PREFACE:

EZ2000 comprises a number of components used in the remediation and testing of CA-IDMS applications for Y2K compliance. Details relating to the individual components may be found in the specific product user guides.

RELATED DOCUMENTATION:

EZ-DB Overview
EZ-DB Integrated **I**nstallation Guide
EZ-DB Information and Error Messages

EZ2000 EZ-Test Date User Guide
EZ2000 EZ-Test **I**DMS User Guide
` **EZ2000** EZ-**I**DMSAGER User Guide
EZ2000 EZ-DATAAGER User Guide
EZ2000 EZ-Bridge User Guide
EZ2000 EZ-WatchGuard User Guide

TABLE OF CONTENTS

FIGURES

Features Overview

- **EZ-Test IDMS**

Virtual date aging for CA-IDMS. Dynamically ages CA-IDMS databases at run-time. Allows users to perform Y2K application validation against existing CA-IDMS databases without the need to physically age the data. Eligible dates are passed to application programs as if they were aged without changing the underlying data. Conduct tests using different *simulated* or *Assumed RunDates* without having to create multiple test databases. Flexible aging criteria may be defined, with all possible date formats supported.

- **EZ-Test Date**

Simulate a different date and time for particular *users* without having to IPL the machine. Supports Batch, CICS and CA-IDMS. Support available within each environment to set date/time for particular entities. For example, CICS terminal, CA-IDMS DBNAME, terminal, user, etc. etc. Fully integrated with EZ-Test IDMS allowing users to execute with *Assumed RunDate* against existing CA-IDMS databases dynamically aged in step with the *Assumed RunDate*.

- **EZ-IDMSAGER**

Physical ager for CA-IDMS. Create physically aged CA-IDMS databases. Updates an existing CA-IDMS database in place, incrementing all eligible date fields according to same flexible rules as used in EZ-Test IDMS. All possible date formats supported.

- **EZ-DATAAGER**

Physical ager for sequential files. Create physically aged sequential files. Updates a sequential file in place, incrementing all eligible date fields according to same flexible rules as used in EZ-Test IDMS. All possible date formats supported.

- **EZ-Bridge**

Using EZ-Bridge *windowing*, there is no need to restructure the database or to encode existing data to incorporate a century indicator. EZ-Bridge will automatically make dates compliant without changing the the database definition - *even in date formats where there is no space for a century indicator.*

EZ-Bridge can also be used to ***avoid mass recompilation*** of programs if you choose date field expansion as your Y2000 solution.

- **EZ-WatchGuard®**

While EZ-Test IDMS and EZ-Bridge provide the most comprehensive tools available for CA-IDMS *Year 2000* testing and rollout, EZ-WatchGuard can provide the *added reassurance* of a run-time component that will ALERT your system administrator if *non-compliant* dates are retrieved from the database, and PHYSICALLY PREVENT programs from storing non-compliant dates on the database.

New Features in release 5.2

This section summarizes changes introduced in release 5.2 (C80101) compared with previous releases.

- **Default CTLTAB name processing.**

If there is no CTLTAB= entry in the //DB2KCTL control file, then the following default names are used depending on the current environment:-

CA-IDMS:	CTLIDMS
CICS:	CTLCICS
Batch:	CTLBTCH

If the appropriate table name is not available for load, then EZ-Test IDMS will attempt to load the default module, DB2KCTL . If that fails also then EZTest is disabled.

This means that different CTLTABs may be generated for IDMS, CICS and Batch without the need to input the table names via //DB2KCTL.

- **New CTLTAB CLASS=SYSTEM**

A new CTLTAB CLASS=SYSTEM has been added. If this is the first parameter in the #CTLTAB then it applies to all users of the system that are not excluded by #EXCTAB. i.e., they use the RUNDATE and BASEDAT in the associated ENTRY=DBEZ2000 .

- **EZ-Bridge DFCAPP Support**

The SDETAB now allows specification of *application* date format code (DFCAPP) in addition to the *schema* date format code (DFC) available in previous releases. DFCAPP may be different to DFC. Support for this feature requires the EZ-Bridge product. Refer to the EZ-Bridge User Guide for further information.

- **Table Loads**

Any tables mentioned in //DB2KCTL are loaded first. Any tables not specified in //DB2KCTL are loaded using the default table names for CTLTAB listed above, DB2KEXE (for CICS DB2KCEX), DB2KDFC and DB2KMON. If CTLTAB contains a BRGTAB table name then that is loaded. All SRLTAB names coded in the CTLTAB(s) are loaded. All SEOTAB names in the SRLTABs are loaded.

The use of //DB2KCTL to list table names should now be pretty much redundant except to list tables not referenced elsewhere and not to be used at run time but required to be viewed via DB2K displays.

- **Pivot Year**

Years containing YY only are deemed to be 20^{th} century if they are not lower than the pivot year, if they are lower then they are deemed to be 21^{st} century. Pivot Year is entered into the system in the first #SDETAB entry PIVOT=nn and can be overridden at element level, the default is 40.

- **Fiscal Dates**

The software attempts to load FISCUXIT at initialization time. If successful, then this module is called before and after date conversion for any fiscal dates. It is up to the user to ensure that FISCUXIT contains the appropriate code to deal with the date. Other user exits are not affected, they are called as normal. The source is supplied in the install libraries.

- **Business Dates**

A new parameter BIZDAY=YES coded in any #SDETAB entry will indicate that this date has to fall on a business day. We calculate the number of business days between RUNDATE and BASEDAT and add that number to all occurrences of that date to ensure that the new date always falls on a business day.

- **EZ-Test IDMS Shutdown Report**

EZ-Test IDMS will produce a shutdown report providing counts by record element of the number of times aged and/or bridged.

- **EZ-Test IDMS Date Validation Errors**

The maximum number of date validation errors reported for a particular date element is now controlled by MAXERR in #SDETAB, the default is 10.

- **EZ-IDMSAGER**

The CA-IDMS physical data ager has been significantly enhanced with this release. Refer to the *EZ-IDMSAGER user guide* for further details.

This section summarizes changes introduced in release 5.2 (C81001) compared with previous releases.

- **Enhanced SVC Hook**

The SVC Hook component of EZ-Test DATE has been significantly enhanced. Refer to the *EZ-Test DATE user guide* for further details.

- **#DFCTAB**

Wild cards may be included in DFC codes using **?** as a wild card. The character in this position will not be affected by aging or bridging.

Non contiguous century indicators are supported. A new parameter **CCOFF=nn** where nn is the displacement from the start of the date to the start of the century indicator will allow aging and bridging of dates where the century indicator has been added to a filler field.

Signed zoned decimal dates are supported. Generate a new DFC entry for each type e.g.,

#DFCTAB PIC=DISPLAY,FORMAT=YYMMDDS.

The S indicates this DISPLAY field is signed.

- **#CTLTAB**

Use RUNDATE=TODAY to set the assumed run date to system start up date.

- **#SRLTAB**

New parameters have been added for SQL support.

- **Fiscal Dates**

CE801.130 in install PTFs provides a sample user exit to support fiscal dates in the format YYYYWW where WW is the week in the year. This can easily be adapted to cater for other fiscal date formats.

Customizing the EZ2000 environment

1 Customizing the EZ2000 environment

1.1 EZ2000 generation overview

This section contains information relevant to **EZ2000** product family including **EZ-Test IDMS**, **EZ-Bridge** and **EZ-WatchGuard**. There is much overlap between these products and in many cases a user of one of these products will also be a user of at least one of the others. For that reason, all of the customisation information is contained herein, and included in the User Guides for each of these products. You should refer to the Product Overview of the particular product you are installing or customizing to determine which of these tables is relevant to that product.

All tables are generated using the supplied assembler macros described in this section. It is not necessary to generate *all* of the tables described here. Some of the tables are supplied pre-generated with default values. You only need to re-generate these tables if you wish to override the supplied defaults. Other tables, namely the CTLTAB and SRLTAB are site dependant and *must* be generated in all cases.

A brief description of each table follows:-

EXCTAB	**Task Exclusion table.** This table defines a list of CA-IDMS or CICS task codes which are to be excluded from EZ2000 processing. Generate this table only if you want to add or remove entries.
MONTAB	**Month Abbreviation table.** This table defines the standard month name abbreviations for alphabetic months in dates. eg. JAN, FEB, MAR etc. Generate this table only if you wish to change the definitions, e.g., for foreign language support.
DFCTAB	**Date Format Code Table.** This table defines all of the expected internal date representations eg. MMDDYY CHAR, 0CYYDDD PACKED etc. The pre-defined table contains an extensive list of possible formats. Generate this table only if your site stores dates in format(s) *not* included in the supplied table.
CTLTAB	**EZ2000 Control Table.** This table is used to activate EZ2000 for particular users/entities and to define the applicable *Assumed RunDate/Time*. Each site must generate this table for EZ2000 to be activated as the definitions are site specific. As you will see later, it is possible to have multiple tables, and also to override the definitions dynamically using online functions.
SRLTAB	**Schema Record Date Element Definition Table.** The SRLTAB is used at run-time by the EZ2000 IDMS component to identify the location and format of date fields contained in IDMS records. Each site must generate this table as the contents are site specific.

SEOTAB

Schema Record Date Element Override Table. This table is used if you wish to override the default EZ-Test IDMS date manipulation for particular elements. (The default action is for all SRLTAB defined dates to be incremented by the *date difference*, where *date difference* is *Assumed RunDate* - *Base Date*).

BIZTAB

Business Holiday Table. This table is used to define weekends and business holidays. It is required by EZ-Test IDMS when aging dates which must always fall on a business day (See BIZDAY=YES parameter). Sample tables are provided containing definitions for certain countries.

BRGTAB

Bridge Table. The BRGTAB is used by EZ-Bridge to identify the load-libraries and dictionary load-areas which contain programs to which the *schema* view applies, and the load-libraries and dictionary load-areas which contain programs to which the *application* view applies.

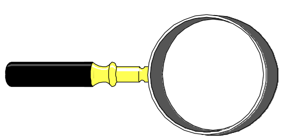
Each of the above tables is now described in detail.

1.2 Generating the EZ2000 Control Table (CTLTAB)

The CTLTAB is the common control point for EZ-Test, EZ-Bridge and EZ-WatchGuard which defines which entities (users, dbnames, programs etc.) are eligible for processing. There are different parameters which may be required depending on which components are installed. For example, for EZ-Test, you may specify the Assumed Rundate required by particular entities.

There are various *environments* for which you can specify *Assumed RunDates* using this table. For example, Batch, CICS and IDMS. Within each *environment*, you can further qualify the *Assumed RunDates* to apply to specific *classes*. For example, within the IDMS environment, you can specify different dates which apply to specific *users*, *dbnames*, *tasks*, etc., etc.

You can specify the applicable dates for multiple environments and classes within a single table. You can also, if you wish, generate multiple tables with different definitions for different environments. Default table names apply in each environment. Also, the name of the particular table to be used at run-time may be specified through the DB2KCTL parameter file (as described in **section 2.1.4**). There is also the ability to modify the active table *dynamically* using the supplied DC/UCF DB2K online function (as described in **section 2.2.3**)



In practise there will be several CTLTAB's in use at the same time. For example, the DB2KSET utility will load a CTLTAB into MVS CSA for use by the SVC HOOK mechanism. The default table batch table loaded is called CTLBTCH. The EZ2000 IDMS initialisation process will load a CTLTAB into the IDMS address space for use by the EZ2000 IDMS component. The default CA-IDMS table is called CTLIDMS. Similarly, the CICS initialization process will load a table (default) CTLCICS. All of these table names may be overridden if required.

The CTLTAB is generated using the #CTLTAB macro. The following illustrates the four classes of #CTLTAB macros which are combined to generate a CTLTAB.

```
#CTLTAB      ENTRY=FIRST,<first parameters>
#CTLTAB      ENTRY=DBEZ2000,<EZ2000 run-date parameters>
```

```
#CTLTAB      SRLTAB=<srltab-2 name>        (optional)
#CTLTAB      CLASS=<class parameters>
#CTLTAB      ENTRY=LAST
```

To generate a valid #CTLTAB you must first code a #CTLTAB ENTRY=FIRST statement with applicable parameters. The ENTRY=FIRST parameters define the table name, and define the *environment* to which the following definitions apply, eg., Batch, CICS, IDMS or ALL. Following the ENTRY=FIRST statement, you then include one or more ENTRY=DBEZ2000 statements which define a particular *Assumed RunDate*. Each ENTRY=DBEZ2000 statement *may* be followed by one or more SRLTAB statements (only if the ENTRY requires more than one SRLTAB), followed by one or more ENTRY=CLASS statements that further define the particular *classes* to which the current *Assumed RunDate* applies, eg., userid, dbname etc. The table definition is terminated with a #CTLTAB ENTRY=LAST statement.

The following figure (**Figure 1**) shows the full syntax of the #CTLTAB ENTRY=FIRST statement.

```
tabname   #CTLTAB ENTRY=FIRST,
              TYPE=IDMS,                        ( IDMS|CICS|BATCH|DB2|ALL )
              XTASK=YES,                        ( YES | NO )
              SVCMODE=KEYZERO,                  ( KEYZERO | KEYUSER )
              BRGTAB=                           ( BRGTAB name )
              ERUSID=C0,                        ( SVCEXTN Identifier )
              ERUSJOB=1                         ( Offset to jobname in SVCEXTN )
```

Figure 1 #CTLTAB ENTRY=FIRST Syntax

Each of the available parameters is described below:-

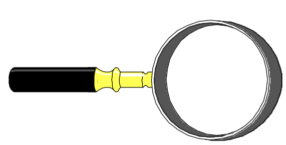
tabname: The name of the table may be specified by including a 1-8 character label in columns 1-8 of the #CTLTAB ENTRY=FIRST macro. If the macro is unlabelled, the default name of DB2KCTL will be generated.

Note: This name should also be the link-edit name of the generated table.

Default table names apply in each environment. These are:-

CTLBTCH	Batch
CTLIDMS	CA-IDMS
CTLCICS	CICS
CTLDB2	DB2

The initialisation process in each environment will look first for the default table name as applicable, and if not found, for a table name of DB2KCTL. If neither table is found, EZ-Test IDMS is disabled in that environment.



The CTLTAB to be loaded can also be overridden by specifying the //DB2KCTL run-time parameter CTLTAB=*tab-name*.

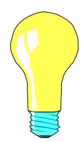
Once a CTLTAB has been loaded it is

checked for SRLTABs and each of those is loaded. Then each SRLTAB is checked and any SEOTABs specified are also loaded. It is still possible to specify all table names through the //DB2KCTL run-time parameters if desired, but it should now be possible to run all environments using the new default CTLTAB names and not have to specify any //DB2KCTL parameters anywhere regardless of how many SRLTAB or SEOTABs are required by CA-IDMS or local mode batch.

TYPE: This parameter defines the environment to which this CTLTAB applies. Valid values are:-

IDMS
CICS
BATCH
DB2
ALL

The default value is BATCH.



Note that it is possible to generate multiple CTLTABs for use in different environments, or a generic CTLTAB for use in *all* environments by specifying TYPE=ALL. At initialisation time, EZ2000 will build a run-time CTLTAB selecting only the relevant definitions for the applicable environment.

XTASK: Specify whether you wish to use a Task Exclusion List (EXCTAB). Refer to section 1.4 for further details about the purpose of the EXCTAB. Default value is YES.

SVCMODE: This parameter is used by the EZ-Test DATE SVC Hook to define whether date simulation should be enabled for all requests (supervisor and problem state) or problem state requests only. The parameter may have the following values:

KEYZERO: ALL requests on behalf of eligible programs will receive the simulated date and time, that is, irrespective of whether the requester is in supervisor or problem state.

This is the default value for this parameter.

KEYUSER Requests on behalf of eligible programs will receive the simulated date and time, only if the requester is in problem state (key nonzero)

This parameter defines the default SVCMODE for all subsequent CLASS=JOB entries in this CTLTAB. This default value may be overridden for particular CLASS=JOB entries by including the SVCMODE parameter on the applicable DBEZ2000 statement.

The SVCMODE may also be overridden at run time using through in-stream JCL parameter specification.

BRGTAB Specify the name of the BRGTAB to be loaded at system initialisation time.

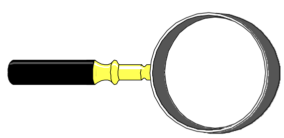
Note**:** The name may also be overridden by the BRGTAB= parameter in the //DB2KCTL run-time parameter file. If not specified there, the BRGTAB specified in the CTLTAB is used.

ERUSID: This parameter specifies the 2-character identifier of the SVC extension to be used by EZ2000 for access to the ERUS jobname information.

The default value is C0, which is the ID of the SVCEXTN created by PERFMON. Therefore, if your site runs CA-PERFMON, there is no need to specify this parameter.

Note: To use the PERFMON SVC extension, the CA-IDMS SVC must have been generated with the option PERFMON=YES.

If you are not a PERFMON user, or you wish to use USER specific SVC extension, specify ERUSID=UA which is the required identifier for a User SVC extension.



This parameter is required for CA-IDMS central version environments only, and where you have specified a CLASS=ERUS in the CTLTAB definition.

ERUSJOB: This parameter defines the offset of the jobname in the SVC Extension. The default value is 1, which is the applicable offset if you are using the PERFMON extension. If you are using a USER SVC extension, specify the offset which applies in your case.

The following figure (**Figure 2**) shows the full syntax of the #CTLTAB ENTRY=DBEZ2000 statement.

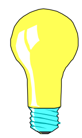
```
      #CTLTAB ENTRY=DBEZ2000,
         SRLTAB=DB2KSRL,          ( SRLTAB Name )
         RUNDATE=YYYYMMDD,        ( Assumed RunDate )
         BASEDAT=,                ( Override for IDMS startup date)
         BASETIM=,                ( Override for IDMS startup time)
         ACTION=INCLUDE           ( INCLUDE | EXCLUDE )
```

Figure 2 #CTLTAB ENTRY=DBEZ2000 Syntax

Each of the available parameters is described below:-

Note: If this is not the first ENTRY=EZ2000 statement and any of the following parameters are omitted, then the previous value specified for this parameter will be used as the default.

SRLTAB: Specify the name of the SRLTAB to which this ENTRY relates. In practise, you will generate an SRLTAB per schema/version. At execution time EZ2000 will use the SRLTAB that applies to the selected CTLTAB entry. The Name may be 1-8 alphanumeric characters. The default name is DB2KSRL, or the last specified value.



There may be a requirement for a particular ENTRY to reference multiple SRLTABs, for example, if a DBNAME includes SEGMENTS from different SCHEMAS. Each SRLTAB is for one schema version only. To include the others, code as many #CTLTAB SRLTAB= statements as required, immediately following this #CTLTAB statement.

RUNDATE: Specify the *Assumed RunDate* (simulated date) to be used for the CLASSES defined in this entry. The date must be specified in format YYYYMMDD. If omitted, defaults to last specified value.

eg. RUNDATE=19991231

You may also specify the special reserved word TODAY

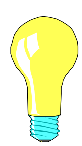
eg RUNDATE=TODAY

This would be used when date simulation and/or data aging is not required, but you are using the CTLTAB to activate EZ-Bridge *windowing* and/or EZ-WatchGuard, *run-time monitoring.*

BASEDAT: This is an optional parameter which allows you to override the IDMS startup date. Specify the startup date in the format YYYYMMDD.

eg. BASEDATE=19990430

BASETIM: This is an optional parameter which allows you to override the IDMS startup time or batch job start time. Specify the startup time in the format HHMM.



The parameters BASEDAT and BASETIM can be used in conjunction with the RUNDATE parameter to affect the way that EZ2000 adjusts the dates in database records. If BASEDAT is *not* specified, EZ2000 calculates the *difference* between the specified RUNDATE and the *actual* date. This *difference* is the number of days added to the applicable date fields in database records. If BASEDAT *is* specified, EZ2000 calculates the *difference* between the required RUNDATE and the *BASEDAT* and uses this difference. Therefore, by using BASEDAT, you can cause EZ2000 to always add the same number of days when adjusting dates, and thereby maintain a consistent view of the test database whatever the *actual* date and *Assumed RunDate*.

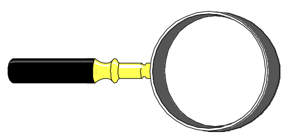
There are various situations where this may prove a useful facility for EZ2000 users. Refer to the Technical Notes section (**section 3.1**) for further discussion about this subject.

ACTION: Specifies whether the subsequent CLASS statements related to this EZ2000 entry are to be included or excluded by EZ2000 for date simulation.

INCLUDE: The RUNDATE specified in this EZ2000 entry applies to the entities as defined by the following CLASS statements.

EXCLUDE: The RUNDATE specified in this EZ2000 entry applies to everything except the entities defined by the following CLASS statements.

This parameter defaults to INCLUDE or the last specified value.



ACTION=EXCLUDE is ignored for CLASS=JOB entries. All CLASS=JOB entries are treated as INCLUDE by the SVC HOOK.

The following figure (**Figure 3**) shows the full syntax of the #CTLTAB SRLTAB= statement.

```
      #CTLTAB SRLTAB=                         ( SRLTAB name )
```

Figure 3 #CTLTAB SRLTAB= Syntax

The available parameter is described below:-

SRLTAB Specify the name of a second or subsequent SRLTAB which may be used by the current #CTLTAB ENTRY=DBEZ2000 statement..

This is an optional statement, and is only required if more than one SRLTAB is required. The name of the first SRLTAB is specified on the #CTLTAB ENTRY=DBEZ2000 statement SRLTAB= parameter.

The following figure (**Figure 4**) shows the full syntax of the #CTLTAB CLASS= statement.

```
      #CTLTAB OR,                          ( AND | OR )
         CLASS=,                           ( CLASS Name )
         NAME=,                            ( entity Name )
```

Figure 4 #CTLTAB CLASS= Syntax

Each of the available parameters is described below:-

AND|OR: This is an optional parameter. As a #CTLTAB ENTRY=DBEZ2000 entry can be followed by one or more CLASS statements, this parameter is used to define whether EZ2000 should treat subsequent CLASS statements as AND or OR conditions. If omitted, defaults to AND. This is best illustrated by example:

```
#CTLTAB CLASS=DBNAME,
      NAME=SYSDICT
#CTLTAB OR,CLASS=USER
      NAME=SOGRB01
#CTLTAB OR,CLASS=USER
      NAME=SOPGC01
```

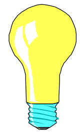
The above example specifies that the current EZ2000 definition applies if the DBNAME is SYSDICT, AND the user is SOGRB01 OR SOPGC01.

CLASS: Specify the CLASS. Certain CLASSES are valid in particular environments only. The following table indicates the valid CLASSES and the environments to which they apply.

	IDMS	**CICS**	**BATCH**
SYSTEM	✔	✔	
USER	✔		
LTERM	✔	✔	
PROGRAM	✔		✔
TASK	✔		
JOB			✔
ERUS	✔		
DBNAME	✔	✔	
FILE			✔
PROGVER	✔		
SUBSCHEMA	✔		
DIALOG	✔		
NODE	✔		

If the CTLTAB contains CLASS statements which do not apply in the current environment, they are simply ignored.

CLASS=SYSTEM is the only CLASS which does not require a NAME. If CLASS=SYSTEM is the first parameter in the #CTLTAB then it applies to all users of the system that are not excluded by #EXCTAB.



You could use CLASS=SYSTEM, for example, to automatically define the RUNDATE for all CICS users at CICS system startup - instead of having to issue DB2K SET transaction.

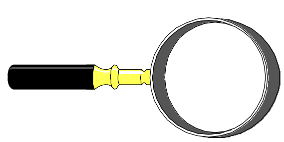
The following notes apply to particular CLASSes in different environments:

CA-IDMS

TASK	In the IDMS environment, this will be matched against either a DC/UCF task code, or in the case of ADS/O applications, the ADS application name.
PROGRAM	In the IDMS environment, this will matched against either:- • a DC/UCF program name. • in the case of ADS/O, the ADS/O DIALOG name. • SSCPNAME - that is the name specified on the Bind Run Unit.
ERUS	In the IDMS CV environment, this will be matched against the jobname of the external run-unit. Refer to the ERUSID parameter of the ENTRY=FIRST statement if you wish to use this option.

BATCH

JOB	In the BATCH environment, this CLASS is matched by the SVC HOOK against the JOB or STC name. All CTLTAB CLASS=JOB entries are treated as TYPE=INCLUDE.
PROGRAM	In the BATCH environment you may specify ACTION=EXCLUDE for CLASS=PROGRAM entries. These entries are matched against the job-step program name, thereby allowing the user to exclude certain steps within a job from date simulation.



The SVC HOOK checks for any EXCLUDE PROGRAM entries before checking for a matching CLASS=JOB entry. The position of the EXCLUDE entries in the CTLTAB is therefore not important.

FILE	This CLASS is used by the DATAAGER only. Refer to *EZ-DATAAGER User*

Guide for further details.

CICS

SYSTEM	CLASS=SYSTEM is used in the CICS environment to set the environment for all users.
DBNAME	In the CICS environment the user may issue the

```
DB2K SET <SYSTEM> DBNAME=
```

command to set the Assumed Rundate etc, to the predefined CTLTAB CLASS=DBNAME entry.

Refer to **section 2.3** for further details.

NAME: Every CLASS statement (except CLASS=SYSTEM) must contain a NAME clause, which explicitly defines the CLASS. The name may be alphanumeric characters of varying length depending on the context, ie. CLASS to which it applies. For example, a JOBNAME is usually 8 characters. A TERMINAL name in IDMS is usually 8 characters. The name of a class type may be abbreviated to specify groups of names all beginning with a particular prefix by including the *catch-all* character * (asterix).

For example, NAME=ABC* - all names beginning ABC.

Following are some examples that illustrate some typical CLASS statements:

```
#CTLTAB CLASS=USER,NAME=SOXGB01
```

The preceding statement will trigger EZ2000 processing for user name SOXGB01.

```
#CTLTAB CLASS=DBNAME,NAME=DB991231
```

The preceding statement will trigger EZ2000 processing for DBNAME DB991231.

```
#CTLTAB CLASS=JOB,NAME=ASOX*
```

The preceding statement will trigger EZ2000 processing for any batch job name commencing with the characters ASOX.

Remember that a single #CTLTAB ENTRY=DBEZ2000 statement may be followed by multiple #CTLTAB CLASS statements to provide flexible trigger conditions for EZ2000 to take effect. Refer to the discussion of the AND | OR clause above.

Furthermore, a single CTLTAB may contain more than one #CTLTAB ENTRY=DBEZ2000 definition, each containing its own selection (trigger) conditions. At run-time, EZ2000 will process the CTLTAB sequentially, and use the first DBEZ2000 entry that applies.

```
       #CTLTAB ENTRY=LAST
```

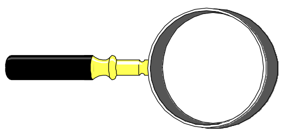
Figure 5 #CTLTAB ENTRY=LAST Syntax

Figure 5 shows the full syntax of the #CTLTAB ENTRY=LAST macro. This statement must be coded as illustrated to signify the end of the CTLTAB.

1.3 Generating the Schema Record Date Element Definitions (SRLTAB)

EZ-Test IDMS - The EZ-Test IDMS component is responsible for shifting dates contained in IDMS records backwards and forwards as applicable when moving records between program variable storage and the DMCL buffer. For this purpose, it needs to know the IDMS record elements which are dates and which need shifting. To facilitate this at run-time, EZ-Test IDMS uses an SRLTAB which defines the applicable elements and their internal format (picture) and location within the relevant IDMS records.

EZ-Bridge - With the introduction of EZ-Bridge, the SRLTAB is also used to identify the record-elements which have a different date format definition within the application and schema view of that record element, and thus require *bridging*, and/or require *windowing*.



In many cases, these record-elements will also be fields which are being *aged* - in which case they will already have SRLTAB definitions. However, it is also possible that fields which are not being aged (which would not previously have required SRLTAB definitions) will now require an SRLTAB definition to support *bridging* and/or *windowing*.

EZ-WatchGuard - The EZ-WatchGuard component is responsible for monitoring dates retrieved and stored on a CA-IDMS database, and enuring that the date format complies with the expected date format. For this purpose the SRLTAB is used to identify the record/elements to be *watched*, and the expected date format and other rules eg, Business day.

Since an SRLTAB relates to a particular CA-IDMS schema and version, in practise you may generate multiple SRLTAB's. The name of the particular SRLTAB(s) to use at run-time are specified in the CTLTAB and may also be specified via the SRLTAB= run-time parameter. (See **section 2.1.4** for details of specifying run-time parameters)

Sample JCL for the various steps involved in the generation of an SRLTAB is included in the installation source library and documented in the appendix of this user guide. The process consists of the following steps:-

- Generate an SDETAB (sample JCL in DB2KSDE)

  This job step generates an SDETAB using the user specified #SDETAB macro statements.

  Sample JCL to generate an SDETAB may be found in the installation source library member DB2KSDE.

- Generate SRLTAB from SDETAB (sample JCL in DB2KSRL)

  Sample JCL to generate an SRLTAB from an SDETAB may be found in the installation source library member DB2KSRL. This job consists of the following steps:

DB2KDICT	Reads the SDETAB and extracts date element information from the data dictionary. Writes necessary data to output file.
DB2KSORT	Sorts the output file generated by DB2KDICT step.
DB2KASMG	Reads sorted output and generates SRLTAB source
DB2KSRLA	Assemble and Link SRLTAB from generated source.
DB2KSRLK	Link-Edit SRLTAB

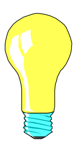
These steps are now described in detail.

1.3.1 Generate an SDETAB

The first step is to generate the SDETAB which is subsequently processed by the DB2KDICT utility. The SDETAB is generated using the #SDETAB macro. The following illustrates the three classes of #SDETAB macros which must be combined to generate an SDETAB.

```
#SDETAB    ENTRY=FIRST,<first options>
#SDETAB    EL=element-name,<element options>
#SDETAB    ENTRY=LAST
```

To generate a valid SDETAB you begin the definition with a #SDETAB ENTRY=FIRST statement. You then include an #SDETAB EL= statement for every date element that you require EZ2000 to manipulate. In practise, this is likely to be *every* element which is a used to store a date except for *fixed* dates which should not be moved *whatever* the Assumed RunDate. For example, *dates of birth* would probably *not* be adjusted. The table definition is terminated by a #SDETAB ENTRY=LAST statement.



The philosophy of EZ2000 makes the assumption that you have already identified all of the date fields in the database as part of the *impact analysis* phase of your Y2K project, so the generation of this list of elements should not be too arduous a task. Of course, if you discover through EZ2000 testing that certain elements have been omitted, then this in itself is a valuable part of the testing phase.

Figure 6 shows the full syntax of the #SDETAB ENTRY=FIRST macro.

```
#SDETAB ENTRY=FIRST,
    SRLNAME=                     ( SRLTAB Name )
    SCHEMA=                      ( Schema Name )
    VERSION=                     ( Schema Version )
    SEOTAB=                      ( SEOTAB name )
    FILE=                        ( File name )
    WINDOW=NO                    ( YES | NO )
    PIVOT=40                     ( Pivot Year )
    AGE=YES                      ( YES | NO )
    CENTIND=1                    ( 21st Century Indicator )
    MAXERR=10                    ( Maximum errors )
```

Figure 6 #SDETAB ENTRY=FIRST Syntax

Each of the available parameters is described below.

SRLNAME: Specify the name of the SRLTAB to be generated. The name may be 1-8 alphanumeric characters.

SCHEMA: Specify the name of the CA-IDMS schema to which this SRLTAB relates.

VERSION: Specify the Schema version number to which this SRLTAB relates. Defaults to one if not specified.

Note: The combination of Schema and Version must represent a valid CA-IDMS schema which exists on the data dictionary.

FILE Specify the name of the *non*-IDMS file to which this SRLTAB relates. Refer to *EZ-DATAAGER User Guide* for further details.

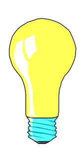
AGE: Specify whether EZ-Test IDMS is to perform *data aging* for the record-element(s) defined in this SRLTAB. This parameter defines the <u>default</u> action for this SRLTAB.

<u>YES</u> EZ-Test IDMS should age records-elements defined within this SRLTAB according to the applicable EZ-Test IDMS criteria.

NO Do not age record elements defined within this SRLTAB.

If omitted defaults to YES.

The parameter may be overridden for particular record-elements by specifying the AGE= parameter on the applicable SDETAB EL= statement.



If a user were running EZ-Test IDMS only, then the user would exclude a record elements from the aging process simply by <u>not defining</u> an SRLTAB entry for that element. However, when using both EZ-Bridge and EZ-Test, it is possible that a date field (for example a date of birth) that did not require aging, *does* require *bridging* or *windowing*. It is therefore necessary to generate an SRLTAB entry to specify the DFC and DFCAPP, but AGE=NO should also be specified to stop EZ-Test IDMS aging the field also.

WINDOW Specify whether EZ-Bridge Windowing is active for record-elements defined within this SRLTAB. This parameter defines the default action for this SRLTAB.

YES Windowing is required by default for all subsequent record-elements defined within this SRLTAB. EZ-Bridge will automatically encode dates to incorporate a century indicator (based upon the PIVOT year parameter) when storing/modifying data. Refer to DFC parameter and *EZ-Bridge User Guide* for further information.

NO Windowing is not required by default for subsequent record-elements defined within this SRLTAB.

If omitted, defaults to NO.

The parameter may be overridden for particular record-elements by specifying the WINDOW= parameter on the applicable SDETAB EL= statement

.PIVOT The PIVOT parameter is used to specify to EZ-Bridge when a two-digit year is to be considered a 20th century or 21st century date. This parameter defines the default PIVOT year for this SRLTAB. The default value is 40.

A 2-digit year less than PIVOT will be considered a 21st century date. A 2-digit year greater than PIVOT will be considered a 20th century date. This parameter is used in conjunction with the WINDOW parameter. For example, PIVOT=40 indicates that the years 00-39 should be considered as 2000-2039, whereas 40-99 should be considered as 1940-1999.

The parameter may be overridden for particular record-elements by specifying the PIVOT= parameter on the applicable SDETAB EL= statement.

CENTIND: Specify the binary value which is used as a century indicator in date fields to indicate a 21st century date. The default value of this parameter is 1.

In most cases it will not be necessary to alter this value. The IBM standard is that a century indicator of zero indicates 20th century, one indicates 21st century etc. There are however rare instances where a site may wish to override this. For example, a user maintaining historical data dating back to the 19th century may use zero to indicate 19th century, one to indicate 20th century and two to indicate 21st century etc.

SEOTAB: Specify the name of the Schema Element Override table to be loaded if required. If an SEOTAB is to be used, the name should be specified here or through the //DB2KCTL SEOTAB= parameter. There is no default.

MAXERR: Specify the maximum number of errors that will be reported for any particular record-element. At run-time, EZ-Test IDMS will write an error message to the DB2KLOG for (or MVS log if DB2KLOG is not defined) whenever it cannot age a record-element due to an unexpected field content (ie. the data is incompatible with the specified Date Format Code for the record element).

The default value of this parameter is 10.

Figure 7 shows the full syntax of the #SDETAB EL= macro.

```
#SDETAB EL=                       ( element name)
    VERSION=                      ( element Version )
    RECNAME=                      ( Record Name )
    DFC=                          ( Date Format Code )
    DFCAPP=                       ( Application Date Format Code )
    YYMM=                         ( ONLY or omitted )
    VALIDATE=YES                  ( YES | NO )
    BIZDAY=                       ( YES | NO )
    PIVOT=40                      ( 21st century window )
    AGE=NO                        ( YES | NO )
    WINDOW=                       ( YES | NO )
    WATCH=NO                      ( YES | NO )
    ABORT=NO                      ( YES | NO )
    HIGHVAL=NO                    ( YES | NO )
    LOWVAL=NO                     ( YES | NO )
    FORCE=NO                      ( YES | NO )
    REPORT=NO                     ( YES | NO )
```

Figure 7 #SDETAB EL= Syntax

Each of the available parameters is described below:

EL: Specify the name of the IDMS schema date element to which this statement applies (within this schema/version). If the element name contains embedded spaces, then the name should be enclosed in quotes.

Note: Only include entries for base element names - not synonyms.

VERSION: Specify the version number of the schema element to which this statement applies. Defaults to ALL if not specified.

Note: The combination of element name and Version must represent a valid CA-IDMS record element definition which exists on the data dictionary for the current schema.

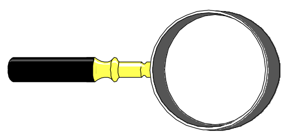
DFC: Specify the Date Format Definition for this record element. The Date Format Code Specified must be a valid Date Format Code

Equate as documented in the Date Format Code DSECT (#DFCDS). For every Date Format Code definition in the Date Format Code Table (DFCTAB), there is a corresponding equate in #DFCDS. You should specify the name of the equate which matches the Date Format used by this element.

EZ-Bridge Specific Information

Unless you are implementing WINDOWING, you should specify the name of the equate which matches the Date Format used by this element.

If you ARE implementing WINDOWING, then it is possible that you may define a DFC to include a century indicator. For example, the actual date format of the record-element on the database is P00YYDDD, but to implement WINDOWING you would specify DFC=P0CYYDDD.

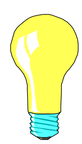


If you are using EZ-Bridge to implement WINDOWING only, i.e, there are no DFCAPP definitions in the SDETAB, then you must generate a dummy BRGTAB (DB2KBRG). Refer to **section 1.9.2** for further information.

end of EZ-Bridge specific information

Note: If you have date format representations which do not exist in the DFCTAB, then you will need to regenerate the DFCTAB to include the appropriate format(s), and add an equate for the code to #DFCDS.

A listing of the supplied default #DFCDS can be found in the appendix.



It is possible to cater for the situation where the same record-element may have *different* date formats within different element versions or different record types by coding multiple #SDETAB EL= statements for each possibility. Refer to the examples later in this section.

VALIDATE: Specify whether DB2KDICT should validate the Date Format Code definition against the data dictionary definition for the element. The default value is YES. Dependant on the value of this parameter the following actions will occur.

YES: DB2KDICT will validate the Date Format Codes against the dictionary definition. If a discrepancy occurs, the following message will be displayed, and no SRL table entry will be generated for that particular *record-element* occurrence.

DB905707 LENGTH/USAGE DOES NOT MATCH DFC TABLE *element-name*

As a particular element may be used in multiple record occurrences, one such message may be produced for each occurrence.

NO: No field validation is performed by DB2KDICT. Generate an SRLTAB entry for each *record-element* occurrence.

Examples:-

1. YY MM DD are defined separately and there is no group date element name. Then add an SDE entry for YY with DFC=NYYMMDD and VALIDATE=NO. Do <u>not</u> add SDE entries for MM and DD.

2. A text field length 20 contains (or may contain) a date in position 12. Add an SDE entry for the text field with VALIDATE=NO and in SEOTAB an entry for the text field specifying an exit which will adjust the actual date address and DFC Code as appropriate.

YYMM: If YYMM=ONLY is specified, then date adjustment occurs only for the Year and Month portion of the date, thus preserving the day within the month.

For example, if the calculated date difference between Assumed RunDate and Base Date is 3 years 2 months and 12 days, then *years* would be incremented by 3 and *months* would be incremented by 2, leaving the *day* within month intact. The day is then checked to see if it is still valid for the month, and if not, set to the last day of the month. Thus 29th Feb 2001 would become 28th Feb 2001.

This is a particularly useful feature if you wish to age data by an exact number of years. Under the default calculation, the date may be aged by the exact number of years plus or minus one day, depending on where leap year(s) fall relative to the dates.

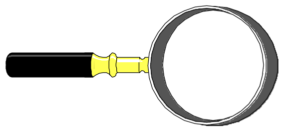
This parameter is omitted of default calculation is required.

RECNAME: Specify the CA-IDMS *record name* which applies to this particular *record-element* definition. This is an optional parameter and provides for greater flexibility in the specification of Date Format Codes. See examples later in this section.

Note: Only include entries for base record names - not synonyms.

BIZDAY:

If BIZDAY=YES is specified, then dates in this record element must fall on Business days only. The default for this parameter is NO.



When aging this field, EZ-Test IDMS will calculate the number of business days between RUNDATE and BASEDAT and add that number to ensure that the new date always falls on a business day. The definition of business holidays is contained in the DB2KBIZ table. Refer to **section 1.8** for further details.

DFCAPP:

Specify the *application* Date Format Definition for this record element. This parameter is only required if the application *may* have a different view of this record-element. If omitted, it is assumed that the definitions are the same.

EZ-Bridge will determine whether the current program is using the DFC or the DFCAPP for this record element depending on the load-library or dictionary LOADLIB from which the program was loaded. Refer to BRGTAB definition in *EZ-Bridge User Guide* for further details.

It is not necessary for the DFCAPP to represent a date of equal length to DFC. For example,

```
DFC=PYYYYDDD,DFCAPP=PYYDDD,
```

specifies that the element is stored on the database as a packed decimal JULIAN date - YYYYDDD (4 bytes) whereas the application program format is packed decimal JULIAN date - YYDDD (3 bytes). Because the Date Format Codes are different, EZ-Bridge will convert the date field as it is moved to and from the database. Any conversions are done in the application's working storage. If the length changes (which is the case in this example), any subsequent data is shifted forward or back as required.

The Date Format Code specified must be a valid Date Format Code equate as documented in the Date Format Code DSECT (#DFCDS). For every Date Format Code definition in the Date Format Code Table (DFCTAB), there is a corresponding equate in #DFCDS. You should specify the name of the equate which matches the Date Format used by this element.

Note: If you have date format representations which do not exist in the DFCTAB, then you will need to regenerate the DFCTAB to include the appropriate format(s), and add an equate for the code to #DFCDS.

Refer to **section 1.6** if you need to regenerate the DFCTAB.

A listing of the supplied default #DFCDS can be found in the appendix.

It is possible to cater for the situation where the same record-element may have *different* DFC and DFCAPP formats within different element versions or different record types by coding multiple #SDETAB EL= statements for each possibility.

Refer to the EZ-Test IDMS User Guide for further details.

AGE: Specify whether EZ-Test IDMS is to perform data aging for this record element.

YES EZ-Test IDMS should age this field according to the applicable EZ-Test IDMS criteria. This is the default value.

NO Do not age this record element.

If a user were running EZ-Test IDMS only, then the user would exclude a record elements from the aging process simply by <u>not defining</u> an SRLTAB entry for that element. However, when using both EZ-Bridge and EZ-Test, it is possible that a date field (for example a date of birth) that did not require aging, *does* require bridging. It is therefore necessary to generate an SRLTAB entry to specify the DFC and DFCAPP, but AGE=NO should also be specified to stop EZ-Test IDMS aging the field also.

Refer to the *EZ-Bridge User Guide* for further details.

PIVOT: The PIVOT parameter is used to specify to EZ-Bridge when a two-digit year is to be considered a 20^{th} century or 21^{st} century date. The default value is 40.

A 2-digit year less than PIVOT will be considered a 21^{st} century date. A 2-digit year greater than PIVOT will be considered a 20^{th} century date. This technique is also known as *windowing*. For example, PIVOT=40 indicates that the years 00-39 should be considered as 2000-2039, whereas 40-99 should be considered as 1940-1999. i.e., a date in program variable storage of 01/12/25 would be stored on the database as 2001/12/25.

Refer to the *EZ-Bridge User Guide* for further details.

WINDOW: Specify whether EZ-Bridge Windowing is active for this entry.

YES Windowing is required for this entry.

NO Windowing is not required for this entry.

If omitted, defaults to the value specified on the SDETAB ENTRY=FIRST statement.

WATCH: Specify whether run-time monitoring is active for this entry.

YES EZ-WatchGuard runtime monitoring is active.

NO EZ-WatchGuard run-time monitoring is inactive.

The default value is NO.

Refer to the *EZ-WatchGuard User Guide* for further details.

ABORT: Specify whether run-units should be prevented from storing non-compliant dates.

YES Stop programs storing non-compliant dates.

In this event, a STORE would get a 1299 return code, a MODIFY would get a 0899 return code. Note: To use ABORT=YES you must also have specified WATCH=YES.

NO If you specify ABORT=NO, then the storing of non-compliant dates is allowed, but (if you have specified WATCH=YES) a warning will be written to EZ-WatchGuard report and/or journal file.

The default value is YES.

Refer to the *EZ-WatchGuard User Guide* for further details.

HIGHVAL This parameter is used to specify whether high values (ie.,hex all FF's) are allowed for this field.

NO means that the field must always be a real and valid date.

YES High values are allowed.

The default value is NO.

Refer to the *EZ-WatchGuard User Guide* for further details.

LOWVAL This parameter is used to specify whether low values (ie.,hex zeroes) are allowed for this field.

NO means that the field must always be a real and valid date.

YES Low values are allowed.

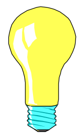
The default value is NO.

Refer to the *EZ-WatchGuard User Guide* for further details.

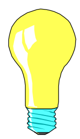
FORCE

Specify FORCE=YES to cause EZ-Bridge to set the century indicator (based upon PIVOT year) to dates that include a century indicator, but where the indicator is not set by the application. For example, a COMP-3 date YYMMDD is stored as X'CYYMMDDS' where C is the century and S is the sign. EZ-Bridge will set the century indicator based upon pivot year.

Thus dates stored by non-compliant applications will be made compliant.



In addition, any Y2K dates already on the database stored by non-compliant applications will have the century indicator set by EZ-Bridge when read from the database, and the next modify will result in the century indicator being written back to the database. Thus, existing non-compliant dates will gradually be made compliant, and transparently to the applications.



If it is required to specify FORCE=YES to ensure that dates stored are compliant, but the application is unable to process dates with the century indicator set, (i.e., the dates must be returned with a century indicator of zero), then specify two DFC codes. For example,

```
DFC=PCYYMMDD,DFCAPP=P0YYMMDD
```

The default is FORCE=NO.

Refer to the *EZ-Bridge User Guide* for further details.

REPORT

Specify REPORT=YES to cause EZ-Bridge to output details of record elements where the century indicator has been set by the FORCE=YES parameter to the DB2KLOG file.

The default is REPORT=NO.

```
#SDETAB ENTRY=LAST
```

Figure 8 #SDETAB ENTRY=LAST Syntax

Figure 8 shows the full syntax of the #SDETAB ENTRY=LAST macro. This statement must be coded as illustrated to signify the end of the SDETAB.

The following figure (**Figure 9**) shows an example of an SDE table definition.

```
      DB2KSDE  #SDETAB ENTRY=FIRST,SCHEMA=DB2KSCHM,SRLNAME=DB2KSRL
        #SDETAB EL=INVOICE-DATE,DFC=PCCYYDDD
        #SDETAB EL=ORDER-DATE,DFC=PYYYYDDD
        #SDETAB EL=TODAYS-DATE,DFC=PCYYMMDD
        #SDETAB EL=DISPLAY-DATE,DFC=AD@MMM@Y
           :
           :
           :
        #SDETAB ENTRY=LAST
```

Figure 9 #SDETAB example

Examples of Specifying different Date Format Codes.

The following examples illustrate the way different SDE tables may be generated to cater for a number of different requirements.

CASE 1 - A Date with a single format that occurs in *n* records.

```
#SDETAB EL=element-name,DFC=dfc-code,<DFCAPP=app-dfc-code>
```

In this case, DB2KDICT would generate SRLTAB entries for all occurrences of the date element within the schema version. If DB2KDICT detects more than one version of the element then a warning message is issued. If the PICTURE or LENGTH of any of the element versions conflicts with the specified DFC, then an error message is produced unless VALIDATE=NO is specified. The same DFCAPP (if specified) applies to all elements.

CASE 2 - A Date with two or more versions each with different PICTURES, but consistent within each date-element-version.

```
#SDETAB EL=element-name,VER=ver-1,DFC=dfc-code-1,<DFCAPP=app-dfccode-1>
#SDETAB EL=element-name,VER=ver-2,DFC=dfc-code-2,<DFCAPP=app-dfccode-2>
```

CASE 3 - A Date with two or more PICTURES within the same element version, but in different record types. For example, *application-A* stores it as YYYYMMDD in *Record-A*, while *application-B* stores it as DDMMYYYY in *Record-B*. *Application-A* and *Application-B* may in addition have the same or different DFCAPP definitions.

```
#SDETAB EL=element-name,DFC=dfc-code-1,RECNAME=record-1,<DFCAPP=app-dfccode-1>
#SDETAB EL=element-name,DFC=dfc-code-2,RECNAME=record-2,<DFCAPP=app-dfccode-2>
```

CASE 4 - A Date with two or more PICTURES within the same element version but in different record types, and a third PICTURE within another date-element-version. For example, *application-A* stores *version-1* of the date as YYYYMMDD in *record-A*, but *application-B* stores it as DDMMYYYY in *record-B*. Other applications store *version-2* of the date as YYYYDDD in other record types. Each definition may additionally specify the same or a different DFCAPP.

```
#SDETAB EL=element-name,VER=ver-1,DFC=dfc-code-1,RECNAME=record-1,<DFCAPP=...
#SDETAB EL=element-name,VER=ver-1,DFC=dfc-code-2,RECNAME=record-2,<DFCAPP=...
#SDETAB EL=element-name,VER=ver-2,DFC=dfc-code-3,<DFCAPP=...
```

Note that in this example, *element-version* is not mandatory in the first two statements, as *record-name* is specified (though it may be good practise to code it anyway). The third definition will cater for all occurrences of *element-ver-2* in any record type.

CASE 5 - A text string *may* or *may not* contain a date depending on other criteria.

```
#SDETAB EL=element-name,DFC=dfc-code,VALIDATE=NO,<DFCAPP=...
```

element-name is the name of the element containing text strings. DFC is the default Date Format Code but may be overridden at run-time.

```
#SEOTAB RECORD=record-name,VER=version
#SEOTAB ELEMENT=element-name,VER=version,EXIT=exitname
```

The user exit will be invoked at run-time to decide whether the element contains a date, its format and whether it is to be processed by EZ2000.

CASE 1-5 - If desired, every occurrence of all versions of an element could be specified as long as they all included VER= and RECNAME=. The Software is thus able to cater for those users with consistent dates that want to input the minimum amount of information, and also those with inconsistent date usage.

1.3.2 Executing the DB2KSRL job

A sample job stream is supplied in the installation source library called DB2KSRL which will generate an SRLTAB from a previously generated SDETAB. The job stream consists of the following job-steps.

DB2KDICT The DB2KDICT program is run to extract the required information from the IDMS data dictionary in order to generate the SRLTAB. DB2KDICT reads the SDETAB generated in the previous step and the IDMS data dictionary (IDD) to build control information required by EZ2000 for each applicable record element for each applicable record for the applicable schema.

DB2KDICT creates the output file DB2KDICT.

DB2KSORT The DB2KDICT output file is SORTED.

DB2KASMG The sorted extract file is read by DB2KASMG to generate the assembler statements required to build the SRLTAB.

DB2KSRLA Assemble generated assembler source.

DB2KSRLK Link-edit SRLTAB.

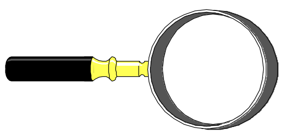
1.4 Generating the Task Exclusion Table (EXCTAB)

When EZ2000 is active on a CA-IDMS CV, there are various DC/UCF task codes that one would normally want to be excluded from date simulation whatever the CTLTAB definition. This is because the primary purpose of EZ2000 is for application testing. For example, if EZ2000 is simulating a date for a particular userid, that user might still prefer to see the real date (todays date) when issuing DCMT commands or using PERFMON. Also, EZ2000 does not adjust the date for certain internal task codes. To facilitate this, EZ2000 uses a control table called the TASK EXCLUSION TABLE (EXCTAB). The EXCTAB is generated using the #EXCTAB macro. The following illustrates the three classes of #EXCTAB macros which must be combined to generate an EXCTAB.

```
#EXCTAB ENTRY=FIRST,<first-parameters>
#EXCTAB ENTRY=TASK,<task-parameters>
#EXCTAB ENTRY=LAST
```

To generate a valid EXCTAB you begin the definition with a #EXCTAB ENTRY=FIRST statement. You then include an #EXCTAB ENTRY=TASK statement for every task code or task code mask that you wish to be excluded from date manipulation. The table definition is terminated by a #EXCTAB ENTRY=LAST statement.

The EZ2000 installation source library includes a member called DB2KEXC which is the job-stream used to generate the default EXCTAB. You only need to regenerate this table if you wish to add or remove entries from the default table. In this case, you can use the supplied source as a base from which to generate your own table(s), or you can generate your own EXCTAB from scratch. The name of the particular EXCTAB to use at run-time may be specified via the EXCTAB= run-time parameter. (See **section 2.1.4** for details of specifying run-time parameters).



There is also a supplied default EXCTAB which is used by the CICS component, called DB2KCEX.

Figure 10 shows the full syntax of the #EXCTAB ENTRY=FIRST macro.

```
tabname      #EXCTAB ENTRY=FIRST
```

Figure 10 #EXCTAB ENTRY=FIRST Syntax

Each of the available parameters is described below.

tabname: The name of the table may be specified by including a 1-8 character label in columns 1-8 of the #EXCTAB ENTRY=FIRST macro. If the macro is unlabelled, the default name of DB2KEXC will be generated.

Note: This name should also be the link-edit name of the generated table.

Figure 11 shows the full syntax of the #EXCTAB ENTRY=TASK macro.

```
       #EXCTAB ENTRY=TASK,
           NAME=                ( task name )
```

Figure 11 #EXCTAB ENTRY=TASK Syntax

Each of the available parameters is described below:

NAME: Specify the name of the IDMS DC/UCF task code to be excluded from EZ2000 date manipulation. Name may be 1-8 alphanumeric characters that explicitly define a DC/UCF task code. It can also include the asterix (*) character as the last character to denote a generic match. For example:-

NAME=ADSA	exclude task code ADSA
NAME=PMDC*	exclude any task code which begins with characters PMDC.

```
      #EXCTAB ENTRY=LAST
```

Figure 12 #EXCTAB ENTRY=LAST Syntax

Figure 12 shows the full syntax of the #EXCTAB ENTRY=LAST macro. This statement must be coded as illustrated to signify the end of the EXCTAB.

The following figure (**Figure 13**) shows an example of a valid EXCTAB definition.

```
DB2KEXC       #EXCTAB ENTRY=FIRST
              #EXCTAB ENTRY=TASK,NAME=ADSA
              #EXCTAB ENTRY=LAST
```

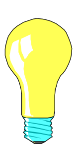
Figure 13 #EXCTAB example

1.5 Generating the Month Abbreviation Table (MONTAB)

The Month Abbreviation table is a table of month name abbreviations for alpha months in dates. The table is generated using the #MONTAB macro. A standard default table is supplied with EZ2000 as illustrated in **Figure 14**. It is only necessary to regenerate this table if you have different month name abbreviations, for example, in non-English speaking countries. #MONTAB allows a maximum of 10 characters for each month.

```
DB2KMON       #MONTAB M01=JAN,
                 M02=FEB,
                 M03=MAR,
                 M04=APR,
                 M05=MAY,
                 M06=JUN,
                 M07=JUL,
                 M08=AUG,
                 M09=SEP,
                 M10=OCT,
                 M11=NOV,
                 M12=DEC
```

Figure 14 #MONTAB example



If you have dates with different abbreviations, eg. 01JAN99 and 01JANUARY99, you will have to decide which abbreviation will be the default abbreviation defined in MONTAB, and then use the user exit to convert the *non-standard* abbreviation to the *standard* abbreviation as defined in MONTAB, allow date conversion to proceed, and then convert the new month back to the *non-standard* format on exit.

The installation source library contains a member DB2KMON which is a sample job-stream used to generate the default MONTAB.

1.6 Generating the Date Format Code Table (DFCTAB)

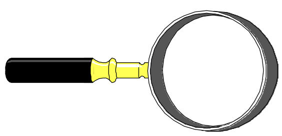
When generating the SRLTAB, you must specify the Date Format Code for every applicable element that EZ2000 needs to adjust. Refer to **section 1.3** for further details. As described in section 1.3, every Date Format Code specified in the SDETAB EL= statement must exist in the DFCTAB otherwise an assembler error will occur. EZ2000 is supplied with a default DFCTAB that in many cases will cater for all of the possible Date Format Codes in use at a given site. However, in some cases there may be Date Format representations which have not been catered for, in which case it will be necessary to regenerate the DFCTAB (or create a new DFCTAB to include the additional definitions).

This section describes how to generate the DFCTAB.

The DFCTAB is generated using the #DFCTAB macro. The following illustrates the three classes of #DFCTAB macros which must be combined to generate a DFCTAB.

```
#DFCTAB ENTRY=FIRST,<first-parameters>
#DFCTAB ENTRY=DATE,<date-parameters>
#DFCTAB ENTRY=LAST
```

To generate a valid DFCTAB you begin the definition with a #DFCTAB ENTRY=FIRST statement. You then include a #DFCTAB ENTRY=DATE statement for every required Date Format Code. The table definition is terminated by a #DFCTAB ENTRY=LAST statement.



The installation source library contains a member DB2KDFC which is a sample job-stream used to generate the default table. If you need to define additional Date Format Codes for your site, you should use the supplied source as your base DFCTAB, and include additional ENTRY=DATE statements <u>after</u> those provided in the default table. Furthermore, you will need to modify the supplied #DFCDS DSECT to include equates for the additional Date Format Codes. A listing of the supplied #DFCDS can be found in the appendix **section 4.2**.

Figure 15 shows the full syntax of the #DFCTAB ENTRY=FIRST macro.

```
tabname      #DFCTAB ENTRY=FIRST
```

Figure 15 #DFCTAB ENTRY=FIRST Syntax

Each of the available parameters is described below.

tabname: The name of the table may be specified by including a 1-8 character label in columns 1-8 of the #DFCTAB ENTRY=FIRST macro. If the macro is unlabelled, the default name of DB2KDFC will be generated.

Note: This name should also be the link-edit name of the generated table.

The name of the particular DFCTAB to use at run-time may be specified via the DFCTAB= run-time parameter. (See **section 2.1.4** for details of specifying run-time parameters).

Figure 16 shows the full syntax of the #DFCTAB ENTRY=DATE macro.

```
#DFCTAB ENTRY=DATE,
    FORMAT=,                            ( Date Format )
    PIC=,                               ( Picture )
    NINES=NO                            ( YES | NO )
    CCOFF=,                             ( century offset )
    YYOFF=,                             ( year offset )
    MMOFF=,                             ( month offset )
    DDOFF=                              ( day offset)
```

Figure 16 #DFCTAB ENTRY=DATE Syntax

Each of the available parameters is described below.

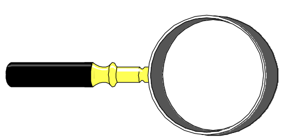
FORMAT: Specify the Date Format Code to be added to the table. Date Format Code describes the internal representation of a date field in an IDMS record element..

The characters “Y”, “M” and “D” are used to indicate **Y**ear, **M**onth and **D**ay character positions respectively. “C” is used to represent **C**entury indicator. Any other character would represent a filler position eg, “/” or “.”.

Examples of possible definitions are as follows:

```
#DFCTAB ENTRY=DATE,FORMAT=CYYMMMDD,PIC=DISPLAY
#DFCTAB ENTRY=DATE,FORMAT=0CYYDDD,PIC=COMP-3
#DFCTAB ENTRY=DATE,FORMAT=YYMM,PIC=HEX
#DFCTAB ENTRY=DATE,FORMAT=YY/MM/DD,PIC=DISPLAY
#DFCTAB ENTRY=DATE,FORMAT=YY?MM?DD,PIC=DISPLAY
```

Any special characters in the DFC, for example '/', would be inserted unchanged into the adjusted date. Use the special character '?' to indicate that you want filler characters to remain unchanged. For example, if the original date was 1999.12.31 and you used a DFC FORMAT=YYYY/MM/DD, then the adjusted date would contain '/' characters instead of '.' characters.



The #DFCTAB macro assumes that MM is numeric, whereas MMM is alpha. Something must be coded for every character position in the record element except the SIGN digit (when the PIC is COMP-3) eg., MM/DD/YY (CL8) or 0YYYYDDD (XL4) or YYYYDDD (PL4). or 0CYYDDD (PL4).

PIC: Defines the Picture, that is internal representation, of this record element. PIC must be a valid data representation, ie., one of the following:

COMP-3	- Packed Decimal
DISPLAY	- Character
HEX	- Hexadecimal
COMP	- Binary

NINES: Specify NINES=YES to process S9 (8) COMP fields that are NINES complement format

If you require nines complement date support,

1) regenerate your DFCTAB with a new date defined (for example) as,

```
#DFCTAB ENTRY=DATE,PIC=COMP,NINES=YES,
    FORMAT=YYYYMMDD
```

(Note: the format could also be MMDDYYYY or DDMMYYYY.)

2. Add a new entry to the corresponding position in #DFCDS, for example

```
BY4MMDD9 EQU nn
```

where *nn* is the next sequential number.

3. Regenerate your SDE and SRL tables using the new date format code where required.

CCOFF: Add the parameter CCOFF=*nn* where *nn* is the offset of the century indicators from the start of the date field

This parameter is required where century indicators have been added to a filler or are otherwise not contiguous to the rest of the date field.

For example: If you have a date YYMMDD followed by 10 characters then a two digit century indicator,

1) regenerate your DFCTAB with a new date defined as,

```
#DFCTAB PIC=DISPLAY,FORMAT=YYMMDDCC,
      CCOFF=16
```

2) Add a corresponding entry to #DFCDS For example,

```
NYMDC16 EQU nn
```

where *nn* is the next seq code.

3) Then change the SDETAB entry to

```
#SDETAB EL=element,DFC=NYMDC16,
```

```
VALIDATE=NO
```

Regenerate DFCTAB, SDETAB and SRLTAB.

The installation source library includes a member DB2KDFC which contains sample JCL used to build the default DFCTAB.

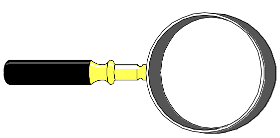
```
#DFCTAB ENTRY=LAST
```

Figure 17 #DFCTAB ENTRY=LAST Syntax

Figure 17 shows the full syntax of the #DFCTAB ENTRY=LAST macro. This statement must be coded as illustrated to signify the end of the DFCTAB.

1.7 Generating the Schema Record Date Element Override table (SEOTAB)

The default functionality of EZ2000 is to adjust every date field defined in the SRLTAB based upon the applicable RUNDATE as defined in the CTLTAB. For example, RUNDATE is 01 JAN 2000 and hence all defined dates get incremented by the difference between *base date* and RUNDATE.



base date defaults to *todays date* unless you have specified a BASEDAT in the CTLTAB definition. Refer to **section 1.2** for further information regarding generating the CTLTAB, and Technical Notes (**section 3.1**) for further information about using the BASEDAT parameter.

There are some date fields stored on the database that do not need adjusting at all, in which case, they are simply omitted from the SRLTAB. There are other date fields that *do* need adjusting, but because of particular circumstances are not adjusted by the default amount, but instead based upon some other criteria as defined in the Schema Record Date Element Override table (SEOTAB).

The SEOTAB is generated using the #SEOTAB macro. The following illustrates the different classes of #SEOTAB macros which must be combined to generate a SEOTAB.

```
#SEOTAB ENTRY=FIRST          <table definition>
#SEOTAB ENTRY=RECORD         <record definition>
#SEOTAB ENTRY=ELEMENT        <element definition>
#SEOTAB ENTRY=LAST
```

To generate a valid SEOTAB you begin the definition with a #SEOTAB ENTRY=FIRST statement. You would then include one or more #SEOTAB ENTRY=RECORD statements (one for each record containing elements which you wish to override). Each ENTRY=RECORD is followed by one or more ENTRY=ELEMENT statements (one for each element within that particular record for which there are overrides). The table is terminated by a #SEOTAB ENTRY=LAST.

This is illustrated by the following example:-

```
#SEOTAB ENTRY=FIRST,......
#SEOTAB ENTRY=RECORD,.....
#SEOTAB ENTRY=ELEMENT,.....
#SEOTAB ENTRY=ELEMENT,.....
#SEOTAB ENTRY=ELEMENT,.....
#SEOTAB ENTRY=RECORD,.....
#SEOTAB ENTRY=ELEMENT,.....
#SEOTAB ENTRY=LAST
```

A member DB2KSEO is included in the installation source library which contains sample JCL which may be modified to generate your own SEOTAB(s) if required.

The following figure (**Figure 18**) shows the full syntax of the #SEOTAB ENTRY=FIRST statement.

```
tabname      #SEOTAB ENTRY=FIRST,
                 SRLTAB=DB2KSRL,                    ( SRLTAB Name )
```

Figure 18 #SEOTAB ENTRY=FIRST Syntax

Each of the available parameters is described below:-

tabname: The name of the table may be specified by including a 1-8 character label in columns 1-8 of the #SEOTAB ENTRY=FIRST macro. If the macro is unlabelled, the default name of DB2KSEO will be generated.

Note: This name should also be the link-edit name of the generated table.

The name of the particular SEOTAB to use at run-time may be specified in the SRLTAB or through the //DB2KCTL SEOTAB= run-time parameter. (See **section 2.1.4** for details of specifying run-time parameters).

SRLTAB: Specify the name of the SRLTAB to which these overrides apply. There can only be one SEOTAB per SRLTAB.

The following figure (**Figure 19**) shows the full syntax of the #SEOTAB ENTRY=RECORD statement.

```
       #SEOTAB ENTRY=RECORD,
           RECORD=,                     ( Record Name )
           ID=                          ( Record Id )
```

Figure 19 #SEOTAB ENTRY=RECORD Syntax

Each of the available parameters is described below:-

RECORD: Specify the name of the CA-IDMS Record to which the subsequent ENTRY=ELEMENT definitions relate. You may specify multiple ENTRY=RECORD statements each followed by one or more ENTRY=ELEMENT statements.

ID: Specify the Record Identifier of this record.

The following figure (**Figure 20**) shows the full syntax of the #SEOTAB ENTRY=ELEMENT statement.

```
#SEOTAB ENTRY=ELEMENT,
    ELEMENT=,                    ( Element Name )
    EXIT=,                       ( User Exit Name )
    ADJUST=,                     ( Forward | Back )
    DEFAULT=YES,                 ( Yes| No )
    BY=,                         ( Number to Adjust )
    DATEFWD=,                    ( Exact Date Forward YYYYMMDD )
    DATEBAK=                     ( Exact Date Back YYYYMMDD )
```

Figure 20 #SEOTAB ENTRY=ELEMENT Syntax

Each of the available parameters is described below:-

ELEMENT: Specify the name of the CA-IDMS date element.

EXIT: Specify the name of a User Exit to be invoked for this date element. If specified, the User Exit is invoked *before* and *after* the EZ2000 date adjustment. ie.,

On retrieving data; As record is moved from DMCL buffer to PVS (program variable storage), user exit is invoked as record is moved from DMCL, then EZ2000 performs date adjustment, then user exit is invoked again prior to record being moved to Program Variable storage.

On storing/modifying data; As record is moved from PVS to DMCL, user exit is invoked as record is moved from PVS, then EZ2000 performs date adjustment, then user exit is invoked again prior to record being moved to DMCL.

Refer to Technical notes (**section 3.4**) for full details of coding and installing EZ2000 user exits.

ADJUST: You can specify adjustments for this date field that EZ2000 will apply *after* or *instead of* performing the standard date adjustment. The function of this parameter varies depending on the values of the DEFAULT and BY parameters described later. This parameter can have the following values:-

FORWARD: If DEFAULT=YES - Perform standard date adjustment then shift *forward* by number (as specified in the BY parameter).

If DEFAULT=NO - Do not perform standard date adjustment. Just shift *forward* by number (as specified in the BY parameter).

BACK: If DEFAULT=YES - Perform standard date adjustment then shift *backward* by number (as

specified in the BY parameter).
If DEFAULT=NO - Do not perform standard date adjustment. Just shift *backward* by number (as specified in the BY parameter).

If the date contains *days*, then BY= number of days to adjust. If the date contains no days but does contain *months*, then BY= number of months to adjust. If date contains only *years*, then BY= number of years to adjust.

BY: Specify the number of *days*, *months* or *years* to adjust. See ADJUST parameter above.

DEFAULT: Specify whether EZ2000 should perform standard date adjustment prior to applying the adjustment (Forward or Back). Refer to the ADJUST parameter previously described.

The default value of this parameter is YES.

DATEFWD: You can explicitly define a date to set for this date field when shifting dates forward (ie on retrieving a record from the database, upon moving the record *to* Host Variable storage). The date is specified in the format YYYYMMDD (although the actual date generated will be set according to the Date Format Code for the particular record element).

The specified date is set <u>instead</u> of the default date which would be generated by adding the *date difference* (ie., RUNDATE - base date).

For example, suppose you have a date field START-FINANCIAL-YEAR which is 1st April 1997, and you set a RUNDATE of 1st Jan 2000. In this case, you would want to ensure that START-FINANCIAL-YEAR gets set to 1st APRIL 1999.

DATEBAK: You can explicitly define a date to set for this record element when shifting dates back (ie on storing/modifying a record, upon moving the record *from* Host Variable storage). The date is specified in the format YYYYMMDD (although the actual date generated will be set according to the Date Format Code for the particular record element).

The specified date is set <u>instead</u> of the default date which would be generated by subtracting the *date difference* (ie., RUNDATE - base date).

```
#SEOTAB  ENTRY=LAST
```

Figure 21 #SEOTAB ENTRY=LAST Syntax

Figure 21 shows the full syntax of the #SEOTAB ENTRY=LAST macro. This statement must be coded as illustrated to signify the end of the SEOTAB.

1.8 Generating the Business Holiday Table (BIZTAB)

If you specify the SDETAB parameter BIZDAY=YES, then EZ-Test IDMS will ensure that any aged dates for that record element will fall on business (working) days only. For this purpose, EZ-Test IDMS requires a BIZTAB which defines non-working days as applicable in that particular environment.

Sample tables are provided in the installation tape load-library (BIZTABUK and BIZTABUS) which define all non-working days for the UK and USA respectively until the year 2009. If you wish to extend the period covered, or define non-working days for other countries, then you will need to generate your own site specific BIZTAB.

The BIZTAB is generated using the #BIZTAB macro. The following illustrates the different classes of #BIZTAB macros which must be combined to generate a BIZTAB.

```
#BIZTAB ENTRY=FIRST           <table definition>
#BIZTAB ENTRY=HOL             <Holiday definition>
#BIZTAB ENTRY=LAST
```

To generate a valid BIZTAB you begin the definition with a #BIZTAB ENTRY=FIRST statement. You would then include one or more #BIZTAB ENTRY=HOL statements for each non-working day to be defined. The table is terminated by a #BIZTAB ENTRY=LAST.

A member DB2KBIZ is included in the installation source library which contains sample JCL which may be modified to generate your own BIZTAB(s) if required.

The following figure (**Figure 22**) shows the full syntax of the #BIZTAB ENTRY=FIRST statement.

```
tabname      #BIZTAB ENTRY=FIRST,
                  FRIDAY=WEEKEND,
                  SATURDAY=WEEKEND,
                  SUNDAY=WEEKEND
```

Figure 22 #BIZTAB ENTRY=FIRST Syntax

Each of the available parameters is described below:-

tabname: The name of the table may be specified by including a 1-8 character label in columns 1-8 of the #BIZTAB ENTRY=FIRST macro. If the macro is unlabelled, the default name of DB2KBIZ will be generated.

Note: This name should also be the link-edit name of the generated table.

The default name of the BIZTAB is DB2KBIZ. Sample tables are provided with names such as BIZTABUS, BIZTABUK etc. You can override the name of the BIZTAB loaded at runtime using the DB2KCTL parameter BIZTAB=. Alternatively, you can rename the required sample table to DB2KBIZ.

FRIDAY=WEEKEND
SATURDAY=WEEKEND
SUNDAY=WEEKEND

Specify the names of the days of the week which are the standard non-working days. Normally this would be Saturday and Sunday, although any permutation is allowed.

The following figure shows the full syntax of the #BIZTAB ENTRY=HOL statement.

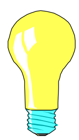
```
              #BIZTAB ENTRY=HOL,
                  BIZHOL=               ( Business Holiday - YYYYMMDD )
```

Figure 23 #BIZTAB ENTRY=HOL Syntax

The available parameters are described below:-

BIZHOL

Specify a date which is a holiday or other *non-working* day, in the format YYYYMMDD. You should include one #BIZTAB ENTRY=HOL statement for every applicable date for the period that EZ-Test IDMS requires.



Holidays which fall on weekends should not be included. Only define holidays which are <u>in addition to</u> normal weekend breaks. For example, if Christmas day falls on a Saturday, it should not be included, but you probably will have an entry for the Monday and Tuesday.

The sample tables included on the installation tape have definitions up to the year 2010. You may need to extend these tables depending on your requirements.

The following figure shows the full syntax of the #BIZTAB ENTRY=LAST statement.

```
              #BIZTAB ENTRY=LAST
```

Figure 24 #BIZTAB ENTRY=LAST Syntax

It should be coded exactly as illustrated.

Refer to **technical notes** section for further information about how business holiday aging is calculated.

1.9 Generating the EZ-Bridge loadlib table (BRGTAB)

The BRGTAB is used by EZ-Bridge to identify the load-libraries and dictionary load-areas which contain programs to which the *schema* view applies, and the load-libraries and dictionary load-areas which contain programs to which the *application* view applies.

NOTE: When using EZ-Bridge to implement windowing only, i.e, there are no DFCAPP definitions included in the SDETAB, then it is necessary to generate a dummy BRGTAB. See section 1.9.2 later.

The BRGTAB is generated using the #BRIDGE macro. The following illustrates the four classes of #BRIDGE macros which must be combined to generate a BRGTAB;

```
#BRIDGE     ENTRY=SRLTAB,<SRLTAB options>
#BRIDGE     ENTRY=LOAD,<LOAD options>
#BRIDGE     ENTRY=DICT,<DICT options>
#BRIDGE     ENTRY=LAST
```

A valid BRGTAB consists of one or more #BRIDGE ENTRY=SRLTAB statements, each followed by one or more #BRIDGE ENTRY=LOAD and/or #BRIDGE ENTRY=DICT statements. The table is terminated by a #BRIDGE ENTRY=LAST statement.

Figure 25 shows the full syntax of the #BRIDGE ENTRY=SRLTAB statement.

```
tabname     #BRIDGE ENTRY=SRLTAB
                  NAME=                    ( SRLTAB Name )
                  LIBLIST=                 ( SCHEMA | NULL )
```

Figure 25 #BRIDGE ENTRY=SRLTAB

Each of the available parameters is described below:-

tabname: The name of the table may be specified by including a 1-8 character label in columns 1-8 of the first #BRIDGE ENTRY=SRLTAB macro. If the macro is unlabelled, the default name of DB2KBRG will be generated.

Note: This name should also be the link-edit name of the generated table.

NAME Specify the name of the SRLTAB to which the following program library definitions apply. An SRLTAB relates to a particular schema/version, and there may be multiple SRLTABs available to a particular CA-IDMS system. At run-time EZ-Bridge will select the BRGTAB entry which relates to the current SRLTAB.

LIBLIST Specify LIBLIST=SCHEMA to indicate that the following library list defines the names of the load-libraries and/or dictionaries which contain the programs/dialogs which use the DFC definition for any applicable record-elements.

If the LIBLIST parameter is omitted, the following library list defines the names of the load-libraries and/or dictionaries which contain the programs/dialogs which use the DFCAPP definition for any applicable record-elements.

The user may define the list in either manner, whichever is more convenient.

```
      #BRIDGE ENTRY=LOAD,
            NAME=                    ( load library name )
```

Figure 26 #BRIDGE ENTRY=LOAD syntax.

Figure 26 shows the full syntax of the #BRIDGE ENTRY=LOAD statement.

Each of the available parameters is described below:-

NAME Enter NAME='DSN' where DSN is the dataset name of an OPSYS load library. The name should be fully qualified and enclosed in quotes.

This library should be included in the CDMSLIB or EZ-Bridge concatenation of the applicable CA-IDMS CV or local job. The library contains application programs which require the schema definition (DFC) or the application definition (DFCAPP) of applicable record-elements, depending on whether the preceding ENTRY=SRLTAB specified LIBLIST=SCHEMA or not.

```
       #BRIDGE ENTRY=DICT,
             NAME=                    ( DICTNAME )
```

Figure 27 #BRIDGE ENTRY=DICT syntax.

Figure 27 shows the full syntax of the #BRIDGE ENTRY=DICT statement.

Each of the available parameters is described below:-

NAME Enter NAME='DICTNAME' , where DICTNAME relates to an alternate dictionary load-area, or NAME='DEFAULT' for the primary dictionary.

The load-area so described contains ADS/O dialogs which require the schema definition (DFC) or the application definition (DFCAPP) of applicable record-elements, depending on whether the preceding ENTRY=SRLTAB specified LIBLIST=SCHEMA or not.

```
       #BRIDGE ENTRY=LAST
```

Figure 28 #BRIDGE ENTRY=LAST

Figure 28 shows the full syntax of the #BRIDGE ENTRY=LAST statement. The parameter should be coded as shown.

1.9.1 BRGTAB example

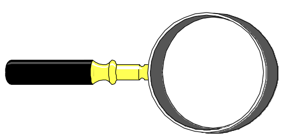
Figure 29 shows an example of a typical BRGTAB definition.

```
     DB2KBRG #BRIDGE ENTRY=SRLTAB,NAME=DB2KSRL,LIBLIST=SCHEMA
*
             #BRIDGE ENTRY=LOAD,NAME='IDMSDEV.C9501.LOADLIB'
             #BRIDGE ENTRY=LOAD,NAME='IDMSDEV.C9501.DBALOAD'
             #BRIDGE ENTRY=LOAD,NAME='IDMSDEV.C9501.DEVLOAD'
             #BRIDGE ENTRY=DICT,NAME='DEFAULT'
*
             #BRIDGE ENTRY=SRLTAB,NAME=DB2KSRLN
             #BRIDGE ENTRY=LOAD,NAME='IDMSDEV.C9501.TESTY2K'
             #BRIDGE ENTRY=DICT,NAME='DICTY2K'
*
             #BRIDGE ENTRY=LAST
```

Figure 29 BRGTAB example

In this example, note that for the first SRLTAB (DB2KSRL) the user has specified LIBLIST=SCHEMA. The library definition for that SRLTAB includes three OPSYS load-libraries and an entry for the default dictionary. Any program loaded from any of these OPSYS libraries or the default dictionary will use the SCHEMA definition of any date field defined in the SRLTAB (ie. DFC). Programs loaded from <u>any other</u> OPSYS library or any other dictionary will use the APPLICATION view of any date field defined in the SRLTAB (ie. DFCAPP).

The table also includes a definition for a second SRLTAB (DB2KSRLN). For this definition, the parameter LIBLIST=SCHEMA is omitted. For this SRLTAB there are just two library definitions, an OPSYS library and a dictionary load-area. Any program loaded from either this OPSYS library or the specified dictionary will use the APPLICATION definition of any date field defined in the SRLTAB (ie. DFCAPP). Programs loaded from <u>any other</u> OPSYS library or any other dictionary will use the SCHEMA view of any date field defined in the SRLTAB (ie. DFC).



Note that for CICS and other ERUS, the load libraries listed must be made available to the CV by adding them to the end of CDMSLIB or to a special DD statement //EZ-Bridge DD DSN=, etc.

If that DD statement exists then it is checked with a BLDL to see if the load module that performs the BIND RUN UNIT is contained within the concatenation. If not and LIBLIST=SCHEMA, or it is and LIBLIST is not =SCHEMA then DFCAPP is used.

If //EZ-Bridge is not included then CDMSLIB is checked.

1.9.2 Dummy BRGTAB for WINDOWING implementation

When using DB-EZ-Bridge to implement windowing on its own, ie there are no DFCAPP= parameters in the SDE table, a dummy bridge table (DB2KBRG) must be supplied.

These parameters will define a suitable dummy table:

```
#BRIDGE ENTRY=SRLTAB
#BRIDGE ENTRY=LOAD,NAME='DUMMY'
#BRIDGE ENTRY=LAST
```

EZ2000 Run-time environment

2 EZ2000 Run-time environment

2.1 Activating EZ2000 for different environments

2.1.1 Executing non-database batch programs

It is not necessary to make *any* JCL changes to batch job JCL in order for them to run with EZ2000 Assumed RunDates. Provided that the DB2KSET utility has been executed in order to install the SVC HOOK and load the CTLTAB, and the batch job name is present in the current CTLTAB, then the batch program will receive the applicable Assumed RunDate and time in response to any application request for the current date and time.

2.1.2 Executing batch programs against a CA-IDMS CV

As in the case of non-database batch programs, it is not necessary to make any modifications to the batch job JCL. However, if the program is retrieving data from a CA-IDMS database managed by a CA-IDMS central version, and you expect the database records presented to the program to have date fields adjusted by EZ2000, then the EZ2000 component installed in the applicable CV must have some way of recognising the requests from the batch program as being EZ2000 eligible. This can be achieved by defining the applicable ERUSJOB name or DBNAME in the CTLTAB in use by the CV. For example, if the CTLTAB has been defined such that different RUNDATEs apply to different DBNAMEs, then the batch job must BIND with the appropriate DBNAME. Refer to **section 1.2** for further details about the CTLTAB.

2.1.3 Local Mode IDMS and CV startup JCL

Local mode CA-IDMS jobs, and Central Version Startup JCL should be modified as illustrated in the following figure.

```
//JOBCARD
//DB2KSET  EXEC PGM=progname
//CDMSLIB  DD DSN=ez2000.loadlib,DISP=SHR  <= may be STEPLIB for local mode
//DB2KLOG  DD SYSOUT=*
//DB2KCTL  DD *
control-statements
```

Figure 30 IDMS Startup/Local Mode JCL

ez2000.loadlib	This is the name of the load-library containing the EZ2000 run-time modules and generated tables. For CA-IDMS startup JCL it should be included in the CDMSLIB library concatenation. For local mode jobs, it should be included in CDMSLIB if applicable, or the jobs STEPLIB. In either case, the library must appear in the concatenation ahead of any other dataset containing an IDMSDBIO load-module.

DB2KLOG — Specify a SYSOUT file to which EZ-Test IDMS will write summary statistics at CV shut-down or local job step termination.

If EZ-Test IDMS encounters any date validation errors at run-time, e.g. internal format is not compatible with specified date format code for the field, then validation errors are also written to the DB2KLOG file.

DB2KCTL — Run-time parameters are specified via the DB2KCTL SYSIN file. The available run-time parameters are described in the next section.

2.1.4 The //DB2KCTL run-time parameter file

Users may include a DB2KCTL file in the run-time JCL to override the names of the various EZ2000 run-time tables.

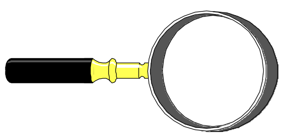


Table Loading; Any tables mentioned in //DB2KCTL are loaded first. Any tables <u>not</u> specified in //DB2KCTL are loaded as follows: -

- CTLTAB - the default table applicable to the environment (i.e., CTLIDMS, CTLCICS etc).
- DB2KEXE (for CICS DB2KCEX), DB2KDFC and DB2KMON.

If CTLTAB contains a BRGTAB table name then that is loaded. All SRLTAB names coded in the CTLTAB(s) are loaded. All SEOTAB names in the SRLTABs are loaded.

As should be apparent, the use of //DB2KCTL to specify table names should not normally be necessary, other than to list tables not referenced elsewhere and not to be used at run time, but required to be viewed via DB2K displays (See CTLTAB and BRGTAB notes below).

If required, the parameters may be included *in-stream* as shown below:-

//DB2KCTL *	
`CTLTAB=`*tabname*	Control Table (one run-time table only - but multiple CTLTAB entries may be coded - see CTLTAB note below)
`EXCTAB=`*tabname*	Task Exclusion Table (one only)
`DFCTAB=`*tabname*	Date Format Code Table (one only)
`SRLTAB=`*tabname*	Schema Record Date Element Table (one per schema version)
`SEOTAB=`*tabname*	Record Date Element Override Table (one per SRLTAB)
`BRGTAB=`*tabname*	Bridge Table (one run-time table only - but multiple BRGTAB entries may be coded - see BRGTAB note below).
`BIZTAB=`*tabname*	Business day table name (one only)

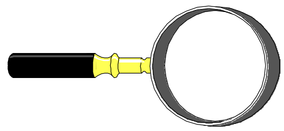
or alternatively, you may specify the parameters through an input dataset e.g.,

//DB2KCTL DD DSN=*parameter.file*,DISP=SHR

where ***parameter.file*** is allocated as RECFM=F or FB and LRECL=80.

Each parameter must be coded on a separate line beginning in column one. The permissible parameters are described below:

CTLTAB= — *tabname* is the name of the EZ2000 Control Table to be used by this job. If this parameter is not specified, the default table for the current environment is loaded.



Only one CTLTAB is used at run time in any given environment. However, multiple entries for CTLTAB= may be input via //DB2KCTL to enable their contents to be displayed via the DB2K display screens. If this is done, then the control table to be used at run time in that environment must be included and be the **first** in the list. The first will always be the one used at run time.

SRLTAB=

tabname is the name of the Schema Record Date Element Definition Table.

There may be multiple occurrences of this parameter for a particular job, as in practice, you may have generated one such table for each available Schema/Version. At system initialisation time, EZ-Test IDMS loads *all* of the specified tables. At run-time, EZ-Test IDMS selects the applicable CTLTAB DBEZ2000 entry for the relevant CLASS and will use the specified SRLTAB for that entry.

If no SRLTAB entries are specified in the //DB2KCTL file, EZ-Test IDMS will load all of the SRLTABs specified in the run-time CTLTAB.

SEOTAB=

tabname is the name of the Schema Record Date Element Override Table.

There may be multiple occurrences of this parameter for a particular job, as an SEOTAB relates to a particular SRLTAB. At system initialisation time, EZ-Test IDMS will load *all* of the specified tables, and at run-time will use the applicable table for the current SRLTAB.

If no SEOTAB entries are specified in the //DB2KCTL file, EZ-Test IDMS will load all of the SEOTAB(s) as specified by the applicable SRLTAB(s).

DFCTAB=

tabname is the name of the Date Format Code Table. If this parameter is not specified, the default name DB2KDFC is used.

There may only ever be one such table in use at a time for a particular job.

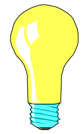
EXCTAB=

tabname is the name of the Task Exclusion Table. If this parameter is not specified, the default name DB2KEXC (or DB2KCEX for CICS) is used.

There may only ever be one such table in use at a time for a particular job.

BIZTAB=

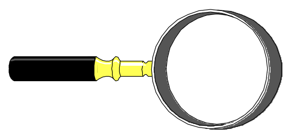
tabname is the name of the Business Day table. If this parameter is not specified, the default name DB2KBIZ is used.



Sample tables are provided in the installation load library containing definitions for particular countries (e.g. BIZTABUK, BIZTABUS etc.). If a table is provided for your country, you can simply rename it to DB2KBIZ. Alternatively, you will need to generate a BIZTAB using the #BIZTAB macro as described in **section 1.8**.

BRGTAB=

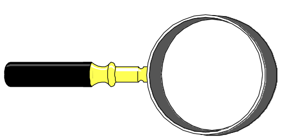
tabname is the name of the EZ-Bridge table to be used. If this parameter is not specified, the default BRGTAB name specified in the run-time CTLTAB is loaded.



Only one BRGTAB is used at run time. However, multiple entries for BRGTAB= may be input via //DB2KCTL to enable their contents to be displayed via the DB2K display screens. If this is done, then the BRGTAB to be used at run time in that environment must be included and be the **first** in the list. The first will always be the one used at run time.

2.1.5 The DB2KLOG SYSOUT File

Users may specify a DB2KLOG file in the run-time JCL to which EZ-Test IDMS will write a shutdown or job-step termination report. Any run-time date validation errors will also be reported in the DB2KLOG file if present.



If DB2KLOG is not present, EZ-Test IDMS will write any run-time error messages directly to the MVS log.

The following figure shows part of a typical shutdown report.

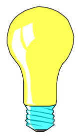
```
 11 MAR 99 12:48                DB-EZ-Test 2000 Shutdown Report                         PAGE 1

                               DB-EZ-Test Control Table: CTLTAB
                            Base Date: 19990311    Assumed Date: 20020101
                            Days Aged:      0         YYMM Aged:  0  0
Schema Table PRODSRL
  Record: RCECOMP              Aged:             0   Restructured:             0
   Dates: ACCDATD                          #Errors:            0 DFC: CYYSDDD      Comp-3
  Record: RCACV1               Aged:             0   Restructured:             0
   Dates: ACV1INDA                         #Errors:            0 DFC: CYYSDDD      Comp-3
          ACV1TRDA                         #Errors:            0 DFC: CYYSDDD      Comp-3
  Record: RCLIST               Aged:             0   Restructured:             0
   Dates: ADATE                            #Errors:            0 DFC: CYYSDDD      Comp-3
  Record: RCQUOTE              Aged:             0   Restructured:             0
   Dates: ADJDATE                          #Errors:            0 DFC: CYYSDDD      Comp-3
          BASED                            #Errors:            0 DFC: CYYSDDD      Comp-3
  Record: RCECOMP              Aged:             0   Restructured:             0
   Dates: CACCYED                          #Errors:            0 DFC: CYYSDDD      Comp-3
  Record: RCWCOU               Aged:             0   Restructured:             0
   Dates: CAV-W5256                        #Errors:            0 DFC: CYYSDDD      Comp-3
  Record: RCLIST               Aged:             0   Restructured:             0
   Dates: CDATE                            #Errors:            0 DFC: CYYSDDD      Comp-3
  Record: RCCLAIM              Aged:             0   Restructured:             0
   Dates: CLINDATE                         #Errors:            0 DFC: CYYSDDD      Comp-3
          CLTRDATE                         #Errors:            0 DFC: CYYSDDD      Comp-3
  Record: RCCURD2              Aged:             0   Restructured:             0
   Dates: CURD-MATD                        #Errors:            0 DFC: CYYSDDD      Comp-3
  Record: RCFEARN              Aged:             0   Restructured:             0
   Dates: CURFYEDT                         #Errors:            0 DFC: CYYSDDD      Comp-3
  Record: RCCDBNM              Aged:             0   Restructured:             0
   Dates: DBCUDATE                         #Errors:            0 DFC: CYYSDDD      Comp-3
  Record: RCUCILH              Aged:             0   Restructured:             0
   Dates: DEADDATE                         #Errors:            0 DFC: CYYSDDD      Comp-3
  Record: RCSECUR              Aged:             0   Restructured:             0
   Dates: DIVCURPD                         #Errors:            0 DFC: CYYSDDD      Comp-3
```

Figure 31 Sample DB2KLOG shutdown report.

2.2 CA-IDMS Online Commands and Displays

The EZ2000 online displays can be invoked by entering the DC/UCF task code **DB2K** from the DC/UCF Enter Next Task Code prompt. On 3270 type terminals, the display will be full colour. If your terminal is not a 3270 but does support extended colour attributes, you can use the task code **DB2KC** to force full colour display.

Either of these task codes will initially display the EZ2000 main menu as described in the next section. On this screen, you are presented with a menu of all of the various screens currently available.



Note that ALL of the following commands/entries will work from ANY screen, and have the same effect.

2.2.1 Screen Navigation

You can jump directly to any other screen by entering either the number corresponding to the screen, or a mnemonic at the command line, as illustrated below:

Number	mnemonic	Jump to
	MENU	Menu Display
1	CTL	CTLTAB display
2	MON	MONTAB display
3	DFC	DFCTAB display
4	EXC	EXCTAB display
5	SRL	SRLTAB display
6	SEO	SEOTAB display
7	BRG	BRGTAB display
8	BIZ	BIZTAB display

The following PF/control keys can also be used and have the following functions:

<PF3>	Return to Menu screen
<PF4>	Exit displays
<PF7>	Up
<PF8>	Down
CLEAR	Exit displays

2.2.2 DB2K Command Line functions

There are a number of commands which may be entered to control the CA-IDMS environment. These may be entered on the command line of any **DB2K** screen.

CTLTAB=
SRLTAB=
To select the SRLTAB or CTLTAB to be displayed, the user can specify the name required using the commands:

```
CTLTAB=ctlname     or
SRLTAB=srlname
```

This can be typed on any screen and remains in force for the users's display session or until modified with another CTLTAB= or SRLTAB= command. The user may use this command to display any of the SRLTABs in use by this particular CV, and the CTLTAB in use by this CV or the SVC Hook.

NEWCOPY
All of the EZ2000 tables can be *new copied* using the command NEWCOPY. This will reload all of the DB2K tables as specified at system startup (named in DB2KCTL file, or the default names). So, for example, the user can regenerate the CTLTAB to add a new DBNAME and then issue the NEWCOPY command. All of the tables, including the new CTL table, will be reloaded and any new/modified definitions will be in effect. NEWCOPY can be entered on any screen, but will always return you to the main menu screen.

SET RUNDATE=YYYYMMDD **SRLTAB=**srltab-name **<BASEDATE=**YYYYMMDD**>** **<BASETIME=**HHMM**>**
The SET command may be used from the command line of any screen to override the current settings in effect for that particular terminal. Parameters shown in brackets <> are optional. The other parameters are mandatory.

The current session settings are shown on the message line of every screen .

SET OFF
The SET OFF command will disable the current override and return to the default value.

SET DISABLE
Switch off EZ2000 processing for the CA-IDMS CV.

SET ENABLE
Switch on EZ2000 processing for the CA-IDMS CV.

EZ2000 will remain active for the life of run-units in progress at the time the DISABLE command was issued. EZ2000 will be inactive for new run-units starting after the DISABLE command was issued. (and vice versa for the ENABLE command).

Note that all command line keywords can be shortened to a minimum of two characters that uniquely identify the keyword.
The above mentioned SET commands can also be entered directly from the DC/UCF *enter next task code* prompt using the DB2KSET task code described later.

2.2.3 Menu Display

```
COGITO EZ-DB:R5.3  Gen Level:C90201                  V0014  13:06:43 2002.001
                         EZ-DB EZ-Test CONTROL PANELS


                                   MENU

                 1  -  Control Table              (CTLTAB)
                 2  -  Month Abbreviations        (MONTAB)
                 3  -  Date Formats               (DFCTAB)
                 4  -  Task Exclusion List        (EXCTAB)
                 5  -  Record/Element Definitions (SRLTAB)
                 6  -  Record/Element Overrides   (SEOTAB)
                 7  -  Bridge Loadlist            (BRGTAB)
                 8  -  Business Holidays          (BIZTAB)

       Current User Session Settings            Products Active
        SRLTAB            : PRODSRL              Date Simulation : Active
        RUNDATE           : 20020101             Logical Ager    : Active
        BASEDATE          : 19990308             Windowing       :
        BASETIME          :                      Bridge          :
        DAYS AGED         :   1030               DBWatch         :
 ENTER COMMAND BELOW>

 SET SRLTAB=DB2KSRL  RUNDATE=20011231 BASEDAT=19970601/12:00
```

This screen illustrates the DB2K main menu. It is the initial screen displayed when you enter the DB2K or DB2KC command. From this screen you can jump directly to any of the other available screens by entering the number corresponding to the screen or the screen mnemonic at the command line.

You can return to this screen from any other screen by entering the <PF03> function key.

The screen shown above illustrates the use of the SET command. Current User Settings are displayed on this screen. In addition, the SET command line will be echoed on the message line of all other screens as a reminder to the user of the settings in effect. If the user has issued the DB2KSET DISABLE command to disable EZ-Test IDMS for the entire CV, then the message “DBEZ-TEST2000 IS CURRENTLY DISABLED" will be shown on the bottom line of every DB2K display.

2.2.4 Control Table Display

```
COGITO EZ-DB (R5.2)  Tape (C80101)                   V0001  11:14:36 1997.101
                       *** EZ-DB DB-EZ2000 Display ***
                           CONTROL TABLE ENTRIES
 CONTROL TABLE NAME : DB2KCTL             ERUS EXTENSION   : UA  OFFSET :    4
               TYPE : IDMS                SYSTEM TASKS     : EXCLUDE

    #  CLASS        VALUE      RULE  I/E  ASSUME-DAT SRL-TABLE START-DATE/TIME
-------------------------------------------------------------------------------
    1  DBNAME       DBN2K001  OR     I     20000101   DB2KSRL   SYSTEM
       DBNAME       DBN2K002  OR
    2  DBNAME       DBN2K003  EQUAL  I     20000102   DB2KSRL   19991231  20:33









-------------------------------------------------------------------------------
 <PF3> BACK  <PF4> EXIT  <PF7> UP  <PF8> DOWN
ENTER COMMAND BELOW>
```

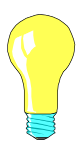
This screen illustrates the DB2K Control table display It can be selected by entering the number **1** or the mnemonic **CTL** at the command line from any other screen.

The screen is used to display the current CTLTAB settings.

By default, this function will display the *active* CTLTAB as currently used by this CA-IDMS CV. You can also display the CTLTAB currently in use by the SVC Hook routine (DB2KSVC) by specifying the following:-

CTLTAB=*ctlname*

at the command line. Where *ctlname* is the name of the CTLTAB loaded by the DB2KSET routine.



CTLTAB= can in fact be entered from any screen and remains in force for the users entire display session, or until modified by another CTLTAB= command.

From this screen you can jump directly to any of the other available screens by entering the number corresponding to the screen or the screen mnemonic at the command line.

2.2.5 Month Abbreviation Table Display

```
COGITO EZ-DB (R5.2)  Tape (C80101)                   V0005 15:14:17 1997.091
                        *** EZ-DB DB-EZ2000 Display ***
                          MONTH ABBREVIATION TABLE

  MONTH ABBREV TABLE  :   DB2KMON

                      MONTH#            ABBREVIATION
                      --------------------------------
                         01        -       JAN
                         02        -       FEB
                         03        -       MAR
                         04        -       APR
                         05        -       MAY
                         06        -       JUN
                         07        -       JUL
                         08        -       AUG
                         09        -       SEP
                         10        -       OCT
                         11        -       NOV
                         12        -       DEC
  <PF3> BACK  <PF4> EXIT <PF7> UP  <PF8> DOWN
 ENTER COMMAND BELOW>
```

This screen illustrates the DB2K Month Abbreviation Table display It can be selected by entering the number **2** or the mnemonic **MON** at the command line from any other screen.

The screen is used to display the current MONTAB settings.

From this screen you can jump directly to any of the other available screens by entering the number corresponding to the screen or the screen mnemonic at the command line.

2.2.6 Date Format Code Table Display

```
COGITO EZ-DB (R5.2)  Tape (C80101)                      V0005  15:14:58 1997.091
                        *** EZ-DB DB-EZ2000 Display ***
                              DATE FORMAT CODES

  DATE FORMAT TABLE  :    DB2KDFC

  CODE  PICTURE                FMAT  LEN | CODE  PICTURE                FMAT  LEN
 -------------------------------------------------------------------------------
      0  SQLDATE                 S     0 |    11  DDDDDDD                 P     7
      1  0CYYDDD                 P     7 |    12  0YY                     P     3
      2  YYYYDDD                 P     7 |    13  CYY                     P     3
      3  YYDDD                   P     5 |    14  CYYDDD                  U     6
      4  00YYDDD                 P     7 |    15  0YYYYDDD                U     8
      5  0YYDDDC                 P     7 |    16  0YYDDD                  U     6
      6  CYYMMDD                 P     7 |    17  000YYDDD                U     8
      7  0YYYYMMDD               P     9 |    18  YYDDDC                  U     6
      8  0YYMMDD                 P     7 |    19  00YYDDDC                U     8
      9  DDD                     P     3 |    20  0CYYMMDD                U     8
     10  DDDDD                   P     5 |    21  YYYYMMDD                U     8
 -------------------------------------------------------------------------------
   <PF3> BACK  <PF4> EXIT <PF7> UP  <PF8> DOWN
 ENTER COMMAND BELOW>
```

This screen illustrates the DB2K Date Format Code Table display It can be selected by entering the number **3** or the mnemonic **DFC** at the command line from any other screen.

The screen is used to display the current DFCTAB definitions.

From this screen you can jump directly to any of the other available screens by entering the number corresponding to the screen or the screen mnemonic at the command line.

2.2.7 Excluded Tasks

```
COGITO EZ-DB (R5.2  Tape (C80101)                   V0005  15:15:29 1997.091
                        *** EZ-DB DB-EZ2000 Display ***
                            EXCLUDED TASKS

  TASK EXCLUDE TABLE :    DB2KEXC



-------------------------------------------------------------------------------
 ADSC       ADSCADSR   ADSCT      ADSOTATU   ALLOC8     B          BYE
 CLIST      CLOD       COPTFS     COSEQ      COSEQD     COSEQDC    COSEQV
 DBBP       DBBPMNTR   DBBU       DBHOT      DBIM       DBIMC      DBIMV
 DCMT       DCPROFIL   DCUF       DEBUG      IDD        IDDM       IDDML
 IDDMT      IDDT       LOOK       MAPC       MAPCT      OCF        OCFT
 OCFX       OLMI       OLP        OPER       PEEK       PEEKC      PEEKX
 PMDC       PMDCHALT   PMDCLDR    PMDCMNTR   PMDCSERV   PMDCVAPP   QUED
 RHDCSTTS   S          SCHEMA     SCHEMAT    SDEL       SEND       SHARE
 SHAREU     SHOWMAP    SIGNOFF    SIGNON     SSB        SSBF       SSBU
 SSC        SSCT       SUSPEND    SYNC       SYSGEN     SYSGENT    TCF
 XAST       XMVS

-------------------------------------------------------------------------------
   <PF3> BACK  <PF4> EXIT <PF7> UP  <PF8> DOWN
 ENTER COMMAND BELOW>
```

This screen illustrates the DB2K Excluded Task table display It can be selected by entering the number **4** or the mnemonic **EXC** at the command line from any other screen.

The screen is used to display the current EXCTAB definitions.

From this screen you can jump directly to any of the other available screens by entering the number corresponding to the screen or the screen mnemonic at the command line.

2.2.8 Record/Element Definitions

```
COGITO EZ-DB:R5.3  Gen Level:C90201                  V0014  13:07:59 2002.001
                        EZ-DB EZ-Test CONTROL PANELS
                         RECORD/ELEMENT DEFINITIONS

  SRL TABLE: PRODSRL  SCHEMA: DSTMPROD           VERS:    1  SEO TABLE: PRODSEO

  RECORD/ELEMENT NAME   REC ID       KEY  DFC1 DFC2 WIN WAT BRG AGE BIZ YYMM SEO
 -------------------------------------------------------------------------------
  RCECOMP                 3300
    ACCDATD                                 81                  Y
  RCACV1                   126
    ACV1INDA                                81                  Y            Y
    ACV1TRDA                                81                  Y
  RCLIST                  1903
    ADATE                                   81                  Y
  RCQUOTE                 3302
    ADJDATE                                 81                  Y
    BASED                                   81                  Y
  RCECOMP                 3300
 -------------------------------------------------------------------------------
  <PF3> BACK  <PF4> EXIT  <PF7> UP  <PF8> DOWN
 ENTER COMMAND BELOW>
```

This screen illustrates the DB2K Schema Record Element definitions. It can be selected by entering the number **5** or the mnemonic **SRL** at the command line from any other screen.

The screen is used to display the current SRLTAB and corresponding SEOTAB setting if applicable. As it is possible to have multiple SRL/SEOTABs, it is possible to select the SRL/SEOTAB to display by entering:

SRLTAB=*srlname*

at the command line.

SRLTAB= can in fact be entered from any screen and remains in force for the users entire display session, or until modified by another SRLTAB= command.

From this screen you can jump directly to any of the other available screens by entering the number corresponding to the screen or the screen mnemonic at the command line.

2.2.9 Record/Element Overrides

```
COGITO EZ-DB:R5.3  Gen Level:C90201                  V0014  13:08:27 2002.001
                         EZ-DB EZ-Test CONTROL PANELS
                           RECORD ELEMENT OVERRIDES

  SEO TABLE: PRODSEO  SRL TABLE: PRODSRL  SCHEMA: DSTMPROD VERSION:    1

  RECORD/ELEMENT NAME   REC ID        KEY  DFLT ADJ   SET-FWD  SET-BAK  USEREXIT
 -------------------------------------------------------------------------------
  RCACV1                   126
    ACV1INDA                                Y

 -------------------------------------------------------------------------------
  <PF3> BACK  <PF4> EXIT  <PF7> UP  <PF8> DOWN
 ENTER COMMAND BELOW>
```

This screen illustrates the DB2K Schema Record Element Overrides. It can be selected by entering the number **6** or the mnemonic **SEO** at the command line from any other screen.

The screen is used to display the current SEOTAB definitions.

From this screen you can jump directly to any of the other available screens by entering the number corresponding to the screen or the screen mnemonic at the command line.

2.2.10 Bridge Table Display

```
 COGITO EZ-DB (R5.2)  Tape (C81001)                   V0005  10:58:13 1999.179
                        *** EZ-DB DB-EZ2000 Display ***
                            BRIDGE TABLE ENTRIES
 BRIDGE TABLE NAME :  DB2KBRG


   #   SRLTAB     VIEW  TYPE   DSN/DICTNAME
-------------------------------------------------------------------------------
    1  DB2KSRL    APPL   DSN   IDMSD2.C71001.LOAD120
                         DSN   CHJXG01.EZ-DB43.LOAD120
                         DSN   IDMSD2.R129506.LOADLIB
                         DSN   DBMSD2.R1201.LOADLIB
                         DICT  **DEFAULT**
    2  SRLBRG     APPL   DSN   CHJXG01.EZ-DB43.LOAD120
                         DSN   DBMSD2.R1201.LOADLIB
                         DSN   IDMSD2.R129506.LOADLIB
                         DSN   IDMSD2.C71001.LOAD120
                         DICT  SYSDICT
                         DICT  APPLDICT
-------------------------------------------------------------------------------
 <PF3> BACK  <PF4> EXIT  <PF7> UP  <PF8> DOWN
ENTER COMMAND BELOW>
```

This screen illustrates the EZ-Bridge LOADLIST display. It can be selected by entering the number **7** or the mnemonic **BRG** at the command line from any other screen.

From this screen you can jump directly to any of the other available screens by entering the number corresponding to the screen or the screen mnemonic at the command line.

2.2.11 Business Holidays

```
COGITO EZ-DB:R5.3  Gen Level:C90201                  V0014  12:59:18 1999.070
                        EZ-DB EZ-Test CONTROL PANELS
                             BUSINESS HOLIDAYS

 BUSINESS HOLIDAY TABLE : DB2KBIZ                        WEEKEND : SAT/SUN

 BUSINESS HOLIDAY DATES (YYYYMMDD)
-------------------------------------------------------------------------------
 19970328  19980525  19991228
 19970331  19980831
 19970505  19981225
 19970526  19981228
 19970825  19990101
 19971225  19990402
 19971226  19990405
 19980101  19990503
 19980410  19990531
 19980413  19990830
 19980504  19991227
-------------------------------------------------------------------------------
 <PF3> BACK  <PF4> EXIT  <PF7> UP  <PF8> DOWN
ENTER COMMAND BELOW>
```

This screen illustrates the DB2K Business Holidays table display It can be selected by entering the number 8 or the mnemonic **BIZ** at the command line from any other screen.

The screen is used to display the current EXCTAB definitions.

2.2.12 Using the DB2KSET command

The various **SET** commands which may be issued from the command line of the **DB2K** displays may also be entered directly at the DC/UCF *enter next task code* prompt through the DC/UCF task code **DB2KSET**.

The syntax is equivalent except that the SET keyword is omitted. For example,

```
DB2KSET SRLTAB=DB2KSRL RUNDATE=20010101
```

Available commands are as follows:

DB2KSET SRLTAB=*srltab* RUNDATE=*rundate* <BASEDAT=*basedat*> <BASETIME=*basetime*>
Set *user specific* overrides. Note SRLTAB and RUNDATE are mandatory. BASEDAT and BASETIME are optional.

DB2KSET DISPLAY
Display the current user settings. The current user settings are also shown on the message line of all DB2K screens.

DB2KSET OFF
Switch off current overrides.

DB2KSET ENABLE
Restart EZ-Test IDMS processing for the CA-IDMS CV.

DB2KSET DISABLE
Switch off EZ-Test IDMS processing for the CA-IDMS CV.

EZ2000 will remain active for the life of run-units in progress at the time the DISABLE command was issued. EZ2000 will be inactive for new run-units starting after the DISABLE command was issued. (and vice versa for the ENABLE command).

2.3 EZ2000 in the CICS environment

2.3.1 Implementing the EZ-Test DATE CICS Component

To implement the EZ-Test DATE CICS component, you must first have completed the standard installation steps as described in the Integrated Installation Guide. In addition, note the extra steps required for CICS systems which are included in the member INSTDB2K in the installation source library.

These steps are described below:-

```
1) ADD CICS PPT ENTRIES FOR:

      DB2KCENA ASSEMBLER RESIDENT EXECKEY CICS
      DB2KCEXE ASSEMBLER RESIDENT
      DB2KCEIA ASSEMBLER RESIDENT EXECKEY CICS
      DB2KCEIC COBOL RESIDENT EXECKEY CICS
      DB2KCEIL LE370 RESIDENT EXECKEY CICS
      DB2KCEAS ASSEMBLER EXECKEY CICS
      DB2KCECS COBOL EXECKEY CICS
      DB2KCELS LE370 EXECKEY CICS
      DB2KCICS ASSEMBLER EXECKEY CICS


2) ADD CICS TRANSACTION:

      TRA(DB2K) PROGRAM(DB2KCICS).
      TRA(D2KN) PROGRAM(DB2KCENA).

3) ADD AN ENTRY FOR DB2KCENA IN THE THIRD PHASE OF PLTPI

For example,

      PLT    DFHPLT TYPE=INITIAL,                                        X
                   SUFFIX=CA
             DFHPLT TYPE=ENTRY,PROGRAM=COGIENAB  ENABLE CICS LINK
             COPY   DFHDELIM                     DELIMIT PHASE 2
             DFHPLT TYPE=ENTRY,PROGRAM=IDMSINTC  CICS IDMS INTERFACE
             DFHPLT TYPE=ENTRY,PROGRAM=IDMSINTL  CICS IDMS INTERFACE
             DFHPLT TYPE=ENTRY,PROGRAM=DB2KCENA  ENABLE DB2K EXIT
             DFHPLT TYPE=FINAL
             END

4) MAKE THE EZ-DB LOAD LIBRARY AVAILABLE TO THE CICS RUN TIME SYSTEM
```

Note: Refer to the current INSTDB2K for full details as this member may supercede this documentation.

2.3.2 CICS Run-time JCL

The CICS runtime JCL should be modified as follows: -

- The load library containing the EZ2000 runtime modules and tables should be made available to the CICS run-time system.
- Optionally, you may include the //DB2KCTL parameter file in the CICS run-time JCL. This will be

read by DB2KCENA at system startup time and allows you to specify the names of the tables loaded. Refer to section 2.1.4 for full details of the available parameters.

2.3.3 Controlling the operation of EZ2000 in the CICS environment

The operation of EZ-Test IDMS within the CICS environment is controlled by the supplied CICS transactions, **DB2K** and **DCUF**, and the standard EZ-Test IDMS tables as described elsewhere in this user guide.

2.3.4 Generating EZ-Test IDMS Run-time tables.

The following run time tables are relevant in the CICS environment.

CTLTAB: The CTLTAB is used to define the *Assumed RunDate(s)* which apply to different CLASSES. In practice, the only relevant CLASS which may be used by the CICS system is DBNAME. The user could then use the supplied **DB2K** transaction to specify which DBNAME entry in the CTLTAB should be used for that user or the entire CICS system.

EXCTAB: The EXCTAB defines the names of the CICS transactions which are to be *excluded* from EZ2000 processing. The installation source library contains a sample job (DB2KCEX) which is used to generate the default CICS exclusion table. This may be modified as required.

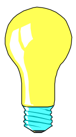
DFCTAB: The Date Format Code table is used by the run time system to perform date translation. The default supplied table is used for this purpose and it should not be necessary to specify this parameter unless you have renamed the default table.

None of the other DB2KCTL runtime parameters should be coded in the CICS environment.

Some of the discussion related to the execution of CICS programs is very similar to the discussion in the previous topic "Executing Batch Programs against a CA-IDMS CV" (**section 2.1.2**). EZ2000 has to perform two distinct functions. Firstly, the user should receive the *Assumed RunDate* and *time* in response to application program date/time requests. Secondly, if the program is retrieving database records from a CA-IDMS CV, the EZ-Test IDMS component needs to recognize the requests in order to adjust the applicable date fields in IDMS records returned to the application programs.

2.3.5 Setting the CA-IDMS DBNAME

To enable a CICS user to set a CA-IDMS DBNAME for a particular terminal without application amendments, the CICS transaction **DCUF** is provided. Note that implementation of the DCUF facility requires modifications to the IDMSINTC and/or IDMSINTL interface. The full description of how to implement the DCUF facility, including the required source modifications to the CA-IDMS CICS interface, is provided in PTF **CD710.002** which can be found in the installation tape PTF library.



The requirement to pass a DBNAME from CICS was originally provided for the **DB-Image** product. Users of that product may already have implemented this feature. The DBNODE parameter is also a requirement of the DB-Image product in special circumstances. Its use requires the implementation of a special *pre-bind exit* (exit-23). Refer to the *DB-Image User Guide* for full details.

Once implemented (as described above) the syntax of the DCUF transaction is as follows:-

`DCUF <SET> DBNAME=``*dbname*` **`DBNODE=`**`*dbnode*`

Note: The **SET** is optional. Either or both of DBNAME and DBNODE may be entered.

DBNAME The CA-IDMS DBNAME is set for a particular terminal to *dbname,* where *dbname* is 1-8 alphanumeric characters which should correspond to a valid DBNAME used by the applicable CA-IDMS CV and also defined in the CTLTAB used by that CV. The *Assumed RunDate* defined for the corresponding DBNAME CLASS in the CTLTAB used by the CA-IDMS CV will determine the dates returned in CA-IDMS records to this user.

All tasks executed from this terminal will use the DBNAME specified until varied by another DCUF transaction. Enter the DCUF transaction with no DBNAME and the DBNAME will no longer be used.

DBNODE This parameter is used by the supplied DCEXIT23 to set a DBNAME for UCF transactions or to override the DBNAME for DB-Image users in special circumstances.

2.3.6 Setting the CICS Date and Time

The CICS *Assumed RunDate* and time is set using the DB2K transaction as follows:

either **DB2K SET** <SYSTEM> **DATE**=*yyyymmdd* <**TIME**=*hhmm*>
or **DB2K SET** <SYSTEM> **DBNAME**=*dbname*

Parameters in <> are optional. The parameters have the following meaning:

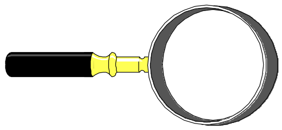
SYSTEM SYSTEM is optional. If specified then other parameters apply to all users under CICS. If omitted, it applies to this terminal only and would override any SYSTEM settings. i.e., you can use SET SYSTEM to set global defaults for everybody and then use SET DATE= etc., to define different values for yourself only.

DATE Specify the required date in YYYYMMDD date format.

TIME Time is optional. Specify the required time in HHMM format. If this parameter is omitted, then the real time is always returned to the user.

DBNAME Specify *dbname* where *dbname* relates to a CLASS=DBNAME entry in the CTLTAB as used by the CICS system. The *Assumed RunDate* will be set according to the applicable CTLTAB entry. Refer to **section 1.2** for further details about the CTLTAB.

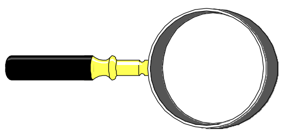
Date/Time affected are EIBTIME and EIBDATE plus the response from an EXEC CICS ASKTIME.



If DATE and TIME are specified then the current values are taken from EIBDATE and EIBTIME and subtracted to give the difference. The difference is then added to EIBDATE and EIBTIME for all subsequent transactions. So you could for example issue the following command:-

```
DB2K SET DATE=19991231 TIME=2359
```

and watch it go over midnight.



You should obviously ensure consistency between the *Assumed RunDate* in affect for the CICS terminal user(s) and the *Assumed RunDate* used by the applicable DBNAME within the CA-IDMS System. This is ensured if you use the same CTLTAB in both systems, and use the

`DB2K SET <system> DBNAME=dbname`

command to control the *Assumed RunDate* within the CICS system.

2.3.7 Other CICS Commands

`DB2K DISPLAY` `<SYSTEM>`

Display current settings for this terminal, or the global settings if the optional SYSTEM keyword is included.

`DB2K STOP` `<SYSTEM>`

Cancel DB2K for this terminal, or SYSTEM if specified.

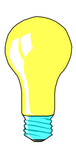
If DB2K SET is used for a terminal, then it stays set for that terminal until a DB2K STOP command is issued.

`D2KN STOP`

Stop all EZ2000 and delete all programs ready for new copy.

`D2KN START`

Reload all programs and restart EZ2000.



In between STOP and START commands the user can issue CEMT commands to new copy any EZ2000 programs defined to CICS. These are DB2KCENA, DB2KCDAT, DB2KCEIA, DB2KCEIC, DB2KCEIL. All other programs involved are loaded by DB2KCENA on the START command using MVS loads. So if maintenance has been applied, or tables regenerated, the user can new copy the changes. DB2KCICS can be new copied at any time using CEMT. There is no need to issue STOP/START.

Technical Notes

3 Technical Notes

3.1 Using BASEDAT and BASETIM

This section contains further information about the optional CTLTAB parameters BASEDAT and BASETIM and explains the circumstances when it is appropriate to use them. Before explaining these parameters, it is worth reviewing how the standard date adjustment is performed when these parameters are not coded.

Generating the Correct *Date Difference*

See also *Dates with no Days* (**Section 3.2**).

Normally the *date difference* (ie. the default number of days to be added to applicable date fields) is calculated as RUNDATE minus *todays date* (where RUNDATE is the Assumed RunDate as defined in the CTLTAB, and *todays date* is usually self-explanatory). However, there are occasions when for some reason the machine may have been IPL'd with a different date. In this case, the *todays date* value returned to EZ2000 will be incorrect, resulting in an incorrect *date difference* being calculated. For example, if *todays date* really is 1 January 1997, but you have IPL'd with a date of 31 December 1999, and you set RUNDATE to 1 January 2000. EZ2000, by default, would calculate a *date difference* of just <u>one day</u> rather than 1095 days (3 years). If you were to use the BASEDAT parameter, setting BASEDAT to 19970101, then EZ2000 would calculate and use the correct date difference of 1095 days.

Keeping an *old* test database up-to-date

It is quite common that a site created a test database many years ago and that this is still the *test-bed* used for application testing. For example, maybe the test database has no data created later than 1994. If the user specifies a RUNDATE of 20000101 (assuming today is 19970101), then EZ2000 would add 3 years to all of the date fields resulting in the dates being bought forward no later than 1997. However, if the User specified a BASEDAT of 19940101, EZ2000 would add 6 years to the dates and therefore make the data truly *current*.

It is obviously possible to use BASEDAT in this way to make old test databases current for testing *today*, not just for *future* RUNDATEs.

Maintaining a consistent view of the database

There is a more subtle use of the BASEDAT parameter which requires a more involved explanation.

Normally, the user would simply specify the CTLTAB parameter RUNDATE which defines the *assumed* "*todays date*". For example, the real "*todays date*" might be **1 January 1997**, but if the user specifies a RUNDATE of 20000101 then, to that user, *todays date* will appear to be **1 January 2000.**

At system initialisation time EZ2000 will normally calculate the difference in days between the RUNDATE and the real system startup date. Using the dates in the above example, the date difference is exactly three years, or 1095 days. This would be the number of days which would be added to any date fields returned in CA-IDMS records (assuming that the user has not specified any non-standard adjustment using the SEOTAB). Similarly, if the user were to modify or store a new record with "todays date" within it, the date would appear to that user as **1 January 2000**, but on the store, EZ2000 would subtract 1095 days, so that on the database physically the date would be stored as **1 January 1997.**

Now consider the implications of this.

Suppose on the same day, a user were to display this record without EZ2000. The date would be displayed as 1 January 1997, which is consistent, because that is *todays* date.

Now suppose that this user is repeating some testing the following *real* day, ie., 2 January 1997, but still using a RUNDATE of 20000101. *Today*, for the user appears to be 1 January 2000, but if he were to re-display that same record, it would contain *yesterdays date.* ie., 31 December 1999. This is because the calculated day difference is now one less, ie., 1094 days.

It could be that this is the desired result, as the record *was* physically stored yesterday. Alternatively, some users may consider this inconsistent. Logically, this record was stored *today* (1 January 2000) and logically *today* is still 1 January 2000! This is a good example of where you could use BASEDAT to maintain a consistent view of a database across an extended time-span.

If BASEDAT is specified in the CTLTAB, then at system initialisation time, instead of subtracting the real startup date, EZ2000 will subtract BASEDAT from RUNDATE to calculate the *date difference*. In this way, the user will circumvent the inconsistent results described above. Consider the following example. Say, todays date is 1 January 1997. the users sets BASEDAT to 19970101 (1 January 1997) and RUNDATE to 20000101 (1 January 2000). The user stores a record with todays date 1 January 2000, which gets physically stored as 1 January 1997. (In this case the date difference is 1095 whether or not we use BASEDAT or system startup date to calculate the difference). The following *real day* (2 January 1997), the user repeats the test still specifying RUNDATE 20000101 and BASEDAT 19970101. By using BASEDAT for the date difference calculation, the date difference is still 1095 days rather than 1094 days. When 1095 days is added to the record (which was physically stored as 1 January 1997) the user will see the record as 1 January 2000, ie. the same as it appeared yesterday!

3.2 Dates with no Days

The rules for date adjustment cater for dates with no days as follows; When the *day difference* is calculated (see above), we also calculate the difference in *Years* and *Months* between the YYYYMMDD values of BASEDAT and RUNDATE, and use the values calculated to adjust the year and month portion of dates that do not contain days. The days do come into this calculation. For example, if BASEDAT is 19970730 and RUNDATE is 20000725, then the difference is 2 years 11 months (the remainder is dropped). If RUNDATE is 20000730, the difference is 3 years 0 months exactly.

The following rules apply:

a) YYYYMM dates - add the year/month difference.
b) MM only - add the MM part of the difference (unlikely ever to be used).
c) YY only - add the YY part of the difference (unlikely ever to be used).

3.3 Special considerations when using YYMM=ONLY

The following tech note was originally issued as CE801.T02.

```
**********************************************************************
*   CZ801.T02 EZ-Test Tech Note                          March 1998 *
*                                                                    *
*   KEYWORDS: EZ-Test YYMM=ONLY                                      *
*                                                                    *
*   YYMM=ONLY may be specified on the #SDETAB entry for a date to   *
*             instruct the EZ-Test software to age the YYMM portion *
*             of a date leaving the days alone.                      *
*                                                                    *
*   This is fine if the day is always 01 or less than 29, however   *
*             inconsistencies will arise with days > 28 as the       *
*             software will not allow an invalid date. If the        *
*             resultant days is higher than the max for the month   *
*             then EZ-Test will set it to the last day of the month.*
*             If the record is then modified the record is stored   *
*             back on the database with the modified days.           *
*                                                                    *
*   Eg INVOICE-DATE=19980131 and we age by 2 Years 1 month then     *
*      INVOICE-DATE=20000231 is invalid so is adjusted to            *
*      INVOICE-DATE=20000229                                         *
*                                                                    *
*   If the record is modified we decrement by 2 Years 1 Month and   *
*      INVOICE-DATE=19980129 is the value stored back on the         *
*      database.                                                     *
*                                                                    *
*   This is a feature of YYMM=ONLY that cannot be prevented as the  *
*      software cannot keep a history of every data occurrence       *
*      aged in case it is modified at some later stage. YYMM=ONLY   *
*      should therefore be used with caution and only when it is    *
*      required and the anomoly descibed above will not occur or    *
*      will not matter to the application.                           *
*                                                                    *
**********************************************************************
```

3.4 Using Date Adjustments User Exits

The SEOTAB is provided as a mechanism for catering for *non-standard* date adjustment for specific record elements. The SEOTAB has a number of options which provide several possible modifications of the standard date adjustment. To provide for even greater flexibility, the SEOTAB also provides the option of specifying a user provided exit routine to be called for a particular record-element. Refer to **section 1.7** for further details of SEOTAB generation and USEREXIT specification. Following is an example of a possible user exit. It demonstrates the function of such an exit, and may be used as a skeleton for creating your own exits if required.

NB. The sample exit shown below can be found in the installation source library member DB2KUXIT.

```
         TITLE 'EZ-Test SAMPLE USER EXIT'
***********************************************************************
*                                                                     *
*        DB2KUXIT: SAMPLE DATE ADJUSTMENT USER EXIT                   *
*                                                                     *
*        ON ENTRY: R0 ==> DFCTAB ENTRY POINT                          *
*                  R1 ==> ISADATE                                     *
*                  R13==> STACK                                       *
*                  R14==> RETURN ADDRESS                              *
```

```
*                 R15==> ENTRY POINT ADDRESS                             *
*                                                                        *
*       ON EXIT   R15 = 0 CONTINUE WITH DATE ADJUSTMENT                  *
*                 R15 = 4 SKIP THIS PARTICULAR DATE                      *
*                                                                        *
*       CALLERS REGISTERS R1 THROUGH R14 MUST BE SAVED AND RESTORED.     *
*                                                                        *
*   NOTES: 1) DB2KDFCA AND DB2KDFCN ARE ALWAYS THE SAME                  *
*          2) DB2KODAT AND DB2KNDAT ARE ALWAYS THE SAME                  *
*          3) IF PROCESSING THE DATE MANUALLY THEN TEST DATEFLG2         *
*             IF DATEBACK IS SET THEN MOVE BACK IN TIME ELSE MOVE        *
*             FORWARD IN TIME.                                           *
*                                                                        *
*   THIS EXAMPLE PROCESSES DATA THAT MAY OR MAY NOT CONTAIN A DATE       *
*   DEPENDING ON THE VALUE OF THE FIRST BYTE OF THE DATA (YOURID).       *
*                                                                        *
*   IF YOURID = 1 THEN THERE IS A DATE AND THE DEFAULT DFCODE USED       *
*                 IN THE #SDETAB DEFINITION FOR THIS ELEMENT IS          *
*                 CORRECT.                                               *
*   IF YOURID = 2 THEN THERE IS NO DATE.                                 *
*   ELSE          THE DATA CONTAINS A DATE BUT OF A DIFFERENT DATE       *
*                 FORMAT CODE.                                           *
*                                                                        *
*   TO PROCESS THIS DATA FIRST CODE A #SDETAB ENTRY WITH                 *
*   #SDETAB EL=YOURID,VALIDATE=NO,DFC= (DEFAULT DFC CODE)                *
*                                                                        *
*   THE USER EXIT IS THEN CALLED WITH THE ADDRESS OF #ISADATE WHICH      *
*   CONTAINS THE ADDRESS OF YOURID AND THE ADDRESS OF THE DEFAULT        *
*   DFCTAB ENTRY. THE USER EXIT TESTS YOURID. IF=1 THEN IT ADJUSTS       *
*   THE ADDRESS OF YOURID TO POINT TO THE REAL DATE INSTEAD OF YOURID *
*   AND EXITS WITH RC 0 TO ALLOW EXTEST TO ADJUST THE DATE. IF=2 THEN *
*   THERE IS NO DATE SO THE RETURN CODE IS SET TO 4 TO FORCE EZ-Test  *
*   TO SKIP THIS OCCURRENCE OF THE DATE. OTHERWISE IT CONTAINS A DATE *
*   BUT OF A DIFFERENT FORMAT SO THE EXIT CHANGES THE ADDRESS OF         *
*   YOURID TO POINT TO THE REAL DATE AND ALSO THE ADDRESS OF THE         *
*   DFCTAB ENTRY TO POINT TO THE CORRECT ENTRY FOR THIS DATE FORMAT.     *
*                                                                        *
**************************************************************************
         START
R0       EQU   0
R1       EQU   1
R2       EQU   2
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R10      EQU   10
R11      EQU   11
R12      EQU   12
R13      EQU   13
R14      EQU   14
R15      EQU   15
DB2KUXIT CSECT
        USING DB2KUXIT,R15          * BASE REG
         B     STRT0000             * BRANCH ROUND EYECATCHER
         DC    CL8'DB2KUXIT'
         DC    C'&SYSDATE '
         DC    C'&SYSTIME '
        DROP  R15
        USING DB2KUXIT,R12          * BASE REG
STRT0000 DS    0H
         SAVE  (14,12)
         LR    R12,R15              * EPA
         LA    R11,18*4(,R13)       * NEW RSA
         ST    R13,4(,R11)          * CHAIN RSAS
         LR    R15,R13              * STACK FIRST ENTRY
         LR    R13,R11              * STACK FIRST ENTRY
         ST    R13,8(,R15)          * FORWARD CHAIN
         TITLE 'EZT - MAIN CONTROL ROUTINE'
**************************************************************************
```

```
*        MAIN CONTROL SECTION                                          *
************************************************************************
         SPACE
MAIN0000 DS    0H
        USING ISADATE,R3
        USING DFC,R4
        USING USERDATE,R5
         LR    R3,R1                   * DATE PARM ADDRESS
         LR    R4,R0                   * DATE FORMAT TABLE
         L     R5,DATEODAT             * ADDRESS OF DATE FIELD
*
         CLI   YOURID,YOURTYP1         * IS IT TYPE 1 ?
         BE    EXITRC0                 * YES SO DEFAULT DFC IS CORRECT
         CLI   YOURID,YOURTYP2         * IS IT TYPE 2 ?
         BE    EXITRC4                 * YES SO IT IS NOT A DATE
*
         LA    R14,NDDMMY4             * NO SO DFCODE=NDDMMY4
         MH    R14,=AL2(DFCCDSLN)      * DFCODE X LENGTH OF ENTRY
         LA    R14,DFCODES(R14)        * ADDRESS CORRECT DFC ENTRY
         ST    R14,DATEDFCA            * UPDATE PARM LIST
         ST    R14,DATEDFCN            * UPDATE PARM LIST
*                                        WITH NEW DFC ENTRY ADDRESS
EXITRC0  DS    0H
         LA    R5,YOURDATE             * ADDRESS THE REAL DATE
         ST    R5,DATEODAT             * UPDATE THE DATE ADDRESS
         ST    R5,DATENDAT             * UPDATE IN PLACE
         SR    R15,R15                 * RC=0, PROCESS THE DATE
         B     MAINEXIT                * RETURN TO CALLER
*
EXITRC4  DS    0H
         LA    R15,4                   * RC=4, SKIP THIS DATE
*
MAINEXIT DS    0H
         L     R13,4(R13)              * BACK UP RSA
         L     R14,12(R13)             * RETURN ADDRESS
         LM    R0,R12,20(R13)          * CALLERS REGS
         BR    R14                     * RETURN TO MVS
         LTORG
************************************************************************
*        USER DATE FIELD DSECT                                         *
*             IF YOURID IS SET TO 1 THEN THE DATE IS YYYYMMDD          *
*             IF YOURID IS SET TO 2 THEN IT IS NOT A DATE              *
*             ELSE IT IS DDMMYYYY                                      *
************************************************************************
USERDATE DSECT
YOURID   DS    CL1                      * DATE ID
YOURTYP1 EQU   C'1'                       TYPE 1 YYYYMMDD
YOURTYP2 EQU   C'2'                       TYPE 2 NOT A DATE
YOURTYP3 EQU   C'3'                       TYPE 3 DDMMYYYY
YOURDATE DS    CL8                      * DATE
************************************************************************
*        DSECTS                                                        *
************************************************************************
         COPY  #ISADATE
         COPY  #DFCDS
         COPY  #SRLDS
         COPY  #K2CDS
         END
```

A number of sample exits are included in the installation source library as follows:-

DB2KXIT1
An element contains XXMMDDYY where MMDDYY is a date and XX must always equal YY. The exit sets XX to YY after aging.

DB2KXIT2
This tests a date for a particular value(s). If it matches the value(s) then the date is ignored. This caters for "special" dates that are not real dates but merely indicate something to the application, e.g.,

20991231 = end of file.

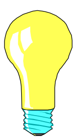
DB2KXIT3
A date is physically held as MMYY but is part of a sort key sorted as YYMM. The exit controls the aging of the sort key when it is held as YYMM in working storage and is used for OBTAIN USING etc. The date format code is changed from MMYY to YYMM.

DB2KXIT4
This exit controls the aging of a single element that will contain a date if a control field is set to a particular value, else it is not a date and is to be ignored.

DB2KXIT5
This exit controls the processing required for a record that may contain many dates in different locations and with different date formats depending on some control values in the database record. The exit identifies each date occurrence in turn, and it's date format, and calls a supplied routine to age the date. When all dates have been processed the exit returns to caller.

This sample exit is supplied in Cobol. The instructions for generating this exit are contained in **INSTXIT5.**

FISCUXIT
This exit deals with all fiscal dates. It identifies the type of fiscal date, eg YYYYWW, and converts it to standard format, eg YYYYDDD. Control is then returned to allow the date to be processed after which it is then invoked again to convert the processed date back into it's original fiscal format.



This exit differs from the other exits in that there is no need to generate an SEOTAB entry. EZ-Test IDMS will attempt to load FISCUXIT at initialization time. If successful then this module is called before and after date conversion for any fiscal dates.

INSTXIT5
Instructions for compiling and link editing DB2KXIT5. If the exit is coded in Cobol, as in the example, then a new version of IGZEOPT must be generated and the exit linked as instructed. If the exit is generated in Assembler then there is no need to generate IGZEOPT but the link edit should be as specified here, with the exception that the INCLUDE for IGZEOPT should be omitted.

3.5 Using EZ2000 with other “Date Simulation” products

It is possible that your site has already purchased another proprietary date simulation product prior to the installation of EZ2000, and you are concerned about the interaction or compatibility of these products with EZ2000. We are aware of the functionality of such products as HOURGLASS and TICKTOCK to name just two, though this in no way implies that there are not other products available.

It is important to distinguish the difference between such products and EZ2000. These other products are only concerned with adjustment of date and time as returned to program requests. For example, through TIME SVC requests, CICS EIBDATE etc. In this respect, any overlap with the functionality of EZ2000 would be with the EZ-Test DATE SVC HOOK and CICS component. Such products are not concerned with the adjustment of dates in database records, nor do they support adjustment of date/time requests within the CA-IDMS address space. (The only product which we are aware of which provides this support is “Date Simulator” from HSL).

This means that you actually have a choice as to whether to use the EZ2000 product in its entirety, or to use the EZ-Test IDMS logical ager in conjunction with another proprietary product for the non-IDMS system date adjustment. In this respect, you would need to compare the functionality of the applicable product with that of EZ-Test to decide which better meets your needs. Obviously, although there will be similarities in the function, the actual implementation and set-up will be different. In some cases, the choice will be made for you based upon other criteria. If you do choose to use another product in conjunction with EZ-Test IDMS, then the following points must be taken into account:-

- There is no need to run the DB2KSET utility to install the SVC Hook.
- If you are a CICS user, and your other product includes CICS support for EXEC CICS ASKTIME, EIBDATE etc., you may choose not to use the EZ-Test DATE CICS support for date/time manipulation within CICS . In which case you would omit the CICS specific implementation instructions. Assuming that your CICS programs are accessing CA-IDMS records, you would complete the rest of the installation as normal. In addition, you may still wish to define the DCUF transaction in order to be able to pass a DBNAME to the CA-IDMS address space.
- Generate the CA-IDMS environment as described in this User Guide. Your CTLTAB definition will only require TYPE=IDMS entries.
- An important point to consider is the way EZ-Test IDMS works out the number of days to adjust dates in database records. A comprehensive discussion of this subject can be found in section 3.1 The point to note is that, if you set the EZ-Test RUNDATE to 20000101, but you are using PRODUCTX to set the IPL date (for example) to 20000101, then EZ-Test IDMS will compute a day difference of zero and no date adjustment will take place in database records! The way around this is to use the BASEDAT parameter also described in section 3.1.

3.6 How Business days are calculated

The following technical note regarding business day calculation was originally issued as PTF technical note **CE801.T15**.

```
**********************************************************************
*   CE801.T15 EZ-Test Tech Note                      October 1998 *
*                                                                    *
*   KEYWORDS: EZ-Test BIZDAY                                         *
*                                                                    *
**********************************************************************
*                                                                    *
*   This Tech Note describes how EZ-Test2000 aging takes place       *
*   when the parameter BIZDAY=YES is specified for a date field.     *
*                                                                    *
**********************************************************************
*                                                                    *
*  Regular default aging (when BIZDAY=NO is specified or allowed     *
*  to default) works by calculating the number of days between       *
*  the rundate and basedate and then adding that number of days      *
*  to each date.  When BIZDAY=YES is specified, the number of        *
*  days to age by is worked out differently.  Unless this is         *
*  properly understood, apparently confusing results can be seen.    *
*                                                                    *
*  The underlying principle is that by specifying BIZDAY=YES         *
*  you are saying that you want the amount of aging to be by the     *
*  number of business days between the rundate and basedate. This    *
*  is not the same as aging normally and then subsequently           *
*  adjusting to ensure that the resultant date falls on a            *
*  business day.                                                     *
*                                                                    *
*  Business day aging is done this way to ensure that dates that     *
*  are a certain number of business days apart will keep that        *
*  relationship after aging.  The process also works both ways;      *
*  ie forward aging on a read and reverse aging for a modify will    *
```

```
*   produce consistant results.  This would not be the case if the    *
*   process was one of simple moving the date forward to the next     *
*   business day.                                                     *
*                                                                     *
*   An effect of this that should be appreciated is that a            *
*   BIZDAY=YES date wil not necessarily age to the same date as a     *
*   BIZDAY=NO date.                                                   *
*                                                                     *
*   One other point to consider is that the basedate and rundate      *
*   should both be business dates themselves.                         *
*                                                                     *
*                                                                     *
*   An example.                                                       *
*   Basedate is 7th April 1998                                        *
*   Rundate is 14th April 1998                                        *
*                                                                     *
*   Default aging is by 7 days.                                       *
*   Business days aging is by 3 working days as the 10th and 13th     *
*   are holidays and there is one weekend in between.                 *
*                                                                     *
*   Date to be aged is 1st April 1998.                                *
*   With BIZDAY=NO, it is aged to 8th April 1998.                     *
*   With BIZDAY=YES, it is aged to 6th April 1998.                    *
*                                                                     *
*   In reality of course, aging will be by a much larger amount       *
*   but this example serves to illustrate the principle.              *
*                                                                     *
***********************************************************************
```

Appendices

4 Appendices

4.1 Sample JCL Included on Installation Tape

4.1.1 INSTDB2K

The Installation source library provides various members containing skeleton JCL which may be modified and used as the basis for all of the necessary jobs which must be run as part of the EZ2000 system generation. The member INSTDB2K is an installation overview and also serves as a cross-reference of the other applicable members in the install source library. As a reference, it is listed here. Some of the other members are also included in this appendix. For the contents of the other members, refer to the source library itself.

```
//*                                                                      00010002
//* EZ-Test-2000 Customisation Guide                                     00010200
//*                                                                      00010300
//* The minimum customisation required for EZ-Test-2000 is the           00010400
//* generation of a DB2KCTL, DB2KSDE and DB2KSRL table. Listed           00010500
//* below are the member names of jobs in the install SRCLIB             00010600
//* for each of the mandatory and optional functions. Please             00010700
//* refer to the product guide for instructions on how to use            00010800
//* each of these tables.                                                00010900
//*                                                                      00011000
//* Prior to running these jobs you should:                              00011100
//*                                                                      00011200
//* 1. Change all occurrences of EZ-DB.cdmslib to the name of the        00011310
//*    run time library containing the EZ-DB load modules.               00011400
//* 2. Change all occurrences of EZ-DB.srclib to the name of the         00011500
//*    install SRCLIB downloaded from the EZ-DB install tape.            00011600
//* 3. Change all occurrences of caidms.loadlib to the name of the       00011700
//*    run time library containing your CA-IDMS software.                00011800
//* 4. Change all occurrences of caidms.maclib to the name of            00011900
//*    CA-IDMS macro library.                                            00012000
//*                                                                      00012300
//* To implement EZ-Test-2000 add EZ-DB.cdmslib to your run time         00012410
//* JCL ahead of caidms.loadlib. It should be added to CDMSLIB           00012500
//* if in use, else STEPLIB.                                             00012600
//*                                                                      00012700
//* EZ-Test-2000 Table Generation Mandatory Steps                        00012800
//* ............................................                         00012900
//*                                                                      00013000
//* DB2KCTL: Generate the DB2KCTL file containing the run time           00013100
//*          control parameters for EZ-Test-2000.                        00013200
//* DB2KSDE: Generate an SDE table of date element names for             00013300
//*          one or more schema/versions.                                00013400
//* DB2KSRL: Generate an SRL table of date element data for              00013500
//*          one or more schema/versions.                                00013600
//*                                                                      00013700
//* EZ-Test-2000 Table Generation Optional Steps                         00013800
//* ...........................................                          00013900
//*                                                                      00014000
//* DB2KSEO: Generate an SEO table of overrides to the default           00014100
//*          action for one or more date elements in the SRL table.      00014200
//* DB2KDFC: Generate additional date format codes.                      00014300
//* DB2KEXC: Modify the default list of IDMS tasks excluded from         00014401
//*          EZ-Test-2000.                                               00014500
//* DB2KCEX: Modify the default list of CICS tasks excluded from         00014601
//*          EZ-Test-2000.                                               00014700
//* DB2KMON: Modify the default list of alphabetic month codes.          00014800
//*                                                                      00014900
```

```
//* EZ-Test-2000 Timer SVC Intercept Install
//* ---------------------------------------
//*
//* 1) Before running the optional steps to install the Timer SVC
//*    Intercept, you must ensure that the Timer SVC intercept programs
//*    are being run from an authorised library.
//*
//*    If you are unable to make the EZ-DB LOADLIB authorised, you must
//*    copy the minimum of the following modules to an authorised
//*    library before running DB2KSET.
//*
//*    Modules which must reside in authorised library:
//*
//*               DB2KANC
//*               DB2KHKI
//*               DB2KHKO
//*               DB2KLOD
//*               DB2KREP
//*               DB2KSET
//*               DB2KSVC
//*               DB2KUAN
//*
//* 2) Run the following steps to install the Timer SVC Intercept
//*
//* DB2KBTCH: Generate a DB2KCTL table for Batch use.
//* DB2KSVCI: Run DB2KSET to install the Timer SVC Intercept hook.
//*
//* 3) You can run the following step to remove the Timer SVC Intercept
//*
//* DB2KSVCO: Run DB2KSET to remove the Timer SVC Intercept
//*
//* NOTE: The Timer SVC Intercept must be reinstalled after each IPL
//*       by running DB2KSVCI again.
//*
//* EZ-Test-2000 CICS Install
//* ------------------------
//*
//* 1) ADD CICS PPT ENTRIES FOR:
//*
//*          DB2KCENA ASSEMBLER RESIDENT EXECKEY CICS
//*          DB2KCEXE ASSEMBLER RESIDENT
//*          DB2KCEIA ASSEMBLER RESIDENT EXECKEY CICS
//*          DB2KCEIC COBOL RESIDENT EXECKEY CICS
//*          DB2KCEIL LE370 RESIDENT EXECKEY CICS
//*          DB2KCEAS ASSEMBLER EXECKEY CICS
//*          DB2KCECS COBOL EXECKEY CICS
//*          DB2KCELS LE370 EXECKEY CICS
//*          DB2KCICS ASSEMBLER EXECKEY CICS
//*
//* 2) ADD CICS TRANSACTION:
//*
//*          TRA(DB2K) PROGRAM(DB2KCICS).
//*          TRA(D2KN) PROGRAM(DB2KCENA).
//*
//* 3) ADD AN ENTRY FOR DB2KCENA IN THE THIRD PHASE OF PLTPI.
//*     For example
PLT     DFHPLT TYPE=INITIAL,                                           X
               SUFFIX=CA
        DFHPLT TYPE=ENTRY,PROGRAM=COGIENAB     ENABLE CICS LINK
        COPY   DFHDELIM              DELIMIT PHASE 2
        DFHPLT TYPE=ENTRY,PROGRAM=IDMSINTC     CICS IDMS INTERFACE
        DFHPLT TYPE=ENTRY,PROGRAM=IDMSINTL     CICS IDMS INTERFACE
        DFHPLT TYPE=ENTRY,PROGRAM=DB2KCENA     ENABLE DB2K EXIT
        DFHPLT TYPE=FINAL
        END
//*
//* 4) MAKE THE EZ-DB LOAD LIBRARY AVAILABLE TO THE CICS RUN TIME
//*    SYSTEM
//*
```

4.1.2 DB2KCTL

```
//jobcard
//*
//* This JCL may be used to create a DB2KCTL table. However see
//* the following members to generate table using the default table
//* names for the various environments.
//*
//* CTLBTCH : Batch JOBS and SVC Intercept
//* CTLCICS : CICS
//* CTLDB2  : DB2
//* CTLIDMS : IDMS
//*
//DB2KCTL EXEC PGM=IEV90,PARM='XREF(SHORT),SYSPARM(12.0)'
//SYSLIB   DD DSN=EZ-DB.SRCLIB,DISP=SHR
//         DD DSN=CAIDMS.MACLIB,DISP=SHR
//         DD DSN=SYS1.MACLIB,DISP=SHR
//         DD DSN=SYS1.MODGEN,DISP=SHR
//         DD DSN=SYS1.AMODGEN,DISP=SHR
//SYSUT1   DD DSN=&&SYSUT1,UNIT=VIO,SPACE=(1700,(600,100))
//SYSUT2   DD DSN=&&SYSUT2,UNIT=VIO,SPACE=(1700,(316,50))
//SYSUT3   DD DSN=&&SYSUT3,UNIT=VIO,SPACE=(1700,(316,50))
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSLIN   DD DUMMY
//SYSPUNCH DD DSN=&&OBJ,UNIT=SYSDA,DISP=(NEW,PASS),
//         DCB=(RECFM=F,LRECL=80,BLKSIZE=80),SPACE=(CYL,(6),,CONTIG)
//SYSIN    DD *
CTLTAB   #CTLTAB TYPE=IDMS,                                          X
               XTASK=YES,                                            X
               ERUSID=UA,                                            X
               ERUSJOB=4
*
         #CTLTAB ENTRY=DBEZ2000,                                     X
               SRLTAB=srltab-1,                                      X
               RUNDATE=19991231,                                     X
               ACTION=INCLUDE
         #CTLTAB SRLTAB=srltab-2           * if required
         #CTLTAB SRLTAB=srltab-3-etc       * if required
         #CTLTAB CLASS=USER,                                         X
               NAME=user-name
*
         #CTLTAB ENTRY=DBEZ2000,                                     X
               SRLTAB=srltab-1,                                      X
               RUNDATE=20000101,                                     X
               ACTION=INCLUDE
         #CTLTAB CLASS=DBNAME,OR,                                    X
               NAME=dbname-1
         #CTLTAB CLASS=DBNAME,OR,                                    X
               NAME=dbname-2-etc
*
         #CTLTAB ENTRY=LAST
//*
//DB2KCTL EXEC PGM=IEWL,REGION=256K,COND=(4,LT),
//         PARM='LIST,XREF,MAP,AMODE=31,RMODE=ANY'
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSPRINT DD SYSOUT=*
//SYSLMOD  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//SYSLIN   DD DSN=&&OBJ,DISP=(OLD,DELETE)
//         DD *
 NAME DB2KCTL(R)
//*
```

4.1.3 DB2KSDE

```
//jobcard
//*
//DB2KSDE EXEC PGM=IEV90,PARM='XREF(SHORT),SYSPARM(12.0)'
//SYSLIB   DD DSN=EZ-DB.SRCLIB,DISP=SHR
//         DD DSN=CAIDMS.MACLIB,DISP=SHR
//         DD DSN=SYS1.MACLIB,DISP=SHR
//         DD DSN=SYS1.MODGEN,DISP=SHR
//         DD DSN=SYS1.AMODGEN,DISP=SHR
//SYSUT1   DD DSN=&&SYSUT1,UNIT=VIO,SPACE=(1700,(600,100))
//SYSUT2   DD DSN=&&SYSUT2,UNIT=VIO,SPACE=(1700,(316,50))
//SYSUT3   DD DSN=&&SYSUT3,UNIT=VIO,SPACE=(1700,(316,50))
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSLIN   DD DUMMY
//SYSPUNCH DD DSN=&&OBJ,UNIT=SYSDA,DISP=(NEW,PASS),
//         DCB=(RECFM=F,LRECL=80,BLKSIZE=80),SPACE=(CYL,(6),,CONTIG)
//SYSIN    DD *
sdename  #SDETAB ENTRY=FIRST,SCHEMA=schema,SRLNAME=srlname
         #SDETAB EL=element-name,DFC=date-format-code
         #SDETAB ENTRY=LAST
//*
//DB2KSDE EXEC PGM=IEWL,REGION=256K,COND=(4,LT),
//         PARM='LIST,XREF,MAP,AMODE=31,RMODE=ANY'
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSPRINT DD SYSOUT=*
//SYSLMOD  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//SYSLIN   DD DSN=&&OBJ,DISP=(OLD,DELETE)
//         DD *
 NAME sdename(R)
/*
```

4.1.4 DB2KSEO

```
//jobcard
//*
//DB2KSEO EXEC PGM=IEV90,PARM='XREF(SHORT),SYSPARM(12.0)'
//SYSLIB   DD DSN=EZ-DB.SRCLIB,DISP=SHR
//         DD DSN=CAIDMS.MACLIB,DISP=SHR
//         DD DSN=SYS1.MACLIB,DISP=SHR
//         DD DSN=SYS1.MODGEN,DISP=SHR
//         DD DSN=SYS1.AMODGEN,DISP=SHR
//SYSUT1   DD DSN=&&SYSUT1,UNIT=VIO,SPACE=(1700,(600,100))
//SYSUT2   DD DSN=&&SYSUT2,UNIT=VIO,SPACE=(1700,(316,50))
//SYSUT3   DD DSN=&&SYSUT3,UNIT=VIO,SPACE=(1700,(316,50))
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSLIN   DD DUMMY
//SYSPUNCH DD DSN=&&OBJ,UNIT=SYSDA,DISP=(NEW,PASS),
//         DCB=(RECFM=F,LRECL=80,BLKSIZE=80),SPACE=(CYL,(6),,CONTIG)
//SYSIN    DD *
seoname  #SEOTAB SRLTAB=srlname
*
         #SEOTAB RECORD=record-name-1,ID=record-id
         #SEOTAB ELEMENT=element-name-1,                               X
               ADJUST=FORWARD,                                         X
               BY=7
         #SEOTAB ELEMENT=element-name-2,                               X
               ADJUST=FORWARD,                                         X
               BY=7
*
         #SEOTAB RECORD=record-name-2,ID=record-id
         #SEOTAB ELEMENT=element-name-3,                               X
               ADJUST=BACK,                                            X
               BY=14
*
         #SEOTAB ENTRY=LAST
//*
//DB2KSEO EXEC PGM=IEWL,REGION=256K,COND=(4,LT),
//         PARM='LIST,XREF,MAP,AMODE=31,RMODE=ANY'
//SYSUT1   DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSPRINT DD SYSOUT=*
//SYSLMOD  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//SYSLIN   DD DSN=&&OBJ,DISP=(OLD,DELETE)
//         DD *
 NAME DB2KSEO(R)
/*
```

4.1.5 DB2KDFC

```
//jobcard
//*
//DB2KDFC EXEC PGM=IEV90,PARM='XREF(SHORT),SYSPARM(12.0)'
//SYSLIB   DD DSN=EZ-DB.SRCLIB,DISP=SHR
//         DD DSN=CAIDMS.MACLIB,DISP=SHR
//         DD DSN=SYS1.MACLIB,DISP=SHR
//         DD DSN=SYS1.MODGEN,DISP=SHR
//         DD DSN=SYS1.AMODGEN,DISP=SHR
//SYSUT1   DD DSN=&&SYSUT1,UNIT=VIO,SPACE=(1700,(600,100))
//SYSUT2   DD DSN=&&SYSUT2,UNIT=VIO,SPACE=(1700,(316,50))
//SYSUT3   DD DSN=&&SYSUT3,UNIT=VIO,SPACE=(1700,(316,50))
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSLIN   DD DUMMY
//SYSPUNCH DD DSN=&&OBJ,UNIT=SYSDA,DISP=(NEW,PASS),
//         DCB=(RECFM=F,LRECL=80,BLKSIZE=80),SPACE=(CYL,(6),,CONTIG)
//SYSIN    DD *
*******************************************************************
*                                                                 *
*   THIS IS THE PREDEFINED LIST OF DATE FORMAT CODES FOR DB-2000. *
*                                                                 *
*   USERS MUST NOT CHANGE THESE CODES OR THEIR ORDER.             *
*                                                                 *
*   ADDITIONAL CODES FOR FORMATS NOT COVERED HERE MUST BE ADDED   *
*   TO THE END OF THIS LIST.                                      *
*                                                                 *
*   C=CENTURY Y=YEAR M=MONTH D=DAY 0=ZERO /=ANY SPECIAL CHARACTER *
*                                                                 *
*   PIC=COMP-3     PACKED DECIMAL FORMAT  COMP-3                  *
*       COMP       BINARY                 COMP                    *
*       DISPLAY    ALPHANUMERIC CHARACTER DISPLAY                 *
*       HEX        HEX                    ASSEMBLER ONLY.         *
*                                                                 *
*   E.G. PL4'1999365'   PIC=COMP-3,FORMAT=YYYYDDD                 *
*        C'01JAN1999'   PIC=DISPLAY,FORMAT=DDMMMYYYY              *
*        F'1999365'     PIC=COMP,FORMAT=YYYYDDD                   *
*        X'01999365'    PIC=HEX,FORMAT=0YYYYDDD                   *
*                                                                 *
*   NOTE THAT FOR PIC=HEX, WHICH IS AN UNUSUAL DATE FORMAT AS IT  *
*   CAN ONLY BE PROCESSED BY ASSEMBLER PROGRAMS, ANY LEADING      *
*   ZEROES MUST BE SPECIFIED IN THE FORMAT CODE.                  *
*                                                                 *
*******************************************************************
*
         #DFCTAB PIC=COMP-3,FORMAT=SQLDATE
         #DFCTAB PIC=COMP-3,FORMAT=0CYYDDD
         #DFCTAB PIC=COMP-3,FORMAT=YYYYDDD
         #DFCTAB PIC=COMP-3,FORMAT=YYDDD
         #DFCTAB PIC=COMP-3,FORMAT=00YYDDD
         #DFCTAB PIC=COMP-3,FORMAT=0YYDDDC
*
         #DFCTAB PIC=COMP-3,FORMAT=CYYMMDD
         #DFCTAB PIC=COMP-3,FORMAT=0YYYYMMDD
         #DFCTAB PIC=COMP-3,FORMAT=0YYMMDD
*
         #DFCTAB PIC=COMP-3,FORMAT=DDD
         #DFCTAB PIC=COMP-3,FORMAT=DDDDD
         #DFCTAB PIC=COMP-3,FORMAT=DDDDDDD
*
         #DFCTAB PIC=COMP-3,FORMAT=0YY
         #DFCTAB PIC=COMP-3,FORMAT=CYY
*
         #DFCTAB PIC=HEX,FORMAT=CYYDDD
         #DFCTAB PIC=HEX,FORMAT=0YYYYDDD
         #DFCTAB PIC=HEX,FORMAT=0YYDDD
         #DFCTAB PIC=HEX,FORMAT=000YYDDD
         #DFCTAB PIC=HEX,FORMAT=YYDDDC
         #DFCTAB PIC=HEX,FORMAT=00YYDDDC
*
```

```
         #DFCTAB PIC=HEX,FORMAT=0CYYMMDD
         #DFCTAB PIC=HEX,FORMAT=YYYYMMDD
         #DFCTAB PIC=HEX,FORMAT=YYMMDD
*
         #DFCTAB PIC=HEX,FORMAT=DDDD
         #DFCTAB PIC=HEX,FORMAT=DDDDDD
         #DFCTAB PIC=HEX,FORMAT=DDDDDDDD
*
         #DFCTAB PIC=DISPLAY,FORMAT=YYDDD
         #DFCTAB PIC=DISPLAY,FORMAT=CYYDDD
         #DFCTAB PIC=DISPLAY,FORMAT=YYDDDC
         #DFCTAB PIC=DISPLAY,FORMAT=YYYYDDD
*
         #DFCTAB PIC=DISPLAY,FORMAT=YYMMDD
         #DFCTAB PIC=DISPLAY,FORMAT=DDMMYY
         #DFCTAB PIC=DISPLAY,FORMAT=MMDDYY
         #DFCTAB PIC=DISPLAY,FORMAT=CYYMMDD
         #DFCTAB PIC=DISPLAY,FORMAT=DDMMYYC
         #DFCTAB PIC=DISPLAY,FORMAT=MMDDYYC
         #DFCTAB PIC=DISPLAY,FORMAT=YYYYMMDD
         #DFCTAB PIC=DISPLAY,FORMAT=DDMMYYYY
         #DFCTAB PIC=DISPLAY,FORMAT=MMDDYYYY
*
         #DFCTAB PIC=DISPLAY,FORMAT=YYMMMDD
         #DFCTAB PIC=DISPLAY,FORMAT=DDMMMYY
         #DFCTAB PIC=DISPLAY,FORMAT=MMMDDYY
         #DFCTAB PIC=DISPLAY,FORMAT=CYYMMMDD
         #DFCTAB PIC=DISPLAY,FORMAT=DDMMMYYC
         #DFCTAB PIC=DISPLAY,FORMAT=MMMDDYYC
         #DFCTAB PIC=DISPLAY,FORMAT=YYYYMMMDD
         #DFCTAB PIC=DISPLAY,FORMAT=DDMMMYYYY
         #DFCTAB PIC=DISPLAY,FORMAT=MMMDDYYYY
*
         #DFCTAB PIC=DISPLAY,FORMAT=DDDD
         #DFCTAB PIC=DISPLAY,FORMAT=DDDDD
         #DFCTAB PIC=DISPLAY,FORMAT=DDDDDD
         #DFCTAB PIC=DISPLAY,FORMAT=DDDDDDD
         #DFCTAB PIC=DISPLAY,FORMAT=DDDDDDDD
*
         #DFCTAB PIC=DISPLAY,FORMAT=YY/MM/DD
         #DFCTAB PIC=DISPLAY,FORMAT=DD/MM/YY
         #DFCTAB PIC=DISPLAY,FORMAT=MM/DD/YY
         #DFCTAB PIC=DISPLAY,FORMAT=YYYY/MM/DD
         #DFCTAB PIC=DISPLAY,FORMAT=DD/MM/YYYY
         #DFCTAB PIC=DISPLAY,FORMAT=MM/DD/YYYY
*
         #DFCTAB PIC=DISPLAY,FORMAT=YY/MMM/DD
         #DFCTAB PIC=DISPLAY,FORMAT=DD/MMM/YY
         #DFCTAB PIC=DISPLAY,FORMAT=MMM/DD/YY
         #DFCTAB PIC=DISPLAY,FORMAT=YYYY/MMM/DD
         #DFCTAB PIC=DISPLAY,FORMAT=DD/MMM/YYYY
         #DFCTAB PIC=DISPLAY,FORMAT=MMM/DD/YYYY
*
         #DFCTAB PIC=COMP,FORMAT=CYYDDD
         #DFCTAB PIC=COMP,FORMAT=YYYYDDD
         #DFCTAB PIC=COMP,FORMAT=YYDDD
         #DFCTAB PIC=COMP,FORMAT=YYDDDC
*
         #DFCTAB PIC=COMP,FORMAT=CYYMMDD
         #DFCTAB PIC=COMP,FORMAT=YYYYMMDD
         #DFCTAB PIC=COMP,FORMAT=YYMMDD
*
         #DFCTAB PIC=COMP,FORMAT=DDD
         #DFCTAB PIC=COMP,FORMAT=DDDD
         #DFCTAB PIC=COMP,FORMAT=DDDDD
         #DFCTAB PIC=COMP,FORMAT=DDDDDD
         #DFCTAB PIC=COMP,FORMAT=DDDDDDD
         #DFCTAB PIC=COMP,FORMAT=DDDDDDDD
*
         #DFCTAB PIC=DISPLAY,FORMAT=YYYY
         #DFCTAB PIC=DISPLAY,FORMAT=YY
*******************************************************************
*   SPECIAL DATE FORMATS WITH EMBEDDED SIGN. THESE ARE REALLY TWO *
```

```
*   CONTIGUOUS PACKED FIELDS: EG. PL2'CYY',PL2'DDD'                 *
********************************************************************
         #DFCTAB PIC=COMP-3,FORMAT=0YYSDDD
         #DFCTAB PIC=COMP-3,FORMAT=CYYSDDD
         #DFCTAB PIC=COMP-3,FORMAT=0YYYYSDDD
********************************************************************
*   CICS ABSOLUTE SECONDS SINCE 1900                                *
********************************************************************
         #DFCTAB PIC=COMP-3,FORMAT=CICSABS
********************************************************************
*   YET MORE DATE FORMATS
********************************************************************
         #DFCTAB PIC=DISPLAY,FORMAT=YYMM
         #DFCTAB PIC=DISPLAY,FORMAT=MM
********************************************************************
*   FISCAL DATE FORMATS CONTAINING PERIOD AND WEEK OR QUARTER
********************************************************************
         #DFCTAB PIC=DISPLAY,FORMAT=YYWW       GREGORIAN WEEK OF YEAR
         #DFCTAB PIC=DISPLAY,FORMAT=YYWW       FISCAL WEEK OF YEAR
         #DFCTAB PIC=DISPLAY,FORMAT=YYPP       FISCAL PERIOD
         #DFCTAB PIC=DISPLAY,FORMAT=YYPPW      FISCAL DATE NO DAY
         #DFCTAB PIC=DISPLAY,FORMAT=YYPPWD     FISCAL DATE WITH DAY
         #DFCTAB PIC=DISPLAY,FORMAT=YYQ        FISCAL QUARTER
         #DFCTAB PIC=DISPLAY,FORMAT=YYMMDD     FISCAL QUARTER BEGIN
         #DFCTAB PIC=DISPLAY,FORMAT=YYMMDD     FISCAL QUARTER END
********************************************************************
*   END OF THE PREDEFINED LIST OF DATE FORMAT CODES FOR DB-2000.   *
*   USER DEFINED ENTRIES MUST BE ADDED HERE BEFORE ENTRY=LAST.     *
********************************************************************
*        #DFCTAB PIC=,FORMAT=          USER DEFINED ENTRY
*
         #DFCTAB ENTRY=LAST
//*
//DB2KDFC EXEC PGM=IEWL,REGION=256K,COND=(4,LT),
//          PARM='LIST,XREF,MAP,AMODE=31,RMODE=ANY'
//SYSUT1    DD UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSPRINT DD SYSOUT=*
//SYSLMOD  DD DSN=EZ-DB.CDMSLIB,DISP=SHR
//SYSLIN    DD DSN=&&OBJ,DISP=(OLD,DELETE)
//          DD *
 NAME DB2KDFC(R)
//*
```

4.2 Date Format Code DSECT (#DFCDS)

```
*********************************************************************
*   #DFCDS  : DB-2000 DATE FORMAT CODE DSECT                        *
*********************************************************************
         SPACE
DFC      DSECT
DFCEYE   DS    CL4                 * EYE CATCHER DFC*
DFCNEXT  DS    A                   * NEXT, ALWAYS ZERO
DFCNAME  DS    CL8                 * TABLE NAME
DFCEND   DS    A                   * ADDRESS OF END OF TABLE
         DS    A                   * SPARE
DFCHDSLN EQU   *-DFC               * LENGTH OF HEADER
*
DFCODES  DS    0F                  * START OF DATE FORMAT CODES
DFCODE   DS    CL24                * DATE FORMAT CODE
DFCODE#  DS    H                   * DATE FORMAT CODE NUMBER
DFCDFMLN DS    H                   * DATE FORMAT CODE LENGTH
DFCTDTLN DS    H                   * TARGET DATE FIELD LENGTH
DFCCCO   DS    H                   * CENTURY OFFSET IN MASK
DFCCCL   DS    H                   * CENTURY LENGTH
DFCYYO   DS    H                   * YEAR OFFSET
DFCYYL   DS    H                   * YEAR LENGTH
DFCMMO   DS    H                   * MONTH OFFSET (CAN ALSO BE PP OR Q)
DFCMML   DS    H                   * MONTH LENGTH
DFCDDO   DS    H                   * DAY OFFSET (CAN ALSO BE WW W OR WD)
DFCDDL   DS    H                   * DAY LENGTH
DFCPIC   DS    X                   * DATE PICTURE
DFCPICC  EQU   X'80'                 CHARACTER
DFCPICP  EQU   X'40'                 PACKED DECIMAL
DFCPICX  EQU   X'20'                 UNSIGNED PACKED DECIMAL
DFCPICB  EQU   X'10'                 BINARY
DFCPSQL  EQU   X'08'                 SQL
DFCPPW   EQU   X'04'                 MM AND DD ARE PERIOD AND WEEK
DFCMMQ   EQU   X'02'                 MM IS FISCAL QUARTER
DFCPSBED EQU   X'01'                 EMBEDDED SIGN
DFCMMPIC DS    X                   * MMM PICTURE
DFCMMPN  EQU   X'80'                 NUMERIC, DEFAULT
DFCMMPC  EQU   X'40'                 CHAR, JAN FEB ETC
DFCMMPS  EQU   X'20'                 SPECIAL CHARACTERS EG YY/MM/DD
DFCCDSLN EQU   *-DFCODES           * LENGTH OF ENTRY
*
DFCDSLEN EQU   *-DFC               * LENGTH OF DSECT
*
***********************************************************************
*        DATE FORMAT CODE EQUATES                                     *
*        THE TABLE MUST BE IN THE FOLLOWING ORDER                     *
* 1ST CHAR: P=PACKED H=HEX N=CHAR NUMERIC A=ALHANUMERIC               *
* SUBSEQUENT CHARACTERS ARE                                           *
* C=CENTURY Y=YEAR M=MONTH D=DAY 0=ZERO /=ANY SPECIAL CHARACTER       *
* PP=PERIOD RANGE 1-13, W=WEEK IN PERIOD RANGE 1-5.                   *
***********************************************************************
SQLDATE  EQU   0                   * SQL FORMAT                        00010001
P0CYYDDD EQU   1                   * PACKED LEN 4 0CYYDDD              00010001
PCCYYDDD EQU   2                   * PACKED LEN 4 CCYYDDD              00010001
PYYYYDDD EQU   2                   * PACKED LEN 4 YYYYDDD              00020001
PY4DDD   EQU   2                   * PACKED LEN 4 YYYYDDD              00020001
PYYDDD   EQU   3                   * PACKED LEN 3 YYDDD                00030001
P00YYDDD EQU   4                   * PACKED LEN 4 00YYDDD              00040001
P0YYDDDC EQU   5                   * PACKED LEN 4 0YYDDDC              00050001
PCYYMMDD EQU   6                   * PACKED LEN 4 CYYMMDD              00060001
P0Y4MMDD EQU   7                   * PACKED LEN 5 0YYYYMMDD            00070001
P0YYMMDD EQU   8                   * PACKED LEN 4 0YYMMDD              00080001
PD3      EQU   9                   * PACKED LEN 2 DDD                  00090001
PD5      EQU   10                  * PACKED LEN 3 DDDDD                00090001
PD7      EQU   11                  * PACKED LEN 4 DDDDDDD              00100001
P0YY     EQU   12                  * PACKED LEN 2 0YY                  00090001
PCYY     EQU   13                  * PACKED LEN 2 CYY                  00090001
XCYYDDD  EQU   14                  * HEX LEN 3 CYYDDD                  00110001
X0Y4DDD  EQU   15                  * HEX LEN 4 0YYYYDDD                00120001
X0YYDDD  EQU   16                  * HEX LEN 3 0YYDDD                  00130001
```

```
X000YDDD EQU   17                * HEX LEN 4 000YYDDD                 00140001
XYYDDDC  EQU   18                * HEX LEN 3 YYDDDC                   00150001
X00YDDDC EQU   19                * HEX LEN 4 00YYDDDC                 00160001
X0CYMMDD EQU   20                * HEX LEN 4 0CYYMMDD                 00170001
XY4MMDD  EQU   21                * HEX LEN 4 YYYYMMDD                 00180001
XYYMMDD  EQU   22                * HEX LEN 3 YYMMDD                   00190001
XD4      EQU   23                * HEX LEN 2 DDDD                     00200001
XD6      EQU   24                * HEX LEN 3 DDDDDD                   00210001
XD8      EQU   25                * HEX LEN 4 DDDDDDDD                 00220001
NYYDDD   EQU   26                * NUMERIC LEN 5 YYDDD                00230001
NCYYDDD  EQU   27                * NUMERIC LEN 6 CYYDDD               00240001
NYYDDDC  EQU   28                * NUMERIC LEN 6 YYDDDC               00250001
NY4DDD   EQU   29                * NUMERIC LEN 7 YYYYDDD              00260001
NYYMMDD  EQU   30                * NUMERIC LEN 6 YYMMDD               00270001
NDDMMYY  EQU   31                * NUMERIC LEN 6 DDMMYY               00280001
NMMDDYY  EQU   32                * NUMERIC LEN 6 MMDDYY               00290001
NCYYMMDD EQU   33                * NUMERIC LEN 7 CYYMMDD              00300001
NDDMMYYC EQU   34                * NUMERIC LEN 7 DDMMYYC              00310001
NMMDDYYC EQU   35                * NUMERIC LEN 7 MMDDYYC              00320001
NY4MMDD  EQU   36                * NUMERIC LEN 8 YYYYMMDD             00330001
NDDMMY4  EQU   37                * NUMERIC LEN 8 DDMMYYYY             00340001
NMMDDY4  EQU   38                * NUMERIC LEN 8 MMDDYYYY             00350001
AYYM3DD  EQU   39                * ALPHANUMERIC LEN 7 YYMMMDD         00360001
ADDMMMYY EQU   40                * ALPHANUMERIC LEN 7 DDMMMYY         00370001
AM3DDYY  EQU   41                * ALPHANUMERIC LEN 7 MMMDDYY         00380001
ACYYM3DD EQU   42                * ALPHANUMERIC LEN 8 CYYMMMDD        00390001
ADDM3YYC EQU   43                * ALPHANUMERIC LEN 8 DDMMMYYC        00400001
AM3DDYYC EQU   44                * ALPHANUMERIC LEN 8 MMMDDYYC        00410001
AY4M3DD  EQU   45                * ALPHANUMERIC LEN 9 YYYYMMMDD       00420001
ADDM3Y4  EQU   46                * ALPHANUMERIC LEN 9 DDMMMYYYY       00430001
AM3DDY4  EQU   47                * ALPHANUMERIC LEN 9 MMMDDYYYY       00440001
ND4      EQU   48                * NUMERIC LEN 4 DDDD                 00450001
ND5      EQU   49                * NUMERIC LEN 5 DDDDD                00460001
ND6      EQU   50                * NUMERIC LEN 6 DDDDDD               00470001
ND7      EQU   51                * NUMERIC LEN 7 DDDDDDD              00480001
ND8      EQU   52                * NUMERIC LEN 8 DDDDDDDD             00490001
AY@M@D   EQU   53                * ALPHANUMERIC LEN 8 YY/MM/DD        00500001
AD@M@Y   EQU   54                * ALPHANUMERIC LEN 8 DD/MM/YY        00510001
AM@D@Y   EQU   55                * ALPHANUMERIC LEN 8 MM/DD/YY        00520001
AY4@MM@D EQU   56                * ALPHANUMERIC LEN 10 YYYY/MM/DD     00530001
ISODATE  EQU   56                * ALPHANUMERIC LEN 10 YYYY/MM/DD     00530001
JISDATE  EQU   56                * ALPHANUMERIC LEN 10 YYYY/MM/DD     00530001
AD@MM@Y4 EQU   57                * ALPHANUMERIC LEN 10 DD/MM/YYYY     00540001
AMM@D@Y4 EQU   58                * ALPHANUMERIC LEN 10 MM/DD/YYYY     00550001
IBMUSADT EQU   58                * ALPHANUMERIC LEN 10 MM/DD/YYYY     00550001
AY@M3@D  EQU   59                * ALPHANUMERIC LEN 9 YY/MMM/DD       00560001
AD@MMM@Y EQU   60                * ALPHANUMERIC LEN 9 DD/MMM/YY       00570001
IBMEURDT EQU   60                * ALPHANUMERIC LEN 9 DD/MMM/YY       00570001
AM3@D@Y  EQU   61                * ALPHANUMERIC LEN 9 MMM/DD/YY       00580001
AY4@M3@D EQU   62                * ALPHANUMERIC LEN 11 YYYY/MMM/DD    00590001
AD@M3@Y4 EQU   63                * ALPHANUMERIC LEN 11 DD/MMM/YYYY    00600001
AM3@D@Y4 EQU   64                * ALPHANUMERIC LEN 11 MMM/DD/YYYY    00610001
BCYYDDD  EQU   65                * BIN LEN 4 CYYDDD                   00110001
BY4DDD   EQU   66                * BIN LEN 4 YYYYDDD                  00120001
BYYDDD   EQU   67                * BIN LEN 4 YYDDD                    00130001
BYYDDDC  EQU   68                * BIN LEN 4 YYDDDC                   00160001
BCYYMMDD EQU   69                * BIN LEN 4 CYYMMDD                  00170001
BY4MMDD  EQU   70                * BIN LEN 4 YYYYMMDD                 00180001
BYYMMDD  EQU   71                * BIN LEN 4 YYMMDD                   00190001
BDAYS3   EQU   72                * BIN LEN 2 DDD                      00200001
BDAYS4   EQU   73                * BIN LEN 2 DDDD                     00200001
BDAYS5   EQU   74                * BIN LEN 4 DDDDD                    00200001
BDAYS6   EQU   75                * BIN LEN 4 DDDDDD                   00200001
BDAYS7   EQU   76                * BIN LEN 4 DDDDDDD                  00200001
BDAYS8   EQU   77                * BIN LEN 4 DDDDDDDD                 00200001
NYYYY    EQU   78                * DECIMAL YYYY                       00200001
NYY      EQU   79                * DECIMAL YY                         00200001
P0YYSDDD EQU   80                * PACKED 0YY EMBEDDED-SIGN DDD       00200001
PCYYSDDD EQU   81                * PACKED CYY EMBEDDED-SIGN DDD       00200001
P0Y4SDDD EQU   82                * PACKED 0YYYY EMBEDDED-SIGN DDD     00200001
PCICSABS EQU   83                * PACKED CICS ABSOLUTE SECS 10-3     00200001
NYYMM    EQU   84                * NUMERIC YYMM                       00200001
NMM      EQU   85                * NUMERIC MM                         00200001
```

```
*
* FISCAL DATES
*
NGYYWW   EQU   86                 * NUMERIC GREGORIAN WEEK OF YEAR      00200001
NFYYWW   EQU   87                 * NUMERIC FISCAL WEEK OF YEAR         00200001
NFYYPP   EQU   88                 * NUMERIC FISCAL PERIOD               00200001
NFYYPPW  EQU   89                 * NUMERIC FISCAL DATE NO DAY          00200001
NFYYPPWD EQU   90                 * NUMERIC FISCAL DATE WITH DAY        00200001
NFYYQ    EQU   91                 * NUMERIC FISCAL QUARTER              00200001
NFYYQBEG EQU   92                 * NUMERIC FISCAL QTR BEGIN YYMMDD     00200001
NFYYQEND EQU   93                 * NUMERIC FISCAL QTR END YYMMDD       00200001
*
* USER DEFINED DATES
*
```

4.3 EZ-WatchGuard Date Validation Journal File (#WTJDS)

The following shows the supplied #WTJDS DSECT which defines the format of the records written to the EZ-WatchGuard journal file if allocated.

```
*********************************************************************
*   #WTJDS  : DB-WATCH DATE VALIDATION JOURNAL FILE                 *
*********************************************************************
         SPACE
WTJ      DSECT
WTJLDAT  DS    PL4                * DATE STAMP JULIAN YYYYDDD
WTJLTIM  DS    CL4                * TIME STAMP HHMM
WTJTASK  DS    CL8                * TASK NAME
WTJPROG  DS    CL8                * PROGRAM NAME
WTJDBN   DS    CL8                * DBNAME
WTJSUB   DS    CL8                * SUBSCHEMA NAME
WTJREC   DS    CL18               * RECORD NAME
WTJELEM  DS    CL32               * ELEMENT NAME
WTJDFC   DS    AL2                * DATE FORMAT CODE
WTJDFCP  DS    XL1                * DATE PICTURE
WTJPICC  EQU   X'80'                CHARACTER
WTJPICP  EQU   X'40'                PACKED DECIMAL
WTJPICX  EQU   X'20'                UNSIGNED PACKED DECIMAL
WTJPICB  EQU   X'10'                BINARY
WTJPSQL  EQU   X'08'                SQL
WTJFLG1  DS    XL1                * ACTION FLAG
WTJUPD   EQU   X'80'                UPDATE APPLIED
WTJABRT  EQU   X'40'                UPDATE ABORTED
WTJCENF  EQU   X'20'                CENTURY SETTING FORCED
WTJIDMS  EQU   X'10'                DATA RETURNED FROM IDMS
WTJUSING EQU   X'08'                USING/CALC KEY
WTJFLG2  DS    XL1                * FLAG 2
WTJVERB  DS    XL1                * DBMS VERB
*
WTJDBKEY DS    XL4                * DBKEY
WTJDATLN DS    H                  * DATE LENGTH
         DS    H                  * SPARE
WTJDATE  DS    XL12               * DATE CONTENTS, LEFT ALIGNED
         DS    F                  * SPARE
*
WTJDSLEN EQU   *-WTJ              * LENGTH OF DSECT
```

4.4 Manually generating an SRLTAB

```
*********************************************************************
*   CZ801.T06 EZ-Test Tech Note                          April 1998 *
*                                                                   *
*   KEYWORDS: EZ-Test DB2KSRL #SRLTAB                               *
*                                                                   *
*   This describes how to manually generate the SRL table for use   *
*   with DATAAGER.                                                  *
*                                                                   *
*   PTF CE801028 (replacement #SRLTAB Macro) is a prerequisite.     *
*                                                                   *
*********************************************************************

Manually generating an SRLTAB
-----------------------------

For non-IDMS users, or for CA-IDMS users that do not wish to add
sequential file definitions to the IDD, it is possible to
manually generate the SRLTAB using the #SRLTAB macro instead of
running the DB2KFRL utility.

The particular disadvantage of doing it this way is that there
will be no cross-validation of the #SRLTAB parameters as would be
inherent in deriving the table from the IDD dictionary.
Particular care must be taken to ensure their accuracy.

The SRLTAB is generated using the #SRLTAB macro. The following
illustrates the four classes of #SRLTAB macro which must be
combined to generate a SRLTAB.

    #SRLTAB ENTRY=FIRST,<first-parameters>
    #SRLTAB ENTRY=RECORD,<record-parameters>
    #SRLTAB ENTRY=ELEMENT,<element-parameters>
    #SRLTAB ENTRY=LAST


To generate a valid SRLTAB you begin the definition with a
#SRLTAB ENTRY=FIRST statement. You would then include one or more
#SRLTAB ENTRY=RECORD statements (one for each record containing
dates to be aged). Each ENTRY=RECORD is followed by one or more
ENTRY=ELEMENT statements (one for each date within that
particular record that is to be aged).  The table is terminated
by a #SRLTAB ENTRY=LAST.

This is illustrated by the following example:-
    #SRLTAB ENTRY=FIRST,......
    #SRLTAB ENTRY=RECORD,.....
    #SRLTAB ENTRY=ELEMENT,.....
    #SRLTAB ENTRY=ELEMENT,.....
    #SRLTAB ENTRY=ELEMENT,.....
    #SRLTAB ENTRY=RECORD,.....
    #SRLTAB ENTRY=ELEMENT,.....
    #SRLTAB ENTRY=LAST



ENTRY=FIRST
-----------

The following shows the full syntax of the #SRLTAB ENTRY=FIRST
statement.

tabname  #SRLTAB ENTRY=FIRST,
              SEOTAB=,              ( SEOTAB NAME )
              FILE=,                ( FILE NAME )
              CENTIND=1,            ( 21ST CENTURY INDICATOR )
              PIVOT=40,             ( PIVOT YEAR )
              MAXERR=               ( MAXIMUM ERRORS REPORTED )

Each of the available parameters is described below:-
```

```
tabname:   The name of the table may be specified by
           including a 1-8 character label in columns 1-8 of
           the #SRLTAB ENTRY=FIRST macro. If the macro is
           unlabelled, the default name of DB2KSRL will be
           generated.
           Note: This name should also be the link-edit name
           of the generated table.

SEOTAB:    Specify the name of the Date Element Override
           table if you require non-standard date adjustment
           for particular date fields.  This is an optional
           parameter and the table will only be used if this
           parameter is specified. Refer to section 4.5 for
           further information about the Date Element
           Override table.

FILE:      The name of the sequential file to which this
           SRLTAB relates.  The name may be 1-8 alphanumeric
           characters. This name must match with a CTLTAB
           CLASS=FILE entry.

CENTIND:   Specify the binary value which is used as a
           century indicator in date fields to indicate a
           21st century date. The default value of this
           parameter is 1.

           In most cases it will not be necessary to alter
           this value. The IBM standard is that a century
           indicator of zero indicates 20th century, one
           indicates 21st century etc. There are however rare
           instances where a site may wish to override this.
           For example, a user maintaining historical data
           dating back to the 19th century may use zero to
           indicate 19th century, one to indicate 20th
           century and two to indicate 21st century etc.

PIVOT:     Specify pivot year as PIVOT=nn.  Years containing
           YY only are deemed to be 20 century if they are
           not lower than the pivot year, if they are lower
           then they are deemed to be 21 century.

           Pivot year can be overridden at element level, the
           default is 40.

MAXERR:    Specify the maximum number of errors that will be
           reported for any particular record element. At
           run-time, EZ-Test will write an error message to
           the DB2KLOG for (or MVS log if DB2KLOG is not
           defined) whenever it cannot age a record element
           due to an unexpected field content (ie. the data
           is incompatible with the specified Date Format
           Code for the record element).

           MAXERR can be overridden at record and/or element
           level, the default value of this parameter is 10.
```

ENTRY=RECORD

As a File may contain multiple record types, it is necessary to identify a field within each record which should be used as the record identifier.

The order of the RECORD definitions within the SRLTAB determines the order that matching takes place during execution of DATAAGER. For example, where a file has a header (low-values in key) and trailer record (high-values in key), these have to be defined before the data records. The data records have NO key value associated with them and so, as their definition acts as a catch-all, it should be last.

The following shows the full syntax of the #SRLTAB ENTRY=RECORD statement.

```
#SRLTAB ENTRY=RECORD,
     RECORD=,               ( RECORD NAME )
     MAXERR=,               ( MAXIMUM ERRORS REPORTED )
     KEYOFF=0,              ( KEY FIELD OFFSET IN RECORD )
     KEYLEN=0,              ( KEY FIELD LENGTH )
     KEYPIC=DISPLAY,        ( KEY FIELD PICTURE )
     KEYVAL=' '             ( KEY FIELD VALUE )
```

Each of the available parameters is described below:-

```
RECORD:    Specify the name of the record to which the
           subsequent ENTRY=ELEMENT definitions relate.  You
           may specify multiple ENTRY=RECORD statements each
           followed by one or more ENTRY=ELEMENT statements.

           Note that this name is used for user reference
           only and can be any user-selected name.

MAXERR:    See under ENTRY=FIRST above.

KEYOFF:    Specify the offset within the record of the key
           which will uniquely identify this record type.

KEYLEN:    Specify the length of the record key.

KEYPIC:    Specify the Cobol picture type of the record key.
           This must be one of: DISPLAY, COMP, COMP-3.

KEYVAL:    Specify the value of the record key.
           This should be specifed as an assembler constant
           consistent with the specification of KEYPIC,
           eg, KEYPIC             FORMAT     EXAMPLES
               DISPLAY (alpha)    CLx' '     CL6' ';CL'ABCD'
               DISPLAY (numeric)  xC' '      10C'9'
               COMP               xX'00'     10X'00';4X'FF'
               COMP-3             PLx'0'     PL5'0';PL7'9'
           where x is the length of the field.
```

ENTRY=ELEMENT

Note that DB2KFILE is only interested in the position and format of the date fields within the record. Therefore, it is not necessary to define every single record element.

The following shows the full syntax of the #SRLTAB ENTRY=ELEMENT statement.

```
#SRLTAB ENTRY=ELEMENT,
     PIVOT=,                ( PIVOT YEAR )
     MAXERR=,               ( MAXIMUM ERRORS REPORTED )
     DATENAM=,              ( DATE FIELD NAME )
     DFC=,                  ( DATE FORMAT CODE - SCHEMA )
     DFCAPP=,               ( DATE FORMAT CODE - APPLICATION)
     YYMM=,                 ( NULL | ONLY )
     BIZDAY=,               ( NULL | YES )
     AGE=YES,               ( YES|NO )
     DATEOFF=,              ( DATE FIELD OFFSET )
     DATELEN=,              ( DATE FIELD LENGTH )
     RPGCNT=0,              ( REPEATING GROUP COUNT )
     RPGOFF=0,              ( REPEATING GROUP OFFSET )
     RPGLEN=0,              ( REPEATING GROUP LENGTH )
     ODCOFF=0,              ( OCCURS DEPEND COUNT OFFSET )
     ODCLEN=0,              ( OCCURS DEPEND COUNT LENGTH )
     ODCPIC=COMP            ( OCCURS DEPEND COUNT PICTURE )
```

Each of the available parameters is described below:-

PIVOT: See under ENTRY=FIRST above.

MAXERR: See under ENTRY=FIRST above.

DATENAM: Specify the field name of the date field to which this statement applies (within this file/record).

Note that this name is used for user reference only and can be any user-selected name.

DFC: Specify the Date Format Definition for this date field. The Date Format Code Specified must be a valid Date Format Code Equate as documented in the Date Format Code DSECT (#DFCDS). For every Date Format Code definition in the Date Format Code Table (DFCTAB), there is a corresponding equate in #DFCDS. You should specify the name of the equate which matches the Date Format used by this field.

Note: If you have date format representations which do not exist in the DFCTAB, then you will need to regenerate the DFCTAB to include the appropriate format(s), and add an equate for the code to #DFCDS.

A listing of the supplied default #DFCDS can be found in the appendix.

DFCAPP: Specify the application Date Format Definition for this date field. This parameter is only required if the application may have a different view of this date field. If omitted, it is assumed that the definitions are the same.

Refer to the EZ-Bridge User Guide for further details.

YYMM: If YYMM=ONLY is specified, then date adjustment occurs only for the Year and Month portion of the date, thus preserving the day within the month.

For example, if the calculated date difference between Assumed RunDate and Base Date is 3 years 2 months and 12 days, then years would be incremented by 3 and months would be incremented by 2, leaving the day within month intact. The day is then checked to see if it is still valid for the month, and if not, set to the last day of the month. Thus 29th Feb 2001 would become 28th Feb 2001.

This is a particularly useful feature if you wish to age data by an exact number of years. Under the default calculation, the date may be aged by the exact number of years plus or minus one day, depending on where leap year(s) fall relative to the dates.

This parameter is omitted if default calculation is required.

BIZDAY: If BIZDAY=YES is specified, then dates in this date field must fall on Business days only. The default for this parameter is NO.

When aging this field, EZ-Test will calculate the number of business days between RUNDATE and BASEDAT and add that number to ensure that the new date always falls on a business day. The definition of business holidays is contained in the DB2KBIZ table.

Refer to section 3.8 for further details.

AGE: Specify whether EZ-Test is to perform data aging for this date field.

- YES — EZ-Test should age this field according to the applicable EZ-Test criteria. This is the default value.
- NO — Do not age this date field.

If a user were running EZ-Test only, then the user would exclude a date fields from the aging process simply by not defining an SRLTAB entry for that field. However, when using both EZ-Bridge and EZ-Test, it is possible that a date field (for example a date of birth) that did not require aging, does require bridging. It is therefore necessary to generate an SRLTAB entry to specify the DFC and DFCAPP, but AGE=NO should also be specified to stop EZ-Test aging the field also.

Refer to the EZ-Bridge User Guide for further details.

The next eight parameters are used in various combinations depending on whether you want to describe a single date field or a date field within a repeating group of fields, which can be repeating either a fixed number of times, or a variable number depending on the value of a control field, ie an occurs depending field.

For a single date field, specify DATEOFF & DATELEN.

DATEOFF: Specify the offset within the record of the date field.

DATELEN: Specify the length of the date field.

For a field repeating a fixed number of times, specify DATEOFF, DATELEN, RPGCNT, RPGOFF & RPGLEN.

For an occurs depending field, specify DATEOFF, DATELEN, RPGOFF, RPGLEN, OCDOFF, ODCLEN & ODCPIC.

DATEOFF: Specify the offset of the date field within the repeating group.

DATELEN: Specify the length of the date field.

RPGCNT: Specify the number of times the group repeats.

RPGOFF: Specify the offset within the record of the start of the repeating group.

RPGLEN: Specify the length of the repeating grooup.

ODCOFF: Specify the offset within the record of the control field which holds the occurs depending count.

ODCLEN: Specify the length of the occurs depending control field.

ODCPIC: Specify the Cobol picture type of the occurs depending control field. This must be one of: DISPLAY, COMP, COMP-3.

The use of these parameters when specifying repeating fields is best illustrated with 2 examples:

Given this Cobol record fragment at offset 10 within the record:

```
03 GROUP OCCURS 4.
   05 FILLER     PIC X(3).
   05 DATE       PIC 9(8).
```

```
   the specification for DATE would be
           DATEOFF=3,
           DATELEN=8,
           RPGCNT=4,
           RPGOFF=10,
           RPGLEN=11

Given this Cobol record fragment at offset 10 within the record:
           03 DATE-COUNT    PIC S9(4) COMP.
           03 GROUP OCCURS 5 DEPENDING ON DATE-COUNT.
              05 FILLER     PIC X(6).
              05 DATE       PIC 9(8).
   the specification for DATE would be
           DATEOFF=6,
           DATELEN=8,
           RPGOFF=12,
           RPGLEN=14,
           ODCOFF=10,
           ODCLEN=2,
           ODCPIC=COMP
```

ENTRY=LAST

The following shows the full syntax of the #SRLTAB ENTRY=LAST statement.

```
         #SRLTAB ENTRY=LAST
```

This statement must be coded as illustrated to signify the end of the SRLTAB.

4.5 Aging IDMS SQL defined database

```
********************************************************************
*   CE801.T14 EZ-Test Tech Note                      September 1998 *
*                                                                   *
*   KEYWORDS: EZ-Test NATIVE SQL VIRTUAL DATA AGING                 *
*                                                                   *
********************************************************************
*                                                                   *
*   This Tech Note describes how to set up an SRL table to allow    *
*   virtual aging of a IDMS SQL defined database.                   *
*                                                                   *
*   PTF CE801.147 implements the keywords HASH and INDEX mentioned  *
*   at the end of this Tech Note.                                   *
*                                                                   *
********************************************************************
```

SQL syntax similar to the following can be used to extract the infomation, from the SQL catalog, required to manually build an SRL table.

```
SET SESSION CURRENT SCHEMA SYSTEM;
SELECT TABLE.NAME,
       COLUMN.NAME,
       COLUMN.TYPE,
       COLUMN.VOFFSET,
       COLUMN.VLENGTH
  FROM TABLE, COLUMN
  WHERE TABLE.SCHEMA = 'DEMOEMPL'
    AND TABLE.NAME = COLUMN.TABLE
    AND TABLE.TYPE = 'T'
    AND COLUMN.NAME LIKE '%DATE%'
  ORDER BY TABLE.NAME, COLUMN.VOFFSET;
```

The resultant output is similar to:

NAME	NAME	TYPE	VOFFSET	VLENGTH
BENEFITS	LAST_REVIEW_DATE	DATE	35	8
BENEFITS	PROMO_DATE	DATE	39	8
COVERAGE	SELECTION_DATE	DATE	7	8
COVERAGE	TERMINATION_DATE	DATE	16	8
EMPLOYEE	START_DATE	DATE	145	8
EMPLOYEE	TERMINATION_DATE	DATE	154	8
EMPLOYEE	BIRTH_DATE	DATE	163	8
INSURANCE_PLAN	EFF_DATE	DATE	152	8
JOB	EFF_DATE	DATE	45	8
POSITION	START_DATE	DATE	8	8
POSITION	FINISH_DATE	DATE	17	8

Information from this output can be used to create and assemble an SRL table using the #SRLTAB macro. The following example builds an SRL table for the EMPLOYEE SQL table in schema DEMOEMPL.

```
SQLSRL   #SRLTAB ENTRY=FIRST,FILE=DEMOEMPL
         #SRLTAB RECORD=EMPLOYEE
         #SRLTAB DATENAM=START_DATE,DATEOFF=145,DATELEN=8,DFC=SQLDATE
         #SRLTAB DATENAM=TERMINATION_DATE,                             X
               DATEOFF=154,DATELEN=8,DFC=SQLDATE
         #SRLTAB DATENAM=BIRTH_DATE,DATEOFF=163,DATELEN=8,DFC=SQLDATE
         #SRLTAB ENTRY=LAST
```

The DATEOFF and DATELEN are taken from the VOFFSET and VLENGTH columns in the SQL query above. Additional tables/columns can be defined as needed.

The additional query:

```
SELECT NAME,
       TABLE,
```

```
       COLUMN
  FROM INDEXKEY
  WHERE SCHEMA = 'DEMOEMPL' AND COLUMN LIKE '%DATE%';
```

can be used to determine whether the additional parameters INDEX=YES and INDEX=YES are required on the column definitions in the SRL. HASH=YES indicates that the column participates in a hash key (NAME=HASH from the above query) and INDEX=YES indicates that the column is part of an index key (NAME not equal to HASH).