBMBR = "WS-PROGRAM-NAME"TCDSDCBL = "TRCENGIN"
```

Generated:

```
01 WS-PROGRAM-NAME PIC X(08) VALUE "TRCENGIN"
```

TCDSDCIS

This variable is similar to TCDSDCBL. Variable TCDSDCIS is the COBOL CICS version used for dynamic invocations of the CA Calendar Routines.

Example:

```
TCDSCPGM = "WS-PROGRAM-NAME"TCDSDCIS = "TRCENGNC"
```

Generated:

```
01 WS-PROGRAM-NAME PIC X(08) VALUE "TRCENGNC"
```

TCDSTLIB

This is the name of the ISPF Table library which contains:

- Member TCDSCBRW - this is a list of COBOL language Reserved Words (for example READ, WRITE, ADD).

  Important! This table member should NEVER be modified, renamed, or removed.

- Member TCDSMTBL - this is a list of CA DATE MASKS (for example YYMMDD--, ---YYDDD).

  Important! This table member should NEVER be modified, renamed, or removed.

- A member for each source program processed by the CA (DLG) generated during DLG Profile Processing.

TCDSEDSN

This is the name of the load library which contains the CA full function Calendar Routines. This library MUST be allocated to the users TSO session. The full function Calendar Routines are executed (in Foreground) during DLG processing.

TCDSEMBR

This is the member name of the full function Calendar Routines stored in the above library.

TCDSHLVS

This variable will normally be set to "". If your installation stores CA Holiday Tables on a VSAM file, this variable MUST be set to the KSDS dataset name (for example nodeout.HOLIDAY.KSDS).

COPYBOOK MBRS & SOURCE INCLUSION STATEMENT(VERB)

The variables in this next section contain values which determine DLG generation of source statement inclusion statements (for example COPY membername).

TCDSCINC = "COPY"

Program language verb used to include program source members (for example copy books).

TCDSACPY = "TRCCONVA"

Assembler copy book name of Calendar Routines Link-Pass area.

TCDSCCPY = "TRCCONVR"

COBOL copy book name of Calendar Routines Link-Pass area.

TCDSPCPY = "TRCCONVP"

PL/1 copy book name of Calendar Routines Link-Pass area.

TCDSU1XP = "TRCU1XPD"

COBOL copy book name of member containing Procedure source statements for functions U11-U13.

TCDSU1XW = "TRCU1XWS"

COBOL copy book name of member containing Working Storage source statements for functions U11-U13.

TCDSMLPD = "TRCSMLPD"

COBOL copy book name of member containing Procedure source statements for limited (TRCSMALL) functions.

TCDSMLWS = "TRCSMLWS"

COBOL copy book name of member containing Working Storage source statements for limited (TRCSMALL) functions.

TCDSLGPD = "TRCCONPD"

COBOL copy book name of member containing Procedure source statements for invoking full (TRCENGIN) functions.

TCDSLGWS = "TRCCONWS"

COBOL copy book name of member containing Working Storage source for invoking full (TRCENGIN) functions.

TCDSCMNP = "TRCCOMNP"

COBOL copy book name of member containing common Procedure statements used by in-line TRCSMALL functions.

TCDSCMNW = "TRCCOMNW"

COBOL copy book name of member containing common Working Storage data elements used by in-line TRCSMALL functions.

TCDSDTES

This is the dataset name of the file which contains program data elements definitions used by CA Date Logic Generator to locate, process and generate program source statements invoking Calendar Routines functions. Set this entry to (""), if no file exists or you do not want to use a data element definitions file. See the appendix “DLG ISPF Table Files”, for more information on data elements definition.

TCDSDEPG / TCDSDELM

If you use a Data Elements File, these two elements define the starting position of the application source program name (optional) and the starting position of the data element name in the Data Elements File record.

Note: Please note the following regarding the Data Elements File:

- If the application source program name is not located (or used) in the Data Elements File record, set variable to TCDSDEPG = 0.
- If the Data Elements File is not used at your organization, set variable to TCDSDELM = 0.

TCDSMCPU

Set this variable to 'N' if you do not want to provide CA Date Logic Generator users the ability of adding, changing or removing Copy File names associated with CA Date Logic Generator. Set the variable to 'Y' if you want to allow user to modify these file names.

Copy File names (1-9)

Enter the names of your installation program source inclusion libraries (COPY Files) in the following nine variables. Set unused variables to (““).

```
TCDSMCP1 = "USER.PDS.COPYLIB"
TCDSMCP2 = "nodeout.TCDS.COPYLIB"
TCDSMCP3 = “”
TCDSMCP4 = “”
TCDSMCP5 = “”
TCDSMCP6 = “”
TCDSMCP7 = “”
```

```
TCDSMCP8 = ""
TCDSMCP9 = ""
```

COPYBOOK PASS AREA DATA ELEMENT NAMES

The following entries are used by DLG during program source generation to specify the data element name of the CA Link-Pass area (for example 01 TRC-CONVR-CONVERSATIONAL).

```
TCDSA01L = "TRCCONVA"
TCDSC01L = "TRC-CONVR-CONVERSATIONAL"
TCDSP01L = "TRC_CONVR_CONVERSATIONAL"
```

TCDSBUSP

Set the value of this variable to 'Y' if you want to allow the CA Date Logic Generator user the ability of changing the CA Calendar Routines Business processing parameters for each program DLG processes (see the *CA Calendar Routines Technical Manual*). Set this variable to 'N' which inhibits changing these values as part of the DLG program profile (startup) process.

TCDSCALL

The setting of this variable determines what Calendar Routines invocation program procedural statements will be generated by CA Date Logic Generator. Set the variable to nulls (""), if you want CA Date Logic Generator to prompt the user to determine what type of statements will be generated (for example one of the following options). Set this variable to C if the Calendar Routines will always be invoked by a CALL statement (for example CALL TRCENGIN USING TRC-CONVR-CONVERSATIONAL). Set this variable to P if you want to PERFORM in-line included source statements for the following functions and CALL the full function Calendar Routines for all others:

Code	Description
C101	Days Between
C102	Date +/- N Days
C201	Get Calendar Next Day
C202	Get Calendar Previous Day
U001	Get Current Date/Time
U002	Validate Date
U003	Determine Leap Year
U006	Get Day of Week
U007	Date-->Absolute Days
U008	Absolute Days-->Date

Code	Description
U009	Reformat Date
U011	Compare Two Dates Same Format
U012	Concatenate Century
U013	Truncate Century

TCDSTEST

The CA Date Logic Generator (DLG) has a testing mechanism which provides the DLG user a method of testing Calendar Routines functions with key-entered data. Testing of functions occurs prior to the generation of Calendar Routines invocation statements. Some organizations mandate testing before generation of source statements can be accomplished. Set this variable to variable to 'Y', to force testing. Set this variable to 'N' to allow optional testing of calendar routines functions. A date/ time stamp is inserted whenever testing has been completed by the DLG user.

See Step 5. Test with Some Sample Data for more information about testing.

```
*-------------------------------------------------------------*
* CA(S) COBOL (U002) VALIDATE DATE  ONP 0=NO,1=YES
* D.L.G. TESTING COMPLETED 1998/04/17 12:50:02
*-------------------------------------------------------------*
MOVE 'U002' TO TRC-CONVR-FUNCTION-CODE
MOVE 'SLIDE-20' TO TRC-CONVR-CENTURY-BREAK . . . . . .
```

TCDSAUTO

COBOL Program source code can be automatically remediated by the CA Date Logic Generator (DLG). Automatic remediation normally occurs when a COBOL program is edited using DLG for the first time. DLG invokes program profile processing which instructs the DLG user to provide information about the program and DLG processing options. Set this variable to 'Y' if you always want to have DLG try to automatically remediate simple COBOL IF statements. Set this variable to 'N', if you do not want DLG to attempt this remediation. Set the variable to nulls (" ") if you want the DLG user to determine if automatic remediation should be attempted.

TCDSIDNL

Set this variable to 'Y' to update ‘date’ data element definitions stored in the application program’s temporary data elements definition file (COBOL only). Set this variable to 'N' to not update the file.

The temporary data elements definition file is used to identify ‘date’ related data elements and the Calendar Routines date masks used by DLG during application program source statement generation.

TCDSRKRS

Set this variable to 'Y' to clear all data element name and value assignments stored by DLG each time a different COBOL program is edited using DLG. Set this variable to 'N' to save all data element name and value assignments while the DLG user is actively editing programs.

Set this variable to 'P' to force display of all data element name and value assignments. The display of the data element information is a pop-up screen appearing at the bottom of an ISPF Edit screen. See TCDSDATA—Assign CA Data Element Values and TCDSDATE—Assign CA Date Element Values.

DLG can store data element names for substitution when DLG generates Calendar Routines invocation COBOL program source statements. The following two examples are COBOL MOVE statements which place data into the Calendar Routines input “FROM” date field of the Link-Pass area.

Example 1: Statement generation without data element name substitution:

```
MOVE 'YYMMDD--' TO TRC-CONVR-FROM-DATE-MASK
MOVE ?-USER-FROM-DATE
TO TRC-CONVR-FROM-DATE-X
PERFORM TRC-CALENDAR-ROUTINES
```

Example 2: Statement generation with data element name substitution:

```
MOVE 'YYMMDD--' TO TRC-CONVR-FROM-DATE-MASKMOVE SALES-DATE TO TRC-CONVR-FROM-
DATE-XPERFORM TRC-CALENDAR-ROUTINES
```

SET DEFAULT CALENDAR BUSINESS PARAMETER VALUES

The next series of variables are the settings used by DLG as defaults for Calendar Routines Business Parameters. Refer to the Business Parameters section of the “Communications” chapter in the *CA Calendar Routines Application Manual* for more information about the following parameters.

Variable	Description
TCDSBHO = '01'	Holiday Table ID
TCDSBFY = '01'	Fiscal Year Begin Month
TCDSBFM = '01'	Fiscal Month Begin Day
TCDSBEN = 'T'	From/To Date End Points Definition
TCDSBDW = '1234567'	Numerical Day-of-Week Setting
TCDSBPD = 'NEEEEEN'	Process Day-of-Week Setting
TCDSBCB = 'SLIDE-20'	Century 'Window' Assignment
TCDSIFRM = 'YYMMDD--'	Input 'FROM' Date Mask
TCDSITOM = 'YYMMDD--'	Input 'TO' Date Mask

Variable	Description
TCDSIO1M = 'CCYYMMDD'	1st Output Date Mask
TCDSIO2M = '--------'	2nd Output Date Mask
TCDSIO3M = '--------'	3rd Output Date Mask
TCDSBCK = 'Y'	Test for null input MASK Values
TCDSBCS = 'M	Set alphabetic output literal case. Set to "M" for mixed case (January 1, 2007) or "U" for uppercase (JANUARY 1, 2007).

TCDSUSTD

Set this variable to 'Y' if you want DLG to generate a move statement of the Calendar Routines standard format date (CCYYMMDD) to a program data element. Set this variable to 'N' to omit generation of the move statement.

```
MOVE TRC-CONVR-STDOUT-DATE-X
TO ?-USER-STANDARD-DATE-OUT
```

COBOL statements

The following applies to COBOL statements generated after a PERFORM (or invocation) of the Calendar Routines has been made:

```
PERFORM TRC-CALENDAR-ROUTINES
IF TRC-CONVR-RETURN-GOOD
. . . . . (DLG generated procedure statements)
ELSEPERFORM ?-USER-ERROR-ROUTINE
```

TCDSEROR

Setting this variable to 'Y' will generate the above ELSE and:

- `PERFORM TRC-ERROR-ROUTINE`

  Or

- `PERFORM ?-USER-ERROR-ROUTINE`

Setting this variable to 'N' generates no statements.

TCDSERPH

Set this variable to the COBOL paragraph name the application program is to PERFORM when an error occurs in Calendar Routines processing. If TCDSEROR is specified, this variable must be set to the error routine paragraph name.

TCDSERTN

This variable contains the name of the error routine copy book program source member. The routine should also contain the COBOL paragraph name specified in TCDSERPH.

TCDSEDIF

Set this variable to 'Y' if the application program is written in COBOL-2 and requires encapsulation of 'IF' statements by the COBOL-2 reserved word END-IF. This variable should be set to 'N' for non-COBOL-2 applications and/or applications which do not use the END-IF verb.

To end the edit session, press PF3 to commit the changes.

Step 3. Concatenate EXEC, ISPTLIB, and ISPPLIB Libraries to Session

If your company uses TRX, follow instructions under item 1 below; otherwise follow the instructions under item 2.

In the following, nodeout has been replaced by the actual name specified in Step 2. Specify CA Date Logic Generator Processing Profile Defaults.

Note: The word TSO is necessary, EXEC may be abbreviated as EX, the quotes around the DSN are necessary, and the parm node out should be replaced by the

1. For TRX, enter the following on the command line:

   ```
   TSO EXEC 'nodeout.TCDS.EXEC(TCDS$TRX)'
   ```

   which will execute these commands:

   ```
   TRX ADD FI(SYSEXEC) DA('nodeout.TCDS.EXEC')
   TRX ADD FI(ISPLIB) DA('nodeout.TCDS.ISPPLIB')
   TRX ADD FI(ISPTLIB) DA('nodeout.TCDS.ISPTLIB')
   ```

2. For non-TRX, enter the following on the command line:

   ```
   TSO EXEC 'nodeout.TCDS.EXEC(TCDS$CAT)'
   ```

   The REXX macro will execute the following commands:

   ```
   ALTLIB ACTIVATE APPLICATION(EXEC)
   ...DATASET('nodeout.TCDS.EXEC') UNCOND
   ISPEXEC LIBDEF ISPPLIB DATASETID('nodeout.TCDS.ISPPLIB')
   ISPEXEC LIBDEF ISPPLIB DATASETID('nodeout.TCDS.ISPTLIB')
   ```

Step 4. Setup PFKEYS in User TSO ISPF EDIT Profile

1. Start ISPF EDIT.
2. Type KEYS on the ==> command line and press the Enter key (or PC 3270 session equivalent). Normally the first key assignments shown are for PF1through OF12. You can view additional key settings by pressing the DOWN (NEXT) key PF8 or view previous key settings by pressing the UP (PREVIOUS) key PF7.

3. Choose key combinations containing the PF keys used to invoke the CA Date Logic Generator and related CA Date Logic Generator functions, while you are editing an application program.

   Most programmers do not choose PF1 through PF12, because they have pre-defined meanings. Instead, most programmers choose PF13 through PF24.

4. After you have cycled to the appropriate key display panel, displaying the keys you want to change, press the TAB key to position the cursor on the key or line, and overtype the definitions as shown:

```
Keylist Utility
File
Private  ISR Keylist ISRSPEC Change    Row 13 to 24 of 24
Command ===>    Scroll ===> PAGE
Make changes and then select File action bar.
Keylist Help Panel Name . . . ISRSPECH
Key    Definition Format Label
F13 . . TCDSDATE  SHORT  TCDSDATE
F14 . . TCDSDATA  SHORT  TCDSDATA
F15 . . TCDSRSET  SHORT  TCDSDATA
F16 . . TCDSRKEY  SHORT  TCDSRKEY
F17 . . TCDSSRCH  SHORT  TCDSSRCH
F18 . . TCDSSPF1  SHORT  TCDSSPF1
F19 . . UP LONG Up
F20 . . DOWN LONG  Down
F21 . . SWAP LONG  Swap
F22 . . LEFT LONG  Left
F23 . . RIGHT   LONG  Right
F24 . . CANCEL   LONG  Cancel
```

   The key assignments are suggested. The CA Date Logic Generator user may wish to choose different key assignments (maybe using a key in the F1 through F12 range). The user may also not want to use specific CA Date Logic Generator functions. It is strongly suggested, for minimal CA Date Logic Generator functionality, that key assignments be made for TCDSSPF1 (F18 - invoke DLG) and TCDSRSET (F19 - reset DLG program profile).Additional information describing each CA Date Logic Generator key related function is located in “Using the CA Date Logic Generator”.

5. Press Enter to apply the change.

6. To end key change, press PF3. This will commit the change for the EDIT function.

Chapter 2: Using the CA Date Logic Generator

This chapter gives a high level overview of how to use the CA Date Logic Generator.

- Using the CA Date Logic Generator
- Resetting the DLG Program Profile Definition
- Optional COBOL TCDS PF Key Functions

Using the CA Date Logic Generator

The following steps detail how to use the CA Date Logic Generator.

Step 1. Editing an Application Program

1. To edit a program, enter Type=2 in the command line on the ISPF Primary Function Menu to get the Edit primary panel, and then specify your program name.
2. Position the cursor where you would like the eventual generated code to be placed.

   Typically this would be the location of defective date logic that you want to replace invoking the location where you'd like to create entirely new date logic from scratch, or the beginning of the program if this is the first time you are editing a program using the CA Date Logic Generator.

Step 2. Invoking the CA Date Logic Generator

1. Press the PF key combination that you defined in an ISPF EDIT session using the CA Date Logic Generator (see Step 4. Setup PFKEYS in User TSO ISPF EDIT Profile in “Installing the CA Date Logic Generator”, for instructions). This PF key invokes the CA Date Logic Generator while in ISPF Edit (i.e. key assigned to TCDSSPF1).
2. The first time a member is edited using the CA Date Logic Generator, the DLG member profile screen is displayed.

The following is an example of the DLG program profile definition for a COBOL application program:

```
--------------------- DLG Program Profile Definition --------------------------
OPTION ===>
          Calendar Routine Customer/Version ABABABABAB 6.00
      Date Logic Generator Customer/Version ABABABABAB 6.00
                Friday, February 12, 1999 15:58:43

LangType.... 1 1)COB 2)COB/CICS 3)PL1 4)PL1/CICS 5)ASM 6)ASM/CICS

TransCentury Engine Name WS-TRCENGIN

Copy Book Member Name...   TRCCONVR
Copy Inclusion Statement   COPY
Copylib File Name(s)....   userlib.PDS.COPYLIB
                           userlib.PDS.COPYLIB
Data Element File Name..   userlib.DATA.ELEMENT(IDNL)
                           Above file name(s) must be fully qualified
Press ENTER to continue, PFK3 to EXIT, PFK7 to Add COPYLIB(s)
Program Type set to Batch COBOL
```

The following is an example of the DLG program profile definition for a PL/1 L application program:

```
--------------------- DLG Program Profile Definition --------------------------
OPTION ===>
           Calendar Routine Customer/Version ABABABABAB 6.00
       Date Logic Generator Customer/Version ABABABABAB 6.00
                 Sunday, February 14, 1999 14:19:58

LangType.... 3 1)COB 2)COB/CICS 3)PL1 4)PL1/CICS 5)ASM 6)ASM/CICS

TransCentury Engine Name TRCENGN

Copy Book Member Name...    TRCCONVP
Copy Inclusion Statement    %INCLUDE
Copylib File Name(s)....    userlib.PDS.COPYLIB
                            userlib.Y2K.COPYLIB
Data Element File Name..    userlib.DATA.ELEMENT(IDNL)
                            Above file name(s) must be fully qualified

Press ENTER to continue, PFK3 to EXIT, PFK7 to Add COPYLIB(s)
Program Type set to Batch PL/1
```

The profile screen collects information used by the CA Date Logic Generator to edit and process a member. This information is permanently stored in an ISPF table file allocated for CA Date Logic Generator use. See Program Profile in the “DLG ISPF Table Files” appendix.

- LangType—Language type (batch and CICS)
- CA Engine Name—Name of the CA calendar processing routine
- Copy Book Member Name—Name of the application program link-pass area copybook member
- Copy Inclusion Statement—Source code statement used to include copybook members during compilation

- Copylib File Name(s)—Copylib file names containing copybook source language (must be standard Partitioned datasets)
- Data Element File Name—Name of the data element file which contains definitions of program data elements which have been identified as date fields by an analysis product (like CA Analysis for MVS). The data element names located in this file are used by the CA Date Logic Generator for automatic remediation of simple “IF” statements (described later in this section). See Program Data Elements in the “DLG ISPF Table Files appendix”

The previous Program Profile Parameters can be modified by the user (possible exceptions are the Copylib File Names).

The message: Press ENTER to continue, PF3 to EXIT, PF7 to Add COPYLIB(s) appears at the bottom of the screen. This message can also contain: PF7 to Add/Change COPYLIB(s).

Press Enter to continue with the next item of Step 2. Warning messages are displayed if invalid data or file names are key-entered on the DLG Profile Definition screen. Correct any errors, or clear screen data and press Enter.

Press PF3 to cancel DLG processing. The following screen display will appear on the standard ISPF Edit screen after PF3 is pressed:

```
File   Edit   Confirm   Menu   Utilities   Compilers   Test   Help
------------------------------------------------------------------------------
EDIT            PDGBO.PDS.COBOL(T1) - 01.99              Columns 00001 00072
Command ===>                                               Scroll ===> CSR
# PROFILE PROCESSING CANCELLED, YOU MUST CANCEL THE EDIT SESSION TO RESTART #
```

The above screen display is only a message. No permanent changes have been made to the program being edited. You can continue normal edit processing, making sure any temporary changes to the program source have been removed. You can also CANCEL the current edit session, ensuring no changes to the original program source are made.

When viewing the DLG Program Profile Definition screen, press PF7 to view and/or edit the Copylib File Names used by the CA Date Logic Generator during this ISPF Edit session. If this key is pressed, the following screen is displayed:

```
-------------------- DLG Copylib File Definition Panel -----------------------
OPTION ===>
  Copylib File Name(s).... ADP.CLIENT.NINEWEST.COPYLIB
                           PDGBO.WRK6000.COPYLIB
                           PDGBO.WRK6000.SAMPLE.COPYLIB
                           ____________________________________________
                           ____________________________________________
                           ____________________________________________
                           ____________________________________________
                           ____________________________________________
Press ENTER to continue processing, PF3 to CANCEL
Program Type set to Batch COBOL
```

This screen provides the user the ability to view and/or modify the Copylib Files used by DLG when processing the application program.

Note: These Copylib filenames must be the same files used by your installation compilers. All changes to Copylib filenames are validated. An error message is displayed at the bottom of the screen if the user key-enters an invalid file name.

The user must correct the entry on the following screen to continue processing.

```
-------------------- DLG Copylib File Definition Panel ----------------------
OPTION ==>
Copylib File Name(s)....     ADP.CLIENT.NINEWEST.COPYLIB
                             PDGBO.WRK6000.COPYLIB
                             PDGBO.WRK6000.SAMPLE.COPYLIB
                             invalid-file-name-4
                             invalid-file-name-5
Press ENTER to continue processing, PF3 to CANCEL
COPYLIB FILE NAME(s) NOT FOUND: 4 5
```

3. The Date Element file, described in item 2 above, is used to specify the name of a file containing definitions of program elements, which have been identified as date fields. The Data Element file, for this version of the CA Date Logic Generator, can be an MVS partitioned dataset or a MVS sequential dataset. The file defines program data elements, which are used in application programs.

   Note: CLEAR the Data Element File Name screen area if you do not use a Data Elements file and proceed to the next step.

   The CA Date Logic Generator uses the data element names stored in the Data Element file to locate procedure statements in COBOL and PL/1 programs. Place the name of a Data Element file and member name (if the Data Element information is located in an MVS partitioned dataset) to the Data Element File Name literal and press the Enter key.

```
-------------------- DLG Program Profile Definition -------------------------
OPTION ==>
                Calendar Routine Customer/Version ABABABABAB 6.00
              Date Logic Generator Customer/Version ABABABABAB 6.00
                        Friday, February 12, 1999 17:25:40
 LangType.... 1 1)COB  2)COB/CICS  3)PL1  4)PL1/CICS  5)ASM  6)ASM/CICS
 TransCentury Engine Name            WS-TRCENGIN
 Copy Book Member Name...   TRCCONVR
 Copy Inclusion Statement   COPY
 Copylib File Name(s)....   userlib.PDS.COPYLIB
                            userlib.Y2K.COPYLIB
 Data Element File Name..   userlib.DATA.ELEMENT.FILE
                            Above file name(s) must be fully qualified
Press ENTER to continue, PFK3 to EXIT, PFK7 to Add COPYLIB(s)
Program Type set to Batch COBOL
```

   A warning message is displayed if the Data Element File Name is invalid or the file does not exist:

```
Data Element File Name..   PDGBO.WRK6000.DATA.X
                           Above file name(s) must be fully qualified
Press ENTER to continue, PFK3 to EXIT, PFK7 to Add COPYLIB(s)
WARNING - DATA ELEMENT FILE NAME NOT FOUND
```

The user can correct the Data Element File Name or press the Enter key. If the user does not correct the file name and subsequently presses the Enter key, the Data Element File Name area is cleared (signifying no Data Element File processing) and processing continues with the next item 4, below.

If the file name is correct, the following will occur based upon the file specified in the Data Element File Name screen area:

- Standard sequential dataset. Proceed to item 4, below
- Partitioned dataset with existing member name specified {for example: usernode.PDS.DATA.ELEMENT(program)}. Proceed to the next step.
- Partitioned dataset with invalid member name specified, display WARNING message at bottom of screen
- Partitioned dataset with NO member name specified, display WARNING message at bottom of screen

If the file name is a Partitioned dataset and NO member name is specified, CA Date Logic Generator will display a standard MVS ISPF Browse member selection list:

```
VIEW   PDGBO.WRK6000.DATA.ELEMENT    Row 00001 of 00005
Command ===>    Scroll ===> CSR
Name        VV MM    Created     Changed          Size   Init Mod .
 EDSIDNL  01.03      98/12/15    99/01/07 15:14   13      6    0 .
 E2RIDNL  01.01      98/12/08    98/12/08 16:13   29      29   0 .
 LRSIDNL.
 SAVIDNL  01.35      97/11/30    98/12/23 12:44   302     8    0 .
 U21IDNL  01.02      98/03/31    98/04/06 00:18   4435    4488 0
 **End**
```

The user can view and subsequently select a member by placing an 'S' to the right of a specific member name and pressing the Enter key. The following screen to appear is an ISPF VIEW (or browse) screen display:

```
VIEW    PDGBO.WRK6000.DATA.ELEMENT(SAVIDNL)
****** ***************************** Top of Data ****************
000001 @LIST  IDNL     DNLIST  C305M1   1997/04/21   11:42
000002        P BASED  BASED   DATE_CHAR_6
000003        P BASED  BASED   DATE_CHAR_8
000004        P BASED  BASED   DATE_DEC_FIXED
000005        P BASED  BASED   DATE_DEC_7
000006        P BASED  BASED   DATE_DECIMAL_FIXED
000007        P BASED  BASED   DATE_EDIT_6PER
000008        P BASED  BASED   DATE_EDIT_6SLH
```

The previous Data Element file display shows the data element definitions for program BASED which are stored in member SAVIDNL. This member does not contain data element entries for the program we are editing: “T”, as shown below:

```
EDIT    PDGBO.PDS.COBOL(T) - 01.99
Command ===>
******   ***************************** Top of Data *
000001     IDENTIFICATION DIVISION.
000002     PROGRAM-ID. T.
000003     ENVIRONMENT DIVISION.
000004     CONFIGURATION SECTION.
000005     SOURCE-COMPUTER. XXX.
000006     OBJECT-COMPUTER. YYY.
```

After viewing and selecting a data element file (and/or member), CA Date Logic Generator scans the data element records for a matching program name. For this example, no data element definitions for program “T” exist in PDGBO.WRK6000.DATA.ELEMENT(SAVIDNL). When this occurs, the DLG Program Profile Definition is redisplayed with the following pop-up message providing additional instructions for the user.

```
 -------------------- DLG PROFILE DEFINITION --------------------------
OPTION ==>
     Calendar Routine Customer/Version ABABABABAB 6.00
  Date Logic Generator Customer/Version ABABABABAB 6.00
             Saturday, February 13, 1999 13:24:26
LangType.... 1 1)COB   2)COB/CICS   3)PL1   4)PL1/CICS   5)ASM   6)ASM/CICS
TransCentury                 ___________________________________________
                    | OPTION ==>                                        |
Copy Book Mem       |                                                   |
Copy Inclusio       |                                                   |
                    | No DATA ELEMENT definitions for Program T         |
Copylib File        | were found in the Data Element file:              |
                    | PDGBO.WRK6000.DATA.ELEMENT(E2RIDNL)               |
Data Element        |                                                   |
                    | 1 You can Exit (PF3) the process and BYPASS       |
                    | use of the Data Element file                      |
                    | 2 Press PF7 to view the Data Element file         |
                    | for the displayed Program ID (see below)          |
                    | 3 Change the contents of the PROGRAM-ID: T        |
                    | 4 Change the Data Element file name (above)       |
                    ____________________________________________________
 Press ENTER  PF3=EXIT   PF7=VIEW
 Program Type
```

4. The CA Date Logic Generator scans the program source being edited looking for a source library inclusion statement which inserts the CA Calendar Routines link/ pass area. Depending on the entry on the DLG Program Profile Definition Copy Inclusion Statement, CA Date Logic Generator scans for the following:

Entry	Scanned For
Cobol	TRCCONVR
PL/1	TRCCONVP
Assembler	TRCCONVA

If CA Date Logic Generator cannot locate the call/pass area copy book inclusion statement in a COBOL or PL/1 program, the DLG Program Profile Definition screen is displayed with a message prompting the user to select a place to insert the statement. A warning message is displayed for Assembler programs, should the call/pass copy book not be found.

```
--------------------- DLG Program Profile Definition -------------------------
OPTION ==>
           Calendar Routine   Customer/Version ABABABABAB 5.00
         Date Logic Generator Customer/Version BETA DLG  5.00
                SATURDAY, APRIL 18, 1998  19:17:52
LangType.... 1  1)COB  2)COB/CICS  3)PL1  4)PL1/CICS  5)ASM  6)ASM/CICS
TransCentury Engine Name             TRCENGIN
Copy Book Member Name...   TRCCONVR
Copy Inclusion Statement   COPY
Copylib File Name(s)..     userlib.PDS.COPYLIB
                           userlib.Y2K.COPYLIB
Date Element File Name .   userlib.DATA.ELEMENT(INDL)
Do You Want To place the COPY TRCCONVR Statement after
            WORKING STORAGE(W), or Not Inserted(N) W
Press ENTER to continue, PFK3 to EXIT, PFK7 to Add COPYLIB(s)
```

5. The DLG Program Profile Definition screen is displayed with a request to the user to view or change CA Date Logic Generator Business Parameter settings for the program being edited.

   Note: If the DLG screen and message are not displayed, go to the next step.

```
--------------------- DLG Program Profile Definition -------------------------
OPTION ==>
              Calendar Routine   Customer/Version ABABABABAB 5.00
            Date Logic Generator Customer/Version BETA DLG  5.00
                   SATURDAY, APRIL 18, 1998  19:17:52
LangType.... 1 1)COB  2)COB/CICS  3)PL1  4)PL1/CICS  5)ASM  6)ASM/CICS
TransCentury Engine Name             TRCENGIN
Copy Book Member Name...  TRCCONVR
Copy Inclusion Statement  COPY
Copylib File Name(s)..    ADP.PLANTINUM.CLIENT.NINEWEST.COPYLIB
                          PDGBO.WRK5000.COPYLIB
                          PDGBO.WRK5000.SAMPLE.COPYLIB
Date Element File Name .  PDGBO.WRK5000.DATA.ELEMENT (INDL)
Do You want to VIEW or CHANGE
TransCentury Calendar Routines Business Parameters (Y OR N) Y
Press ENTER to continue, PFK3 to EXIT, PFK7 to Add COPYLIB(s)
```

The user can key-enter an 'N' to bypass Business Parameters processing. Go to the next step if 'N' is entered. The user must enter a 'Y' and press the Enter key to view or update the DLG Business Parameters. When this is done, the CA Calendar Routines Business and DLG Generator Parameters screen is displayed.

```
--- TransCentury Calendar Routines Business and DLG Generator Parameters----
OPTION ==>
                  Holiday Table ID  01        BUSINESS PARAMETERS:
            Fiscal Year Begin Month  01
             Fiscal Month Begin Day  01
From/To Date End Points Definition  T         B=both, N=none, T=to, F=from
     Numerical Day-of-Week Setting  1234567   Sunday-Saturday (e.g. 1234567)
       Process Day-of-Week Setting  NEEEEEN   Sunday-Saturday (e.g. NEEEEEN)
       Century 'Window' Assignment  19/20;50  19/20;68 (fixed) or SLIDE-20
            INPUT 'From' Date Mask  YYMMDD--  (e.g. 'YYMMDD--' alphabetic)
              INPUT 'To' Date Mask  YYMMDD--  (e.g. '--YYMMDD' numeric)
              1st Output Date Mask  CCYYMMDD  (e.g. 'CCYYMMDD')
              2nd Output Date Mask  --------  (use '--------' if NO output)
              3rd Output Date Mask  --------  (use '--------' if NO output)
   Test for null input MASK Values  Y         ('Y' Test, 'N' No, ' ' Ignore)
          Alphabetic Literal CASE  M         ('U' Upper, 'M' Mixed)
                                             (DLG GENERATOR PARAMETERS:)
  PERFORM some functions or "CALL"  P         ('P' PERFORM, 'C' force CALL)
     Create TRC Standard-Date code  N         ('Y' generate MOVE statement)
      Create "ENDIF" when required  N         ('Y' generate ENDIF statement)
 Create PERFORM User-error-routine  N         ('Y' generate PERFORM)
Press enter to continue, PF3 to exit, PF5 RESET to installation default
```

This screen displays the default Business Parameters which will be used when the application program being edited processes CA Calendar Routines functions. The user can modify these parameters using the instructions contained in the *CA Calendar Routines Applications Manual*.

Any changes to the displayed Business Parameters are edited by CA Date Logic Generator. The CA Calendar Routines Business and DLG Generator Parameters screen is redisplayed with the appropriate error message when key-enter data is not valid. The user must correct any errors before processing can continue.

```
     --- TransCentury Calendar Routines Business and DLG Generator Parameters----
      OPTION ==>
                      Holiday Table ID  01        BUSINESS PARAMETERS:
                Fiscal Year Begin Month  01
                 Fiscal Month Begin Day  01
    From/To Date End Points Definition  T         B=both, N=none, T=to, F=from
         Numerical Day-of-Week Setting  1234567   Sunday-Saturday (e.g. 1234567)
           Process Day-of-Week Setting  NEEEEEN   Sunday-Saturday (e.g. NEEEEEN)
           Century 'Window' Assignment  19/20;50  19/20;68 (fixed) or SLIDE-20
|IVALID INPUT  | INPUT 'From' Date Mask |YYMMDDxx| (e.g. 'YYMMDD--' alphabetic)
|FROM DATE MASK|   INPUT 'To' Date Mask  YYMMDD--  (e.g. '--YYMMDD' numeric)
                  1st Output Date Mask  CCYYMMDD  (e.g. 'CCYYMMDD')
                  2nd Output Date Mask  --------  (use '--------' if NO output)
                  3rd Output Date Mask  --------  (use '--------' if NO output)
       Test for null input MASK Values  Y         ('Y' Test, 'N' No, ' ' Ignore)
              Alphabetic Literal CASE  M         ('U' Upper, 'M' Mixed)
                                                 (DLG GENERATOR PARAMETERS:)
      PERFORM some functions or "CALL"  P         ('P' PERFORM, 'C' force CALL)
         Create TRC Standard-Date code  N         ('Y' generate MOVE statement)
          Create "ENDIF" when required  N         ('Y' generate ENDIF statement)
     Create PERFORM User-error-routine  N         ('Y' generate PERFORM)
    Press enter to continue, PF3 to exit, PF5 RESET to installation default
|ERROR MESSAGE|--------------|INVALID FROM DATE MASK|
```

The Business and DLG Generator Parameters shown above can be reset to your installation default settings by pressing PF5.

6. COBOL—A scan is made by the CA Date Logic Generator if the program source code is COBOL and the user elects to dynamically invoke the CA Calendar Routines (see the name key-entered on the following screen as the CA Engine Name). Placing a data element name (which contains more than eight characters and/or uses COBOL naming conventions {-}), instructs the CA Date Logic Generator to create source statements which dynamically CALL the CA Calendar Routines. The scan of the COBOL program application source is conducted searching for a Data Element definition matching the CA Engine Name.

   If the CA Date Logic Generator does not locate the Data Element name in the application source program, the DLG Program Profile Definition screen is displayed with a message at the bottom of the screen display. The message prompts the user to specify the location for insertion of the Data Element definition (reply 'W') or non-insertion (reply 'N').

   The following example shows the DLG Program Profile Definition screen display requesting COBOL program statement insertion of WS-TRCENGIN data element definition.

```
--------------------- DLG Program Profile Definition ----------------------
OPTION ==>
        Calendar Routine Customer/Version ABABABABAB 6.00
     Date Logic Generator Customer/Version ABABABABAB 6.00
              Tuesday, February 16, 1999 13:19:06
LangType.... 1  1)COB  2)COB/CICS  3)PL1  4)PL1/CICS  5)ASM  6)ASM/CICS
TransCentury Engine Name          WS-TRCENGIN
Copy Book Member Name... TRCCONVR
Copy Inclusion Statement COPY
Copylib File Name(s).... PDGBO.PDS.COPYLIB
                         PDGBO.WRK6000.COPYLIB
Data Element File Name..
                         Above file name(s) must be fully qualified
Do You want to place the 01 WS-TRCENGIN CALL-program definition statement
       after WORKING STORAGE(W), or Not Inserted(N) W
Press ENTER to continue, PFK3 to EXIT, PFK7 to Add COPYLIB(s)
```

```
EDIT    PDGBO.PDS.COBOL(T) - 01.99
Command ===>    Scroll ===> CSR
000020
000021    WORKING-STORAGE SECTION.
000022
000023    01 INLINE-PASS-AREA.
000024       05 INLINE-DATEIN.
000025          10 INLINE-DATEIN-9 PIC 9(06).
000030    /
000031    *-------------------------------------------------------*
000032    *      TransCentury(W) WARNING - DO NOT REMOVE COPYBOOKS          *
000033    *             AND/OR GENERATED STATEMENTS.                *
000034    *-------------------------------------------------------*
000035    01 WS-TRCENGIN PIC X(08) VALUE 'TRCENGIN'.
000036       COPY TRCCONVR.
000037    PROCEDURE DIVISION.
000038
000039   IF USER-TO-DATE-9 NOT = ZEROS
```

PL/1—A scan is made by the CA Date Logic Generator if the program source code is PL/1 for a data element definition defining an external definition for the CA Calendar Routines (see the name key-entered on the following screen as the CA Engine Name).

If the CA Date Logic Generator does not locate the Data Element definition in the application source program, the DLG Program Profile Definition screen is displayed with a message at the bottom of the screen display. The message prompts the user to specify the location for insertion of the Data Element definition (reply 'Y') or non-insertion (reply 'N').

```
----------------- DLG Program Profile Definition -----------------------
OPTION ==>
           Calendar Routine Customer/Version ABABABABAB 6.00
    Date Logic Generator Customer/Version ABABABABAB 6.00
             Tuesday, February 16, 1999 15:47:43
LangType.... 3  1)COB  2)COB/CICS  3)PL1  4)PL1/CICS  5)ASM  6)ASM/CICS
TransCentury Engine Name           TRCENGN
Copy Book Member Name... TRCCONVP
Copy Inclusion Statement %INCLUDE
Copylib File Name(s).... PDGBO.PDS.COPYLIB
                         PDGBO.WRK6000.COPYLIB
Data Element File Name.. PDGBO.WRK6000.DATA.ELEMENT(SAVIDNL)
                         Above file name(s) must be fully qualified
Do You want To place the DCL statement for TRCENGN inserted after
     the PROC OPTIONS statement(Y), or Not Inserted(N)  Y
Press ENTER to continue, PFK3 to EXIT, PFK7 to Add COPYLIB(s)
```

```
EDIT    PDGBO.PDS.PL1(BASED) - 01.44 Columns 00001
Command ===>    Scroll ===> CSR
****** ********************** Top of Data ************************
000100    BASED: PROC OPTIONS(MAIN);
000110   /*-------------------------------------------------------*
000120    * TransCentury(W) WARNING - DO NOT REMOVE COPYBOOKS              *
000130    *-------------------------------------------------------*/
000131   DCL TRCENGN EXT ENTRY OPTIONS(ASM);
000140 %INCLUDE TRCCONVP;;
000200
000300   DCL SYSPRINT FILE STREAM OUTPUT;
000400
```

7. The CA Date Logic Generator can automatically replace all simple COBOL or PL/1 “IF” statements with program source statements which will move the from/to data elements and/or literals to the CA link/pass area and invoke the CA Calendar Routines COMPARE function (see Step 3—COBOL and PL/1 “IF” Statement Automated Modification).

   The DLG Program Profile Definition screen is displayed with a message prompting the user to convert (Y) or bypass (N) conversion of “IF” statements.

DLG Program Profile Definition processing will terminate if the user replies with an 'N' to the display prompt or if the program being edited is not COBOL or PL/1 (seethe next step).

```
-------------------- DLG Program Profile Definition ------------------------
OPTION ==>
         Calendar Routine      Customer/Version ABABABABAB 5.00
         Date Logic Generator  Customer/Version BETA DLG  5.00
                    SATURDAY, APRIL 18, 1998  19:17:52
TransCentury Engine Name            TRCENGIN
Copy Book Member Name...  TRCCONVR
Copy Inclusion Statement  COPY
Copylib File Name(s)..    ADP.PLANTINUM.CLIENT.NINEWEST.COPYLIB
                          PDGBO.WRK5000.COPYLIB
                          PDGBO.WRK5000.SAMPLE.COPYLIB
Date Element File Name .  PDGBO.WRK5000.DATA.ELEMENT (INDL)
Do You want to automatically CONVERT Simple IF statements (Y OR N) Y
Press ENTER to continue, PFK3 to EXIT, PFK7 to Add COPYLIB(s)
NOTE; If you automatically CONVERT, your current CURSOR position IS lost!
```

a. Key enter 'N' and press the Enter key if you do not want to modify simple “IF” statements (proceed to the next step).

   Or

b. Key enter 'Y' and press Enter to modify simple “IF” statements.

   A pop-up window appears in the middle of the screen when the user key enters a Y to the above screen prompt. This pop-up window displays the maximum estimated time required to automatically convert all simple “IF” statements contained in the program currently being edited.

   The following example shows the DLG estimated time to convert simple IF statement time estimates screen:

```
-------------------- DLG Program Profile Definition ------------------------
OPTION ==>
            Calendar Routine Customer/Version ABABABABAB 5.00
          Date Logic Generator Customer/Version BETA DLG  5.00
                   SATURDAY, APRIL 18, 1998  19:17:52
LangType.... 1  1)COB  2)COB/CICS  3)PL1  4)PL1/CICS  5)ASM  6)ASM/CICS
TransCentury
                         TransCentury Date Logic Generator
Copy Book Me
Copy Inclusi          Automated simple “IF” change processing
Copylib File            may take up to 1 Minute to complete.
                             (Press ENTER to continue)
Date Element
Do You want to automatically CONVERT Simple IF statements (Y OR N) Y

Press ENTER to continue, PFK3 to EXIT, PFK7 to Add COPYLIB(s)
NOTE; If you automatically CONVERT, your current CURSOR position IS lost!
```

The automatic conversion of “IF” statement function examines the COBOL or PL/1 program source currently being edited and locates “IF” statements. The “IF” statements are examined determining the format of the comparison.

“IF” statements will be converted to the CA Calendar Routines Compare Two Dates (Same Format) function when the following applies:

a. One of the data elements has a matching entry in the Data Elements file (see Data Element File Name). No statements with indexed data elements will be converted.

b. The IF statement contains a single comparison (e.g. “IF” data-element > ‘97123’ THEN). NO COMPOUND statements will be converted (e.g. AND, OR, ELSE IF).

   A pop-up window appears in the middle of the screen at the conclusion of automatic “IF” statement conversion. The pop-up window displays results of “IF” statement conversion processing.

```
------------------- DLG Program Profile Definition -------------------------
OPTION ==>
         Calendar Routine Customer/Version ABABABABAB 6.00
       Date Logic Generator Customer/Version ABABABABAB 6.00
                 Saturday, February 27, 1999 18:41:06
LangType.... 1 1)COB 2)COB/CICS 3)PL1 4)PL1/CICS 5)ASM 6)ASM/CICS
TransCentury
                                        TransCentury Date Logic Generator
Copy Book Me
Copy Inclusi                        Automated simple "IF" Change processing
Copylib File                   Data Element matches 10   Statements Changed 8
Data Element                           (Press ENTER to continue)

Do You want to automatically CONVERT Simple IF statements (Y/N) y

Press ENTER to continue, PFK3 to EXIT, PFK7 to Add COPYLIB(s)
NOTE; If you automatically CONVERT, your current CURSOR position IS lost!
```

8. The program currently being edited will not be automatically modified if the program is designated (see Resetting the DLG Program Profile Definition) as Assembler. The program will also not normally be automatically modified if the program has been previously edited using the CA Date Logic Generator.

   Most automated changes occur to COBOL or PL/1 programs during the initial execution of the CA Date Logic Generator. Automatically modified COBOL or PL/1 programs have matching entries in the Data Elements file and contain simple “IF” statements referencing these elements.

   The following examples show a COBOL program and PL/1 program before and after automatic DLG changes.

```
IDENTIFICATION DIVISION.
PROGRAM-ID. LOADTABL.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. XXX.
OBJECT-COMPUTER. YYY.
*---------------------------------
*
*      TEST QUOTE VERSION "5.0"
*
*---------------------------------
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT IN-FILE ASSIGN TO UT-S-INP.
DATA DIVISION.
FILE SECTION.
FD IN-FILE.
01  IN-REC.
    05  USER-FROM-DATE.
```

```
        10  USER-FROM-DATE-9 PIC 9(06).
    05  USER-TO-DATE.
        10  USER-TO-DATE-9 PIC 9(06).
    05  USER-STDOUT PIC X(08).
    05  USER-OUT1  PIC X(08).
    05  USER-OUT2  PIC X(08).
    05  USER-OUT3  PIC X(08).
    05  USER-DOW  PIC 9(01 ).
    05  USER-HOLIDAY PIC 9(01 ).
    05  USER-DAY-COUNT   PIC 9(06).
    05  USER-DAY-NAME   PIC X(25).
WORKING-STORAGE SECTION.
01  INLINE-PASS-AREA.
    05  INLINE-DATEIN.
        10  INLINE-DATEIN-9 PIC 9(06).
    05  INLINE-RETURN-CODE PIC 9(01 ) VALUE 0.
        88 INLINE-GOOD-RETURN   VALUE 0.
    05  INLINE-HOLIDAY   PIC 9(01 ).

  PROCEDURE DIVISION.

IF USER-TO-DATE-9 > USER-FROM-DATE-9
   DISPLAY '"ERRORS"'. IF USER-FROM-DATE > USER-TO-DATEDISPLAY 'ERRORS'.
IF USER-FROM-DATE = USER-TO-DATE
   DISPLAY 'LAST'.
IF USER-FROM-DATE NOT > USER-TO-DATE
   DISPLAY 'NEXT'.
MOVE USER-FROM-DATE TO INLINE-DATEIN.
CALL 'XYZ' USING INLINE-PASS-AREA.
IF INLINE-GOOD-RETURN
   MOVE INLINE-HOLIDAY TO USER-HOLIDAY.
IF USER-FROM-DATE > USER-TO-DATE
   DISPLAY '" THE NEXT ITEM YOU WILL SEE"'.
ALL-DONE.
    GOBACK.
```

The following example shows a COBOL program after ISPFEDIT-CA Date Logic Generator profile processing:

```
IDENTIFICATION DIVISION.
PROGRAM-ID. LOADTABL.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. XXX.
OBJECT-COMPUTER. YYY.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT IN-FILE ASSIGN TO UT-S-INP.
DATA DIVISION.
FILE SECTION.
FD IN-FILE.
01  IN-REC.
    05  USER-FROM-DATE.
        10  USER-FROM-DATE-9 PIC 9(06).
    05  USER-TO-DATE.
        10  USER-TO-DATE-9 PIC 9(06).
    05  USER-STDOUT PIC X(08).
    05  USER-OUT1  PIC X(08).
    05  USER-OUT2  PIC X(08).
    05  USER-OUT3  PIC X(08).
    05  USER-DOW  PIC 9(01 ).
    05  USER-HOLIDAY PIC 9(01 ).
    05  USER-DAY-COUNT   PIC 9(06).
    05  USER-DAY-NAME   PIC X(25).

WORKING-STORAGE SECTION.

01  INLINE-PASS-AREA.
    05  INLINE-DATEIN.
        10  INLINE-DATEIN-9 PIC 9(06).
    05  INLINE-RETURN-CODE PIC 9(01 ) VALUE 0.
```

```
        88  INLINE-GOOD-RETURN   VALUE 0.
    05  INLINE-HOLIDAY   PIC 9(01 ).
/
        COPY TRCCONVR.
/
        COPY TRCU1XWS.
/
**=> IF USER-TO-DATE-9 > USER-FROM-DATE-9
*-------------------------------------------------------*
*     TransCentury(S) COBOL (U01 1) 'QUICK COMPARE FROM/TO'       *
*-------------------------------------------------------*
    MOVE '--YYMMDD' TO TRC-CONVR-FROM-DATE-MASK
    MOVE '--YYMMDD' TO TRC-CONVR-TO-DATE-MASK
    MOVE 'SLIDE-20' TO TRC-CONVR-CENTURY-BREAK
    MOVE USER-TO-DATE-9 TO TRC-CONVR-FROM-DATE-9
    MOVE USER-FROM-DATE-9 TO TRC-CONVR-TO-DATE-9
    MOVE 'U011'   TO TRC-CONVR-FUNCTION-CODE
    PERFORM TRC-CALENDAR-ROUTINES
    IF TRC-CONVR-FROM-GREATER-THAN-TO
*-------------------------------------------------------*
*            END OF GENERATED TransCentury(E) CODE           *
*-------------------------------------------------------*
            DISPLAY '"ERRORS"'.
**=> IF USER-FROM-DATE > USER-TO-DATE
*-------------------------------------------------------*
*     TransCentury(S) COBOL (U011) 'QUICK COMPARE FROM/TO'        *
*-------------------------------------------------------*
MOVE 'GENR-X' TO TRC-CONVR-FROM-DATE-MASK
MOVE 'GENR-X' TO TRC-CONVR-TO-DATE-MASK
MOVE 'SLIDE-20' TO TRC-CONVR-CENTURY-BREAK
MOVE USER-FROM-DATE TO TRC-CONVR-FROM-DATE-E
MOVE USER-TO-DATE TO TRC-CONVR-TO-DATE-E
MOVE 'U011'   TO TRC-CONVR-FUNCTION-CODE
PERFORM TRC-CALENDAR-ROUTINES
IF TRC-CONVR-FROM-GREATER-THAN-TO

*-------------------------------------------------------*
*   END OF GENERATED TransCentury(E) CODE                     *
*-------------------------------------------------------*
DISPLAY 'ERRORS'.
**=> IF USER-FROM-DATE = USER-TO-DATE
*-------------------------------------------------------*
*     TransCentury(S) COBOL (U011) 'QUICK COMPARE FROM/TO'        *
*-------------------------------------------------------*
    MOVE 'GENR-X' TO TRC-CONVR-FROM-DATE-MASK
    MOVE 'GENR-X' TO TRC-CONVR-TO-DATE-MASK
    MOVE 'SLIDE-20' TO TRC-CONVR-CENTURY-BREAK
    MOVE USER-FROM-DATE TO TRC-CONVR-FROM-DATE-E
    MOVE USER-TO-DATE TO TRC-CONVR-TO-DATE-E
    MOVE 'U011'   TO TRC-CONVR-FUNCTION-CODE
    PERFORM TRC-CALENDAR-ROUTINES IF TRC-CONVR-FROM-EQUALS-TO
*-------------------------------------------------------*
*       END OF GENERATED TransCentury(E) CODE                   *
*-------------------------------------------------------*
DISPLAY 'LAST'.
**=> IF USER-FROM-DATE NOT > USER-TO-DATE
*-------------------------------------------------------*
*     TransCentury(S) COBOL (U011) 'QUICK COMPARE FROM/TO'        *
*-------------------------------------------------------*
    MOVE 'GENR-X' TO TRC-CONVR-FROM-DATE-MASK
    MOVE 'GENR-X' TO TRC-CONVR-TO-DATE-MASK
    MOVE 'SLIDE-20' TO TRC-CONVR-CENTURY-BREAK
    MOVE USER-FROM-DATE TO TRC-CONVR-FROM-DATE-E
    MOVE USER-TO-DATE TO TRC-CONVR-TO-DATE-E
    MOVE 'U011'   TO TRC-CONVR-FUNCTION-CODE
    PERFORM TRC-CALENDAR-ROUTINES
    IF TRC-CONVR-FROM-NOT-GREATER-TO
*-------------------------------------------------------*
*       END OF GENERATED TransCentury(E) CODE                   *
*-------------------------------------------------------*
        DISPLAY 'NEXT'.
    MOVE USER-FROM-DATE TO INLINE-DATEIN.
    CALL 'XYZ' USING INLINE-PASS-AREA.
    IF INLINE-GOOD-RETURN
```

```
    MOVE INLINE-HOLIDAY TO USER-HOLIDAY.
**=> IF USER-FROM-DATE > USER-TO-DATE
*-------------------------------------------------------*
*     TransCentury(S) COBOL (U011) 'QUICK COMPARE FROM/TO'        *
*-------------------------------------------------------*
    MOVE 'GENR-X' TO TRC-CONVR-FROM-DATE-MASK
    MOVE 'GENR-X' TO TRC-CONVR-TO-DATE-MASK
    MOVE 'SLIDE-20' TO TRC-CONVR-CENTURY-BREAK
    MOVE USER-FROM-DATE TO TRC-CONVR-FROM-DATE-E
    MOVE USER-TO-DATE TO TRC-CONVR-TO-DATE-E
    MOVE 'U011'   TO TRC-CONVR-FUNCTION-CODE
    PERFORM TRC-CALENDAR-ROUTINES
    IF TRC-CONVR-FROM-GREATER-THAN-TO
*-------------------------------------------------------*
*      END OF GENERATED TransCentury(E) CODE                    *
*-------------------------------------------------------*
          DISPLAY '" THE NEXT ITEM YOU WILL SEE"'.
    ALL-DONE.
        GOBACK.
    /
        COPY TRCU1XPD.
    /
     TRC-STANDARD-CALL.
        IF TRC-CONVR-RETURN-GOOD
           NEXT SENTENCE.
```

The following example shows a PL/1 program before ISPFEDIT-CA Date Logic Generator profile processing screen

```
BASED: PROC OPTIONS(MAIN);
     DCL SYSPRINT FILE STREAM OUTPUT;
     DCL TTMMJJ     PIC '999999';
     DCL MMDDYY     CHAR(6);
     DCL SUM1   FLOAT DEC(6);
     DCL SUM2   FIXED BIN(31);
     DCL TABLE1(10)   FLOAT DEC (6);
     DCL TABLE2(20)   FIXED BIN(15) BASED(TABLE_POINTER) %INCLUDE SYSLIB(COPY1);;
     OPEN FILE(SYSPRINT);
     GET LIST(TABLE1);
     SUM1 = SUM(TABLE1);
     PUT DATA(SUM1);
     IF TITLE ¬-> MMDDYY THEN TABLE_POINTER = ADDR(TABLE1);
     IF DATE_GRP1S1 = DATE_8 THEN
         TABLE_POINTER = ADDR(TABLE1);
     IF DATE_GRP1F1 = DATE_8 THEN
         DO;
           TABLE_POINTER = ADDR(TABLE1);
           GET LIST(TABLE2);
           SUM2 = SUM(TABLE2);
           PUT DATA(SUM2);
         END;
     IF DATE_GRP1F2 = DATE_8 THEN
         TABLE_POINTER = ADDR(TABLE1);
     IF DATE_GRP1F3 = DATE_8 THEN
         TABLE_POINTER = ADDR(TABLE1);
     IF DATE_GRP1F4 = DATE_8 THEN
         TABLE_POINTER = ADDR(TABLE1);
     IF DATE_GRP1F6 = DATE_8 THEN
         TABLE_POINTER = ADDR(TABLE1);
     IF DATE_GRP1F7 = DATE_8 THEN
         TABLE_POINTER = ADDR(TABLE1);
     IF DATE_GRP1F12 = DATE_8 THEN
         TABLE_POINTER = ADDR(TABLE1);
     IF JUL_DT = DATE_STR2 THEN
```

The following examples shows the PL/1 program after ISPFEDIT-CA Date Logic Generator profile processing:

```
BASED: PROC OPTIONS(MAIN);    00010000
/*-------------------------------------------------------*      00011000
 * TransCentury(W) WARNING - DO NOT REMOVE COPYBOOKS             *
00012000
 *-------------------------------------------------------*/     00013000
   DCL TRCENGN EXT ENTRY OPTIONS(ASM);                          00013100
%INCLUDE TRCCONVP;;                                             00014000
                                                                00020000
   DCL SYSPRINT FILE STREAM OUTPUT;                             00030000
                                                                00040000
   DCL TTMMJJ     PIC '999999';                                 00050000
   DCL MMDDYY     CHAR(6);                                      00060000
   DCL SUM1   FLOAT DEC(6);                                     00070000
   DCL SUM2   FIXED BIN(31);                                    00080000
   DCL TABLE1(10)   FLOAT DEC (6);                              00090000
   DCL TABLE2(20)   FIXED BIN(15) BASED(TABLE_POINTER);         00100000
   DCL TABLE_POINTER POINTER;                                   00110000
                                                                00120000
   %INCLUDE SYSLIB(COPY1);;                                     00130000
                                                                00140000
   DCL (VERIFY,SUBSTR,PLIDUMP,ADDR,LENGTH,NULL,INDEX,SUM) BUILTIN;
                                                                00150000
                                                                00160000
   OPEN FILE(SYSPRINT);                                         00170000
                                                                00180000
   GET LIST(TABLE1);                                            00190000
   SUM1 = SUM(TABLE1);                                          00200000
   PUT DATA(SUM1);                                              00210000
/*=>   IF TITLE ^> MMDDYY THEN TABLE_POINTER = ADDR(TABLE1);
*/                                                              00220000
/*-------------------------------------------------------*/    00221000
/* TransCentury(S) PL-1 (U011) 'QUICK COMPARE FROM/TO'  */              00222000
/*-------------------------------------------------------*/    00223000
   TRC_CONVR_FROM_DATE_MASK = 'GENR-X';                         00224000
   TRC_CONVR_TO_DATE_MASK = 'GENR-X';                           00225000
   TRC_CONVR_CENTURY_BREAK = '19/20;50';                        00226000
   TRC_CONVR_FROM_DATE_E   = TITLE;                             00227000
   TRC_CONVR_TO_DATE_E    = MMDDYY;                             00228000
   TRC_CONVR_FUNCTION_CODE = 'U011';                            00229000
   CALL 'TRCENGN' (TRC_CONVR_CONVERSATIONAL);                   00229100
   IF TRC_CONVR_OUT_NUMERIC_PARM ^= +1 THEN                     00229200
/*-------------------------------------------------------*/    00229300
/*   END OF GENERATED CA(E) CODE */                             00229400
/*-------------------------------------------------------*/    00229500
   TABLE_POINTER = ADDR(TABLE1);                                00229600
                                                                00230000
/*=>   IF DATE_GRP1S1 = DATE_8 THEN   */                        00240000
/*-------------------------------------------------------*/    00241000
/*   TransCentury(S) PL-1 (U011) 'QUICK COMPARE FROM/TO'  */              00242000
/*-------------------------------------------------------*/    00243000
   TRC_CONVR_FROM_DATE_MASK = 'GENR-X';                         00244000
   TRC_CONVR_TO_DATE_MASK = 'GENR-X';                           00245000
   TRC_CONVR_CENTURY_BREAK = '19/20;50';                        00246000
   TRC_CONVR_FROM_DATE_E   = DATE_GRP1S1;                       00247000
   TRC_CONVR_TO_DATE_E    = DATE_8;                             00248000
   TRC_CONVR_FUNCTION_CODE = 'U011';                            00249000
   CALL 'TRCENGN' (TRC_CONVR_CONVERSATIONAL);                   00249100
   IF TRC_CONVR_OUT_NUMERIC_PARM = +0 THEN                      00249200
/*-------------------------------------------------------*/    00249300
/*   END OF GENERATED TransCentury(E) CODE */                           00249400
/*-------------------------------------------------------*/    00249500
   TABLE_POINTER = ADDR(TABLE1);                                00250000
                                                                00260000
/*=>   IF DATE_GRP1F1 = DATE_8 THEN    */                      00270000
/*-------------------------------------------------------*/    00271000
/* TransCentury(S) PL-1 (U011) 'QUICK COMPARE FROM/TO'  */              00272000
/*-------------------------------------------------------*/    00273000
   TRC_CONVR_FROM_DATE_MASK = 'GENR-X';                         00274000
   TRC_CONVR_TO_DATE_MASK = 'GENR-X';                           00275000
   TRC_CONVR_CENTURY_BREAK = '19/20;50';                        00276000
   TRC_CONVR_FROM_DATE_E   = DATE_GRP1F1                        00277000
   TRC_CONVR_TO_DATE_E    = DATE_8;                             00278000
   TRC_CONVR_FUNCTION_CODE = 'U011';                            00279000
   CALL 'TRCENGN' (TRC_CONVR_CONVERSATIONAL);                   00279100
```

```
   IF TRC_CONVR_OUT_NUMERIC_PARM = +0 THEN                    00279200
/*-------------------------------------------------------*/    00279300
/*   END OF GENERATED TransCentury(E) CODE */                          00279400
/*-------------------------------------------------------*/    00279500
   DO;                                                        00280000
      TABLE_POINTER = ADDR(TABLE1);                           00290000
      GET LIST(TABLE2);                                       00300000
      SUM2 = SUM(TABLE2);                                     00310000
      PUT DATA(SUM2);                                         00320000
   END;                                                       00330000
                                                              00340000
/*=>   IF DATE_GRP1F2 = DATE_8 THEN  */                       00350000
/*-------------------------------------------------------*/    00351000
/* TransCentury(S) PL-1 (U011) 'QUICK COMPARE FROM/TO'  */            00352000
/*-------------------------------------------------------*/    00353000
   TRC_CONVR_FROM_DATE_MASK = '---YYDDD';                     00354000
   TRC_CONVR_TO_DATE_MASK = '---YYDDD';                       00355000
   TRC_CONVR_CENTURY_BREAK = '19/20;50';                      00356000
   TRC_CONVR_FROM_DATE_9   = DATE_GRP1F2;                     00357000
   TRC_CONVR_TO_DATE_9    = DATE_8;                           00358000
   TRC_CONVR_FUNCTION_CODE = 'U011';                          00359000
   CALL 'TRCENGN' (TRC_CONVR_CONVERSATIONAL);                 00359100
   IF TRC_CONVR_OUT_NUMERIC_PARM = +0 THEN                    00359200
/*-------------------------------------------------------*/    00359300
/*   END OF GENERATED TransCentury(E) CODE */                          00359400
/*-------------------------------------------------------*/    00359500
   TABLE_POINTER = ADDR(TABLE1);                              00360000
                                                              00370000
/*=>   IF DATE_GRP1F3 = DATE_8 THEN    */                     00380000
/*-------------------------------------------------------*/    00381000
/* TransCentury(S) PL-1 (U011) 'QUICK COMPARE FROM/TO'  */            00382000
/*-------------------------------------------------------*/    00383000
   TRC_CONVR_FROM_DATE_MASK = '--YYMMDD';                     00384000
   TRC_CONVR_TO_DATE_MASK = '--YYMMDD';                       00385000
   TRC_CONVR_CENTURY_BREAK = '19/20;50';                      00386000
   TRC_CONVR_FROM_DATE_9   = DATE_GRP1F3;                     00387000
   TRC_CONVR_TO_DATE_9    = DATE_8;                           00388000
   TRC_CONVR_FUNCTION_CODE = 'U011';                          00389000
   CALL '   TRCENGN' (   TRC_CONVR_CONVERSATIONAL);           00389100
   IF    TRC_CONVR_OUT_NUMERIC_PARM = +0 THEN                 00389200
/*-------------------------------------------------------*/    00389300
/*   END OF GENERATED TransCentury(E) CODE */                          00389400
/*-------------------------------------------------------*/    00389500
   TABLE_POINTER = ADDR(TABLE1);                              00390000
                                                              00400000
   IF DATE_GRP1F4 = DATE_8 THEN                               00410000
   TABLE_POINTER = ADDR(TABLE1);                              00420000
                                                              00430000
/*=>   IF DATE_GRP1F6 = DATE_8 THEN */                        00440000
/*-------------------------------------------------------*/    00441000
/* TransCentury(S) PL-1 (U011) 'QUICK COMPARE FROM/TO'  */            00442000
/*-------------------------------------------------------*/    00443000
   TRC_CONVR_FROM_DATE_MASK = 'GENR-X';                       00444000
   TRC_CONVR_TO_DATE_MASK = 'GENR-X';                         00445000
   TRC_CONVR_CENTURY_BREAK = '19/20;50';                      00446000
   TRC_CONVR_FROM_DATE_E   = DATE_GRP1F6;                     00447000
   TRC_CONVR_TO_DATE_E    = DATE_8;                           00448000
   TRC_CONVR_FUNCTION_CODE = 'U011';                          00449000
   CALL '   TRCENGN' (   TRC_CONVR_CONVERSATIONAL);           00449100
   IF    TRC_CONVR_OUT_NUMERIC_PARM = +0 THEN                 00449200
/*-------------------------------------------------------*/    00449300
/*   END OF GENERATED TransCentury(E) CODE */                          00449400
/*-------------------------------------------------------*/    00449500
   TABLE_POINTER = ADDR(TABLE1);                              00450000
                                                              00460000
/*=>   IF DATE_GRP1F7 = DATE_8 THEN    */                     00470000
/*-------------------------------------------------------*/    00471000
/* TransCentury(S) PL-1 (U011) 'QUICK COMPARE FROM/TO'  */            00472000
/*-------------------------------------------------------*/    00473000
   TRC_CONVR_FROM_DATE_MASK = 'GENR-X';                       00474000
   TRC_CONVR_TO_DATE_MASK = 'GENR-X';                         00475000
   TRC_CONVR_CENTURY_BREAK = '19/20;50';                      00476000
   TRC_CONVR_FROM_DATE_E   = DATE_GRP1F7;                     00477000
   TRC_CONVR_TO_DATE_E    = DATE_8;                           00478000
```

```
   TRC_CONVR_FUNCTION_CODE = 'U011';                        00479000
   CALL '   TRCENGN' (   TRC_CONVR_CONVERSATIONAL);         00479100
   IF    TRC_CONVR_OUT_NUMERIC_PARM = +0 THEN               00479200
/*---------------------------------------------------------*/ 00479300
/*   END OF GENERATED TransCentury(E) CODE */                        00479400
/*---------------------------------------------------------*/ 00479500
TABLE_POINTER = ADDR(TABLE1);                               00480000
                                                            00490000
/*=>   IF DATE_GRP1F12 = DATE_8 THEN   */                   00500000
/*---------------------------------------------------------*/ 00501000
/* TransCentury(S) PL-1 (U011) 'QUICK COMPARE FROM/TO'  */          00502000
/*---------------------------------------------------------*/ 00503000
   TRC_CONVR_FROM_DATE_MASK = 'GENR-X';                     00504000
   TRC_CONVR_TO_DATE_MASK = 'GENR-X';                       00505000
   TRC_CONVR_CENTURY_BREAK = '19/20;50';                    00506000
   TRC_CONVR_FROM_DATE_E   = DATE_GRP1F12;                  00507000
   TRC_CONVR_TO_DATE_E    = DATE_8;                         00508000
   TRC_CONVR_FUNCTION_CODE = 'U011';                        00509000
   CALL '   TRCENGN' (   TRC_CONVR_CONVERSATIONAL);         00509100
   IF    TRC_CONVR_OUT_NUMERIC_PARM = +0 THEN               00509200
/*---------------------------------------------------------*/ 00509300
/*   END OF GENERATED TransCentury(E) CODE */                        00509400
/*---------------------------------------------------------*/ 00509500
   TABLE_POINTER = ADDR(TABLE1);                            00510000
                                                            00520000
/*=>   IF JUL_DT = DATE_STR2 THEN     */                   00530000
/*---------------------------------------------------------*/ 00531000
/* TransCentury(S) PL-1 (U011) 'QUICK COMPARE FROM/TO'  /          00532000
/*---------------------------------------------------------*/ 00533000
   TRC_CONVR_FROM_DATE_MASK = 'GENR-X';                     00534000
   TRC_CONVR_TO_DATE_MASK = 'GENR-X';                       00535000
   TRC_CONVR_CENTURY_BREAK = '19/20;50';                    00536000
   TRC_CONVR_FROM_DATE_E   = JUL_DT;                        00537000
   TRC_CONVR_TO_DATE_E    = DATE_STR2;                      00538000
   TRC_CONVR_FUNCTION_CODE = 'U011';                        00539000
   CALL '   TRCENGN' (   TRC_CONVR_CONVERSATIONAL);         00539100
   IF    TRC_CONVR_OUT_NUMERIC_PARM = +0 THEN               00539200
/*---------------------------------------------------------*/ 00539300
/*   END OF GENERATED TransCentury(E) CODE */                        00539400
/*---------------------------------------------------------*/ 00539500
   TABLE_POINTER = ADDR(TABLE1);                            00540000
                                                            00550000
END BASED;                                                  02110000
```

Step 3. COBOL and PL/1 “IF” Statement Automated Modification

This CA Date Logic Generator function can replace COBOL and PL/1 “IF” statements with program source statements which will move the from/to data elements and/or literals to the CA link/pass area and invoke the CA Calendar Routines compare function. This function is used when:

- A data element file does not contain all data elements, which are date fields,

  Or

- When a data element file does not exist or is not used.

To invoke this function, perform the steps below.

Note: This function can be performed as part of the Program Profile processing described in item 4 when a Data Element file is used.

Place the cursor under (or on) the desired “IF” statement and press the Program Function Key (PF key) assigned to the CA Date Logic Generator (e.g., PF18).

Note: The cursor must be under or on the “I” in “IF”.

The following screen is the ISPF EDIT screen before invoking the CA Date Logic Generator “IF” statement processing function:

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
EDIT   PDGBO.PDS.COBOL(DEMOCOB2) - 01.05     Columns 00001 00072
Command ===> Scroll ===> CSR
Cursor is placed under or on the ‘I’ of the first IF
000100
000101
000102
  WHEN TRC-CARDS-REC-CD EQUAL 1
 TRC-CARDS-FROM-DATE  = TRC-CARDS-TO-DATE
statement
000103
  DISPLAY ERROR-MSG
000104  END-IF
000105
000106  WHEN TRC-CARDS-REC-CD EQUAL 2
000107 IF TRC-CARDS-FROM-DATE GREATER
000108 TRC-CARDS-TO-DATE
000109 MOVE “*” TO TRC-CARDS-REC-WS
000110 MOVE TRC-CARDS-REC-WS TO
000111 TRC-DRIVR-TRANS-RECORD
000112  END-IF
000113
000114  WHEN TRC-CARDS-REC-CD EQUAL 3
000115 IF TRC-CARDS-FROM-DATE-9 NOT = 971231
000116 SET TRC-CARDS-EOF-YES TO TRUE
000117  END-IF
000118
000119  WHEN OTHER
```

The CA Date Logic Generator will examine the COBOL language statement to determine from/to data element or literal type.

The CA Date Logic Generator will transfer processing to Standard functions (see Step 4. Edit the Selection through Step 6. Generate the Code into the Application Program) should the Generator not successfully be able to generate valid COBOL or PL/1 source statements to invoke the CA calendar Compare Two Dates (Same Format) function.

The following screen appears after the terminal user invokes the CA Date Logic Generator, by pressing the key assigned to TCDSSPF1 (see Step 4. Setup PFKEYS in User TSO ISPF EDIT Profile in “Installing the CA Date Logic Generator”).

```
 File Edit Confirm Menu Utilities Compilers Test Help
EDIT    PDGBO.PDS.COBOL(DEMOCOB2) - 01.06  Columns 00001 00072
Command ===>    Scroll ===> CSR
000100  WHEN TRC-CARDS-REC-CD EQUAL 1
000101 ***=>  IF TRC-CARDS-FROM-DATE =
000102 ***=>    TRC-CARDS-TO-DATE
000103 ************************************************************
000104 *  BEGINNING OF CA CODE   *
000105 ************************************************************
000106  MOVE “GENR-X” TO TRC-CONVR-FROM-DATE-MASK
000107  MOVE “GENR-X” TO TRC-CONVR-TO-DATE-MASK
000108  MOVE “19/20;68” TO TRC-CONVR-CENTURY-BREAK
000109  MOVE TRC-CARDS-FROM-DATE TO TRC-CONVR-FROM-DATE-E
000110  MOVE TRC-CARDS-TO-DATE TO TRC-CONVR-TO-DATE-E
000111  MOVE “U011”   TO TRC-CONVR-FUNCTIONS-CODE
000112  PERFORM TRC-CALENDAR-ROUTINES
000113
000114  IF TRC-CONVR-FROM-EQUALS-T
O000115 ************************************************************
000116 *  THE END OF CA CODE   *
000117 ************************************************************
000118   DISPLAY ERROR-MSG
000119     END-IF
```

The original “IF” statement has been changed to program comments. Program source code to invoke the CA Calendar compare function “replaces” the original “IF” statement.

Step 4. Edit the Selection

Note: Steps 4-6 apply to all non-COBOL / PL/1 “IF” statement CA Calendar processing functions, COBOL / PL/1 “IF” statements that the CA Date Logic Generator could not process, or non-IF statement application program source code.

The instructions in this step apply to the majority of CA Calendar Routines date processing functions. Seven sub steps guide the user through CA Calendar Routines Function Selection and Function Specification.

The remainder of this section provides instructions for testing the DLG function and generating application program source statements to invoke the DLG function.

Note: The examples used in this section are for a PL/1 program.

To edit the selection

1. While editing a program using the CA Date Logic Generator, place the cursor on a program source statement line and press the PF key assigned to TCDSSPF1 (F18).

   The following screen shows the PL/1 source program with ISPF EDIT cursor located under “C” of CURR_DATE:

```
File Edit Confirm Menu Utilities Compilers Test Help
EDIT   PDGBO.PDS.PL1(BASED) - 01.25   Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
*********************************Top of Data*********************************
BASED: PROC OPTIONS (MAIN);
DCL SYSPRINT FILE STREAM OUTPUT;
DCL DAT1 PIC '99.99.99'   INIT ('0')   STATIC;
DCL DAT2 PIC '99/99/99';
DCL TTMMJJ   PIC '999999';
DCL MMDDYY    CHAR (6);
DCL SUM1  FLOAT DEC (6);
DCL SUM2  FIXED BIN (31);
DCL TABLE1 (10) FLOAT DEC (6);
DCL TABLE2 (20) FLOAT BIN (15) BASED (TABLE_POINTER);
DCL TABLE_POINTER   POINTER;
%INCLUDE SYSLIB (COPY1);;
%INCLUDE SYSLIB (TRCCONVP);;
. . . . . .
. . . . . .
. . . . . . (Additional PL/1 Statements)
. . . . . .
. . . . . .
IF TRC_CONVR_FROM_DATE_S = TRC_CONVR_TO_DATE_S THEN
DO;
TABLE_POINTER = ADDR (TABLE1);
GET LIST (TABLE2);
SUM2 = SUM(TABLE2);
PUT DATA (SUM2);
END;;
IF CURR_DATE = MMDDYY THEN
ISPF Edit Cursor
TABLE_POINTER = ADDR (TABLE1);
IF TITLE3 > MMDDYY THEN TABLE_POINTER = ADDR (TABLE1);
IF TITLE4   = JUL_DT THEN
DO;
TABLE_POINTER = ADDR (TABLE1);
GET LIST (TABLE2);
SUM2 = SUM (TABLE2);
PUT DATA (SUM2);
```

2. The DLG Function Selection screen appears which is used to specify the CA processing selections. Choose a selection from this screen by placing an “X” in the appropriate screen display field(s).

The following example shows the DLG function selection screen:

```
====================== DLG FUNCTION SELECTION ======================
A-UTILITY  OR B-BASIC  OR C-EXTENDED   OR D-CALENDAR -360
  CURRENT....     BETWEEN..X   NEXT DAY..  365->360..
  VALID DATE.     +/- DAYS..     PREV DAY..  360->365..
LEAP YEAR..     +/- WEEKS.     SINCE BOM.  BETWEEN...
HOLIDAY....     +/- MONTHS     UNTIL EOM.  +/-DAYS...
PROCESS....     +/- YEARS.     BOM +/- N.  +/-MONTHS.
DAY OF WK..     EOM +/- N.  +/-YEARS..
->ABS DAY..     NTH DOW...
ABS DAY->..
RE-FORMAT..
AGE BETWEEN
COMPARE....
CONCATENATE
TRUNCATE...
OR E-ADVANCED FUNCTIONS (CHOOSE ONE FROM EACH E? OPTION. E1 -> E5)
E1---DATE-ID.. OR SINCE/UNTIL. OR RELATIVE... OR NTH-DOW...
E2---BEGINNING OR ENDING......
E3---PREVIOUS. OR CURRENT..... OR NEXT.......
E4---CALENDAR. OR FISCAL......
E5---YEAR..... OR QTR......... OR MONTH......
F1=HELP      F3-EXIT    F4-BACK-1-LEVEL
```

3. Press Enter to edit the selection.

   If the edits are bad, an error message appears. The user must correct any errors before processing can continue. If the edits are good, the DLG Function Specification screen appears.

4. On the DLG Function Specification screen, choose the options and enter parameters as requested.

   The parameters are specific to the selection made on the DLG Function Selection screen; therefore the contents of the DLG Function Specification screen will vary from date function to date function.

   The following example shows the DLG function specification screen:

```
======================== DLG FUNCTION SPECIFICATION ==================
FUNCTION-IN-PROCESS= P101  DAYS BETWEEN
CHOOSE EITHER --- CALENDAR-DAYS.   OR PROCESSING-DAYS. X
ENTER BUSINESS-PARAMETERS-  ** ENTER INPUT-PARAMETERS-- PRES ENTER OR F5*
FROM-DATE-MASK..= YYMMDD-- =*=> FROM-DATE-X.......=   *
TO-DATE-MASK....= --YYMMDD =*=> TO-DATE-9.........=   *
* *
* *
*******************************************
END-POINTS-DEF........= T  (B=BOTH, N=NEITHER, F=FROM ONLY, T=TO ONLY)
1234567
HOLIDAY-TABLE.........= 01  (ALPHA-NUMERIC 00...THRU... ZZ)
PROCESSING-DAY-DEF....= NEEEEEN  (A=ALWAYS, N=NEVER, E=EXCLUDE, I=INCLUDE)
CENTURY-BREAK.........= 19/20;68 (E.G. 19/20;68 IF YY >= 68 THEN 19 ELSE 20)
F1=HELP   F4=BACK-1-LEVEL   F5=ENTER-DATA  F6=GENERATE
```

5. Press the Enter key to show additional parameters that are dependent on some of the entered data fields. Any dependent fields are shown below the dashed line on the lower part of the screen.

6. Enter or change the data in these fields as required based on the type of date calculation being performed.
7. Either press PF5 to test the CA Calendar processing function with sample data, (see Step 5. Test with Some Sample Data).

   Or

   Press PF6 to generate the code into the program source program (.

Step 5. Test with Some Sample Data

1. If you press PF5 to test the date logic with live sample data, the right portion of the DLG Function Specification screen turns into a dialog, allowing you to enter test data values.

   The following example shows DLG function testing with key-entered data.

```
========================= DLG FUNCTION SPECIFICATION ====================
FUNCTION-IN-PROCESS= P101  DAYS BETWEEN
Enter test data values in this portion of the screen
CHOOSE EITHER --- CALENDAR-DAYS.   OR PROCESSING-DAYS. X
ENTER BUSINESS-PARAMETERS-
FROM-DATE-MASK..= YYMMDD—
TO-DATE-MASK....= --YYMMDD **
ENTER INPUT-PARAMETERS-- PRESS ENTER OR F5* =*=>
FROM-DATE-X.......= * =*=> TO-DATE-9.........= *
* *
* *
*******************************************
END-POINTS-DEF........= T  (B=BOTH, N=NEITHER, F=FROM ONLY, T=TO ONLY)
-------------------------------------------------------------------------------
HOLIDAY-TABLE.........= 01 (ALPHA-NUMERIC 00...THRU... ZZ)
PROCESSING-DAY-DEF....= NEEEEEN  (A-ALWAYS, N-NEVER, E-EXCLUDE, I=INCLUDE)
CENTURY-BREAK.........= 19/20;68 (E.G. 19/20;68 IF YY >= 68 THEN 19 ELSE 20)
F1=HELP  F4=BACK-1-LEVEL   F5=ENTER-DATA  F6=GENERATE
```

   The fields and parameters that you are required to fill in depend upon the type of date calculation you have selected. Edits will continually be performed on the business parameters and data values that you key enter.
2. When you press PF5 or press Enter, the middle right portion of the screen displays the results of calling the CA Calendar Routines. Specifically, the RETURN-CODE, OUTPUT-DATEs and other VALUES are displayed for review.
3. Review these results. If you change the Masks, Dates, Parameters or other Values and press Enter, new results are displayed.
4. If these new results conform to your expectations, press PF6 to generate code which is inserted into your source program.

Step 6—Generate the Code into the Application Program

For Cobol programs, press PF6 to generate code which will PERFORM the TRC-CALENDAR-ROUTINES execution “shell.” All DLG functions for release 5.0 and above will PERFORM a “shell” paragraph which DGL inserts at the bottom of the application COBOL program.

The “shell” paragraph will determine (based on function and installation default CA Calendar Routines processing settings) whether a “CALL” to CA Calendar Routines will be invoked, or a PERFORM to “in-line” CA COBOL procedure statements will be made. The “in-line” CA COBOL procedure statements are included in the application source program by insertion of COPY (member) statements (see the appendix “COBOL CA Invocation Architecture”).

For Assembler or PL/1 programs, press PF6 to generate code which will CALL the CA Calendar Routines.

The following example shows the PL/1 application invoking a CA function.

```
File Edit Confirm Menu Utilities Compilers Test Help
EDIT   PDGBO.PDS.PL1(BASED) - 01.26   Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
If CURR_DATE= MMDDYY THEN
/*-------------------------------------------------------------------------*/
/* TransCentury(S) PL-1 (P101) DAYS BETWEEN */
/*-------------------------------------------------------------------------*/
TRC_CONVR_CONVERSATIONAL = ‘’;
TRC_CONVR_FUNCTION_CODE = ‘P101’;
TRC_CONVR_HOLIDAY_TBL = ‘TCDSBHO’;
TRC_CONVR_END_PNTS_DEF = ‘TCDSBEN’;
TRC_CONVR_PROC_DAY_DEF = ‘TCDSBPD’;
TRC_CONVR_CENTURY_BREAK = ‘TCDSBCB’;
TRC_CONVR_FROM_DATE_MASK = ‘TCDSIFRM’;
TRC_CONVR_FROM_DATE_X = ?_USER_FROM_DATE;
TRC_CONVR_TO_DATE_MASK = ‘TCDSITOM’;TRC_CONVR_TO_DATE_X = ?_USER_TO_DATE;
CALL TRCENGN (TRC_CONVR_CONVERSATIONAL) ;
IF TRC_CONVR_RETURN_GOOD = 0 THEN
DO;
?-USER-OUT-NUMERIC = TRC_CONVR_OUT_NUMERIC_PARM;
END;
ELSE
DO;
CALL ?_USER_ERROR_ROUTINE;
END;
/*-----------------------------------------------------------------*/
/* END OF GENERATED TransCentury(E) CODE  */
/*-----------------------------------------------------------------*/
TABLE_POINTER = ADDR (TABLE1);
IF TITLE3 > MMDDYY THEN
TABLE_POINTER = ADDR (TABLE1);
IF TITLE4   = JUL_DT THEN
  DO;
TABLE_POINTER = ADDR (TABLE1);
GET LIST (TABLE2);
SUM2 = SUM (TABLE2);
PUT DATA (SUM2);
END;
```

Resetting the DLG Program Profile Definition

The user can reset the Program Profile Definition values while editing an application program by pressing the ISPF Edit Function key assigned to TCDSRSET (F15).

Pressing the TCDSRSET suspends the normal ISPF Edit process and invokes DLG Program Profile Definition processing. (See Step 2. Invoking the CA Date Logic Generator for instructions).

The following example shows the DLG program profile definition display of an existing application DLG profile.

```
---------------------- DLG Program Profile Definition -------------------------
OPTION ==>
Calendar Routine   Customer/Version ABABABABAB 5.00
TransCentury Date Logic Generator Customer/Version BETA DLG  5.00
SATURDAY, APRIL 18, 1998  19:17:52
LangType.... 1 1)COB 2)COB/CICS 3)PL1 4)PL1/CICS 5)ASM 6)ASM/CICS
TransCentury Engine Name  TRCENGIN
Copy Book Member Name...  TRCCONVR
Copy Inclusion Statement COPY
Copylib File Name(s)..   ADP.PLANTINUM.CLIENT.NINEWEST.COPYLIB
PDGBO.WRK5000.COPYLIB
PDGBO.WRK5000.SAMPLE.COPYLIB
Date Element File Name .  PDGBO.WRK5000.DATA.ELEMENT(INDL)
Press ENTER to continue, PFK3 to EXIT, PFK7 to Add COPYLIB(s)
Program Type set to Batch COBOL
```

Optional COBOL TCDS PF Key Functions

The following example is a screen display of the ISPF Edit function key settings for a COBOL application program processed using the CA Date Logic Generator.

```
   Keylist Utility
File
Private   ISR Keylist ISRSPEC Change Row 13 to 24 of 24
Command ===> Scroll ===> PAGE
 Make changes and then select File action bar.
 Keylist Help Panel Name . . . ISRSPECH
 Key    Definition    Format Label
F13 . . TCDSDATE  SHORT TCDSDATE
F14 . . TCDSDATA  SHORT TCDSDATA
F15 . . TCDSRSET  SHORT TCDSDATA
F16 . . TCDSRKEY  SHORT TCDSRKEY
F17 . . TCDSSRCH  SHORT TCDSSRCH
F18 . . TCDSSPF1  SHORT TCDSSPF1
F19 . . UP  LONG Up
F20 . . DOWN    LONG Down
F21 . . SWAP    LONG Swap
F22 . . LEFT    LONG Left
F23 . . RIGHT    LONG Right
F24 . . CANCEL   LONG Cancel
```

Function keys assigned to TCDSSPF1 and TCDSRSET have been previously described in this guide (see Step 2. Invoking the CA Date Logic Generator and Resetting the DLG Program Profile Definition). The remainder of this section describes the use of Edit Function keys assigned to TCDSSRCH, TCDSDATE, TCDSDATA, and TCDSRKEY.

TCDSDATE—Assign CA Date Element Values

The CA Calendar Routines perform functions which can process one to five unique input/output date fields:

- FROM DATE—An application program date element name or variable value for DLG COBOL program statement generation (reference to the input from date).
- TO DATE—An application program date element name or variable value for DLG COBOL program statement generation (reference to the input to date).
- OUT1/2/3 DATE—An application program date element name(s) for DLG COBOL program statement generation (reference to OUT1/2/3 dates).

The following example shows sample application program file definitions:

```
File Edit Confirm Menu Utilities Compilers Test Help
EDIT   PDGBO.PDS.COBOL(T) 0- 01.99    Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
FILE SECTION.
FD IN-FILE.
01 IN-REC.
05 USER-FROM-DATE.
10 USER-FROM-DATE-9 PIC 9(06).
05 USER-TO-DATE
10 USER-TO-DATE-9 PIC 9(06).
05 USER-STDOUT PIC X(08).
05 USER-OUT1  PIC X(08).
05 USER-OUT2  PIC X(10).
05 USER-OUT3  PIC X(32).
05 USER-DOW   PIC 9(01).
05 USER-HOLIDAY PIC 9(01).
05 USER-DAY-COUNT   PIC 9(06).
05 USER-DAY-NAME PIC X(25).  WORKING-STORAGE SECTION.
01 PASS-AREA.
05 PASS-FROM-DATE
10 PASS-FROM-DATE-9 PIC 9(06).
```

The TCDSDATE function provides a method of “point and assign” of application program data elements or literals (input only) to COBOL application program data elements used by the CA Calendar Routines.

The following example shows the code that will be replaced by invoking the CA Calendar Routines “DAYS BETWEEN” date function. The user places the cursor under the data element name USER-FROM-DATE-9 and presses the PF key assigned to TCDSDATE (F13).

```
File Edit Confirm Menu Utilities Compilers Test Help
EDIT   PDGBO.PDS.COBOL(T) 0- 01.99    Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
             MOVE ❶USER-FROM-DATE-9   TO
                  PASS-FROM-DATE-9.
             MOVE  USER-TO-DATE-9    TO
                  PASS-TO-DATE-9.
             CALL ‘XYZ’ USING PASS-AREA.
             IF INLINE-GOOD-RETURN
```

❶ Edit cursor

The TCDSDATE function scans the application program source for the definition of USER-FROM-DATE-9 (see the sample application program file definitions example). The TCDSDATE function uses the Working Storage definition from the application program to assign the CA Calendar Routines date mask (--YYMMDD) to this program data element.

The screen is redisplayed with a pop-up window appearing at the bottom of the screen (see the following example). The CA date appears below the data element name (USER-FROM-DATE-9).

```
File Edit Confirm Menu Utilities Compilers Test Help
EDIT  PDGBO.PDS.COBOL(T) 0- 01.99 Columns 00001 00072
Command ===> TCDSDATE FROM    Scroll ===> CSR
/
PROCEDURE DIVISION.
  MOVE USER-FROM-DATE-9 TO
       PASS-FROM-DATE-9.
  MOVE USER-TO-DATE-  9 TO
       PASS-TO-DATE-9.
OPTION ==>
                            SET Input/Output date field or value
   Data Element: USER-FROM-DATE-9
TCDS Date Mask: --YYMMDD <=== Enter MASK value
  05 USER-FROM-DATE.
     10 USER-FROM-DATE-9 PIC 9(06).
  05 USER-TO-DATE.
     10 USER-TO-DATE-  9 PIC 9(06).
  05 USER-STDOUT         PIC X(08).
  05 USER-OUT1           PIC X(08).
PF1=From PF2=To PF3=EXIT PF4=Out-1 PF5=Out-2 PF6=Out-3 PF7=Std
```

The date mask, automatically assigned by the TCDSDATE function, can be changed by the user. The key-entered date mask is matched to a table of valid date masks. Incorrect date masks must be corrected by the user before processing can continue.

The following is an example of a pop-up display showing a data element assigned to “FROM” date and CA Calendar Routines incorrect date mask entered by the DLG user.

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
EDIT  PDGBO.PDS.COBOL(T) 0- 01.99                        Columns 00001 00072
Command ===> TCDSDATE FROM                                  Scroll ===> CSR
       /
       PROCEDURE DIVISION.
           MOVE USER-FROM-DATE-9 TO
                PASS-FROM-DATE-9.
           MOVE USER-TO-DATE-9   TO
                PASS-TO-DATE-9.
OPTION ==>
                                    SET Input/Output date field or value
   Data Element: USER-FROM-DATE-9
TCDS Date Mask: --ABCDEF INVALID Date Mask
  05 USER-FROM-DATE.
     10 USER-FROM-DATE-9 PIC 9(06).
  05 USER-TO-DATE.
     10 USER-TO-DATE-9   PIC 9(06).
  05 USER-STDOUT         PIC X(08).
  05 USER-OUT1           PIC X(08).
PF1=From PF2=To PF3=EXIT PF4=Out-1 PF5=Out-2 PF6=Out-3 PF7=Std
```

The following example shows an ISPF EDIT display of sample procedure source code. The highlighted data element name and date mask information displayed in the top right corner (FROM date:--YYMMDD) is saved when the user presses the (PF1=From) PFkey.

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
EDIT   PDGBO.PDS.COBOL(T) 0- 01.99                        FROM date:--YYMMDD
Command ===> ____________________________________________ Scroll ===> CSR
             MOVE USER-FROM-DATE-9   TO
                  PASS-FROM-DATE-9.
             MOVE USER-TO-DATE-9     TO
                  PASS-TO-DATE-9.
           CALL ‘ABC’ USING PASS-AREA.
           IF INLINE-GOOD-RETURN
```

In the following example, the DLG user placed the cursor under a data element and pressed the PFkey assigned to TCDSDATE (F13). For this example, the data element definition was removed from the application program. TCDSDATE scanned the application program source for a matching data element definition. No match was found, DLG processing was suspended, and the highlighted error message appeared on the ISPF Edit Display screen.

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
EDIT   PDGBO.PDS.COBOL(T) 0- 01.99                  Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
## CURSOR is positioned on a Undefined Data Element ##
               ❶ PASS-TO-DATE-9.
             CALL ‘XYZ’ USING PASS-AREA.
             IF INLINE-GOOD-RETURN
```

❶ Edit cursor

The DLG user corrected the problem by placing the cursor under a different data element and pressing the (PF2=To) PFkey.

The next screen to appear is a pop-up window displaying data element CA Calendar Routines information.

```
File Edit Confirm Menu Utilities Compilers Test Help
------------------------------------------------------------------------------
EDIT PDGBO.PDS.COBOL(T) 0- 01.99                          Columns 00001 00072
Command ===> TCDSDATE TO                                      Scroll ===> CSR
/
PROCEDURE DIVISION.
   MOVE USER-FROM-DATE-9 TO
        PASS-FROM-DATE-9.
   MOVE USER-TO-DATE-9 TO
        PASS-TO-DATE-9.
OPTION ==>
SET Input/Output date field or value
Data Element: USER-TO-DATE-9
TCDS Date Mask: --YYMMDD <=== Enter MASK value
   05 USER-FROM-DATE.
      10 USER-FROM-DATE-9 PIC 9(06).
   05 USER-TO-DATE.
      10 USER-TO-DATE-9 PIC 9(06).
   05 USER-STDOUT PIC X(08).
   05 USER-OUT1 PIC X(08).
PF1=From PF2=To PF3=EXIT PF4=Out-1 PF5=Out-2 PF6=Out-3 PF7=Std
```

After pressing Enter, the ISPF EDIT screen is redisplayed with the following message appearing in the top right portion of the screen display:

```
TO date: --YYMMDD
File Edit Confirm Menu Utilities Compilers Test Help
EDIT   PDGBO.PDS.COBOL(T) 0- 01.99     TO date: --YYMMDD
Command ===> ____________________________________________________ Scroll ===> CSR
   MOVE USER-TO-DATE-9 TO
        PASS-TO-DATE-9. CALL 'XYZ' USING PASS-AREA.
IF INLINE-GOOD-RETURN
```

The next series of examples involve assignment of data elements to CA Calendar Routines output data fields. The example “ABC” date function inputs a “FROM” date and outputs 1 to 3 output dates in different user specified date formats.

After assigning the “FROM” and “TO” date fields, the next step for the user is to assign data element names and CA Calendar Routines date masks to output dates 1, 2, and 3. The user places the cursor under each application date element name (USER-OUT1, USER-OUT2, USER-OUT3), and presses the TCDSDATE (F13) PFkey.

```
File Edit Confirm Menu Utilities Compilers Test Help
EDIT   PDGBO.PDS.COBOL(T) 0- 01.99  OUT1 date:--YYMMDD
Command ===> ____________________________________________________ Scroll ===> CSR
MOVE USER-FROM-DATE-9   TO
PASS-FROM-DATE-9.
MOVE USER-TO-DATE-9    TO
 PASS-TO-DATE-9.
  CALL 'ABC' USING PASS-AREA.
  IF INLINE-GOOD-RETURN
    MOVE PASS-OUT1 TO USER-OUT1
    MOVE PASS-OUT2 TO USER-OUT2
    MOVE PASS-OUT3 TO  USER-OUT3.
```

In the following example, the user pressed the TCDSDATE PFkey (F13), and the TCDSDATE function scanned the application program source, located the data element name and resolved the date element definition to the CA date mask (CCYYMMDD).

```
File Edit Confirm Menu Utilities Compilers Test Help
EDIT   PDGBO.PDS.COBOL(T) v- 01.99    Columns 000001 000072
Command ===> _____________________________________________ Scroll ===> CSR
IF INLINE-GOOD-RETURN
   MOVE PASS-OUT1 TO  USER-OUT1
   MOVE PASS-OUT2 TO  USER-OUT2
   MOVE PASS-OUT3 TO  USER-OUT3.
****************************** Bottom of Data ******************************
 OPTION ==>
                                        SET Input/Output date field or value
   Data Element: USER-OUT1
TCDS Date Mask: CCYYMMDD <=== Enter MASK value
  05 USER-STDOUT  PIC X(08).
  05 USER-OUT1 PIC X(08).
  05 USER-OUT2 PIC X(10).
  05 USER-OUT3 PIC X(32).
  05 USER-DOW PIC 9(01).
  05 USER-HOLIDAY  PIC 9(01).
PF1=From PF2=To PF3=EXIT PF4=Out-1 PF5=Out-2 PF6=Out-3 PF7=Std
```

Both data element name and date mask are saved by DLG as the CA Output Date-1 definition.

The user next attempts to assign the Output-2 date field by placing the cursor under the USER-OUT2 data element and pressing the TCDSDATE PF key (F13).

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
EDIT     PDGBO .PDS.COBOL(T) 0- 01.99                       Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
             MOVE USER-FROM-DATE-9   TO
                  PASS-FROM-DATE-9.
             CALL 'ABC' USING PASS-AREA.
             IF INLINE-GOOD-RETURN
                MOVE PASS-OUT1 TO  USER-OUT1
                MOVE PASS-OUT2 TO  USER-OUT2
                MOVE PASS-OUT3 TO  USER-OUT3.
```

In the following example, the TCDSDATE function scanned the program source and located the data element name. The function could not resolve the data element definition. The screen is redisplayed with a pop-up window appearing in the middle of the screen as shown in the following example.

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
EDIT  PDGBO.PDS.COBOL(T) 0- 01.99                              Columns 0001 0002
Command ===> _____________________________________________________ Scroll ===> CSR
            MOVE USER-FROM-DATE-9  TO
                 PASS-FROM-DATE-9.
            CALL 'ABC' USING PASS-AREA.
            IF INLINE-GOOD-RETURN
               MOVE PASS-OUT1 TO USER-OUT1
               MOVE PASS-OUT2 TO USER-OUT2
               MOVE PASS-OUT3 TO USER-OUT3.
   OPTION ==>
                                          SET Input/Output date field or value
     Data Element: USER-OUT2
TCDS Date Mask:  <=== Enter MASK value
  05 USER-OUT1         PIC X(08).
  05 USER-OUT2         PIC X(10).
  05 USER-OUT3         PIC X(32).
  05 USER-DOW          PIC 9(01).
  05 USER-HOLIDAY      PIC 9(01).
  05 USER-DAY-COUNT    PIC 9(06).
PF1=From PF2=To PF3=EXIT PF4=Out-1 PF5=Out-2 PF6=Out-3 PF7=Std
```

The following example shows a pop-up display where a user has supplied a date mask definition for USER-OUT2.

```
File    Edit    Confirm     Menu     Utilities     Compilers     Test     Help
EDIT  PDGBO.PDS.COBOL(T) - 01.99                            Columns 0001 0072
Command ===> TCDSDATE OUT2                                    Scroll ===> CSR
               IF INLINE-GOOD-RETURN
                  MOVE PASS-OUT1 TO USER-OUT1
                  MOVE PASS-OUT2 TO USER-OUT2
                  MOVE PASS-OUT3 TO USER-OUT3.
****************************** Bottom of Data ******************************
OPTION ==>
SET Input/Output date field or value Data Element: USER-OUT2
TCDS Date Mask: E-M/D/Y <=== Enter MASK value
05 USER-OUT1    PIC X(08).
05 USER-OUT2   PIC X(10).
05 USER-OUT3    PIC X(32).
05 USER-DOW    PIC 9(01).
05 USER-HOLIDAY     PIC 9(01).
05 USER-DAY-COUNT    PIC 9(06).
PF1=From PF2=To PF3=EXIT PF4=Out-1 PF5=Out-2 PF6=Out-3 PF7=Std
```

The DLG user key enters e-m/d/y in the date mask area of the pop-up window and presses the (PF5=Out-2) PFkey. The ISPF edit screen reappears with the following message displayed on the top right corner of the window:

```
OUT2 date: E-M/D/Y
```

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
EDIT   PDGBO .PDS.COBOL(T) 0- 01.99                          OUT2 date:E-M/D/Y
Command ===> ____________________________________________ Scroll ===> CSR
             MOVE USER-FROM-DATE-9   TO
                  PASS-FROM-DATE-9.
             CALL 'ABC' USING PASS-AREA.
             IF INLINE-GOOD-RETURN
                MOVE PASS-OUT1 TO  USER-OUT1
                MOVE PASS-OUT2 TO USER-OUT2
                MOVE PASS-OUT3 TO  USER-OUT3.
```

The DLG user moves the cursor under the USER-OUT3 data element and presses the TCDSDATE PFkey (F13). A similar process is required to assign the data element name and date mask for the OUT-3 data element.

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
EDIT  PDGBO.PDS.COBOL(T) 0- 01.99                    Columns 00001 00072
Command ===> TCDSDATE OUT3                              Scroll ===> CSR
            MOVE USER-FROM-DATE-9  TO
                 PASS-FROM-DATE-9.
            CALL 'ABC' USING PASS-AREA.
            IF INLINE-GOOD-RETURN
                 MOVE PASS-OUT1 TO USER-OUT1
                 MOVE PASS-OUT2 TO USER-OUT2
                 MOVE PASS-OUT3 TO USER-OUT3.
   OPTION ==>
                                      SET Input/Output date field or value
      Data Element: USER-OUT3
TCDS Date Mask:     <=== Enter MASK value
  05 USER-OUT1         PIC X(08).
  05 USER-OUT2         PIC X(10).
  05 USER-OUT3         PIC X(32).
  05 USER-DOW          PIC 9(01).
  05 USER-HOLIDAY      PIC 9(01).
  05 USER-DAY-COUNT    PIC 9(06).
PF1=From PF2=To PF3=EXIT PF4=Out-1 PF5=Out-2 PF6=Out-3 PF7=Std
```

The following example shows a pop-up display requesting the user to supply date mask definition for USER-OUT3.

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
EDIT  PDGBO.PDS.COBOL(T) 0- 01.99                    Columns 00001 00072
Command ===> TCDSDATE OUT3                              Scroll ===> CSR
             IF INLINE-GOOD-RETURN
                MOVE PASS-OUT1 TO USER-OUT1
                MOVE PASS-OUT2 TO USER-OUT2
                MOVE PASS-OUT3 TO USER-OUT3.
OPTION ==>
                                      SET Input/Output date field or value
     Data Element: USER-OUT3
TCDS Date Mask: e-tds-45 <=== Enter MASK value
  05 USER-OUT1         PIC X(08).
  05 USER-OUT2         PIC X(10).
  05 USER-OUT3         PIC X(32).
  05 USER-DOW          PIC 9(01).
  05 USER-HOLIDAY      PIC 9(01).
  05 USER-DAY-COUNT    PIC 9(06).
   PF1=From PF2=To PF3=EXIT PF4=Out-1 PF5=Out-2 PF6=Out-3 PF7=Std
```

The following example shows an application program source redisplayed with data element name and date mask assigned by TCDSDATE (PF6=Out-3) PFkey.

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
EDIT   PDGBO.PDS.COBOL(T) 0- 01.99                          OUT3 date: E-TDS-45
Command ===> ____________________________________________ Scroll ===> CSR
             MOVE USER-FROM-DATE-9
                  TO  PASS-FROM-DATE-9.
             CALL 'ABC' USING PASS-AREA.
             IF INLINE-GOOD-RETURN
                MOVE PASS-OUT1 TO  USER-OUT1
                MOVE PASS-OUT2 TO USER-OUT2
                MOVE PASS-OUT3 TO USER-OUT3.
```

After positioning the cursor in the application program source, the user presses the PFkey assigned to TCDSSPF1 (F18). Pressing the TCDSSPF1 key initiates the Date Logic Function Selection process.

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
EDIT   PDGBO.PDS.COBOL(T) 0- 01.99                          Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
             MOVE USER-FROM-DATE-9  TO
                  PASS-FROM-DATE-9.
             CALL 'ABC' USING PASS-AREA.
             IF INLINE-GOOD-RETURN
                MOVE PASS-OUT1 TO  USER-OUT1
                MOVE PASS-OUT2 TO  USER-OUT2
                MOVE PASS-OUT3 TO  USER-OUT3.
```

The following example shows the DLG Selection Screen where the DLG user has selected the RE-FORMAT function:

```
=================== DLG FUNCTION SELECTION ====================
A-UTILITY OR B-BASIC OR C-EXTENDED OR D-CALENDAR-360
CURRENT....     BETWEEN... NEXT DAY. . 365->360..
VALID DATE.     +/- DAYS.. PREV DAY.. 360->365..
LEAP YEAR..     +/- WEEKS. SINCE BOM. BETWEEN...
HOLIDAY....     +/- MONTHS UNTIL EOM. +/-DAYS...
PROCESS....     +/- YEARS. BOM +/- N. +/-MONTHS.
DAY OF WK..                EOM +/- N. +/-YEARS..
->ABS DAY..                NTH DOW...
ABS DAY->..
RE-FORMAT.. X
AGE BETWEEN
COMPARE....
CONCATENATE
TRUNCATE...
OR E-ADVANCED FUNCTIONS (CHOOSE ONE FROM EACH E? OPTION. E1 -> E5)
E1---DATE-ID.. OR SINCE/UNTIL. OR RELATIVE... OR NTH-DOW...
E2---BEGINNING OR ENDING......
E3---PREVIOUS. OR CURRENT..... OR NEXT.......
E4---CALENDAR. OR FISCAL......
E5---YEAR..... OR QTR......... OR MONTH......
F1=HELP F3=EXIT F4=BACK-1-LEVEL
```

The user selects the function which he/she wants to perform by placing an X in the appropriate selection screen area. The next screen to appear is the DLG Function Specification screen. The FROM and OUTPUT 1-3 date masks have been previously assigned by the TCDSDATE DLG function.

```
================}DLG FUNCTION SPECIFICATION ================
FUNCTION-IN-PROCESS= U009        REFORMAT DATE
 ENTER BUSINESS-PARAMETERS-  ** ENTER INPUT-PARAMETERS-- PRESS ENTER OR F5*
 FROM-DATE-MASK..= --YYMMDD = *=> FROM-DATE-9.......= 00001231            *
                              *                                           *
                              *                                           *
                              *                                           *
                              *** RETURN-CODE = 000 **********************
                              *  STDOUT-DATE-X..... = 20001231            *
OUT1-DATE-MASK.= CCYYMMDD =   *=> OUT1-DATE-X.......= 20001231            *
OUT2-DATE-MASK.= E-M/D/Y =    *=> OUT2-DATE-E.......= 12/31/00            *
OUT3-DATE-MASK.= E-TDS-45 =   *=> OUT3-DATE-E.......= SUNDAY, DECEMBER 31,*
                              ******************************************
CENTURY-BREAK.........= SLIDE-20 ( WINDOW RANGE FROM => 1978 TO => 2077)
F1=HELP   F4=BACK-1-LEVEL   F5=ENTER-DATA  F6=GENERATEGOOD RETURN
```

The user can press F5=ENTER-DATA to test various data values for this CA Function (U009-REFORMAT DATE).

After testing (which may be optional), the user can request DLG to generate the appropriate source statements which will instruct the application program to invoke the CA Calendar Routines (DLG user press PF6 key).

```
File    Edit     Confirm    Menu    Utilities    Compilers    Test    Help
EDIT  PDGBO.PDS.COBOL(T) 0- 01.99                          Columns 00001 00072
Command ===> ____________________________________________ Scroll ===> CSR
             * MOVE USER-FROM-DATE-9  TO
             * PASS-FROM-DATE-9.
             * CALL 'ABC' USING PASS-AREA.
             * IF INLINE-GOOD-RETURN
             * MOVE PASS-OUT1 TO USER-OUT1.
             * MOVE PASS-OUT2 TO USER-OUT2.
             * MOVE PASS-OUT3 TO USER-OUT3.
             *-----------------------------------------------*
             * TransCentury(S) COBOL (U009) REFORMAT DATE
             * D.L.G. TESTING COMPLETED 1998/04/19 17:43:35
             *-----------------------------------------------*
                  MOVE 'U009'   TO TRC-CONVR-FUNCTION-CODE
                  MOVE 'SLIDE-20' TO TRC-CONVR-CENTURY-BREAK
                  MOVE 'CCYYMMDD' TO TRC-CONVR-OUT1-DATE-MASK
                  MOVE 'E-M/D/T' TO TRC-CONVR-OUT2-DATE-MASK
                  MOVE 'E-TDS-45' TO TRC-CONVR-OUT3-DATE-MASK
                  MOVE 'Y' TO TRC-CONVR-VALIDATE-NULL
                  MOVE '--YYMMDD' TO TRC-CONVR-FROM-DATE-MASK
                  MOVE USER-FROM-DATE-9
                                TO TRC-CONVR-FROM-DATE-9
                  PERFORM TRC-CALENDAR-ROUTINES
                  IF TRC-CONVR-RETURN-GOOD
                     MOVE TRC-CONVR-OUT1-DATE-X
                          TO USER-OUT1
                     MOVE TRC-CONVR-OUT2-DATE-E
                          TO USER-OUT2
                     MOVE TRC-CONVR-OUT3-DATE-E
                          TO USER-OUT3
                  ELSE
                     PERFORM TRC-ERROR-ROUTINE
                  END-IF
             *
             *-----------------------------------------------*
             * END OF GENERATED TransCentury (E) CODE *
             *-----------------------------------------------*
```

Literals as Date Fields

Both numeric and alpha-numeric literals can be used as date fields. The following examples show the process used by DLG to assign a numeric literal as a date value.

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
--------------------------------------------------------------------------------
EDIT PDGBO.PDS.COBOL(T) 0- 01.99                                TO date: CCYYMMDD
Command ===> _____________________________________________ Scroll ===> CSR
             MOVE USER-FROM-DATE-9 TO
                  PASS-FROM-DATE-9.
             MOVE 20000101 TO
                  PASS-TO-DATE-9.
             CALL 'XYZ' USING PASS-AREA.
             IF INLINE-GOOD-RETURN
                MOVE PASS-NUM TO USER-DAY-COUNT
```

After placing the cursor under the numeric literal, the user pressed the TCDSDATE PF key (F13) and the following pop-up window appears.

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
EDIT  PDGBO.PDS.COBOL(T) - 01.99  Columns 00001 00072
Command ===>   Scroll ===> CSR/ PROCEDURE DIVISION.
   MOVE USER-FROM-DATE-9  TO   PASS-FROM-DATE-9.
MOVE 20000101  TO
   PASS-TO-DATE-9.
   OPTION ==>
 Literal: 20000101
Date Mask: CCYYMMDD   SET Input date literals    PF1=From PF2=To PF3=EXIT
```

After pressing the Enter key, the ISPF EDIT screen appears. The user places the cursor under the USER-DAY-COUNT field and presses the PF key assigned to TCDSDATA (F14, see TCDSDATA—Assign CA Data Element Values).

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
EDIT  PDGBO.PDS.COBOL(T) - 01.99                              O/P Numparm
Command ===> TCDSDATE TO                                 Scroll ===> CSR
      /
      PROCEDURE DIVISION.
           MOVE USER-FROM-DATE-9  TO
                PASS-FROM-DATE-9.
           MOVE 20000101  TO
                PASS-TO-DATE-9.
           CALL 'XYZ' USING PASS-AREA.
           IF INLINE-GOOD-RETURN
              MOVE PASS NUM TO USER-DAY-COUNT
```

After the user completes TCDSDATE/TCDSDATA data element PF key processing, the user invokes the CA Date Logic Generator by pressing the PF key assigned to TCDSSPF1 (F16). The next series of screens show the processing required for generating COBOL source statements required to invoke the “DAYS BETWEEN” function.

```
==================== DLG FUNCTION SELECTION ====================
A-UTILITY  OR B-BASIC  OR     C-EXTENDED OR D-CALENDAR-360
  CURRENT....    BETWEEN... X  NEXT DAY..    365->360..
  VALID DATE.    +/- DAYS..    PREV DAY..    360->365..
  LEAP YEAR..    +/- WEEKS.    SINCE BOM.    BETWEEN...
  HOLIDAY....    +/- MONTHS    UNTIL EOM.    +/-DAYS...
  PROCESS....    +/- YEARS.    BOM +/- N.    +/-MONTHS.
  DAY OF WK..                  EOM +/- N.    +/-YEARS..
  ->ABS DAY..                  NTH DOW...
  ABS DAY->..
  RE-FORMAT..
  AGE BETWEEN
  COMPARE....
  CONCATENATE
  TRUNCATE...
OR E-ADVANCED FUNCTIONS (CHOOSE ONE FROM EACH E? OPTION. E1 -> E5)
E1---DATE-ID.. OR SINCE/UNTIL. OR RELATIVE...  OR NTH-DOW...
E2---BEGINNING OR ENDING......
E3---PREVIOUS. OR CURRENT..... OR NEXT.......
E4---CALENDAR. OR FISCAL......
E5---YEAR..... OR QTR......... OR MONTH......
F1=HELP F3=EXIT      F4=BACK-1-LEVEL
```

The following example shows the DLG Function Specification screen with input “FROM” and “TO” date masks assigned using TCDSDATE PF key information.

```
============= DLG FUNCTION SPECIFICATION ===========
FUNCTION-IN-PROCESS= P101 DAYS BETWEEN
 CHOOSE EITHER --- CALENDAR-DAYS. OR PROCESSING-DAYS. X
 ENTER BUSINESS-PARAMETERS-  ** ENTER INPUT-PARAMETERS-- PRESS ENTER OR F5*
  FROM-DATE-MASK..= --YYMMDD =* => FROM-DATE-9.......=                    *
  TO-DATE-MASK....= CCYYMMDD =* => TO-DATE-X.........=                    *
                             *                                            *
                             *                                            *
                             **********************************************
 END-POINTS-DEF........= T (B=BOTH, N=NEITHER, F=FROM ONLY, T=TO ONLY)
     - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
    HOLIDAY-TABLE.........= AA (ALPHA-NUMERIC 00...THRU... ZZ)
    PROCESSING-DAY-DEF....= NEEEEEN (A=ALWAYS, N=NEVER, E=EXCLUDE, I=INCLUDE)
CENTURY-BREAK.........= SLIDE-20 ( WINDOW RANGE FROM => 1978 TO => 2077)
 F1=HELP   F4=BACK-1-LEVEL  F5=ENTER-DATA  F6=GENERATE
```

The following example shows a program source modified to include statements which will set CA Calendar Routines CALL/PASS area fields, invoke the Calendar Routines, and process information after the invocation:

```
File Edit Confirm Menu Utilities Compilers Test Help
EDIT  PDGBO.PDS.COBOL(T) - 01.99 Columns 00001 00072Command ===>  Scroll ===>
CSR/ PROCEDURE DIVISION.
   MOVE USER-FROM-DATE-9 TO  PASS-FROM-DATE-9.   MOVE 20000101 TO
  PASS-TO-DATE-9.
   CALL 'XYZ' USING PASS-AREA.*-------------------------------------------------
-----------------** TransCentury(S) COBOL (C101) DAYS BETWEEN ** D.L.G. TESTING
COMPLETED 1998/05/05 17:47:30  **-----------------------------------------------
-------------------*
   MOVE 'C101'     TO TRC-CONVR-FUNCTION-CODE
   MOVE 'T' TO TRC-CONVR-END-PNTS-DEF
   MOVE 'SLIDE-20' TO TRC-CONVR-CENTURY-BREAK
   MOVE 'Y' TO TRC-CONVR-VALIDATE-NULL
   MOVE '--YYMMDD' TO TRC-CONVR-FROM-DATE-MASK
   MOVE USER-FROM-DATE-9
                   TO TRC-CONVR-FROM-DATE-9
   MOVE 'CCYYMMDD' TO TRC-CONVR-TO-DATE-MASK
   MOVE 20000101
                   TO TRC-CONVR-TO-DATE-X
   PERFORM TRC-CALENDAR-ROUTINES
   IF TRC-CONVR-RETURN-GOOD
      MOVE TRC-CONVR-OUT-NUMERIC-PARM
           TO USER-DAY-COUNT
      ELSE
           PERFORM TRC-ERROR-ROUTINE
   END-IF
*                                                                 *
*-----------------------------------------------------------------*
* END OF GENERATED TransCentury(E) CODE                                  *
*-----------------------------------------------------------------*
   IF INLINE-GOOD-RETURN
      MOVE PASS-NUM    TO USER-DAY-COUNT
/
   COPY TRCCOMNP.
/
   COPY TRCSMLPD.
/
   COPY TRCERROR.
/
 TRC-STANDARD-CALL.
   IF TRC-CONVR-RETURN-GOOD  NEXT SENTENCE.
****************************** Bottom of Data ********************************
```

As shown in Optional COBOL TCDS PF Key Functions, input numeric, output numeric, and output alphabetic application program data elements can be assigned by the TCDSDATA function during an ISPF Edit session using DLG. The TCDSDATA function saves the data element name (or literal) for DLG COBOL program statement generation.

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
--------------------------------------------------------------------------------
EDIT PDGBO.PDS.COBOL(T) 0- 01.99                          Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
FILE SECTION.
FD IN-FILE.
01 IN-REC.
   05 USER-FROM-DATE.
      10 USER-FROM-DATE-9 PIC 9(06).
   05 USER-TO-DATE.
      10 USER-TO-DATE-9 PIC 9(06).
   05 USER-STDOUT PIC X(08).
   05 USER-OUT1 PIC X(08).
   05 USER-OUT2 PIC X(08).
   05 USER-OUT3 PIC X(08).
   05 USER-DOW PIC 9(01).
   05 USER-HOLIDAY PIC 9(01).
   05 USER-DAY-COUNT PIC 9(06).
   05 USER-DAY-NAME PIC X(25).
WORKING-STORAGE SECTION.
01 PASS-AREA.
   05 PASS-FROM-DATE.
     10 PASS-FROM-DATE-9 PIC 9(06)
```

In the following example, the DLG user saves a numeric application data element name by placing the cursor under a program data element name and pressing the TCDSDATA PFkey (F14).

```
File    Edit    Confirm    Menu    Utilities    Compilers    Test    Help
EDIT  PDGBO.PDS.COBOL(T) 0- 01.99 Command ===>
_____________________________________________ Scroll ===> CSR
   MOVE USER-FROM-DATE-9  TO   PASS-FROM-DATE-9.   MOVE USER-TO-DATE-9 TO
        PASS-TO-DATE-9.
   CALL 'XYZ' USING PASS-AREA.
   IF INLINE-GOOD-RETURN
      MOVE PASS-NUM  TO USER-DAY-COUNT
```

TCDSDATA displays a pop-up window at the bottom of the screen. The DLG user selects the CA Calendar Routine Link/Pass variable which will be moved-to or moved-from by pressing one of the PFkeys listed at the bottom of the pop-up window (e.g., PF2=Output Numeric).

```
EDIT    PDGBO.PDS.COBOL(T2) - 01.55  Columns 00001 00072
Command ===> TCDSDATA      Scroll ===>
CSR
000069   MOVE USER-FROM-DATE-9 TO
000070   PASS-FROM-DATE-9.  0
00071   MOVE USER-TO-DATE-9 TO
000072   PASS-TO-DATE-9.
000073   CALL 'XYZ' USING INLINE-PASS-AREA.
000074   IF INLINE-GOOD-RETURN
000075
000076    OPTION ==>
000077
000078  Data Element: USER-DAY-COUNT
000079  SET Alpha/Numeric Parameter field or value
000080
```

```
000081 05 USER-HOLIDAY  PIC 9(01).
000082 05 USER-DAY-COUNT PIC 9(06).
000083 05 USER-DAY-NAME     PIC X(25).
000084    01 PASS-AREA.
000085 05 PASS-FROM-DATE.
000086 10 PASS-FROM-DATE-9 PIC 9(06).
000087
PF1=Input Numeric PF2=Output Numeric PF3=EXIT PF5=Output Alpha 000088
```

The following example shows an application program display—cursor is positioned on the source statement line where DLG generated code will be inserted (AFTER):

```
File Edit Confirm Menu Utilities Compilers Test Help
EDIT  PDGBO.PDS.COBOL(T) 0- 01.99 Numeric O/PCommand ===>
_____________________________________________ Scroll ===> CSR
   MOVE USER-FROM-DATE-9  TO   PASS-FROM-DATE-9.   MOVE USER-TO-DATE-9 TO
   PASS-TO-DATE-9.
   CALL 'XYZ' USING PASS-AREA.
   IF INLINE-GOOD-RETURN
 MOVE PASS-NUM  TO USER-DAY-COUNT
```

After positioning the cursor in the application program source, the user presses the PFkey assigned to TCDSSPF1 (F18). Pressing the TCDSSPF1 key initiates the CA Date Logic Generator Function Selection process.

```
=================== DLG FUNCTION SELECTION ===================

 A-UTILITY   OR B-BASIC OR C-EXTENDED
OR D-CALENDAR-360 CURRENT....
BETWEEN... X
NEXT DAY.. 365->360..
VALID DATE. +/- DAYS..
PREV DAY.. 360->365..
LEAP YEAR.. +/-
WEEKS. SINCE BOM.
BETWEEN... HOLIDAY.... +/-
MONTHS UNTIL EOM. +/-DAYS...
PROCESS.... +/- YEARS. BOM +/- N. +/-
MONTHS. DAY OF WK.. EOM +/- N. +/-
YEARS.. ->ABS DAY.. NTH DOW...
ABS DAY->.. RE-FORMAT..
AGE BETWEEN COMPARE....
CONCATENATE TRUNCATE...
OR
E-ADVANCED FUNCTIONS
(CHOOSE ONE FROM EACH E? OPTION. E1 -> E5)
E1---DATE-ID..  OR SINCE/UNTIL.  OR RELATIVE...  OR NTH-DOW...
E2---BEGINNING  OR ENDING......
E3---PREVIOUS.  OR CURRENT.....  OR
NEXT.......
E4---CALENDAR.  OR FISCAL......
E5---YEAR.....  OR QTR.........  OR
MONTH......
F1=HELP   F3=EXIT   F4=BACK-1-LEVEL
```

The user selects the function which he/she wants to perform by placing an X in the appropriate selection screen area.

The next screen to appear is the DLG Function Specification screen. The FROM/TO input date elements and date masks, along with the output numeric data element have been previously assigned by the TCDSDATE and TCDSDATA PFkey DLG functions.

```
================ DLG FUNCTION SPECIFICATION ================
FUNCTION-IN-PROCESS= P101  DAYS BETWEEN
 CHOOSE EITHER --- CALENDAR-DAYS.   OR  PROCESSING-DAYS. X
 ENTER BUSINESS-PARAMETERS-  ** ENTER INPUT-PARAMETERS-- PRESS
ENTER OR F5* FROM-DATE-MASK..
= --YYMMDD =*=>
FROM-DATE-9.......
= 00971231 * TO-DATE-MASK....
= --YYMMDD =*=> TO-DATE-X.........
= 20001231 * * * * *  *** RETURN-CODE = 000
**********************
* *
* *
* *
* *
*** OUTPUT-NUMERIC-PARM..= +00000768 ********
END-POINTS-DEF........
= T    (B=BOTH, N=NEITHER, F=FROM ONLY, T=TO ONLY)
HOLDIAY-TABLE........
.= 98    (ALPHA-NUMERIC 00...THRU... ZZ)
PROCESSING-DAY-DEF....
= NEEEEEN (A=ALWAYS, N=NEVER, E=EXCLUDE, I=INCLUDE)CENTURY-BREAK.........
= SLIDE-20 ( WINDOW RANGE FROM => 1978 TO => 2077)
F1=HELP   F4=BACK-1-LEVEL   F5=ENTER-DATA  F6=GENERATEGOOD RETURN
```

The user can press F5=ENTER-DATA, to test various data values for this CA function (P101-DAYS BETWEEN).

After testing, which may be optional, the user can request DLG to generate appropriate source statements which will instruct the application program to invoke the CA Calendar Routines.

```
File Edit Confirm Menu Utilities Compilers Test Help
EDIT   PDGBO.PDS.COBOL(T) 0- 01.99    Columns 00001 00072
Command ===> _____________________________________________ Scroll ===> CSR
* MOVE USER-FROM-DATE-9 TO
 * PASS-FROM-DATE-9.
* MOVE USER-TO-DATE-9  TO
 * PASS-TO-DATE-9.
* CALL 'XYZ' USING PASS-AREA.
 * IF INLINE-GOOD-RETURN
 *   MOVE PASS-NUM TO  USER-DAY-COUNT.
*-----------------------------------------------------------------*
 * TransCentury(S) COBOL (U009) REFORMAT DATE
 * D.L.G. TESTING COMPLETED 1998/04/19 17:43:35
 *-----------------------------------------------------------------* MOVE
'P101'   TO TRC-CONVR-FUNCTION-CODE
MOVE '98' TO TRC-CONVR-HOLIDAY-TBL
MOVE 'T' TO TRC-CONVR-END-PNTS-DEF
MOVE 'NEEEEEN' TO TRC-CONVR-PROC-DAY-DEF
MOVE 'SLIDE-20' TO TRC-CONVR-CENTURY-BREAK
MOVE 'Y' TO TRC-CONVR-VALIDATE-NULL
MOVE '--YYMMDD' TO TRC-CONVR-FROM-DATE-MASK
MOVE USER-FROM-DATE-9   TO TRC-CONVR-FROM-DATE-9
MOVE 'CCYYMMDD' TO TRC-CONVR-TO-DATE-MASK
MOVE 20000101   TO TRC-CONVR-TO-DATE-X PERFORM
TRC-CALENDAR-ROUTINES IF TRC-CONVR-RETURN-GOOD
Output numeric
MOVE TRC-CONVR-OUT-NUMERIC-PARM
data element
TO USER-DAY-COUNT assignment
```

```
ELSE
PERFORM TRC-ERROR-ROUTINE
 END-IF *
*-------------------------------------------------*
* END OF GENERATED CA (E) CODE   *
*-------------------------------------------------*
/
 COPY TRCCONPD.
/
COPY TRCERROR.Statements
/
  TRC-STANDARD-CALL.

added by DLG
 CALL 'TRCENGIN' USING TRC-CONVR-CONVERSATIONAL
******************************** Bottom of Data ****************************
```

In this example, the user places the cursor under an output alphabetic data element and stores the data element name by pressing the TCDSDATA PF key.

```
File Edit Confirm Menu Utilities Compilers Test Help
EDIT   PDGBO.PDS.COBOL(T) 0- 01.99   Alpha O/P
Command ===> _____________________________________________ Scroll ===> CSR
            MOVE USER-FROM-DATE-9   TO
                 PASS-FROM-DATE-9.
            CALL 'XYZ' USING PASS-AREA.
            IF INLINE-GOOD-RETURN
               MOVE PASS-NUM TO USER-DOW.
               MOVE PASS-NAME TO USER-DAY-NAME.
            Edit Cursor
```

TCDSDATA displayed the following pop-up window and the DLG user pressed PFkey (PF5=Output Alpha).

```
EDIT    PDGBO.PDS.COBOL(T2) - 01.55  Columns
00001
00072 Command ===> TCDSDATA      Scroll ===> CSR
000069   MOVE USER-FROM-DATE-9 TO
000070   PASS-FROM-DATE-9.
000071   MOVE USER-TO-DATE-9 TO
000072   PASS-TO-DATE-9.
000073   CALL 'XYZ' USING INLINE-PASS-AREA.
000074   IF INLINE-GOOD-RETURN
000075
000076    OPTION ==>
000077
000078  Data Element: USER-DAY-NAME
000079  SET Alpha/Numeric Parameter field or value
000080
000081 05 USER-HOLIDAY  PIC 9(01).
000082 05 USER-DAY-COUNT    PIC 9(06).
000083 05 USER-DAY-NAME     PIC X(25).
000084    01 PASS-AREA.
000085 05 PASS-FROM-DATE.
000086 10 PASS-FROM-DATE-9 PIC 9(06).
000087    PF1=Input Numeric PF2=Output Numeric PF3=EXIT PF5=Output Alpha
000088
```

After setting the data elements variables for use by DLG, the user must place the cursor under a statement in the application program where he/she wants to insert the application program source required to invoke the CA Calendar Routines.

```
==================== DLG FUNCTION SELECTION ====================
 A-UTILITY   OR B-BASIC OR C-EXTENDED  OR D-CALENDAR-360
CURRENT....     BETWEEN... NEXT DAY..   365->360..
VALID DATE.     +/- DAYS.. PREV DAY..  360->365..
LEAP YEAR..     +/- WEEKS. SINCE BOM.  BETWEEN...
HOLIDAY....     +/- MONTHS UNTIL EOM.  +/-DAYS...
PROCESS....     +/- YEARS. BOM +/- N.  +/-MONTHS.
DAY OF WK..X               EOM +/- N.  +/-YEARS..
->ABS DAY..                  NTH DOW...
ABS DAY->..
RE-FORMAT..
AGE BETWEEN
COMPARE....
CONCATENATE
TRUNCATE...
OR E-ADVANCED FUNCTIONS
(CHOOSE ONE FROM EACH E? OPTION. E1 -> E5)
E1---DATE-ID..  OR SINCE/UNTIL.  OR RELATIVE...  OR NTH-DOW...
E2---BEGINNING  OR ENDING......
E3---PREVIOUS.  OR CURRENT.....  OR NEXT.......
E4---CALENDAR.  OR FISCAL......
E5---YEAR.....  OR QTR.........  OR MONTH......
F1=HELP   F3=EXIT   F4=BACK-1-LEVEL
```

The following example shows the DLG Function Specification screen for U006 DAY OF WEEK with test data entered after pressing F5=ENTER DATA :

```
================DLG FUNCTION SPECIFICATION ================
FUNCTION-IN-PROCESS= U006  DAY OF WEEK ONP = DAY #
ENTER BUSINESS-PARAMETERS- ** ENTER INPUT-PARAMETERS-- PRESS ENTER OR F5
*FROM-DATE-MASK..= --YYMMDD =
*=> FROM-DATE-9.......= 00980419
* *
* *
* *
*  *** RETURN-CODE = 000 ************************
* *
* OUT1-DATE-E.......= SUNDAY
* *
* *
*  *** OUTPUT-NUMERIC-PARM..= +00000001 ********
DAY-OF-WEEK-STRING....= 1234567 (SUN MON TUE WED THU FRI SAT E.G. 1234567)
CENTURY-BREAK.........= SLIDE-20 ( WINDOW RANGE FROM => 1978 TO => 2077)
F1=HELP   F4=BACK-1-LEVEL   F5=ENTER-DATA  F6=GENERATE
GOOD RETURN
```

The following example shows an application program modified by DLG after F5 testing and pressing F6=GENERATE:

```
File Edit Confirm Menu Utilities Compilers Test Help
EDIT  PDGBO.PDS.COBOL(T) 0- 01.99 Columns 00001 00072
Command ===> _____________________________________________________ Scroll ===> CSR
* MOVE USER-FROM-DATE-9  TO
 * PASS-FROM-DATE-9.
* CALL 'XYZ' USING PASS-AREA.
 * IF INLINE-GOOD-RETURN
 *  MOVE PASS-NUM TO USER-DOW.
 *  MOVE PASS-NAME TO USER-DAY-NAME.
*-----------------------------------------------------------------*
 * TransCentury(S) COBOL (U006) DAY OF WEEK ONP = DAY #
```

```
 * D.L.G. TESTING COMPLETED 1998/04/19 17:43:35
 *-----------------------------------------------------------------*
MOVE ‘U006’   TO TRC-CONVR-FUNCTION-CODE
MOVE ‘1234567’ TO TRC-CONVR-DOW-STRING
MOVE ‘SLIDE-20’ TO TRC-CONVR-CENTURY-BREAK
MOVE ‘CCYYMMDD’ TO TRC-CONVR-OUT1-DATE-MASK
MOVE ‘Y’ TO TRC-CONVR-VALIDATE-NULL
MOVE ‘--YYMMDD’ TO TRC-CONVR-FROM-DATE-MASK
MOVE USER-FROM-DATE-9  TO TRC-CONVR-FROM-DATE-9PERFORM
TRC-CALENDAR-ROUTINESIF TRC-CONVR-RETURN-GOOD
MOVE TRC-CONVR-OUT1-DATE-EUSER-DAY-NAME
MOVE TRC-CONVR-OUT-NUMERIC-PARMTO USER-DOWELSE  PERFORM TRC-ERROR-ROUTINE
END-IF *
*-------------------------------------------------*
* END OF GENERATED TransCentury (E) CODE *
*-------------------------------------------------*
```

TCDSRKEY—Reset (Clear) CA Data/Date Element Values

The screen display of the ISPF Edit function key settings for a COBOL application program (see Optional COBOL TCDS PF Key Functions) includes a PF key setting for F16 “TCDSRKEY”.

Pressing F16 invokes DLG TCDSRKEY function. This function is used to display and/or reset DLG data and date element “saved” variable names and definitions. The saved DLG information is used when DLG generates the COBOL application program source to invoke CA Calendar Routines functions.

The following examples display the process used to “set” DLG data and date variable information. The last examples in the series display the DLG screens used to view and/or reset DLG variable information.

The following example shows a COBOL application program:

```
File Edit Confirm Menu Utilities Compilers Test Help
 EDIT    PDGBO.PDS.COBOL(T) - 01.99  Columns 00001 00072
Command ===>    Scroll ===> CSR
000056   MOVE USER-FROM-DATE TO INLINE-DATEIN.
000057   CALL 'XYZ' USING USER-FROM-DATE
000058  USER-TO-DATE
000059  USER-STDOUT
000060  USER-OUT1
000061  USER-OUT2
000062  USER-OUT3
000063  USER-DOW
000064  USER-HOLIDAY
000065  USER-DAY-COUNT
000066  USER-DAY-NAME.
000067   IF INLINE-GOOD-RETURN
000068  MOVE INLINE-HOLIDAY TO USER-HOLIDAY.
```

The following example shows a user assigned DLG “FROM” date field:

```
EDIT
PDGBO.PDS.COBOL(T) - 01.99
FROM date:--MMDDYY
Command ===>    Scroll ===> CSR
000056   MOVE USER-FROM-DATE TO INLINE-DATEIN.
000057   CALL 'XYZ' USING USER-FROM-DATE
```

```
000058  USER-TO-DATE
000059  USER-STDOUT
000060  USER-OUT1
000061  USER-OUT2
000062  USER-OUT3
000063  USER-DOW
000064  USER-HOLIDAY
000065  USER-DAY-COUNT
000066  USER-DAY-NAME.
```

The following example shows a user assigned DLG “TO” date field:

```
EDIT    PDGBO.PDS.COBOL(T) - 01.99  TO date:CCYYMMDD
Command ===>    Scroll ===> CSR
000056   MOVE USER-FROM-DATE TO INLINE-DATEIN.
000057   CALL 'XYZ' USING USER-FROM-DATE
000058  USER-TO-DATE
000059  USER-STDOUT
000060  USER-OUT1
000061  USER-OUT2
000062  USER-OUT3
000063  USER-DOW
000064  USER-HOLIDAY
000065  USER-DAY-COUNT
000066  USER-DAY-NAME.
000067   IF INLINE-GOOD-RETURN
000068  MOVE INLINE-HOLIDAY TO USER-HOLIDAY.
```

The following example shows a user assigned DLG input “NUMREIC” field:

```
EDIT    PDGBO.PDS.COBOL(T) - 01.99   Numeric I/P
 Command ===>    Scroll ===> CSR
 000056   MOVE USER-FROM-DATE TO INLINE-DATEIN.
 000057  CALL 'XYZ' USING USER-FROM-DATE
 000058  USER-TO-DATE
 000059  USER-STDOUT
 000060  USER-OUT1
 000061  USER-OUT2
 000062  USER-OUT3
 000063  USER-DOW
 000064  USER-HOLIDAY
 000065  USER-DAY-COUNT
 000066  USER-DAY-NAME.
```

The following example shows a user assigned DLG “OUTPUT-1” date field:

```
EDIT    PDGBO.PDS.COBOL(T) - 01.99  OUT1 date:CCYYMMDD
Command ===>    Scroll ===> CSR
000056    MOVE USER-FROM-DATE TO INLINE-DATEIN.
000057    CALL 'XYZ' USING USER-FROM-DATE
000058   USER-TO-DATE
000059   USER-STDOUT
000060   USER-OUT1
000061   USER-OUT2
000062   USER-OUT3
000063   USER-DOW
000064   USER-HOLIDAY
000065   USER-DAY-COUNT
000066   USER-DAY-NAME.
```

The following example shows a user assigned DLF “OUTPUT-2” date field:

```
EDIT    PDGBO.PDS.COBOL(T) - 01.99   OUT2 date:CCYYMMDD
Command ===>    Scroll ===> CSR
000056    MOVE USER-FROM-DATE TO INLINE-DATEIN.
000057    CALL 'XYZ' USING USER-FROM-DATE
000058   USER-TO-DATE
000059   USER-STDOUT
000060   USER-OUT1
000061   USER-OUT2
000062   USER-OUT3
000063   USER-DOW
000064   USER-HOLIDAY
000065   USER-DAY-COUNT
000066   USER-DAY-NAME.
```

The following example shows a user assigned “OUTPUT-3” date field:

```
EDIT    PDGBO.PDS.COBOL(T) - 01.99 OUT3 date:CCYYMMDD
Command ===>    Scroll ===> CSR
000056   MOVE USER-FROM-DATE TO INLINE-DATEIN.
000057   CALL 'XYZ' USING USER-FROM-DATE
000058  USER-TO-DATE
000059  USER-STDOUT
000060  USER-OUT1
000061  USER-OUT2
000062  USER-OUT3
000063  USER-DOW
000064  USER-HOLIDAY
000065  USER-DAY-COUNT
000066  USER-DAY-NAME.
```

The following example shows a user assigned DLG output “NUMERIC” field

```
EDIT    PDGBO.PDS.COBOL(T) - 01.99  Numeric O/P
Command ===>    Scroll ===> CSR
000056   MOVE USER-FROM-DATE TO INLINE-DATEIN.
000057   CALL 'XYZ' USING USER-FROM-DATE
000058  USER-TO-DATE
000059  USER-STDOUT
000060  USER-OUT1
000061  USER-OUT2
000062  USER-OUT3
000063  USER-DOW
000064  USER-HOLIDAY
000065 USER-DAY-COUNT
000066  USER-DAY-NAME.
```

The following example shows a user assigned DLG output “ALPHABETIC” field:

```
EDIT    PDGBO.PDS.COBOL(T) - 01.99  Alpha O/P
Command ===>    Scroll ===> CSR
000056   MOVE USER-FROM-DATE TO INLINE-DATEIN.
000057   CALL 'XYZ' USING USER-FROM-DATE
000058  USER-TO-DATE
000059  USER-STDOUT
000060  USER-OUT1
000061  USER-OUT2
000062  USER-OUT3
000063  USER-DOW
000064  USER-HOLIDAY
000065  USER-DAY-COUNT
000066  USER-DAY-NAME.
```

The following example shows display of the ISPF Edit function of the key assigned to TCDSRSET PFKey (F24):

```
 File Edit Confirm Menu Utilities Compilers Test Help
EDIT   PDGBO.PDS.COBOL(T) - 01.99  Columns 00001 00072
Command ===> TCDSRKEY  Scroll ===> CSR
000056 MOVE USER-FROM-DATE TO INLINE-DATEIN.
000057 CALL 'XYZ' USING USER-FROM-DATE
000058  USER-TO-DATE
000059  USER-STDOUT
000060  USER-OUT1
000061  USER-OUT2
000062
000063 PF KEY   TCDS
000064 Assignment  Data Element Name (or literal)  MASK
000065 TCDSFROM   USER-FROM-DATE --MMDDYY
000066 TCDSTO    USER-TO-DATE  CCYYMMDD
000067 TCDSDATA INUM USER-DOW
000068 TCDSDATA ONUM USER-DAY-COUNT
000069 TCDSDATA ALPH USER-DAY-NAME
000070 TCDSDATE STND
000071 TCDSDATE OUT1 USER-OUT1 CCYYMMDD
000072 TCDSDATE OUT2 USER-OUT2 CCYYMMDD
000073 TCDSDATE OUT3 USER-OUT3 CCYYMMDD
000074 PF3=EXIT, PF9=Clear All and EXIT
000075
```

After setting and using the DLG data and date functions, the user pressed the PFkey assigned to TCDSRKEY. A pop-up screen display appears at the bottom of the ISPF Edit screen. The DLG user can clear all CA data element assignments by pressing PFKey PF9 = Clear All.

The DLG user can clear or change individual data element assignments by placing the CURSOR under a specific Data Element Name (or literal) and overtyping or clearing the information on the screen. The DLG user can also modify the TCDS MASK assignment values by placing the CURSOR under a specific date mask and overtyping the screen information. Press the ENTER key after making changes on the screen to process the changes.

The user presses the PF3 key to EXIT this function.

```
EDIT    PDGBO.PDS.COBOL(T) - 01.99  Columns 00001 00072
Command ===>    Scroll ===> CSR
## TCDS PFKEY VALUES FOR DATES AND DATA HAVE BEEN RESET ##
000057   CALL 'XYZ' USING USER-FROM-DATE
000058  USER-TO-DATE
000059  USER-STDOUT
000060  USER-OUT1
000061  USER-OUT2
000062  USER-OUT3
000063  USER-DOW
000064  USER-HOLIDAY
000065  USER-DAY-COUNT
000066  USER-DAY-NAME.
```

TCDSSRCH—Find (Next) Procedure Statement Containing a Matching Element on the Data Element File

The screen display of the ISPF Edit function key settings for a COBOL application program includes a PFkey setting for F17 “TCDSSRCH”.

Pressing F17 invokes DLG TCDSSRCH function. This function is used to search forward in the COBOL application program procedure division source code for the next occurrence of a data element name which has a matching entry in the Data Elements file.

The following examples display the DLG application program settings and Data Element file information required to enable the DLG TCDSSRCH function.

After processing the DLG Program Profile definition, the user can press the TCDSSRCH PFkey to locate application program procedure statements containing data elements stored in a Data Elements file.

```
 --------------------- DLG Program Profile Definition -------------------------
- OPTION ==>   Calendar Routine Customer/Version ABABABABAB 5.00      Date
Logic Generator Customer/Version ABABABABAB 5.00      MONDAY, APRIL 27, 1998
11:29:08
 LangType.... 1 1)COB 2)COB/CICS 3)PL1 4)PL1/CICS 5)ASM 6)ASM/CICS
 TransCentury Engine Name TRCENGIN
 Copy Book Member Name... TRCCONVR   Copy Inclusion Statement COPY  Copylib
File Name(s).... ADP.CLIENT.NINEWEST.COPYLIB
 PDGBO.WRK5000.COPYLIB  PDGBO.WRK5000.SAMPLE.COPYLIB  Data Element File Name..
PDGBO.WRK5000.DATA.ELEMENT(IDNL)
Press ENTER to continue, PFK3 to EXIT, PFK7 to Add COPYLIB(s)Program Type set
to Batch COBOL
```

The following example shows a Data Element file containing entries for the application program being edited Menu Utilities Compilers Help:

```
 BROWSE  PDGBO.WRK5000.DATA.ELEMENT(IDNL) - 01.21  Line 00000000 Col 001 080
 Command ===>    Scroll ===> CSR
******************************** Top of Data ********************************
@LIST  IDNL   DNLIST C305M1  1997/04/21 11:42   TSJNOA
 00006XC T   TRCHLGEN USER-FROM-DATE --MMDDYY
000069C T   TRCHLGEN USER-FROM-DATE-9
00008XC T   TRCHLGEN USER-OUT1 0
0010XC T   TRCHLGEN USER-OUT2
00032XC T   TRCHLGEN USER-OUT3
000089C T   TRCHLGEN USER-STDOUT
00008XC T   TRCHLGEN USER-TO-DATE
000089C T   TRCHLGEN USER-TO-DATE-9
00006GC T   NIN103 XRANDT-DATE
000755 C D
000050 1 00002XC T   NIN103 XRANDT-DAY-DD
000758 C D
000050 1
00002XC T   NIN103 XRANDT-DAY-MM
000757 C D
000050 1
```

Application program statement highlighted after user presses the DLG TCDSSRCH PFkey (F17)

```
File Edit Confirm Menu Utilities Compilers Test Help
EDIT    PDGBO.PDS.COBOL(T) - 01.99  Columns 00001 00072
Command ===>    Scroll ===> CSR
000042 /
000043 PROCEDURE DIVISION.
000044
000045 IF USER-TO-DATE-9 > USER-FROM-DATE-9
000046  DISPLAY '"ERRORS"'.
000047
```

If Program Execution is Suspended

If the CA Date Logic Generator appears to hang up indefinitely while trying to pass test data to the CA Calendar Routines engine, then you may have to recompile the Calendar Routines engine to terminate execution with a GOBACK statement rather than an EXIT PROGRAM statement.

Appendix A: CA Date Logic Generator ISPF Table Files

This appendix gives a description of the ISPF Tables used by the CA Date Logic Generator.

- Overview
- Program Data Elements
- COBOL Reserve Words
- CA Calendar Routines Date Masks
- Program Profile

Overview

The CA Date Logic Generator accesses ISPF tables to retrieve and store information used during an ISPF CA Date Logic Generator EDIT session. The ISPF Table services are used to maintain sets of dialog variables. A table is a two-dimensional array of information in which each column of a table corresponds to a dialog variable, and each row of a table contains a set of values for those variables.

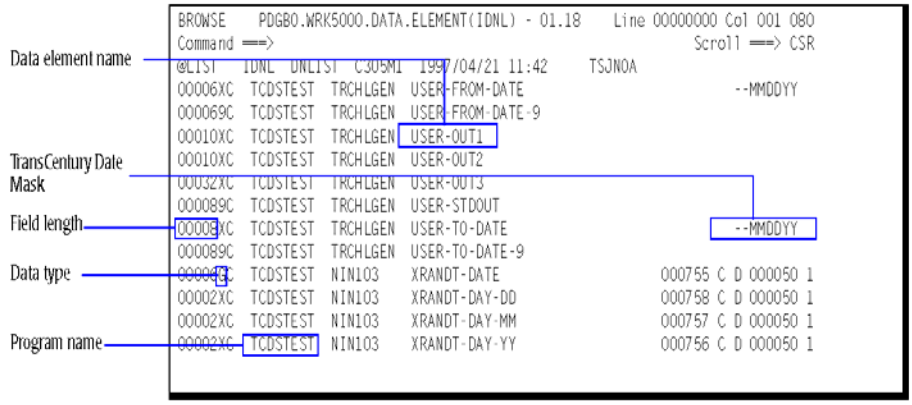
Both temporary and permanent tables are used by CA Date Logic Generator. A temporary table is used to store program data elements extracted from a Data Elements Definition (and usage) file. The other tables used by CA Date Logic Generator (Cobol Reserve Words, Date Masks, and Program Profile) are permanent tables. The CA Date Logic Generator permanent tables are maintained in one or more table libraries (depending on installation preferences) which reside on direct access storage.

Program Data Elements

The Data Elements file is a list of program storage variables identified by an analysis process as date fields (CA Analysis for z/OS produces a VBASE file containing data element names and definitions which have been identified as “date” fields). This file also can contain the data element field definitions.

The CA Date Logic Generator Data Element table is a temporary table which is created during CA Date Logic Generator Program Profile processing (once, each time a program is edited by ISPF EDIT invoking CA Date Logic Generator). This table is not written to disk storage.

CA Analysis for z/OS VBASE FILE (extract):



The above file is used by the CA Date Logic Generator to build a temporary ISPF Table during ISPF EDIT/DLG Program Profile processing. The temporary table contains:

KEY	CA Date Mask
data element name	derived from data type and length or valid mask supplied in VBASE
USER-FROM-DATE	--MMDDYY
USER-OUT1	CCYY-MM-DD
USER-TO-DATE-9	CCYYMMDD

COBOL Reserve Words

The DLF IPSF COBOL Reserve Word table contains the following list of COBOL language statements. This table is used by CA Date Logic Generator to eliminate non-data elements when processing scans for automated remediation of simple “IF” statements and/or “FIND” element processing. A reserved word is a COBOL word which has been given a special meaning within the COBOL language. The word is reserved for the special purpose which has been defined for the word and cannot be used as a user-defined data element.

ACCEPT	ACCESS	ADD	ADDRESS
ADVANCING	AFTER	ALL	ALPHABET
ALPHABETIC	ALPHABETIC-LOWER	ALPHABETIC-UPPER	ALPHANUMERIC
ALPHANUMERIC-EDITED	ALSO	ALTER	ALTERNATE
AND	ANY	APPLY	ARE
AREAS	ARITHMETIC	ASCENDING	ASSIGN
AT	AUTHOR	B-AND	B-EXOR
B-LESS	B-NOT	B-OR	BASIS
BEFORE	BEGINNING	BINARY	BIT
BITS	BLANK	BLOCK	BOOLEAN
BOTTOM	BY	CALL	CANCEL
CBL	CD	CF	CH
CHARACTER	CHARACTERS	CLASS	CLOCK-UNITS
CLOSE	COBOL	CODE	CODE-SET
COLLATING	COLUMN	COM-REG	COMMA
COMMIT	COMMUNICATION	COMP	COMP-1
COMP-2	COMP-3	COMP-4	COMP-5
COMP-6	COMP-7	COMP-8	COMP-9
COMPUTATIONAL	COMPUTATIONAL-1	COMPUTATIONAL-2	COMPUTATIONAL-3
COMPUTATIONAL-4	COMPUTATIONAL-5	COMPUTATIONAL-6	COMPUTATIONAL-7
COMPUTATIONAL-8	COMPUTATIONAL-9	COMPUTE	CONFIGURATION
CONNECT	CONTAINED	CONTAINS	CONTENT
CONTINUE	CONTROL	CONTROLS	CONVERTING
COPY	CORR	CORRESPONDING	COUNT

CURRENCY	CURRENT	DATA	DATE
DATE-COMPILED	DATE-WRITTEN	DAY	DAY-OF-WEEK
DB	DB-ACCESS-CONTROLKEY	DB-DATA-NAME	DB-EXCEPTION
DB-RECORD-NAME	DB-SET-NAME	DB-STATUS	DBCS
DE	DEBUG-CONTENTS	DEBUG-ITEM	DEBUG-LINE
DEBUG-NAME	DEBUG-SUB-1	DEBUG-SUB-2	DEBUG-SUB-3
DEBUGGING	DECIMAL-POINT	DECLARATIVES	DEFAULT
DELETE	DELIMITED	DELIMITER	DEPENDING
DESCENDING	DESTINATION	DETAIL	DISABLE
DISCONNECT	DISPLAY	DISPLAY-1	DISPLAY-2
DISPLAY-3	DISPLAY-4	DISPLAY-5	DISPLAY-6
DISPLAY-7	DISPLAY-8	DISPLAY-9	DIVIDE
DIVISION	DOWN	DUPLICATE	DUPLICATES
DYNAMIC	EGCS	EGI	EJECT
ELSE	EMI	EMPTY	ENABLE
END	END-ADD	END-CALL	END-COMPUTE
END-DELETE	END-DISABLE	END-DIVIDE	END-ENABLE
END-EVALUATE	END-IF	END-MULTIPLY	END-OF-PAGE
END-PERFORM	END-READ	END-RECEIVE	END-RETURN
END-REWRITE	END-SEARCH	END-SEND	END-START
END-STRING	END-SUBTRACT	END-TRANCEIVE	END-UNSTRING
END-WRITE	ENDING	ENTER	ENTRY
ENVIRONMENT	EOP	EQUAL	EQUALS
ERASE	ERROR	ESI	EVALUATE
EVERY	EXACT	EXCEEDS	EXCEPTION
EXCLUSIVE	EXIT	EXTEND	EXTERNAL
FALSE	FD	FETCH	FILE
FILE-CONTROL	FILLER	FINAL	FIND
FINISH	FIRST	FOOTING	FOR
FORMAT	FREE	FROM	FUNCTION
GENERATE	GET	GIVING	GLOBAL
GO	GOBACK	GREATER	GROUP

HEADING	HIGH-VALUE	HIGH-VALUES	I-O
I-O-CONTROL	ID	IDENTIFICATION	IF
IN	INDEX	INDEX-1	INDEX-2
INDEX-3	INDEX-4	INDEX-5	INDEX-6
INDEX-7	INDEX-8	INDEX-9	INDEXED
INDICATE	INITIAL	INITIALIZE	INITIATE
INPUT	INPUT-OUTPUT	INSERT	INSPECT
INSTALLATION	INTO	INVALID	IS
JUST	JUSTIFIED	KANJI	KEEP
KEY	KEY	LABEL	LAST
LD	LEADING	LEFT	LENGTH
LESS	LIMIT	LIMITS	LINAGE
LINAGE-COUNTER	LINE	LINE-COUNTER	LINES
LINKAGE	LOCALLY	LOCK	LOW-VALUE
LOW-VALUES	MEMBER	MEMORY	MERGE
MESSAGE	MODE	MODIFY	MODULES
MORE-LABELS	MOVE	MULTIPLE	MULTIPLY
NATIVE	NEGATIVE	NEXT	NO
NONE	NOT	NULL	NULLS
NUMBER	NUMERIC	NUMERIC-EDITED	OBJECT-COMPUTER
OCCURS	OF	OFF	OMITTED
ON	ONLY	OPEN	OPTIONAL
OR	ORDER	ORGANIZATION	OTHER
OUTPUT	OVERFLOW	OWNER	PACKED-DECIMAL
PADDING	PAGE	PAGE-COUNTER	PARAGRAPH
PASSWORD	PERFORM	PF	PH
PIC	PICTURE	PLUS	POINTER
POSITION	POSITIVE	PRESENT	PRINTING
PRIOR	PROCEDURE	PROCEDURE-POINTER	PROCEDURES
PROCEED	PROCESSING	PROGRAM-ID	PROTECTED
PURGE	QUEUE	QUOTE	QUOTES
RANDOM	RD	READ	READY

REALM	RECEIVE	RECONNECT	RECORD
RECORD-NAME	RECORDING	RECORDS	REDEFINES
REEL	REFERENCE	REFERENCES	RELATION
RELATIVE	RELEASE	RELOAD	REMAINDER
REMOVAL	RENAMES	REPEATED	REPLACE
REPLACING	REPORT	REPORTING	REPORTS
RERUN	RESERVE	RESET	RETAINING
RETRIEVAL	RETURN	RETURN-CODE	REVERSED
REWIND	REWRITE	RF	RH
RIGHT	ROLLBACK	ROUNDED	RUN
SAME	SD	SEARCH	SECTION
SECURITY	SEGMENT	SEGMENT-LIMIT	SELECT
SEND	SENTENCE	SEPARATE	SEQUENCE
SEQUENTIAL	SERVICE	SESSION-ID	SET
SHARED	SHIFT-IN	SHIFT-OUT	SIGN
SIZE	SKIP1	SKIP2	SKIP3
SORT	SORT-CONTROL	SOR-CORE-SIZE	SORT-FILE-SIZE
SORT-MERGE	SORT-MESSAGE	SORT-MODE-SIZE	SORT-RETURN
SOURCE	SOURCE-COMPUTER	SPACE	SPACES
SPECIAL-NAMES	STANDARD	STANDARD-1	STANDARD-2
STANDARD-3	STANDARD-4	START	STATUS
STOP	STORE	STRING	SUB-QUEUE-1
SUB-QUEUE-2	SUB-QUEUE-3	SUB-SCHEMA	SUBTRACT
SUM	SUPPRESS	SYMBOLIC	SYNC
SYNCHRONIZED	TABLE	TALLY	TALLYING
TAPE	TENANT	TERMINAL	TERMINATE
TEST	TEXT	THAN	THEN
THROUGH	THRU	TIME	TIMES
TITLE	TO	TOP	TRACE
TRAILING	TRANSCEIVE	TRUE	TYPE
UNEQUAL	UNIT	UNSTRING	UNTIL
UP	UPDATE	UPON	USAGE
USAGE-MODE	USE	USING	VALID

VALIDATE	VALUE	VALUES	VARYING
WAIT	WHEN	WHEN-COMPILED	WITH
WITHIN	WORDS	WORKING-STORAGE	WRITE
WRITE-ONLY	ZERO	ZEROES	ZEROS
<	<=	+	*
**	-	/>	>=
=			

CA Calendar Routines Date Masks

The CA DATE LOGIC GENERATOR IPSF COBOL CA Calendar Routines Date Masks table contains the following list of CA “date” masks. These masks are used by CA Calendar Routines when processing input/output date fields. The masks are used to interpret the contents of input “date” fields and create formatted output date fields (see the CA Application Manual for more information on “date” masks).

	DDMMCCYY	E-TDS-22	MMDDYY-
---DDDYY	DDMMYY-	E-TDS-23	MMDDYYCC
---KKKYY	DDMMYYCC	E-TDS-24	MMYYCCDD
---YYDDD	DDYYCCMM	E-TDS-25	MMYYDD-
---YYKKK	DDYYMM-	E-TDS-26	MMYYDDCC
--DDMMYY	DDYYMMCC	E-TDS-27	RDDDYY-
--DDYYMM	E-A4*DMY	E-TDS-28	RMMDDYY
--HDDDYY	E-A4*EUR	E-TDS-29	RYYDDD-
--HYYDDD	E-A4*ISO	E-TDS-30	RYYMMDD
--MMDDYY	E-A4*JIS	E-TDS-31	S-ANSI89
--MMYYDD	E-A4*JUL	E-TDS-32	S-CICS-A
--RDDDYY	E-A4*MDY	E-TDS-33	S-COB370
--RYYDDD	E-A4*USA	E-TDS-34	S-DB2
--SDDDYY	E-A4*YMD	E-TDS-35	S-LE370
--SYYDDD	E-CY-M-D	E-TDS-36	S-TDS
--TDDDYY	E-D-M-CY	E-TDS-37	S-1900G
--TYYDDD	E-D-M-Y	E-TDS-38	S-1900NG

--YYDDMM	E-DB2EUR	E-TDS-39	SDDDYY-
--YYMMDD	E-DB2ISO	E-TDS-40	SMMDDYY
-CCDDDYY	E-DB2JIS	E-TDS-41	SYYDDD-
-CCYYDDD	E-DB2USA	E-TDS-42	SYYMMDD
-CCYYKKK	E-DMONCY	E-TDS-43	SNNNNNNN
-DDDCCYY	E-DMONY	E-TDS-44	TDDDYY-
-DDDYYCC	E-M-D-CY	E-TDS-45	TMMDDYY
-HMMDDYY	E-M-D-Y	E-TDS-46	TYYDDD-
-HYYMMDD	E-M-D-CY	E-TDS-47	TYYMMDD
-KKKCCYY	E-M-D-Y	E-TDS-48	YYCCDDD
-RMMDDYY	E-M/D/Y	E-TDS-49	YYCCDDMM
-RYYMMDD	E-TDS-01	E-TDS-50	YYCCMMDD
-SMMDDYY	E-TDS-02	E-TDS-51	YYDDCCMM
-SYYMMDD	E-TDS-03	E-TDS-52	YYDDD--
-TMMDDYY	E-TDS-04	E-TDS-53	YYDDDCC
-TYYMMDD	E-TDS-05	E-TDS-54	YYDDMM-
-YYCCDDD	E-TDS-06	E-TDS-55	YYDDMMCC
-YYDDDCC	E-TDS-07	E-TDS-56	YYKKK--
CCDDDYY	E-TDS-08	E-TDS-57	YYMMCCDD
CCDDMMYY	E-TDS-09	E-TDS-58	YYMMDD-
CCDDYYMM	E-TDS-10	E-Y-M-D	YYMMDDCC
CCMMDDYY	E-TDS-11	E-Y/M/D	9-A-CYJ
CCMMYYDD	E-TDS-12	E-YY-DDD	9-A-MDY
CCYYDDD	E-TDS-13	HDDDYY-	9-A-YJ
CCYYDDMM	E-TDS-14	HMMDDYY	9-A-YMD
CCYYKKK	E-TDS-15	HYYDDD-	9-CYMD
CCYYMMDD	E-TDS-16	HYYMMDD	9-N-CYJ
DDCCMMYY	E-TDS-17	KKKCCYY	9-N-MDY
DDCCYYMM	E-TDS-18	KKKYY--	9-N-YJ
DDDCCYY	E-TDS-19	MMCCDDYY	9-N-YMD
DDDYY--	E-TDS-20	MMCCYYDD	
DDDYYCC	E-TDS-21	MMDDCCYY	

Program Profile

The CA Date Logic Generator IPSF Program Profile table contains an entry for every program edited by ISPF EDIT using CA Date Logic Generator version 5.0 or greater. The Program Profile table contains information required by the CA Date Logic Generator to 'remediate' the program for Year 2000 processing compliance issues (e.g. windowing 6-character dates to enable "IF" statements to process Year "00") and/or invoke CA Calendar Routines functions.

CA Date Logic Generator processes three program language types. Each language requires different instructions when invoking the CA Calendar Routines. See the CA Application Manual for more information regarding specific contents of the table.

Elements	CA Date Logic Generator Settings for each program
APOST	"Y" COBOL program uses apostrophes
BCB	Calendar Routines "Window Parameter"
BCK	Calendar Routines "Validate date null positions"
BDW	Calendar Routines "Day-of-Week string (1234567)"
BEN	Calendar Routines "End Points Definition"
BFM	Calendar Routines "Fiscal Year (month) Start"
BFY	Calendar Routines "Fiscal Year Month (day) Start"
BHO	Calendar Routines "Holiday Table ID"
BPD	Calendar Routines "Business Day-of-Week (NEEEEN)"
CONVRNME	Name of Link/Pass area (01 TRC-CONVR-CONVERSATIONAL)
COPYBK	Name of Link/Pass copy book
COPYST	Source statement inclusion verb (COPY)
COPY1-9	Copylib File data set name(s)
DATAEDS	Data Element file name (e.g. VBASE-Analysis of z/OS)
ENGNAME	Program name DLG will execute in Foreground
F0	Program language type (1 - Batch Cobol)
GENENDIF	"Y/N" Generate "END-IF" COBOL statements when required
GENERROR	"Y/N" Generate COBOL PERFORM of Standard Error Routine
GENPERF	"C/P" Generate statements to always CALL Calendar Routines or PERFROM/CALL based upon function

Elements	CA Date Logic Generator Settings for each program
GENSTND	“Y/N” Generate COBOL statements to move CA Standard Date (CCYYMMDD) to output data element
IFRM	Calendar Routines “From Date Mask”
I01M	Calendar Routines “Output #1 Date Mask”
IO2M	Calendar Routines “Output #2 Date Mask”
IO3M	Calendar Routines “Output #3 Date Mask”
ITOM	Calendar Routines “To Date Mask”
LINKAGR	COBOL “LINKAGE SECTION” statement number
PROCEDR	COBOL “PROCEDURE DIVISION” statement number
QUOTE	Quote “Y” COBOL program uses quote
WRKSTGR	COBOL “WORK STORAGE” statement number

Appendix B: COBOL CA Invocation Architecture

This appendix describes the various methods of invoking the CA Calendar Routines by application programs written in COBOL.

- Overview
- Execution Shell
- Standard CALL
- PERFORM U011, U012, or U013 Functions

Overview

The CA Date Logic Generator generates COBOL source code which updates the CA CALL-Pass area data elements and invokes the Calendar Routines. Earlier versions of the CA Date Logic Generator created COBOL application source statements which always invoked the Calendar Routines using a standard CALL statement, passing the CA Calendar Routines CALL/Pass area as an argument (see the following example). This version of the CA Date Logic Generator can create COBOL application source code which CALLs the CA Calendar Routines or PERFORMS 'in-line' application CA Calendar Routines source code.

```
*-------------------------------------------------------------
* CA COBOL (P533) DATE ID FISCAL MONTH BEGIN
*-------------------------------------------------------------
MOVE 'P533'   TO TRC-CONVR-FUNCTION-CODE
MOVE '01' TO TRC-CONVR-HOLIDAY-TBL
MOVE '01' TO TRC-CONVR-FISCYR-START
MOVE '01' TO TRC-CONVR-FISCMO-START
MOVE 'NEEEEEN' TO TRC-CONVR-PROC-DAY-DEF
MOVE '19/20;68' TO TRC-CONVR-CENTURY-BREAK
MOVE 'CCYYMMDD' TO TRC-CONVR-OUT1-DATE-MASK
MOVE '--YYMMDD' TO TRC-CONVR-OUT2-DATE-MASK
MOVE '-CCYYDDD' TO TRC-CONVR-OUT3-DATE-MASK
MOVE 'YYMMDD--' TO TRC-CONVR-FROM-DATE-MASK
MOVE ?-USER-FROM-DATE
TO TRC-CONVR-FROM-DATE-X
CALL 'TRCENGIN' USING TRC-CONVR-CONVERSATIONAL
IF TRC-CONVR-RETURN-GOOD
MOVE TRC-CONVR-STDOUT-DATE-X
TO ?-USER-STDOUTMOVE TRC-CONVR-OUT1-DATE-X    TO ?-USER-OUT1
MOVE TRC-CONVR-OUT2-DATE-9
TO ?-USER-OUT2
MOVE TRC-CONVR-OUT3-DATE-9
TO ?-USER-OUT3 ELSEPERFORM ?-USER-ERROR-ROUTINE
 END-IF
*
```

Execution Shell

This version of the CA-Date Logic Generator establishes an ‘execution’ Shell Paragraph which is PERFORMED for all COBOL application programs. This shell invokes all CA Calendar Routines functions (350+) using over190 date format masks (see the *CA Calendar Routines Applications Manual* for a description of the functions and date masks).

The CA Date Logic Generator Shell Paragraph can determine which of two possible methods to invoke the CA Calendar Routines. This appendix describes methods of invoking the CA Calendar Routines and the automated CA Date Logic Generator COBOL statement generation processes.

The following example is a sample program which will be used to depict the various methods of invoking the CA Calendar routines by the CA Date Logic Generator.

```
IDENTIFICATION DIVISION.
PROGRAM-ID. LOADTABL.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. XXX.
OBJECT-COMPUTER. YYY.
*-----------------------------------
*
* TEST QUOTE VERSION “5.0”*
*-----------------------------------
 INPUT-OUTPUT SECTION.
FILE-CONTROL.
 SELECT IN-FILE ASSIGN TO UT-S-INP.
DATA DIVISION.
FILE SECTION. FD IN-FILE.
01 IN-REC.
05 USER-FROM-DATE.
10 USER-FROM-DATE-9 PIC 9(06).
05 USER-TO-DATE.
10 USER-TO-DATE-9 PIC 9(06).
05 USER-STDOUT PIC X(08).
05 USER-OUT1  PIC X(08).
05 USER-OUT2  PIC X(08).
05 USER-OUT3  PIC X(08).
05 USER-DOW  PIC 9(01).
05 USER-HOLIDAY PIC 9(01).
05 USER-DAY-COUNT  PIC 9(06).
05 USER-DAY-NAME   PIC X(25).
 WORKING-STORAGE SECTION.
01 INLINE-PASS-AREA.
05 INLINE-DATEIN.
10 INLINE-DATEIN-9 PIC 9(06).
05 INLINE-RETURN-CODE PIC 9(01) VALUE 0.
88 INLINE-GOOD-RETURN   VALUE 0.
05 INLINE-DOW PIC 9(01).
05 INLINE-HOLIDAY  PIC 9(01).
 PROCEDURE DIVISION.
IF USER-TO-DATE-9   GREATER USER-FROM-DATE-9  DISPLAY ‘”ERRORS”’.
MOVE USER-FROM-DATE TO INLINE-DATEIN.
CALL ‘GETDOW’ USING INLINE-PASS-AREA.
IF INLINE-GOOD-RETURN    MOVE INLINE-DOW  TO USER-DOW.
MOVE USER-FROM-DATE TO INLINE-DATEIN.
CALL ‘TESTHOL’ USING INLINE-PASS-AREA.
IF INLINE-GOOD-RETURN    MOVE INLINE-HOLIDAY TO USER-HOLIDAY.
 ALL-DONE.    GOBACK.
```

Two methods of invoking CA Calendar Routines are available. The first method is similar to the original CALL invocation shown in the previous example. The application performs the 'execution' shell paragraph. The 'execution' shell paragraph CALLs CA Calendar Routines.

The second method PERFORMs CA Calendar Routines 'in-line' procedure statements inserted into the application program by the CA Date Logic Generator.

The form of CA Calendar Routines invocation is established when the product is installed and/or changed by the CA Systems Administrator. The CA Date Logic Generator user has no control of how your installation chooses to invoke the CA Calendar Routines.

For COBOL application programs, most installations will choose to have the CA Calendar Routines invoked by PERFORMing the following functions from an 'execution' Shell Paragraph.

Code	Description
C101	Days Between
C102	Date +/- N Days
C201	Get Calendar Next Day
C202	Get Calendar Previous Day
U001	Get Current Date/Time
U002	Validate Date
U003	Determine Leap Year
U006	Get Day of Week
U007	Date --> Absolute Days
U008	Absolute Days --> Date
U009	Reformat Date
U011	Compare Two Dates Same Format
U012	Concatenate Century
U013	Truncate Century

The remaining CA Calendar Routines functions are CALLed from the 'execution' shell paragraph. The CA Date Logic Generator includes several source statement copybooks into COBOL application programs to facilitate CA Calendar Routines invocation. The next series of examples depict various methods the CA Date Logic Generator uses to invoke CA Calendar Routines.

Standard CALL

The example function being executed in this section is a function which cannot be PERFORMed due to the complexity of the function and/or the requirement of the function to access a CA Calendar Routines Holiday Table.

When generating COBOL application program statements, the CA Date Logic Generator will include the following in your application program:

1. COBOL COPY inclusion statement for the CA Calendar Routines copybook CALL/PASS area
2. COBOL COPY inclusion statement for the CA Calendar Routines copybook Working Storage variables used to process CALLed functions
3. COBOL application source statements required to set input values in the CA Calendar Routines CALL/PASS area, PERFORM the 'execution' shell, and process output data after CA Calendar Routines processing has been completed.
4. (Optional) COBOL application source statements to PERFORM an installation specific Error Routine
5. COBOL COPY inclusion statement for the CA Calendar Routines copybook Procedure Division for the 'execution' shell Paragraph
6. (Optional) COBOL COPY inclusion statement for the installation specific Error Routine (Paragraph header and procedure statements)
7. COBOL application source statements used by the 'execution' shell

The following example shows a COBOL program modified by CA Date Logic Generator to process a CALLed CA Calendar Routines function:

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID. LOADTABL.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER. XXX.
 OBJECT-COMPUTER. YYY.
*---------------------------------- *
* TEST QUOTE VERSION "5.0"  *
*----------------------------------
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
   SELECT IN-FILE ASSIGN TO UT-S-INP.
 DATA DIVISION.
 FILE SECTION.
FD IN-FILE.
 01 IN-REC.
05 USER-FROM-DATE.
10 USER-FROM-DATE-9 PIC 9(06).
05 USER-TO-DATE.
10 USER-TO-DATE-9  PIC 9(06).
05 USER-STDOUT   PIC X(08).
05 USER-OUT1 PIC X(08).
05 USER-OUT2 PIC X(08).
05 USER-OUT3 PIC X(08).
05 USER-DOW PIC 9(01).
05 USER-HOLIDAY  PIC 9(01).
05 USER-DAY-COUNT PIC 9(06).
05 USER-DAY-NAME  PIC X(25).
WORKING-STORAGE SECTION.
 01 INLINE-PASS-AREA.
05 INLINE-DATEIN.
10 INLINE-DATEIN-9  PIC 9(06).
   . . . . . .
   COPY TRCCONVR.
   COPY TRCCONWS.
 PROCEDURE DIVISION.
   . . . . . .
* MOVE USER-FROM-DATE TO INLINE-DATEIN.
* CALL 'TESTHOL' USING INLINE-PASS-AREA.
* IF INLINE-GOOD-RETURN
*  MOVE INLINE-HOLIDAY TO USER-HOLIDAY.
*-----------------------------------------------------------------*
* CA(S) COBOL (U004) HOLIDAY     ONP 0=NO,1=YES
* D.L.G. TESTING COMPLETED 1998/05/03 14:30:03
*-----------------------------------------------------------------*
MOVE 'U004'   TO TRC-CONVR-FUNCTION-CODE
MOVE '01' TO TRC-CONVR-HOLIDAY-TBL
MOVE 'SLIDE-20' TO TRC-CONVR-CENTURY-BREAK
MOVE 'Y' TO TRC-CONVR-VALIDATE-NULL
MOVE 'YYMMDD--' TO TRC-CONVR-FROM-DATE-MASK
MOVE ?-USER-FROM-DATE
3
 TO TRC-CONVR-FROM-DATE-X
PERFORM TRC-CALENDAR-ROUTINES IF TRC-CONVR-RETURN-GOOD
 MOVE TRC-CONVR-OUT-NUMERIC-PARM
TO ?-USER-OUT-NUMERIC     ELSE
4
 PERFORM TRC-ERROR-ROUTINE
   END-IF *
*-------------------------------------------------*
* END OF GENERATED CA(E) CODE  *
*-------------------------------------------------*
 5 (see Figure B-6)
   COPY TRCCONPD./
6 (see Figure B-7)
   COPY TRCERROR./ TRC-STANDARD-CALL.7
   CALL 'TRCENGIN' USING TRC-CONVR-CONVERSATIONAL.
```

The following example shows an extraction from the CA Calendar Routines CALL/PASS are copybook member TRCCONVR:

```
01 TRC-CONVR-CONVERSATIONAL.
*
05 TRC-CONVR-BUSINESS-PARAMETERS.
10 TRC-CONVR-HOLIDAY-TBL  PIC X(02) VALUE '01'.
10 TRC-CONVR-FISCYR-START PIC X(02) VALUE '01'.
10 TRC-CONVR-FISCMO-START PIC X(02) VALUE '01'.
10 TRC-CONVR-END-PNTS-DEF PIC X(01) VALUE 'T'.
10 TRC-CONVR-DOW-STRING   PIC X(07) VALUE '1234567'.
10 TRC-CONVR-PROC-DAY-DEF PIC X(07) VALUE 'NEEEEEN'.
10 TRC-CONVR-CENTURY-BREAK PIC X(08) VALUE '19/20;68'.
10 TRC-CONVR-FROM-DATE-MASK PIC X(08) VALUE 'YYMMDD--'.
10 TRC-CONVR-TO-DATE-MASK PIC X(08) VALUE '--YYMMDD'.
10 TRC-CONVR-OUT1-DATE-MASK PIC X(08) VALUE 'YYMMDD--'.
10 TRC-CONVR-OUT2-DATE-MASK PIC X(08) VALUE '--------'.
10 TRC-CONVR-OUT3-DATE-MASK PIC X(08) VALUE '--------'.
10 TRC-CONVR-VALIDATE-NULL PIC X(01) VALUE 'N'.
88 TRC-CONVR-DO-NOT-EDIT-NULLS  VALUE 'N'.
88 TRC-CONVR-EDIT-NULLS   VALUE 'Y'.
```

The following example shows the working storage copybook member TRCCONWS required when processing called functions:

```
*********************************************
*
*   CACHE AREAS
*
* NOTE: THESE AREAS MUST CORRESPOND
* PRECISELY TO TRCCONVR.
*
*********************************************
*
01 TRC-CACHE-RESULT-SW  PIC X.
88 TRC-CACHE-RESULT-FOUND  VALUE 'Y'.
88 TRC-CACHE-RESULT-NOT-FOUND   VALUE 'N'.
*
01 TRC-CACHE-ALPHA-YES-VAL     PIC X(01) VALUE 'Y'.
01 TRC-CACHE-ALPHA-NO-VAL  PIC X(01) VALUE 'N'.
*
01 TRC-CACHE-AREA-TEMP  PIC X(11000).
*
01 TRC-CACHE-AREAS.
05 TRC-CACHE-AREA-DEF-1.
10 TRC-CACHE-ENTRY-1  PIC X(1000).
10 TRC-CACHE-ENTRY-2-12 PIC X(11000).
05 TRC-CACHE-AREA-DEF-2 REDEFINES TRC-CACHE-AREA-DEF-1.
10 TRC-CACHE-ENTRY OCCURS 12 TIMES INDEXED BY TRC-CACHE-IDX.
15 TRC-CACHE-BUSINESS-PARAMETERS PIC X(89).
15 TRC-CACHE-INPUT-PARAMETERS  PIC X(56).
15 TRC-CACHE-OUTPUT-PARAMETERS  PIC X(144).
15 TRC-CACHE-EXTENDED-IO-AREAS.
20 TRC-CACHE-FROM-DATE-E   PIC X(50).
20 TRC-CACHE-TO-DATE-E PIC X(50).
20 TRC-CACHE-OUT1-DATE-E   PIC X(50).
20 TRC-CACHE-OUT2-DATE-E   PIC X(50).
20 TRC-CACHE-OUT3-DATE-E   PIC X(50).
15 FILLER   PIC X(461).
```

The following example shows an extraction from the CA Calendar Routines procedure division copybook member TRCCONPD:

```
TRC-CALENDAR-ROUTINES.
 PERFORM 020000-CHECK-CACHE
IF TRC-CACHE-RESULT-NOT-FOUND
PERFORM TRC-STANDARD-CALLMOVE TRC-CACHE-ENTRY-2-12
TO TRC-CACHE-AREA-TEMPMOVE TRC-CACHE-AREA-TEMP TO TRC-CACHE-AREA-DEF-1
MOVE TRC-CONVR-CONVERSATIONAL
TO TRC-CACHE-ENTRY (12).  /
*************************************************************
*  SEE IF THIS CALCULATION WAS PERFORMED IN THE LAST
*  TWELVE ITERATIONS.
*************************************************************
 020000-CHECK-CACHE. *
MOVE TRC-CACHE-ALPHA-NO-VAL TO TRC-CACHE-RESULT-SW.   *
************************************************************* *
IF THE FUNCTION INVOLVES IMPLICIT OR EXPLICIT  *
*  CURRENT DATE RETIRIVAL PERFORM A FRESH CALL EACH TIME. *
************************************************************* *
IF ((TRC-CONVR-FUNCTION-CODE = 'U001') OR (TRC-CONVR-FROM-DATE-X = 'CURRDATE')
OR (TRC-CONVR-FROM-DATE-E = 'CURRDATE') OR (TRC-CONVR-TO-DATE-X = 'CURRDATE')
OR (TRC-CONVR-TO-DATE-E = 'CURRDATE'))
NEXT SENTENCE   ELSE
PERFORM 022000-LOOK-AT-INDIV-CACHE
VARYING TRC-CACHE-IDX
FROM 1
BY 1
UNTIL TRC-CACHE-IDX > 12
OR TRC-CACHE-RESULT-FOUND.
/ *************************************************************
*  CHECK THE INPUT AND BUSINESS PARMS ONLY
************************************************************* *
 022000-LOOK-AT-INDIV-CACHE.   * IF ((TRC-CACHE-BUSINESS-PARAMETERS
(TRC-CACHE-IDX) =  TRC-CONVR-BUSINESS-PARAMETERS)    AND
(TRC-CACHE-INPUT-PARAMETERS (TRC-CACHE-IDX) =   TRC-CONVR-INPUT-PARAMETERS)
AND  (TRC-CACHE-FROM-DATE-E (TRC-CACHE-IDX) =   TRC-CONVR-FROM-DATE-E)  AND
(TRC-CACHE-TO-DATE-E (TRC-CACHE-IDX) =    TRC-CONVR-TO-DATE-E)) *  MOVE TRC-
CACHE-ALPHA-YES-VAL   TO TRC-CACHE-RESULT-SW    MOVE TRC-CACHE-ENTRY (TRC-
CACHE-IDX)     TO TRC-CONVR-CONVERSATIONAL* ELSENEXT SENTENCE.
```

The following example shows an extraction from the CA Calendar Routines procedure division copybook member TRCERROR:

```
****************************************
*
*   TRCERROR - ERROR PROCESSING ROUTINE
*
****************************************
****************************************
*
*
 TRC-ERROR-ROUTINE.
 DISPLAY
'*-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-*'.
DISPLAY  '*'.
DISPLAY  '* CA BUSINESS PROCESSING PARAMETERS:'.
DISPLAY  '*   HOLIDAY TABLE '   TRC-CONVR-HOLIDAY-TBL.
```

PERFORM U011, U012, or U013 Functions

The example function being executed in this section is one of the 'Bridging' functions. Function U011 (Compare, Same Date format), U012 (Concatenate Century), and U013 (Truncate Century) perform NO validation of input data and only work with specific date masks (see the *CA Calendar Routines Applications Manual* for a description of these functions and date masks which the function can process).

These function are PERFORMed due to the simple nature of the processing requests. When generating COBOL application program statements, the CA Date Logic Generator will include the following in your application program:

1. COBOL COPY inclusion statement for the CA Calendar Routines copybook CALL/PASS area
2. COBOL COPY inclusion statement for the CA Calendar Routines copybook Working Storage variables used to process PERFORMed functions
3. COBOL application source statements required to set input values in the CA Calendar Routines CALL/PASS area, PERFORM the 'execution' shell, and process output data after CA Calendar Routines processing has been completed
4. COBOL COPY inclusion statement for the CA Calendar Routines copybook Procedure Division for the 'execution' shell Paragraph
5. COBOL application source statements used by the 'execution' shell

The following example shows a COBOL program modified by CA Date Logic Generator to process the PERFORM CA Calendar Routines function U011 (Compare, Same Date Format).

```
WORKING-STORAGE SECTION.
01 INLINE-PASS-AREA.
05 INLINE-DATEIN.
10 INLINE-DATEIN-9  PIC 9(06).
   . . . . . .
COPY TRCCONVR.
COPY TRCU1XWS.
PROCEDURE DIVISION.
   . . . . . .
**=> IF USER-TO-DATE-9 < USER-FROM-DATE-9
*-------------------------------------------------------*
* CA(S) COBOL (U011) 'QUICK COMPARE FROM/TO' *
*-------------------------------------------------------*
MOVE '--YYMMDD' TO TRC-CONVR-FROM-DATE-MASK
MOVE '--YYMMDD' TO TRC-CONVR-TO-DATE-MASK
MOVE 'SLIDE-20' TO TRC-CONVR-CENTURY-BREAK
MOVE USER-TO-DATE-9 TO TRC-CONVR-FROM-DATE-9
MOVE USER-FROM-DATE-9 TO TRC-CONVR-TO-DATE-9
MOVE 'U011'   TO TRC-CONVR-FUNCTION-CODE
PERFORM TRC-CALENDAR-ROUTINES
IF TRC-CONVR-TO-LESS-THAN-FROM
*-------------------------------------------------------*
* END OF GENERATED CA(E) CODE   *
*-------------------------------------------------------*
DISPLAY '"PROCESS DATE INVALID"'.
   . . . . . .
 ALL-DONE.
GOBACK.
/
COPY TRCU1XPD.
/
TRC-STANDARD-CALL.
MOVE TRC-RTNCD-I-BAD-ROUTINE-FUNC TO
TRC-CONVR-RETURN-CODE
TRC-INTRL-RETURN-CODE.
```

The following example shows an extraction from Working Storage copybook member TRCU1XWS required when processing PERFORMED Functions U011, U012, U013.

```
*
01 TRC-ANSI-DATE   PIC 9(06).
01 TRC-ANSI-DATE-REDEF REDEFINES
TRC-ANSI-DATE.
05 TRC-ANSI-YEAR    PIC 9(02).
05 TRC-ANSI-MONTH   PIC 9(02).
05 TRC-ANSI-DAY PIC 9(02).
01 TRC-ANSI-TIME-9  PIC 9(08).
01 TRC-ANSI-TIME-HHMMSSHH REDEFINES   TRC-ANSI-TIME-9.
05 TRC-ANSI-HOUR    PIC X(02).
05 TRC-ANSI-MIN PIC X(02).
05 TRC-ANSI-SEC PIC X(02).
05 TRC-ANSI-HUNDR   PIC X(02).
01  TRC-ENGIN-PROGRAM-MISC.
*
05 TRC-ENGIN-DEFAULT-NULL-DATE-X PIC X(08) VALUE ZEROS.
05 TRC-ENGIN-DEFAULT-NULL-DATE-E PIC X(08) VALUE SPACES.
*
05 TRC-ENGIN-DEFAULT-NULL-NUMERIC PIC S9(8) COMP SYNC VALUE
+0.
*
05 TRC-ENGIN-COMPARE-IS-LESS PIC S9(8) COMP SYNC VALUE
-1.
05 TRC-ENGIN-COMPARE-IS-EQUAL   PIC S9(8) COMP SYNC VALUE
+0.
05 TRC-ENGIN-COMPARE-IS-GREATER  PIC S9(8) COMP SYNC VALUE
+1.
*
*
05 TRC-ENGIN-FUNC-CODE-U011 PIC X(04)  VALUE 'U011'.
05 TRC-ENGIN-FUNC-CODE-U012 PIC X(04)  VALUE 'U012'.
05 TRC-ENGIN-FUNC-CODE-U013 PIC X(04)  VALUE 'U013'.
```

The following example shows an extraction from the CA Calendar Routines procedure division copybook member TRCU1XPD required when processing PERFORMed Functions U011, U012, U013.

```
TRC-CALENDAR-ROUTINES.
*
IF TRC-CONVR-FUNCTION-CODE = TRC-ENGIN-FUNC-CODE-U011
PERFORM TRC-030000-SPECIAL-COMPARE  ELSE
IF TRC-CONVR-FUNCTION-CODE = TRC-ENGIN-FUNC-CODE-U012
PERFORM TRC-032000-PROCESS-CENTURY  ELSE
IF TRC-CONVR-FUNCTION-CODE = TRC-ENGIN-FUNC-CODE-U013
PERFORM TRC-033000-TRUNCATE-DATE
ELSE
```

PERFORM Limited Functions

The example function being executed in this section of Appendix B is request to obtain the day-of-week number derived from an input date. The function is one of the CA Calendar Routines functions which can be performed from included CA Calendar Routines source statements in your application program. Performing a 'limited' set of routines is desirable when application program performance is of major importance at your installation. The single drawback using this method is the need to 'recompile' the application program whenever the CA Calendar Routines have been modified with changes involving 'Limited' functions. See the CA Calendar Routines Applications Manual for a description of these functions.

Limited functions can be PERFORMed due to the simple nature of the processing requests and NO Holiday Table access. When generating COBOL application program statements, the CA Date Logic Generator will include the following in your application program:

1. COBOL COPY inclusion statement for the CA Calendar Routines copybook CALL/PASS area
2. COBOL COPY inclusion statement(s) for the CA Calendar Routines copybook Working Storage variables used to process PERFORMed functions
3. COBOL application source statements required to set input values in the CA Calendar Routines CALL/PASS area, PERFORM the 'execution' shell, and process output data after CA Calendar Routines processing has been completed.
4. (Optional) COBOL application source statements to PERFORM an installation specific Error Routine
5. COBOL COPY inclusion statement(s) for the CA Calendar Routines copybook Procedure Division statements for PERFORMED functions and PERFORMing the 'execution' shell
6. COBOL application source statements used by the 'execution' shell
7. (Optional) COBOL COPY inclusion statement for the installation specific Error Routine (Paragraph header and procedure statements)

The following example shows a COBOL program modified by CA Date Logic Generator to process the PERFORM CA Calendar Routines Function U006 (Day-of-Week).

```
 WORKING-STORAGE SECTION.
01 INLINE-PASS-AREA.
05 INLINE-DATEIN.
10 INLINE-DATEIN-9 PIC 9(06).
   . . . . . .
 /
   COPY TRCCONVR.
/
   COPY TRCSMLWS.
/
   COPY TRCCOMNW.
/

 PROCEDURE DIVISION.     . . . . . .
* MOVE USER-FROM-DATE TO INLINE-DATEIN.
* CALL 'GETDOW' USING INLINE-PASS-AREA.
* IF INLINE-GOOD-RETURN
* MOVE INLINE-DOW TO USER-DOW.
*-------------------------------------------------------
* CA(S) COBOL (U006) DAY OF WEEK   ONP = DAY #
* D.L.G. TESTING COMPLETED 1998/05/03 15:26:03
*-------------------------------------------------------
MOVE 'U006'   TO TRC-CONVR-FUNCTION-CODE
MOVE '1234567' TO TRC-CONVR-DOW-STRING
MOVE 'SLIDE-20' TO TRC-CONVR-CENTURY-BREAK
MOVE 'CCYYMMDD' TO TRC-CONVR-OUT1-DATE-MASK
MOVE 'Y' TO TRC-CONVR-VALIDATE-NULL
MOVE 'YYMMDD--' TO TRC-CONVR-FROM-DATE-MASK
MOVE ?-USER-FROM-DATE
 TO TRC-CONVR-FROM-DATE-X
PERFORM TRC-CALENDAR-ROUTINES
IF TRC-CONVR-RETURN-GOOD
 MOVE TRC-CONVR-OUT1-DATE-E
TO ?-USER-OUT1
MOVE TRC-CONVR-OUT-NUMERIC-PARM
TO ?-USER-OUT-NUMERIC
ELSE
PERFORM TRC-ERROR-ROUTINE
   END-IF *
*-------------------------------------------------*
* END OF GENERATED CA(E) CODE  *
*-------------------------------------------------*
   . . . . . .
ALL-DONE.
GOBACK. /
   COPY TRCCOMNP.
/
   COPY TRCSMLPD.
/
   COPY TRCERROR.
/
TRC-STANDARD-CALL.
   MOVE TRC-RTNCD-I-BAD-ROUTINE-FUNC TO
TRC-CONVR-RETURN-CODE
TRC-INTRL-RETURN-CODE.
```

The following example shows an extraction from Working Storage copybook member TRCSMLWS required when processing PERFORMed “Limited” functions:

```
*****************************************************************
*
*
*   TRCSMLWS - IN-LINE ‘PERFORM’ WORKING STORAGE *
*
*****************************************************************
01 TRC-VERSION-ID  PIC X(04) VALUE ‘6.00’.
01 TRC-CUSTOMER-ID PIC X(10) VALUE ‘ABABABABAB’.
*
01 TRC-INLINE-TRANSACTION-FLAG PIC X(01) VALUE ‘N’.
88 TRC-INLINE-TRANSACTION
VALUE ‘Y’.
```

The following example shows an extraction from Working Storage copybook member TRCCOMNW used by CA Calendar Routines common date functions:

```
*****************************************
*
*   TRCCOMNW - COMMON WORK AREAS   *
*****************************************
*****************************************
*
01 TRC-COMNW-ALL-AREAS.
*
*****************************************
*
*   ABSDN - USED WITH 801200-TRC-ABSDN
*
*****************************************
*
05 TRC-ABSDN-ABS-IN  PIC S9(8) COMP SYNC.
*
05 TRC-ABSDN-ABS-OUT  PIC S9(8) COMP SYNC.
```

The following example shows an extraction from Procedure Division statements of common PERFORMed CA Calendar Routines date functions:

```
* ABTJL - CONVERT ABSOLUTE DAYS TO JULIAN DATE  *
*************************************************
*************************************************** * *
 800100-TRC-ABTJL.
*
MOVE TRC-ABTJL-ABS-IN
  TO TRC-ABTJL-ABS-WORK.
*
*
 MOVE ZEROS
TO TRC-ABTJL-TOTAL-PREV-YRS.
*
DIVIDE  TRC-ABTJL-ABS-WORK
BY 1168775
GIVING  TRC-ABTJL-YRS-DIV-BY-3200
```

The following example shows an extraction from Procedure Division statements of “Limited” PERFORMed CA Calendar Routines date functions:

```
 TRC-CALENDAR-ROUTINES.
 PERFORM 020000-CHECK-CACHE.
IF TRC-CACHE-RESULT-NOT-FOUND
PERFORM TRC-DETERMINE-FUNCTION-TYPE
MOVE TRC-CACHE-ENTRY-2-12
TO TRC-CACHE-AREA-TEMP
MOVE TRC-CACHE-AREA-TEMP
TO TRC-CACHE-AREA-DEF-1
MOVE TRC-CONVR-CONVERSATIONAL
TO TRC-CACHE-ENTRY (12).
 TRC-DETERMINE-FUNCTION-TYPE.
*
MOVE 'N' TO TRC-INLINE-TRANSACTION-FLAG.
*
MOVE TRC-CONVR-FUNCTION-CODE TO TRC-INTRL-FUNCTION-CODE.
 IF TRC-INTRL-FUNC-NBR-BRIDGE
PERFORM TRC-VALIDATE-BRIDGE-TRANS
ELSE
IF TRC-INTRL-FUNC-NBR-INLINE
PERFORM TRC-VALIDATE-INLINE-TRANS
ELSE  NEXT SENTENCE.
*
IF TRC-INLINE-TRANSACTION
PERFORM TRC-CALENDAR-ROUTINES-SMALL
ELSE  PERFORM TRC-STANDARD-CALL.
```

Index