CA MICS® Resource Management

Tandem Option Guide

Release 12.9

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: Overview

The CA MICS Application Extension for Tandem Systems formats data recorded by the Tandem MEASURE(TM) facility into the CA MICS Database. The Application Extension for Tandem Systems provides standard data conversion. It:

- converts data from ASCII to EBCDIC
- converts time units to seconds
- decodes device types
- calculates derived elements like response time, percent busy, and rates per second

In addition to storing the MEASURE data in the CA MICS Database, the Application Extension for Tandem Systems has interfaces to the CA MICS Capacity Planner and to CA MICS Accounting and Chargeback. Users of the CA MICS Capacity Planner have access to a Tandem CPU Planning Standard Application that allows IS management to better forecast the load on their Tandem processors. The interface to CA MICS Accounting and Chargeback permits accounting for Tandem usage.

Application Extensions

Application extensions provide specific data source support for a given technology. The purpose of application extensions is to provide supplemental information for the CA MICS management support applications. This information is used for financial management, capacity management, storage management, and performance management.

The CA MICS System is composed of three types of products that operate within the CA MICS platform: management support applications, data integration applications, and application extensions.

Management support applications deliver solutions for IS business management requirements, such as accounting and chargeback, capacity management, and management reporting using the CA MICS Database as their input source. Data integration applications process data for integration into the CA MICS Database and its subsequent use by management support applications. In addition, a DIA is a solution for managing a specific technology by providing reports, an extensive data dictionary, a normalized data model, and exception analysis.

Application extensions provide data formatting and data management services necessary to load specific data into the CA MICS Database. Once in the database, the application extension data is used to extend the scope and value of the management support applications. CA provides application extensions priced according to their functionality. The application extensions enable CA to address specific market niches that would otherwise not warrant development of a full function DIA. The combined breadth of DIAs and application extensions provide the customer a product mix consistent with their management requirements and cost expectations.

Application Extension Categories

Application extensions are characterized by the technologies they represent as well as the IS business management applications they support.

o Technologies
  - Distributed Computing Environments (DCEs)
  - Network Architectures
  - Mainframe Environments
  - Processing Systems

o IS Business Management Applications
  - Financial Management
  - Capacity Management
  - Storage Management
  - Performance Management
  - Service Management
  - Operations Management
  - Quality Management

Benefits of Application Extensions

Application extensions enhance the value of applications

written on the CA MICS platform by adding specific information about a technology. The value of a specific application extension is increased when used with CA-provided management support applications, such as our Accounting and Chargeback, Capacity Planner, and StorageMate products. The benefits that application extensions provide include:

o Reduced time and cost to support and maintain

o Better use of scarce people resources

o Increased return on investments with only a marginal increase in cost

o Improved operational effectiveness

Application Extension Features

Application extensions possess a common set of functions and features. Individual application extensions may have features beyond the common set described here.

o Supported Files

An application extension creates one or more files in the CA MICS Database to support a specific data source. Generally, an application extension follows a one-to-one mapping of the raw data source to the CA MICS Database.

o Integration With CA MICS Applications and Other CA MICS Database Information

Information from application extension data sources can also be used in combination with information provided through other DIAs and AEs. This combination provides a more complete picture of the enterprise computing environment and can be used in almost any type of user-developed reports or for standard analysis to support capacity planning, chargeback, or performance management. MICF Production Reporting provides the basic reporting facilities for application extensions.

o Management Support Application Interfaces

CA MICS application extensions provide the necessary interface to other CA MICS applications. Application extensions use the standard application programming interfaces to make their information available for use by the management support applications appropriate for

the technology being supported.

Easy Installation

CA MICS application extensions are designed to integrate with the CA MICS product line as a standard product. Installation of application extensions follows standardized checklists. These checklists provide step-by-step instructions to ensure a smooth and complete installation, which allows your staff to begin using the application extension immediately rather than building an application from scratch.

Minimized Maintenance

Application extensions, like other CA MICS products, reduce your maintenance effort. CA provides the technology changes as underlying data sources change. You can redirect your efforts to more productive activities such as analysis and management of the enterprise computing environment.

CA MICS Platform

You receive all of the benefits of the CA MICS architecture, including:

o Data integration into a single database

o Ability to add other data sources

o Data filtration, transformation, and summarization

o Online documentation

Technology Requirements

Application extensions operate under the current levels of the CA MICS System and its prerequisite technologies. For information about specific data sources for each application extension, consult your CA account manager or local branch office.

Chapter 2: Usage Guidelines - CA MICS Product Interfaces

This section describes the CA MICS Application Extension for Tandem Systems' interface with CA MICS Accounting and Chargeback and the CA MICS Capacity Planner. These interfaces extend IS management capabilities in the areas of accounting and chargeback and capacity planning for Tandem processors.

CA MICS Accounting and Chargeback

The CA MICS Application Extension for Tandem Systems provides a direct interface for the following files to CA MICS Accounting and Chargeback:

- TDMCPE - CPU Utilization File
- TDMPRS - Process Activity File

Accounting journal files and charging elements have not been defined because we have not determined a definitive methodology to employ MEASURE data for accounting. These can be added using the CA MICS Accounting Journal File Eligibility and Charging Element panels. Refer to the CA MICS Accounting and Chargeback Guide for details.

CA MICS Capacity Planner

The CA MICS Capacity Planner has a Tandem CPU Planning Standard Application that provides a predefined capacity plan for Tandem processors. The standard application uses data from the MEASURE CPU Entity File (TDMCPE). Refer to the CA MICS Capacity Planner Guide for more information on the Tandem CPU Planning Standard Application.

In addition to the Tandem CPU Planning Standard Application, all of this application extension's files are accessible to the Capacity Planner for building resource element files.

Chapter 3: Reports

Five sample reports are included with the CA MICS Application Extension for Tandem Systems. The inquiries are in the MICF catalog group PERFORM in the MICF Shared Inquiry Catalog. You can execute them on request or schedule the reports through the MICF Inquiry Production Interface. The reports are:

- TDMCD1 - An overlay color plot of CPU utilization, by CPU number, by system.
- TDMCX1 - A color chart of CPU utilization by process group.
- TDMPX1 - A printer chart of CPU utilization by process group.

For more information about MICF inquiries, see the CA MICS MICF Reference Guide.

Chapter 4: Exceptions

This chapter is intentionally left blank for future use.

Chapter 5: Files

The CA MICS Application Extension for Tandem Systems stores its data in the Tandem Information Area (TDM). It has one CA MICS file for each Tandem MEASURE Entity File, and a System Profile File (TDMSYP), which summarizes data across CPU entity records.

To minimize DASD usage, only files that are built from the most commonly recorded entities are turned on when this application extension is shipped. You can turn files on or off using the procedures in Complex Level Parameters, Section 7.1 of this guide.

This table shows the relationship between MEASURE Entities, LOADIDs, and CA MICS files and indicates whether the file is turned on or off when the product is shipped. LOADIDs are the Tandem Entity Types (that is, record types). LOADID is also used as a keyword in the TDMOPS parameter, which is discussed in Section 7.2.1.

Tandem Entity Type	LOADID	CA MICS File	Shipped Status
CLUSTER*	CLUS	TDMCLE	ON
CONTROLLER	CNTL	TDMCTL	ON
CPU	CPU	TDMCPE	ON
DEVICE	DEV	TDMDVE	ON
DISC	DISC	TDMDSC	ON
DISCOPEN	DISO	TDMDSO	ON
DISKFILE	DISF	TDMDSF	ON
FILE	FILE	TDMFLE	ON
LINE*	LINE	TDMLNE	ON
NETLINE	NETL	TDMNTL	ON
OPDISK*	ODSK	TDMOPD	OFF
PROCESS	PROC	TDMPRS	ON
PROCESSH	PROH	TDMPRH	OFF
SERVERNET	SNET	TDMSVN	OFF
SQLPROC	SQLP	TDMSQP	OFF
SQLSTMT	SQLS	TDMSQS	OFF
SYSTEM	SYST	TDMSYE	OFF
TERMINAL*	TERM	TDMTMN	ON
TMF	TMF	TDMTME	OFF
		TDMSYP	ON
		TDMSYP	ON
		TDMSYP	ON

*Note: This entity record is no longer supported in

```
   current release version updates.


All files include the Tandem common elements:

  TDMMVER  - MEASURE version number
  TDMCPUNO - CPU Number

  And the following CA MICS common elements:

  DAY      - Day of Month
  DYNAME   - Name of Day of Week
  DURATION - Recording Interval Time
  ENDTS    - End Time Stamp
  HOUR     - Hour of Day
  INTERVLS - Number of Recording Intervals
  MONTH    - Month of Year
  ORGSYSID - Originating System Identification
  STARTTS  - Start Time Stamp
  SYSID    - System Identifier
  WEEK     - Week of Year
  YEAR     - Year of Century
  ZONE     - Time Zone

 A brief description of each file in this application
 extension follows:

 o Tandem Cluster Activity (TDMCLE) File

The TDMCLE file quantifies the communication activity over
the fiber optic link (FOX) that connects multiple systems,
including messages sent/received and bytes transferred.  This
file is derived from Tandem MEASURE CLUSTER Entity records.

The CLUSTER entity is not supported in H-series and J-series
release version updates.  If you are running either of these
series, you must deactivate the TDMCLE file in
sharedprefix.MICS.GENLIB(TDMGENIN).

For more information about file tailoring, see Chapter 6 of
the System Modification Guide.

o Tandem CPU Utilization (TDMCPE) File

The TDMCPE file quantifies the CPU level utilization, memory
usage, swaps, cache activity, disk I/Os, transactions, and
response time for each individual CPU.  This file is derived
from Tandem MEASURE CPU Entity records.
```

o Tandem Controller Activity (TDMCTL) File

The TDMCTL file quantifies the I/O activity undertaken by controllers of disks, tape, terminals and other I/O devices. This file is derived from Tandem MEASURE CONTROLLER Entity records.

o Tandem Disc Activity (TDMDSC) File

The TDMDSC file reports the activity of disks on the local system. It supplies the reads, writes, seeks, swaps, cache hits/misses, and I/O rates for each volume. This file is derived from Tandem MEASURE DISC Entity records.

o Tandem Diskfile Activity (TDMDSF) File

The TDMDSF file is very similar in structure TDMDSF, but does not break down accesses to the disk by process id, and thus provides a higher level view of disk activity. The TDMDSF file also provides some data not found in TDMDSO, including file size at the end of each recording interval. This file is derived from Tandem DISKFILE Entity records.

o Tandem Disc File Openings Activity (TDMDSO) File

The TDMDSO file quantifies the I/O operations performed on a file by physical access. It supplies the reads, writes, and cache hits for each file accessed by disk processes. This file is derived from Tandem DISCOPEN Entity records.

o Tandem Device Activity (TDMDVE) File

The TDMDVE file quantifies the I/O activity for all devices on the local system except disks, communication lines, and terminals. It supplies the reads, writes, bytes transferred, and busy time for each device. This file is derived from Tandem MEASURE DEVICE Entity records.

o Tandem File Openings Activity (TDMFLE) File

The TDMFLE file quantifies the I/O operations performed on a file by logical access. It supplies the reads, writes, I/O rates, and file usage for each file accessed by user processes. This file is derived from Tandem FILE Entity records.

o Tandem Line Activity (TDMLNE) File

The TDMLNE file reports the I/O activity on communication lines, including counts of read/write operations, busy time, and response time. This file is derived from Tandem MEASURE LINE Entity records.

The LINE entity is no longer valid as most connections are either TCP/IP or Fiber Channel. You must deactivate the TDMLNE file in sharedprefix.MICS.GENLIB(TDMGENIN).

For more information about file tailoring, see Chapter 6 of the System Modification Guide.

o Tandem Network Communication Activity (TDMNTL) File

The TDMNTL file reports the activity on network communication lines, including counts of read/write operations, bytes transferred, busy time, and percentage of utilization. This file is derived from Tandem NETLINE Entity records.

o Tandem Optical Disk Activity (TDMOPD) File

The TDMOPD file quantifies the I/O activity for optical disk systems. It supplies the reads, writes, bytes transferred, and busy time for each optical disk volume within an optical disk jukebox. This file is derived from Tandem MEASURE OPDISK Entity records.

Optical disks are no longer used and the OPDISK entity is no longer valid. You must deactivate the TDMOPD file in sharedprefix.MICS.GENLIB(TDMGENIN).

For more information about file tailoring, see Chapter 6 of the System Modification Guide.

o Tandem Process Code Range Usage (TDMPRH) File

The TDMPRH file reports how often a code range within a process executes. This file is derived from Tandem MEASURE PROCESSH Entity records.

o Tandem Process Activity (TDMPRS) File

The TDMPRS file reports the activity and resource usage for processes. It supplies the dispatch count, page faults, working set size, messages transferred, and queue time for processes at the DETAIL timespan and for process group at higher timespans. This file is derived from Tandem PROCESS Entity records.

o Tandem SQL Process Activity (TDMSQP) File

The TDMSQP file reports the activity of SQL processes. It supplies the compiled time and counts of object recompiles, statement recompiles, and OPEN calls for each individual SQL process. This file is derived from Tandem MEASURE SQLPROC Entity records.

o Tandem SQL Statement Usage (TDMSQS) File

The TDMSQS file quantifies the activity of all SQL statements within an SQL process. It supplies the calls, recompiles, sorts, run time, recompiled time, sort time, records used, and records accessed. This file is derived from Tandem MEASURE SQLSTMT Entity records.

o Tandem Servernet Activity (TDMSVN) File

The TDMSVN file quantifies the I/O activity undertaken by ServerNet Addressable Controllers (SAC). This file is derived from Tandem MEASURE SERVERNET Entity records.

o Tandem Network Systems Activity (TDMSYE) File

The TDMSYE file reports the network traffic through EXPAND line handlers. It supplies the identification of the remote system, messages sent, wait time, and counts of packets sent/received. This file is derived from Tandem MEASURE SYSTEM Entity records.

o Tandem System Profile (TDMSYP) File

The TDMSYP file quantifies Tandem system level utilization, processor count, memory usage, swaps, cache activity, disk I/Os, transactions, and response time. This file is derived from Tandem MEASURE CPU Entity records through the TDMCPE file.

o Tandem TMF (TDMTME) File

The TDMTME file reports the activity of home and remote transactions on Transaction Monitoring Facility. It supplies the transaction counts and wait time. This file is derived from Tandem MEASURE TMF Entity records.

o Tandem Terminal Activity (TDMTMN) File

The TDMTMN file reports the terminal I/O activity, including requests, reads, writes, bytes transferred, transaction

count, and response time. This file is derived from Tandem MEASURE TERMINAL Entity records.

The TERMINAL entity is no longer valid as terminal connections are now replaced with PCs having a multi-threaded common process. You must deactivate the TDMTMN file in sharedprefix.MICS.GENLIB(TDMGENIN).

For more information about file tailoring, see Chapter 6 of the System Modification Guide.

Chapter 6: Data Sources

```
The CA MICS Application Extension for Tandem Systems
processes MEASURE data.  MEASURE is the monitor information
on the GUARDIAN(TM) operating system and produces interval
records called entities.  Each MEASURE entity file is
processed as a separate record type in CA MICS.  Users
usually record a subset of the available record types due to
the volume of records produced.  This application extension
supports MEASURE versions C10, C20, C30, D10, D20, D30, D40,
D45, G06, G07, G08, G09, G10, G11, G12, H06 and J06.  Both
styles, LEGACY and ZMS, are also supported.

This section contains the following topics:

     1 - Data Collection
     2 - Data Transfer
     3 - LRECL Lengths of Files
     4 - Processing LEGACY and ZMS Style Measure Data
```

This section contains the following topics:

6.1 Data Collection

MEASURE provides an option to specify a value in the identification string field, LOADID, in each entity record. MEASURE leaves the LOADID value as blank by default. CA MICS uses the LOADID value to identify the MEASURE record type. To specify a value other than blank, use the LOADID clause of the MEASCOM LIST command when you write the entity file. For more information about this procedure, see the MEASURE Reference Guide For the LOADID values that this application extension expects to find in the data, see Chapter 5 of this guide.

If entity files are populated with the specified LOADID, concatenate the data as one file to be input to CA MICS. If entity files are not populated with the LOADID, read each file type as a separate data set to be input on a separate input dd statement. For parameter specifications in TDMOPS and DDNAMES in INPUTTDM to identify whether or not LOADID contains the entity type value, see Sections 7.2.1 and 7.2.2 of this guide.

The MEASURE recording interval should be synchronized to some time before the hour to ensure data is summarized into the correct hour. CA MICS determines the HOUR value from the ENDTS of the entity record. When the record is written a fraction of a second before the hour, the statistics for the hour will be recorded in that hour. When the record is written a fraction of a second after the hour, statistics for the previous hour are recorded in the next hour. The greatest impact on the summarized data is when the recording interval is one hour. For example, data collected between 10 and 11 a.m. with an ENDTS of 10:59:59.99 will be recorded in hour 10. Data collected with an ENDTS of 11:00:00.01 will be reported in hour 11 although the statistics are for hour 10.

Below is an example of starting a measurement to record needed data for this application extension. For more information, see the MEASURE Reference Manual.

```
ENTER: MEASCOM
ENTER: ADD CPU *
ENTER: ADD DISC *
ENTER: ADD PROCESS *
ENTER: START MEASDATA, FROM 23:59:55, FOR 24 HOURS,
       INTERVAL 30 MINUTES
ENTER: EXIT
```

After you have created a measure file and the specified

```
duration has ended, you will need to take this unstructured
file and create a structured file.  Below is an example of
creating structured files for CPU, DISC, and PROCESS
measurement data.

ENTER: MEASCOM
ENTER: ADD MEASDATA
ENTER: SET REPORT FORMAT STRUCTURED
ENTER: LIST CPU *,FROM <previous date>,23:59:55,
       FOR 30 MINUTES, LOADID CPU
ENTER: LIST CPU *,FROM <current date>,00:29:55,
       FOR 30 MINUTES, LOADID CPU
ENTER: LIST CPU *,FROM <current date>,00:59:55,
       FOR 30 MINUTES, LOADID CPU
(Repeat this line for each interval; Change FROM time)

ENTER: LIST DISC *,FROM <previous date>,23:59:55,
       FOR 30 MINUTES, LOADID DISC
ENTER: LIST DISC *,FROM <current date>,00:29:55,
       FOR 30 MINUTES, LOADID DISC
(Repeat this line for each interval; Change FROM time)

ENTER: LIST PROCESS *,FROM <previous date>,23:59:55,
       FOR 30 MINUTES, LOADID PROC
ENTER: LIST PROCESS *,FROM <current date>,00:29:55,
       FOR 30 MINUTES, LOADID PROC
  (Repeat this line for each interval; Change FROM time)

ENTER: EXIT

The above MEASCOM commands can be entered interactively or
through an EXEC file.

The structured files that have been created can be merged
into one file which is transferred to MVS in a VB format for
CA MICS processing. The following example combines files CPU,
DISC, and PROCESS into one file called TESTDATA:

ENTER: FUP
ENTER: PURGEDATA TESTDATA
ENTER: LOAD PROCESS,TESTDATA
ENTER: COPY DISC,TESTDATA
ENTER: COPY CPU,TESTDATA
ENTER: EXIT

The TESTDATA file in the above example must be created
previously with a record size of the largest record in
MEASURE data.  You may add additional bytes for future
expansion.  The following is an example of defining file
```

```
attributes for the TESTDATA file:

ENTER: FUP
ENTER: SET TYPE E
ENTER: SET EXTENT(50 PAGES, 100 PAGES)
ENTER: SET REC 300
ENTER: SET BLOCK 4096
ENTER: CREATE TESTDATA
ENTER: EXIT


LEGACY or ZMS Style Structured Data
-----------------------------------

The MEASCOM SET REPORT STYLE command indicates whether
structured data should be created using the external
interface previous to G11 known as LEGACY style or the
external interface G11 or later known as ZMS style.

The following is an example of how you can create
(LEGACY style) records:

ENTER: MEASCOM
ENTER: SET REPORT FORMAT STRUCTURED
ENTER: SET REPORT STYLE LEGACY
ENTER: ADD MEASDATA
ENTER: LIST measure entities etc .....
ENTER: EXIT

The following is an example of how you can create
(ZMS style) records:

ENTER: MEASCOM
ENTER: SET REPORT FORMAT STRUCTURED
ENTER: SET REPORT STYLE ZMS
ENTER: ADD MEASDATA
ENTER: LIST measure entities etc .....
ENTER: EXIT

In the G-series and earlier release version updates, the
default SET REPORT STYLE is LEGACY; in H-series release
version updates the default SET REPORT STYLE is ZMS.
```

6.2 Data Transfer

Data must be transferred from the Tandem system to MVS for processing in CA MICS. There are transfer programs available to do this. Refer to the MEASURE Reference Manual and related products for more information. If no transfer program is available at your site, the data can be dumped to tape on the Tandem processor and the tape transported to the MVS system.

Data must be transferred in its original ASCII format; the CA MICS Application Extension for Tandem Systems will perform the conversion to EBCDIC. The file(s) passed to MVS should be in VB or VBS format. If all records are combined on one file, use a record size of the largest record plus additional bytes for future added fields to the Tandem MEASURE data. The file blocksize should be optimum for the device type.

MEASURE structured files put to tape must be in a fixed block format in order to be read by IBM systems. The variable record lengths in the structured files will need to be padded with blanks until they are large enough to contain the largest record in the MEASURE file. Then using the following FUP COPY command, you can copy this data to tape on the Tandem system.

```
FUP COPY <structure filename>,<tape device name>,
  BLOCKOUT 6000,RECOUT 300,PAD " "
```

Then on the MVS side when you read the tape created on Tandem, you read it as DCB=(RECFM=FB,BLKSIZE=6000,LRECL=300) using IEBGENER.

If the LOADID field is populated with the entity type, as described in Section 6.1, you can combine all records into one file. If LOADID is not populated with the entity type, each entity type must be a separate file. Refer to Sections 7.2.1 and 7.2.2 of this guide for examples of coding the OPTIONS and input dd statements when the LOADID is and is not populated.

6.3 LRECL Lengths of Files

While performing the data transfer task, you will need to know the LRECL length of the data as you are reading the tape on the MVS side.

The following chart lists each raw data file available from the Measure product, the file's logical record length (LRECL) for each Measure release, and the CA MICS PSP product change that first supported the Measure release.

Tandem Measure File Name	Tandem Measure Release				
	D45 TDM6150	G06 TDM6220	G08 TDM6240	FUTURE	FUTURE
CNTL	73	Obsolete in G-Series			
CPU	262	684	740		
DVE	104	384	384		
DISC	394	674	746		
DISO	160	168	176		
DISF	182	190	206		
FLE	194	226	226		
LINE	128	170	170		
NETL	152	210	210		
ODSK	138	138	138		
PROC	342	400	422		
PROH	164	164	164		
SNET		290	340		
SQLP	126	126	126		
SQLS	178	178	178		
SYST	84	84	84		

TERM	96	112	112		
TMF	102	102	102		

Note: If the field for the corresponding file and Measure release is blank, assume that no new elements were introduced for said release and use the last available LRECL length.

Note: For the TDMSVN (Servernet) File, there is no release for the D45 Measure Release because this file was not introduced until Measure Release G06.

6.4 Processing LEGACY and ZMS Style Data

MEASURE provides an option to specify the structure or style of MEASURE data. The MEASCOM SET REPORT STYLE {LEGACY|ZMS} command indicates whether MEASURE data is formatted using LEGACY style external interface (pre-G11) or using the ZMS style external interface (G11 or later).

It is possible to create both LEGACY and ZMS style MEASURE data from the same system and recording intervals. The statistics collected in both styles will contain the same information. The data created in both styles can be written to separate files or data sets. It is important to note, if both styles of MEASURE data are processed together from the same system and recording intervals into the same CA MICS unit, redundant information will be stored in the database.

Chapter 7: Parameters

Parameter specifications enable CA MICS sites to easily tailor CA MICS products to their environments. This section contains the complex and unit level parameters provided with this application extension.

This section contains the following topics:

7.1 Complex Level Parameters

This application extension's generation definition statement member is sharedprefix.MICS.GENLIB(TDMGENIN). This member defines the Tandem Information Area (TDM), its files, whether or not you can compress the files, and the elements in each file.

File tailoring can be done at the FILE, CYCLES, or NAME/NAMX statement level.

To turn files on or off in timespans, change N to Y or vice versa.

For example, to turn on all online timespans for the Network Systems Activity File (TDMSYE), you need to change the FILE statement from:

```
FILE SYE 00 1 N N N N N N N N Network Systems Activity File
```

to:

```
FILE SYE 00 1 Y Y Y Y Y N N N Network Systems Activity File
```

If the DBMODEL FILE statement is missing, or if no value has been coded for a timespan, you may modify the default CYCLES retention, by timespan, for each file, by modifying the CYCLES statement following the FILE statement.

For example, to change the default CYCLES retention specification for the Network Systems Activity File (TDMSYE) to 7 cycles for the DAYS timespan, you need to change the CYCLES statement (which follows the FILE SYE... and FOPT... statements) from:

```
CYCLES 07 15 18 24 02 00 000 000
```

to:

```
CYCLES 07 07 18 24 02 00 000 000
```

To turn individual elements on or off in timespans, change N to 0 (on) or 0 to N (off). You may also use NAME/NAMX to turn elements on and off for all timespans, if your complex is generated as ESSENTIAL ONLY or INITIAL.

For example, to turn the element SYENRMSG off at the DAYS and higher timespans, change the NAME statement from:

```
  NAME SYENRMSG     00  0 0 0 0 0 Messages

  to:

  NAME SYENRMSG     00  0 N N N N Messages
```

There are two options to turn the element SYENRMSG off at all timespans. You can use either of the following:

- Change all 0s to Ns as follows:

```
        NAME SYENRMSG     00  N N N N N Messages
```

- If your complex is generated as ESSENTIAL ONLY or INITIAL, change NAME to NAMX as follows:

```
        NAMX SYENRMSG     00  0 0 0 0 0 Messages
```

The OPTION Statement

The OPTION statement defines which data element clusters are included or excluded when generating the data element contents. You should change the definition to meet your needs, as described below.

OPTION statement keywords for the Tandem Analyzer are the following:

LEGACY/NOLEGACY - LEGACY style data elements and the following files:

- Tandem Cluster Activity (TDMCLE)
- Tandem Line Activity (TDMLNE)
- Tandem Terminal Activity (TDMTMN)
- Tandem Optical Disk Activity (TDMOPD)

The Cluster Code assignment in TDMGENIN is 35.

You must specify one keyword:

The recommended OPTION for Tandem in an installation using the MEASCOM SET REPORT STYLE LEGACY:

```
OPTION LEGACY
```

The recommended OPTION for Tandem in an installation using the MEASCOM SET REPORT STYLE ZMS:

```
    OPTION NOLEGACY
```

For more information about these statements and file tailoring, see Chapters 4 and 6 of the System Modification Guide.

You must run a TDMCGEN to activate changes to TDMGENIN.

7.2 Unit Level Parameters

Unit level parameters control the processing of this application extension within each database unit. They are stored in the prefix.MICS.PARMS library associated with each database unit you define. This section contains the following subsections:

1 - Options Definition (TDMOPS)
2 - Input DD Statements (INPUTTDM)
3 - Database Space Modeling (DBMODEL)

7.2.1 Processing Options (TDMOPS)

```
This section shows you how to specify the operational
statements that control CA MICS Application Extension for
Tandem processing.

Operational statements are stored in the prefix.MICS.PARMS
cccOPS member, where ccc is the component identifier, and are
incorporated into the CA MICS system by running the
prefix.MICS.CNTL(cccPGEN) job.

************************************************************
*                                                          *
*  NOTE:  CHANGES to prefix.MICS.PARMS(cccOPS) members     *
*         REQUIRE EXECUTION of prefix.MICS.CNTL(cccPGEN)   *
*         to take effect.                                  *
*                                                          *
*         In addition, any change to parameters that       *
*         impact the DAILY operational job JCL such as,    *
*                                                          *
*         o  changing RESTART NO to RESTART YES,           *
*                                                          *
*         o  WORK parameter changes when RESTART NO is in  *
*            effect,                                       *
*                                                          *
*         o  Specifying TAPEfff (if this product supports  *
*            a DETAIL level TAPE option),                  *
*                                                          *
*         o  or changes to prefix.MICS.PARMS(INPUTccc),    *
*                                                          *
*         will require regeneration of the DAILY job by    *
*         executing prefix.MICS.CNTL(JCLGEND) or by        *
*         specifying DAILY in prefix.MICS.PARMS(JCLGENU)   *
*         and executing prefix.MICS.CNTL(JCLGENU).         *
*                                                          *
*         Refer to the checklist (if provided) for updating *
*         cccOPS parameters and running required generation *
*         jobs.                                            *
************************************************************
The TDMOPS member contains the following statements,
described in detail below.

Required Statements
-------------------

  OPTIONS
  ENTITIES
```

```
Optional Statements
-------------------
   WORK
   MULTWORK|NOMULT
   RESTART
   INCRUPDATE
```

7.2.1.1 OPTIONS Statement

The OPTIONS statement is required and one OPTIONS statement for each Tandem system. The required fields on the OPTIONS statement are:

orgsysid	4 character ORGSYSID that matches the ORGSYSID in the SYSID parameter
system-name	system name value in the Tandem MEASURE data (see NOTE below)
LOADID/NOLOADID	keyword indicates whether the entity type is specified in the LOADID field

NOTE: If a system-name is found in the data that does not have a matching ORGSYSID defined in TDMOPS, a default value equal to the first four characters of the system-name will be used as the ORGSYSID. If the ORGSYSID is not defined in prefix.MICS.PARMS(SYSID), the job will abend with a U310 code.

7.2.1.2 ENTITIES Statement

```
The ENTITIES statement is required if NOLOADID is specified
on at least one OPTIONS statement.  Entry is free form.  List
the entity types to be processed, separated by at least one
blank space, using the LOADID values shown in Section 2.3:
CLUS, CPU, DEV, DISC, DISO, FILE, LINE, NETL, ODSK, PROC,
PROH, SNET, SQLP, SQLS, SYST, TERM, and TMF.  Each LOADID
value should only be specified once.  You may use multiple
ENTITIES statements if all LOADIDs will not fit on one line.
If all OPTIONS statements have LOADID specified, the ENTITIES
statement is not required.


Examples of TDMOPS follow:

Example 1:  One system with entity type in the LOADID field.

     OPTIONS TDM1 \SYSTEM1 LOADID

Example 2:  One  system without the entity type in the LOADID
            field.

     OPTIONS TDM2 \SYSTEM2 NOLOADID
     ENTITIES CPU DISC TMF PROC NETL

Example 3:  Three systems, two with LOADID values and one
            without.  The ENTITIES statement must match the
            entity types for the system without LOADID
            information.

     OPTIONS TDM1 \SYSTEM1 LOADID
     OPTIONS TDM2 \SYSTEM2 NOLOADID
     OPTIONS TDM3 \SYSTEM3 LOADID
     ENTITIES CPU DISC TMF PROC NETL

    NOTE:  A TDMPGEN must be run to makes changes to this
           member effective.  Changes to INPUTTDM may also be
           required when changes to this member are made that
           affect the input ddnames required.  Refer to
           Section 7.2.2 for more information.
```

7.2.1.3 WORK, MULTWORK, and NOMULT Statements

```
WORK
----

This statement is optional.  It enables sites experiencing
either SAS WORK space allocation problems or out of work
space conditions during DAYnnn or INCRnnn (where nnn is the
job step number), daily or incremental update processing, to
allocate multiple WORK files.

You can allocate multiple WORK files for use during the daily
and/or incremental update job step.  The maximum number of
WORK files you can allocate varies by product.  These
additional work files are used in conjunction with the single
work data set allocated by default using the JCLDEF
parameters WORKUNIT and WORKSPACE.

Because the individual space allocation requirement for each
WORK file is typically much smaller, it is more likely to be
satisfied.

To take advantage of multiple WORK files support, edit
prefix.MICS.PARMS(cccOPS) and insert a WORK statement as
shown below:

WORK n data_set_allocation_parameters

where n is the number of WORK data sets

        NOTE:  The default is zero (0).
               The maximum is nine (9).

        data_set_allocation_parameters is one or more data
        set allocation parameters (for example, STORCLAS or
        SPACE) separated by spaces.

You can also specify the WORK parameter as the following:

WORK n XXX pppp ssss

where:

  n    is the number of WORK data sets
  XXX  is TRK or CYL
  pppp is the primary allocation
  ssss is the secondary allocation

Note:  When allocating any number of SAS WORK data sets, be
```

aware that one additional SAS WORK data set is automatically allocated to facilitate sorting. For example, if you allocate six SAS WORK data sets, you will actually get seven.

If you omit the data_set_allocation_parameters or the WORK parameter, the work data sets are allocated according to the values you specified for the WORKUNIT and WORKSPACE parameters in prefix.MICS.PARMS(JCLDEF). Use the data_set_allocation_parameters to override this default, either to alter the space allocation or to use System Managed Storage (SMS) parameters to control data set placement and characteristics.

Note: If you allocate insufficient space for the WORK data sets, DAYnnn and/or INCRnnn processing will fail and can only be restarted from the beginning.

Note: If internal step restart is active, you can override the WORK data set allocation parameters at execution-time using the //PARMOVRD facility. For more information about execution-time override of dynamic data set allocation parameters, see the PIOM, section 2.3.6.

Specify data set allocation parameters, separated by blanks, according to SAS LIBNAME statement syntax. If you need multiple lines, repeat the WORK keyword on the continuation line.

WORK accepts the engine/host options documented in the SAS Companion for the z/OS environment, including STORCLAS, UNIT, SPACE, BLKSIZE, DATACLAS, MGMTCLAS, and VOLSER.

Important! Do not specify the DISP parameter.

Example 1:

```
WORK n STORCLAS=MICSTEMP SPACE=(XXX,(pppp,ssss),RLSE)
```

where:

```
  n        - is the number of WORK data sets.
  STORCLAS - specifies a storage class for a new data set.
             The name can have up to 8 characters.
  SPACE    - specifies how much disk space to provide for
             a new data set being allocated.
  XXX      - is TRK or CYL.
  pppp     - is the primary allocation.
  ssss     - is the secondary allocation.
  RLSE     - specifies that free-space should be released
```

```
              when the data set is closed.

Example 2:

WORK n XXX pppp ssss

where:

  n        - is the number of WORK data sets.
  XXX      - is TRK or CYL.
  pppp     - is the primary allocation.
  ssss     - is the secondary allocation.

Example 3 (multiple lines):

WORK n STORCLAS=MICSTEMP UNIT=SYSDA
WORK   SPACE=(xxxx,(pppp,ssss),,,ROUND))

where:

  n        - is the number of WORK data sets.
  STORCLAS - specifies a storage class for a new data set.
             The name can have up to eight characters.
  UNIT     - specifies the generic unit for a new data set.
             The name can have up to eight characters.
  SPACE    - specifies how much disk space to provide for
             a new data set being allocated.
  XXX      - is TRK or CYL.
  pppp     - is the primary allocation.
  ssss     - is the secondary allocation.

Note:  Since there is some performance impact when using
multiple WORK files, you should specify the minimum number of
WORK data sets to meet your work space requirements.  As a
start, try incrementing the number gradually beginning from
the default.


WORK Considerations
--------------------

How Much Space Should You Allocate?

o  First Time Implementation of Multiple Work Files

   If this is the first time you are implementing multiple
   work files for this product in this unit, review
   prefix.MICS.PARMS(JCLDEF) and find the WORKSPACE
   parameter.  It will resemble this sample statement:
```

```
WORKSPACE      TRK 500 250
```

The value shows the current SAS WORK space allocation for the unit as a single data set. It also serves as the default value used in the unit's DAYnnn daily update (and/or INCRnnn incremental update) step unless you provide a WORK parameter.

To achieve the equivalent work space allocation of WORKSPACE TRK 500 250 using multiple WORK data sets that will collectively share the work space requirements of the daily and/or incremental update step, you could code either one of these:

```
WORK 2 SPACE=(TRK,(250,125))
```

```
WORK 5 SPACE=(TRK,(100,50))
```

To determine the total work space, multiply the number of WORK files (n) by the primary (pppp) and secondary (ssss) values specified.

Note: To simplify the example, only the SPACE parameter is shown above. You can follow either with data set allocation parameters like UNIT or STORCLAS as required for your site.

o Adjusting Allocation for Existing Multiple WORK Files

If you have previously implemented multiple WORK file support for this product in this unit, and you want to change either the number of WORK files or the space allocations, examine prefix.MICS.PARMS(cccOPS) and find the existing WORK statement.

- If the existing WORK statement only specifies the number of WORK files but does not contain space allocation information as shown below:

  ```
  WORK 5
  ```

  Then each of the multiple WORK files is allocated using the values from the WORKSPACE parameter of prefix.MICS.PARMS(JCLDEF), as described earlier under First Time Implementation of Multiple Work Files.

  To increase workspace, you can increase the number of WORK files (for example, change WORK 5 to WORK 6,7,8,

or 9), increase the space allocation in the WORKSPACE parameter, or do both.

To decrease workspace, you can decrease the number of WORK files (for example, change WORK 5 to WORK 4,3,2, or 1), decrease the space allocation in the WORKSPACE parameter, or do both.

You can also elect to explicitly specify the multiple WORK file space allocation by adding the space allocation values directly to the WORK statement. This will remove the link to the prefix.MICS.PARMS(JCLDEF) WORKSPACE parameter for multiple WORK file space allocation. This is recommended as it serves to clearly document, in one place, how multiple WORK files are allocated.

- If the existing WORK statement does include space allocation as shown in the examples below:

  WORK 5 TRK 200 100

  or

  WORK 5 SPACE=(TRK,(200,100)) STORCLAS=MICSTEMP

  Simply change the values to meet your needs.

  If you need more work space, you can increase the number of WORK files (for example, change WORK 5 to WORK 6,7,8, or 9), increase the space allocation (for example, change TRK 200 100 to TRK 250 120), or do both.

  To decrease work space, you can decrease the number of WORK files (for example, change WORK 5 to WORK 4,3,2, or 1), decrease the space allocation (for example, change TRK 200 100 to TRK 150 80), or do both.

Note: If internal step restart is NOT active (RESTART NO) and you change the WORK parameter, you must:

o Run cccPGEN
o Run JCLGENU for DAILY (to regenerate DAILY) and, if incremental update is enabled, INCRccc

When internal step restart is active, (RESTART YES), then, when you change WORK and run cccPGEN, changes take effect immediately. There is no need to run JCLGENU.

```
SASWORK
-------

This statement is optional.

The WORK DD statement in the CA MICS procedures allocates
a temporary data set where SAS keeps its temporary data
files and other items that SAS uses during processing of
the current job.

By default, the space allocated is defined in the member
prefix.MICS.PARMS(JCLDEF) with the WORKSPACE and WORKUNIT
parameters, then generated into all the JCL procedures for
a given unit.

With the SASWORK statement you have the option to override
this unit-wide definition to specify the space allocation
individually for the current step.

The format of the SASWORK statement is:

SASWORK data_set_allocation_parameters

where data_set_allocation_parameters is one or more data set
allocation parameters (for example, STORCLAS or SPACE)
separated by spaces.

You can also specify the SASWORK parameter as the following:

SASWORK XXX pppp ssss

where:

  XXX  is TRK or CYL
  pppp is the primary allocation
  ssss is the secondary allocation

If you omit the data_set_allocation_parameters or the SASWORK
statement, the WORK data set is allocated according to the
values you specified for the WORKUNIT and WORKSPACE
parameters in prefix.MICS.PARMS(JCLDEF).  Use the
data_set_allocation_parameters to override this default,
either to alter the space allocation or to use System Managed
Storage (SMS) parameters to control data set placement and
characteristics.

Specify data set allocation parameters, separated by blanks,
```

```
according to SAS LIBNAME statement syntax.  If you need
multiple lines, repeat the SASWORK keyword on the
continuation line.

Example:

SASWORK STORCLAS=MICSTEMP SPACE=(XXX,(pppp,ssss))

where:

  STORCLAS - specifies a storage class for a new data set.
             The name can have up to 8 characters.
  SPACE    - specifies how much disk space to provide for
             a new data set being allocated.
  XXX      - is TRK or CYL.
  pppp     - is the primary allocation.
  ssss     - is the secondary allocation.


Note:  If you change the SASWORK parameter, you must:

o  Run cccPGEN
o  Run JCLGENU for DAILY (to regenerate DAILY) and, if
   incremental update is enabled, INCRccc


MULTWORK|NOMULT fff fff ... fff
-------------------------------

Since multiple work files usage impacts performance, this
product provides these optional parameters so you can
restrict multiple work files usage to only those files having
excessive space requirements.

Note:  You can only use one of these optional parameters with
the WORK statement, NOT both.

The MULTWORK parameter restricts the use of multiple WORK
files to ONLY those listed after the MULTWORK keyword.

MULTWORK fff fff ... fff

where fff is the unique three character identifier

If you need multiple lines, repeat the MULTWORK on the
continuation line.

The NOMULT parameter forces the use of multiple WORK files
for all files EXCEPT those specified after the NOMULT
```

```
keyword.

NOMULT fff fff ... fff

where fff is the unique three character identifier

If you need multiple lines, repeat the NOMULT on the
continuation line.

The default is

MULTWORK CLE CTL CPE DSC DSO DSF DVE FLE LNE NTL OPD
MULTWORK PRH PRS SVN SQP SQS SYE TME TMN

if neither MULTWORK or NOMULT parameters are specified.

The following files are eligible for multiple WORK support:

CLE    TANDEM Cluster Activity File
CTL    TANDEM Controller Activity File
CPE    TANDEM CPU Utilization File
SYP    TANDEM System Profile File
DSO    TANDEM Disc File Openings Activity
DSF    TANDEM Disk File Aggregate Activity
DSC    TANDEM Disc Activity File
DVE    TANDEM Device Activity File
FLE    TANDEM File Openings Activity
LNE    TANDEM Line Activity File
NTL    Network Communication Activity
OPD    TANDEM Optical Disk Activity File
PRH    TANDEM Process Code Range Usage File
PRS    TANDEM Process Activity File
SVN    TANDEM ServerNet Activity File
SQP    TANDEM SQL Process Activity File
SQS    TANDEM SQL Statement Usage File
SYE    TANDEM Network Systems Activity File
TME    TANDEM TMF File
TMN    TANDEM Terminal Activity File

The following section discusses changing the WORK option:

    1 - Change the Number of Work Files
```

7.2.1.3.1 Change the Number of Work Files

```
To change the number of work files used in CA MICS Tandem
Application Extension processing in Step DAY076, follow the
checklist provided below for each unit:

   ************************************************
   *                                              *
   *           CHANGE NUMBER OF WORK FILES        *
   *                                              *
   ************************************************

___ 1.  Update the WORK statement in
        prefix.MICS.PARMS(cccOPS), where (ccc) is the
        component identifier, to specify the number of work
        data sets required.  Below is an example:

        WORK n STORCLAS=MICSTEMP SPACE=(XXX,(pppp,ssss))

        where:

        n        - is the number of WORK data sets.
        STORCLAS - specifies a storage class for a new data
                   set.  The name can have up to eight
                   characters.
        SPACE    - specifies how much disk space to provide
                   for a new data set being allocated.
        XXX      - is TRK or CYL.
        pppp     - is the primary allocation.
        ssss     - is the secondary allocation.

        You should specify the minimum number of WORK data
        sets to meet your work space requirements.  As a
        start, try incrementing the number gradually
        beginning from the default.

___ 2.  If this is the first time you are implementing
        multiple work files for this product, then continue
        with Step 2.  If you are just changing the number
        currently in use, or simply the space definitions,
        then proceed to Step 3 of this checklist.

        Browse sharedprefix.MICS.PROTOLIB(DYcccnnn) and
        sharedprefix.MICS.PROTOLIB(cccINCR), where (nnn) is
        the job step number and (ccc) is the product ID for
        this product, checking for the presence of the WORK
        symbolic on the EXEC statement to determine if you
        have previously modified this product to increase the
        allocation of SAS WORK space.
```

___ 2a. If you find a WORK symbolic, simply divide the primary and secondary allocation values from the WORK symbolic by the number of work files specified above (value of n on the WORK statement coded in Step 1).

Coding the resulting values will yield the same aggregate space allocation you have been using with a single WORK file. To double your available WORK space, carry out the division, double the results and use the values in the WORK definition above.

___ 2b. If you did not find a WORK symbolic in PROTOLIB, examine prefix.MICS.PARMS(JCLDEF) for each CA MICS unit containing this product. Find the WORKSPACE keyword. The space allocation specified is used for a single SAS WORK file. Perform the same division as described in the previous paragraph to determine the quantity that will yield equivalent total allocation with multiple WORK files. Then adjust the values upward to meet your needs.

___ 3. Submit the job in prefix.MICS.CNTL(cccPGEN).

___ 4. If you specified RESTART YES in the product's cccOPS, you are done. Otherwise, you must do Steps 5, 6, and 7.

___ 5. Edit prefix.MICS.PARMS(JCLGENU) so that it contains a single line that reads:

DAILY

or, if incremental update is enabled for this product in this unit database, specify:

DAILY INCRccc

where ccc is the product ID.

___ 6. Submit the job in prefix.MICS.CNTL(JCLGENU). Ensure that there are no error messages in MICSLOG or SYSTSPRT, that the MICSLOG contains the normal termination message, BAS10999I, and that the job completes with a condition code of zero.

___ 7. The following operational job(s) have changed:

DAILY

INCRccc (if incremental update is enabled)

If your site has implemented the operational CA MICS processes in a scheduling product, the JCL may have to be refreshed in that product. See the scheduling product's administrator for the exact processes involved in updating that product's representation of the CA MICS jobs.

7.2.1.4 Internal Step Restart Statements

```
RESTART YES/NO
--------------

This statement is optional.  Specify this to activate
internal step restart for this product's DAILY and/or INCRccc
database update job steps:

RESTART YES


If you do not specify or enable the RESTART parameter, then
this option defaults to the following and internal step
restart is disabled:

RESTART NO


************************************************************
*                                                          *
*  Note:  Changing the RESTART parameter (either from NO   *
*         to YES or from YES to NO) requires regeneration  *
*         of the DAILY operational job by executing        *
*         prefix.MICS.CNTL(JCLGEND) or by specifying       *
*         DAILY in prefix.MICS.PARMS(JCLGENU) and          *
*         executing prefix.MICS.CNTL(JCLGENU).             *
*                                                          *
*         If incremental update is active for this product, *
*         you must also regenerate the INCRccc job.        *
*                                                          *
************************************************************

Internal step restart can significantly reduce time and
resource usage to recover from daily and/or incremental
update processing failures.  CA MICS uses a
checkpoint/restart technique.

o  When internal step restart is activated, the database
   update job step "checkpoints" (or saves) intermediate
   results (work file contents) and the operational
   environment at the end of each processing phase.

o  Then, if required, the database update step can resume
   execution at the beginning of the processing phase in
   which the failure occurred.

o  Restart is accomplished by restoring the operational
```

environment from the last checkpoint, bypassing completed processing phases, and resuming execution using intermediate results (work files) from the last checkpoint.

Note: When you activate internal step restart (RESTART YES), the following optional restart parameters are enabled. These parameters have no effect if restart is disabled (RESTART NO). For more details, see the individual parameter descriptions later in this section.

- RESTARTCKPT data_set_allocation_parameters
- RESTARTWORK data_set_allocation_parameters
- DYNAMWAIT minutes

Processing Phases:

This product employs three database update processing phases followed by the two common roll-up phases.

Phase	Description
FORMAT	Read raw input data, convert to SAS format, and output to intermediate work files.
SORT	Sort intermediate work file contents, eliminate duplicate input data, and prepare for DETAIL cycle creation.
DBUPDATE	Detail cycle creation
DYSUM	Summarize DETAIL data to create new DAYS cycles and to update current week-to-date and month-to-date cycles.
DYAGE	Cutover new database cycles to production and "age" existing cycles.

RESTART Considerations

- Overhead

Enabling internal step restart adds some overhead to the database update job step -- the cost of taking checkpoints and managing saved materials. Since this overhead is relatively constant and independent of input data volume, you may find that costs outweigh potential savings when input data volume is low, for example in a test unit. For high volume, production units, internal step restart support overhead should be a minor portion of total resource usage.

- Cataloged Work Files

  When internal step restart is enabled, the SAS work data set, internal step restart control data set, and multiple work file data sets are allocated and cataloged with permanent dataset names so they will be retained for use in restart if the step abends. These data sets are deleted when the step completes successfully.

  Prior to enabling internal step restart support, these data sets were probably allocated on system "scratch" space with a temporary, system assigned data set names. If your installation standards do not allow "permanent" data sets on DASD volumes used for temporary work space, you may need to use the WORK, RESTARTCKPT, and RESTARTWORK parameters to direct the internal step restart data sets to a generic unit or storage class that allows cataloged data sets.

- Dynamic Allocation

  When internal step restart is active, dynamic allocation is employed for the work data sets. If your installation restricts dynamic allocation of large, cataloged data sets, you may need to use the WORK, RESTARTCKPT, and RESTARTWORK parameters to direct work data set allocation to a generic unit or storage class where dynamic allocation is allowed.

- Data Set Names

  The SAS work data set, internal step restart control data set, and multiple work file data sets are allocated and cataloged according to the standard CA MICS unit database data set name conventions. The default DDNAME and data set names are:

  - SAS work data set,

```
            //cccXWORK DD DSN=prefix.MICS.cccXWORK,.....

      o Internal step restart control data set,
            //cccXCKPT DD DSN=prefix.MICS.cccXCKPT,.....

      o Multiple work file data sets,
            //WORKnn   DD DSN=prefix.MICS.cccWRKnn,.....

    Since these data sets conform to the same data set name
    conventions as your existing CA MICS data sets, there
    should be few, if any, data set name related allocation
    issues.  However, it is possible to override the data set
    names if required.  Please contact CA MICS Product
    Support for assistance if you must alter data set names.


RESTARTCKPT
-----------

This statement is optional.  Specify the following to
override default data set allocation parameters for the
internal step restart checkpoint data set:

RESTARTCKPT   data_set_allocation_parameters

Note:  RESTARTCKPT is ignored when you specify RESTART NO.

The internal step restart checkpoint data set (or cccXCKPT
data set) contains processing status, control, and SAS
environmental information for internal step restart
processing checkpoints.  This includes a copy of the SAS WORK
format and macro catalogs, current macro variable values, and
a description of work files that may be needed to restart
DAYnnn processing.

By default, the cccXCKPT data set is allocated according to
the values you specified for the WORKUNIT and WORKSPACE
parameters in prefix.MICS.PARMS(JCLDEF).  Specify RESTARTCKPT
to override this default, either to alter the space
allocation or to use System Managed Storage (SMS) parameters
to control data set placement and characteristics.

Note:  If you allocate insufficient space for the cccXCKPT
data set, DAYnnn processing will fail and can only be
restarted from the beginning.

Note:  You can override the RESTARTCKPT data set allocation
parameters at execution-time using the //PARMOVRD facility.
For more information about execution-time override of dynamic
```

```
data set allocation parameters, see the PIOM, section 2.3.6.

Specify data set allocation parameters, separated by blanks,
according to SAS LIBNAME statement syntax.  If you need
multiple lines, repeat the RESTARTCKPT keyword on the
continuation line.

RESTARTCKPT accepts the engine/host options documented in the
SAS Companion for the z/OS Environment, including STORCLAS,
UNIT, SPACE, BLKSIZE, DATACLAS, MGMTCLAS, and VOLSER.

Important!  DO NOT SPECIFY THE DISP PARAMETER.

Example 1:

RESTARTCKPT  STORCLAS=MICSTEMP SPACE=(xxxx,(pp,ss),,,ROUND)

where:

  STORCLAS - specifies a storage class for a new data set.
             The name can have up to eight characters.

  SPACE    - specifies how much disk space to provide for
             a new data set being allocated, where:

             xxxx is TRK, CYL, or blklen
             pp is the primary allocation
             ss is the secondary allocation

             and ROUND specifies that the allocated space be
             "rounded" to a cylinder boundary when the unit
             specified was a block length.  ROUND is ignored
             with the TRK or CYL options.

Example 2 (multiple lines):

RESTARTCKPT  STORCLAS=MICSTEMP UNIT=SYSDA
RESTARTCKPT  SPACE=(xxxx,(pp,ss),,,ROUND)

where:

  STORCLAS - specifies a storage class for a new data set.
             The name can have up to eight characters.

  UNIT     - specifies the generic unit for a new data set.
             The name can have up to eight characters.

  SPACE    - specifies how much disk space to provide for
             a new data set being allocated.
```

```
RESTARTWORK
-----------

This statement is optional.  Specify the following to
override default data set allocation parameters for the
internal step restart WORK data set:

RESTARTWORK   data_set_allocation_parameters

Note:  RESTARTWORK is ignored when you specify RESTART NO.

The internal step restart WORK data set (or cccXWORK data
set) contains the intermediate work files that are not
enabled to multiple work file support, including those files
you may have specified on the optional NOMULT statement.

By default, the cccXWORK data set is allocated according to
the values you specified for the WORKUNIT and WORKSPACE
parameters in prefix.MICS.PARMS(JCLDEF).  Specify RESTARTWORK
to override this default, either to alter the space
allocation or to use System Managed Storage (SMS) parameters
to control data set placement and characteristics.

Note:  If you allocate insufficient space for the cccXWORK
data set, DAYnnn processing will fail and can only be
restarted from the beginning.

Note:  You can override the RESTARTWORK data set allocation
parameters at execution-time using the //PARMOVRD facility.
For more information about execution-time override of dynamic
data set allocation parameters, see the PIOM, section 2.3.6.

Specify data set allocation parameters, separated by blanks,
according to SAS LIBNAME statement syntax.  If you need
multiple lines, repeat the RESTARTWORK keyword on the
continuation line.

RESTARTWORK accepts the engine/host options documented in
"SAS Companion for the z/OS Environment", including STORCLAS,
UNIT, SPACE, BLKSIZE, DATACLAS, MGMTCLAS, and VOLSER.

Important!  DO NOT SPECIFY THE DISP PARAMETER.

Example 1:

  RESTARTWORK  STORCLAS=MICSTEMP SPACE=(xxxx,(pp,ss),,,ROUND)

where:
```

```
  STORCLAS - specifies a storage class for a new data set.
             The name can have up to eight characters.

  SPACE    - specifies how much disk space to provide for
             a new data set being allocated, where:

             xxxx is TRK, CYL, or blklen
             pp is the primary allocation
             ss is the secondary allocation

             and ROUND specifies that the allocated space be
             "rounded" to a cylinder boundary when the unit
             specified was a block length.  ROUND is ignored
             with the TRK or CYL options.

Example 2 (multiple lines):

  RESTARTWORK  STORCLAS=MICSTEMP UNIT=SYSDA
  RESTARTWORK  SPACE=(xxxx,(pp,ss),,,ROUND)

where:

  STORCLAS - specifies a storage class for a new data set.
             The name can have up to eight characters.

  UNIT     - specifies the generic unit for a new data set.
             The name can have up to 8 characters.

  SPACE    - specifies how much disk space to provide for
             a new data set being allocated.
The following section discusses enabling this option:

    1 - Enable Internal Step Restart
```

7.2.1.4.1 Enable Internal Step Restart

To enable the internal step restart in CA MICS Tandem Application Extension, follow the checklist provided below:

```
   ************************************************
   *                                              *
   *          ENABLE INTERNAL STEP RESTART        *
   *                                              *
   ************************************************
```

___ 1. Edit prefix.MICS.PARMS(cccOPS), where (ccc) is the component identifier, and specify:

RESTART YES

For additional information on related topic, review the documentation for this product on WORK, RESTARTWORK, and RESTARTCKPT parameters to override default data set allocation parameters.

___ 2. Submit the job in prefix.MICS.CNTL(cccPGEN).

___ 3. Edit prefix.MICS.PARMS(JCLGENU) so that it contains a single line that reads:

DAILY

or, if incremental update is enabled for this product in this unit database, specify:

DAILY INCRccc

where ccc is the product ID.

___ 4. Submit the job in prefix.MICS.CNTL(JCLGENU). Ensure that there are no error messages in MICSLOG or SYSTSPRT, that the MICSLOG contains the normal termination message, BAS10999I, and that the job completes with a condition code of zero.

___ 5. The following operational job(s) have changed:

DAILY

INCRccc (if incremental update is enabled)

If your site has implemented the operational CA MICS processes in a scheduling product, the JCL may have

```
to be refreshed in that product.  See the scheduling
product's administrator for the exact processes
involved in updating that product's representation of
the CA MICS jobs.
```

7.2.1.5 Incremental Update Statements

```
INCRUPDATE
----------

This statement is optional.  Specify this to enable
incremental update for this product:

INCRUPDATE YES

If you do not specify or enable the INCRUPDATE parameter,
then this option defaults to this and incremental update is
disabled:

INCRUPDATE NO

************************************************************
*                                                          *
*  Note:  Changing the INCRUPDATE parameter (either from NO *
*         to YES or from YES to NO) requires regeneration   *
*         of the DAILY operational job by executing         *
*         prefix.MICS.CNTL(JCLGEND) or by specifying        *
*         DAILY in prefix.MICS.PARMS(JCLGENU) and           *
*         executing prefix.MICS.CNTL(JCLGENU).              *
*                                                          *
*         If you specify INCRUPDATE YES, you must also      *
*         generate the INCRccc, cccIUALC, and cccIUGDG jobs *
*         (where ccc is the 3 character product ID).        *
*         Depending on the options you select, you may also *
*         need to execute the cccIUALC and/or cccIUGDG      *
*         jobs.                                             *
*                                                          *
************************************************************

Incremental update can significantly reduce time and resource
usage in the DAILY job by letting you split out a major
portion of daily database update processing into multiple,
smaller, incremental updates executed throughout the day.

o  Standard CA MICS database update processing involves (1)
   reading and processing raw input data to generate DETAIL
   and DAYS level CA MICS database files, followed by (2)
   summarization of DETAIL/DAYS level data to update
   week-to-date and month-to-date database files.

o  When you activate incremental update:

   -  You can execute the first-stage processing (raw data
      input to create DETAIL/DAYS files) multiple times
```

```
      throughout the day, each time processing a subset of
      the total day's input data.

   -  Then, during the final update of the day (in the
      DAILY job), the incremental DETAIL/DAYS files are
      "rolled-up" to the database DETAIL and DAYS
      timespans, and then summarized to update the
      week-to-date and month-to-date files.

o  Incremental update is independent of your internal step
   restart or DBSPLIT specifications.  You have the option
   to perform incremental updates with or without internal
   step restart support.

o  Incremental update is activated and operates
   independently by product.  The incremental update job
   for this product, INCRccc (where ccc is the product ID),
   can execute concurrently with the incremental update job
   for another product in the same unit database.

o  The CA MICS database remains available for reporting and
   analysis during INCRccc job execution.

************************************************************
*                                                          *
*  Note:  CA MICS is a highly configurable system          *
*         supporting up to 36 unit databases, each of which *
*         can be configured and updated independently.     *
*         Incremental update is just one of the options you *
*         can use to configure your CA MICS complex.       *
*                                                          *
*         All efforts should be made to employ CA MICS     *
*         configuration capabilities to minimize issues    *
*         prior to activating incremental update.  For     *
*         example:                                         *
*                                                          *
*         o  Splitting work to multiple units is an        *
*            effective way to enable parallel database     *
*            update processing                             *
*                                                          *
*         o  Adjusting account code definitions to ensure  *
*            adequate data granularity while minimizing    *
*            total database space and processing time      *
*                                                          *
*         o  Tailoring the database to drop measurements   *
*            and metrics of lesser value to your           *
*            data center, thereby reducing database update *
*            processing and resource consumption           *
*                                                          *
```

```
*         While incremental update is intended to reduce    *
*         DAILY job elapsed time, total resource usage of   *
*         the combined INCRccc and DAILY jobs steps can     *
*         increase due to the additional processing         *
*         required to maintain the incremental update       *
*         "to-date" files and for roll-up to the unit       *
*         database.  The increased total resource usage     *
*         will be more noticeable with small data volumes,  *
*         where processing code compile time is a greater   *
*         percentage of total processing cost.              *
*                                                           *
*************************************************************

Note:  When you activate incremental update (INCRUPDATE YES),
the following optional incremental update parameters are
enabled.  These parameters have no effect if incremental
update is disabled (INCRUPDATE NO).  For more details, see
the individual parameter descriptions later in this section.

o INCRDB     PERM/TAPE/DYNAM

o INCRDETAIL data_set_allocation_parameters

o INCRDAYS   data_set_allocation_parameters

o INCRCKPT   data_set_allocation_parameters

o INCRSPLIT  USE/IGNORE data_set_allocation_parameters

Incremental update processing reads and processes raw
measurement data to create and maintain DETAIL and DAYS level
"to-date" files for the current day.

o  These incremental update database files are maintained on
   unique z/OS data sets, independent of the standard CA MICS
   database files, and independent of any other product's
   incremental update database files.  There is one data set
   each for DETAIL and DAYS level "to-date" data and a single
   incremental update checkpoint data set for this product in
   this unit.

o  The incremental update DETAIL and DAYS files can be
   permanent DASD data sets, or they can be allocated
   dynamically as needed and deleted after DAILY job
   processing completes.  Optionally, you can keep the
   incremental update DETAIL and DAYS files on tape, with
   the data being loaded onto temporary DASD space as
   needed for incremental update or DAILY job processing.
   See the INCRDB PERM/TAPE/DYNAM option for more
```

information.

After activating incremental update, you will use three incremental update facility jobs found in prefix.MICS.CNTL (Note that ccc is the product ID):

o cccIUALC

  You execute this job to allocate and initialize the incremental update checkpoint file, and optionally the incremental update DETAIL and DAYS database files. cccIUALC is generally executed just ONE time.

o cccIUGDG

  You execute this job to add generation data group (GDG) index definitions to your system catalog in support of the INCRDB TAPE option. cccIUGDG is generally executed just ONE time.

o INCRccc

  This is the job you execute for each incremental update. You will integrate this job into your database update procedures for execution one or more times per day to process portions of the total day's measurement data.

  Note: The DAILY job is run once at the end of the day. It will perform the final incremental update for the day's data, and then roll-up the incremental DETAIL/DAYS files to the database DETAIL and DAYS timespans and update the week-to-date and month-to-date files.

INCRUPDATE Considerations
---

o Overhead

  Incremental update is intended to reduce DAILY job resource consumption and elapsed time by offloading a major portion of database update processing to one or more executions of the INCRccc job. In meeting this objective, incremental update adds processing in the INCRccc and DAILY jobs to accumulate data from each incremental update execution into the composite "to-date" DETAIL and DAYS incremental update files, and also adds processing in the DAILY job to copy the incremental update files to the unit database DETAIL and DAYS

timespans. The amount of this overhead and the savings in the DAILY job are site-dependent, and will vary based on input data volume and on the number of times INCRccc is executed each day.

In addition, activating incremental update will cause additional compile-based CPU time to be consumed in the DAYnnn DAILY job step. The increase in compile time is due to additional code included for each file structure in support of the feature. This increase should be static based on the scope of the CA MICS data integration product in terms of files. This compile-time increase does not imply an increase in elapsed or execution time. Incremental update allows I/O bound, intensive processing (raw data inputting, initial CA MICS transformation, etc.) to be distributed outside of the DAILY job. I/O processing is the largest contributor to elapsed time in large volume applications. Thus, the expected overall impact is a decrease in the actual runtime of the DAYnnn job step.

- Increased "Prime Time" Workload

  By offloading work from the DAILY job to one or more INCRccc executions throughout the day, you are potentially moving system workload and DASD work space usage from the "off-hours," (when the DAILY job is normally executed) to periods of the day where your system resources are in highest demand. You should schedule INCRccc executions carefully to avoid adverse impact to batch or online workloads. For example, if your site's "prime shift" is 8:00 AM to 5:00 PM, you might choose to schedule incremental updates for 7:00 AM (just before "prime shift") and 6:00 PM (just after "prime shift"), with the DAILY job executing just after midnight.

- Increased DASD Usage

  The DASD space required for the incremental update DETAIL and DAYS database files is in addition to the DASD space already reserved for the CA MICS database. By default, the incremental update database files are permanently allocated, making this DASD space unavailable for other applications. In general, you can assume that the incremental update database files will require space equivalent to two cycles of this product's DETAIL and DAYS timespan files.

  Alternatively, the incremental update database files can

be allocated in the first incremental update of the day and deleted by the DAILY job (see the INCRDB DYNAM option later in this section). This approach reduces the amount of time that the DASD space is dedicated to incremental update, and lets the amount of DASD space consumed increase through the day as you execute each incremental update.

A third option is to store the incremental update database files on tape (see the INCRDB TAPE option). With this approach, the DASD space is required just for the time that each incremental update or DAILY job step is executing. Note that while this alternative reduces the "permanent" DASD space requirement, the total amount of DASD space required while the incremental update or DAILY jobs are executing is unchanged. In addition, the TAPE option adds processing to copy the incremental update files to tape, and to reload the files from tape to disk.

Note: The incremental update checkpoint file is always a permanently allocated disk data set. This is a small data set and should not be an issue.

- Operational Complexity

  Incremental update expands your measurement data management and job scheduling issues. You must ensure that each incremental update and the DAILY job processes your measurement data chronologically; that is, each job must see data that is newer than the data processed by the prior job. By incrementally updating the database, you have more opportunities to miss a log file, or to process a log out of order.

- Interval End Effects

  Each incremental update processes a subset of the day's measurement data, taking advantage of early availability of some of the day's data, for example, when a measurement log fills and switches to a new volume. This can cause a problem if the measurement log split occurs while the data source is logging records for the end of a measurement interval, thus splitting the data for a single measurement interval across two log files. When an incremental update processes the first log file, the checkpoint high end timestamp is set to indicate that this split measurement interval has been processed. Then, when the rest of the measurement interval's data is

encountered in a later update, it can be dropped as duplicate data (because data for this measurement interval end timestamp has already been processed).

Appropriate scheduling of log dumps and incremental updates can avoid this problem. For example, if you plan to run incremental updates at 7:00 AM and 6:00 PM, you could force a log dump in the middle of the measurement interval just prior to the scheduled incremental update executions. This is an extension of the procedure you may already be using for end-of-day measurement log processing. The objective is to ensure that all records for each monitor interval are processed in the same incremental update.

- Dynamic Allocation

  When you activate incremental update and specify TAPE or DYNAM for the INCRDB parameter, dynamic allocation is employed for the incremental update database files. If your site restricts dynamic allocation of large, cataloged data sets, you must use the INCRDETAIL and INCRDAYS parameters to direct incremental update data set allocation to a generic unit or storage class where dynamic allocation is allowed.

- Data Set Names

  The incremental update database files are allocated and cataloged according to standard CA MICS unit database data set name conventions. The DDNAME and default data set names are (where ccc is the product ID):

  - Incremental update checkpoint file,
    ```
    //IUCKPT   DD DSN=prefix.MICS.ccc.IUCKPT,.....
    ```
  - Incremental update DETAIL
    ```
    //IUDETAIL DD DSN=prefix.MICS.ccc.IUDETAIL,.....
    ```
  - Incremental update DAYS
    ```
    //IUDAYS   DD DSN=prefix.MICS.ccc.IUDAYS,....
    ```

  Since these data sets conform to the same data set name conventions as your existing CA MICS data sets, there should be few, if any, data-set-name-related allocation issues. However, it is possible to override the data set names if required. Contact Technical Support at http://ca.com/support for assistance if you must change data set names.

```
INCRDB
------

This statement is optional.  The default is this:

INCRDB PERM

Note:  INCRDB is ignored when you specify INCRUPDATE NO.

Specify this statement or take the default, to keep the incremental
update database DETAIL and DAYS files on permanently
allocated DASD data sets:

INCRDB PERM

Execute the prefix.MICS.CNTL(cccIUALC) job to allocate the
incremental update database files.

************************************************************
*                                                          *
*  Note:  The incremental update checkpoint file is always  *
*         a permanently allocated DASD data set.           *
*                                                          *
************************************************************


Specify this to offload the incremental update DETAIL and
DAYS files to tape between incremental update executions:

INCRDB TAPE #gdgs UNIT=name

With the TAPE option, the incremental update DETAIL and DAYS
DASD data sets are dynamically allocated at the beginning of
the incremental update job or DAILY job step, and then are
deleted after the job step completes.

o  The first incremental update job of the day allocates
   and initializes the incremental update database files.
   At the end of the job, the DETAIL and DAYS files are
   copied to a new (+1) generation of the incremental
   update tape data sets.  Then the DASD files are deleted.

o  Subsequent incremental update jobs restore the DASD
   incremental update database files from the current, (0)
   generation, incremental update tape data sets before
   processing the input measurement data.  At the end of
   the job, the DETAIL and DAYS files are copied to a new
```

(+1) generation of the incremental update tape data sets. Then the DASD files are deleted.

- The DAILY job step also restores the DASD incremental update database files from the (0) generation tape files before processing the input data, but does NOT copy the incremental update database files to tape. Thus, the DAILY job actually creates a new, null (+1) generation.

- Use the #gdgs parameter to specify the maximum number of incremental update tape generations. The minimum is 2 and the maximum is 99, with a default of 5.
  Set the number of generations equal to or greater than the number of incremental updates, including the DAILY job you plan to execute each day. This facilitates restart and recovery if you encounter problems requiring you to reprocess portions of the daily measurement data.

- Use the optional UNIT=name parameter to specify a tape unit name for the incremental update database output tapes. The default is to use the same tape unit as the input tapes.

- A special index must be created in your system catalog for each of the incremental update tape data set generation data groups. The prefix.MICS.CNTL(cccIUGDG) job will generate the statements to create the incremental update GDG index definitions.

  - Before each index is built, it is deleted. These DLTX (or DELETE) statements causes an error message if no entry exists. This is done so that you can change the number of entries without having to delete each of the index entries.

  - DLTX and BLDG (or DELETE and DEFINE) fail if there is a cataloged data set with the same index. IDCAMS (or IEHPROGM) issues a message and gives a return code of 8. This issue is not a problem for non-GDG entries or if the GDG already has the desired number of entries.

  - If you want to change the number of entries kept in a GDG with cataloged data sets, do the following:

    1. Uncatalog any existing entries in the GDG.
    2. Delete the index with a DLTX (or DELETE).
    3. Create the index with a BLDG (or DEFINE).

```
      4. Catalog any entries that are uncataloged in step 1.

o  The incremental update tape data set names are as follows,
   where ccc is the product ID:

   -  Incremental update tape DETAIL file
         tapeprefix.MICS.ccc.IUXTAPE.GnnnnV00

   -  Incremental update tape DAYS file
         tapeprefix.MICS.ccc.IUDTAPE.GnnnnV00

************************************************************
*                                                          *
*  Note:  The INCRDETAIL and INCRDAYS parameters are       *
*         required when you specify INCRDB TAPE.           *
*                                                          *
************************************************************


Specify this parameter to allocate dynamically the
incremental update  DETAIL and DAYS DASD data sets in the first
incremental update of the day, and then delete these data sets
at the end of the DAILY job step:

INCRDB DYNAM


o  With this option, no space is used for the incremental
   update database files during the time between the end of
   the DAILY job step and the beginning of the next day's
   first incremental update.

o  With this approach, you can set the data set allocation
   parameters so that the incremental update DETAIL and DAYS
   data sets start out with a minimum allocation and then
   grow through secondary allocations as more space is
   required for subsequent incremental updates. For
   example, enough space for one incremental update.

************************************************************
*                                                          *
*  Note:  The INCRDETAIL and INCRDAYS parameters are       *
*         required when you specify INCRDB DYNAM.          *
*                                                          *
************************************************************



INCRDETAIL
```

```
----------

This statement is required if you specify either of these:

INCRDB TAPE

INCRDB DYNAM

Otherwise, this statement is optional.  There is no default.


Specify this to define data set allocation parameters for the
incremental update DETAIL data set (IUDETAIL):

INCRDETAIL    data_set_allocation_parameters

Note:  INCRDETAIL is ignored when you specify INCRUPDATE NO.

The incremental update DETAIL data set (IUDETAIL) contains
the current incremental update detail-level database files,
and the DETAIL "to-date" data for the current daily update
cycle.  You should allocate DASD space equivalent to two
cycles of this product's DETAIL timespan data.

If you specified INCRDB PERM (the default), your INCRDETAIL
parameter specifications are used in generating the cccIUALC
job (where ccc is the product ID).

o  You will execute the cccIUALC job to allocate and
   initialize the incremental update database and checkpoint
   files.

o  Omit the INCRDETAIL parameter if you prefer to specify
   data set allocation parameters directly in the generated
   prefix.MICS.CNTL(cccIUALC) job.

If you specified INCRDB TAPE or INCRDB DYNAM, your INCRDETAIL
parameter specifications are used in incremental update
DETAIL data set dynamic allocation during incremental update
or DAILY job step execution.

o  The INCRDETAIL parameter is required for the TAPE or
   DYNAM option.

o  Specify data set allocation parameters, separated by
   blanks, according to SAS LIBNAME statement syntax.  If
   you need multiple lines, repeat the INCRDETAIL keyword
   on the continuation line.
```

```
o  INCRDETAIL accepts the engine/host options documented in
   the SAS Companion for the z/OS Environment, including
   STORCLAS, UNIT, SPACE, BLKSIZE, DATACLAS, MGMTCLAS, and
   VOLSER.

   Important!  DO NOT SPECIFY THE DISP PARAMETER.

o  You can override the INCRDETAIL data set allocation
   parameters at execution-time using the //PARMOVRD
   facility.  For more information about execution-time
   override of dynamic data set allocation parameters, see
   the PIOM, section 2.3.6.

Example 1:

INCRDETAIL   STORCLAS=MICSTEMP SPACE=(xxxx,(pp,ss),,,ROUND)

where:

  STORCLAS - specifies a storage class for a new data set.
             The name can have up to eight characters.

  SPACE    - specifies how much disk space to provide for
             a new data set being allocated, where:

             xxxx is TRK, CYL, or blklen
             pp   is the primary allocation
             ss   is the secondary allocation

             and ROUND specifies that the allocated space be
             "rounded" to a cylinder boundary when the unit
             specified was a block length.  ROUND is ignored
             with the TRK or CYL options.

Example 2 (multiple lines):

INCRDETAIL   STORCLAS=MICSTEMP UNIT=SYSDA
INCRDETAIL   SPACE=(xxxx,(pp,ss),,,ROUND)

where:

  STORCLAS - specifies a storage class for a new data set.
             The name can have up to eight characters.

  UNIT     - specifies the generic unit for a new data set.
             The name can have up to eight characters.

  SPACE    - specifies how much disk space to provide for
             a new data set being allocated.
```

```
INCRDAYS
--------

This statement is required if you specify either of these:

INCRDB TAPE

INCRDB DYNAM

Otherwise, this statement is optional.  There is no default.


Specify this to define data set allocation parameters for the
incremental update DAYS data set (IUDAYS):

INCRDAYS      data_set_allocation_parameters

Note:  INCRDAYS is ignored when you specify INCRUPDATE NO.

The incremental update DAYS data set (IUDAYS) contains the
current incremental update days-level database files, and the
DAYS "to-date" data for the current daily update cycle.  You
should allocate DASD space equivalent to two cycles of this
product's DAYS timespan data.

If you specified INCRDB PERM (the default), your INCRDAYS
parameter specifications are used in generating the cccIUALC
job (where ccc is the product ID).

o  You will execute the cccIUALC job to allocate and
   initialize the incremental update database and checkpoint
   files.

o  Omit the INCRDAYS parameter if you prefer to specify
   data set allocation parameters directly in the generated
   prefix.MICS.CNTL(cccIUALC) job.

If you specified INCRDB TAPE or INCRDB DYNAM, your INCRDAYS
parameter specifications are used in incremental update DAYS
data set dynamic allocation during incremental update or
DAILY job step execution.

o  The INCRDAYS parameter is required for the TAPE or DYNAM
   option.

o  Specify data set allocation parameters, separated by
```

```
   blanks, according to SAS LIBNAME statement syntax.  If
   you need multiple lines, repeat the INCRDAYS keyword on
   the continuation line.

o  INCRDAYS accepts the engine/host options documented in the
   SAS Companion for the z/OS Environment, including
   STORCLAS, UNIT, SPACE, BLKSIZE, DATACLAS, MGMTCLAS, and
   VOLSER.

   Important!  DO NOT SPECIFY THE DISP PARAMETER.

o  You can override the INCRDAYS data set allocation
   parameters at execution-time using the //PARMOVRD
   facility.  For more information about execution-time
   override of dynamic data set allocation parameters, see
   the PIOM, Section 2.3.6.

Example 1:

INCRDAYS     STORCLAS=MICSTEMP SPACE=(xxxx,(pp,ss),,,ROUND)

where:

  STORCLAS - specifies a storage class for a new data set.
             The name can have up to eight characters.

  SPACE    - specifies how much disk space to provide for
             a new data set being allocated, where:

             xxxx is TRK, CYL, or blklen
             pp   is the primary allocation
             ss   is the secondary allocation

             and ROUND specifies that the allocated space be
             "rounded" to a cylinder boundary when the unit
             specified was a block length.  ROUND is ignored
             with the TRK or CYL options.

Example 2 (multiple lines):

INCRDAYS     STORCLAS=MICSTEMP UNIT=SYSDA
INCRDAYS     SPACE=(xxxx,(pp,ss),,,ROUND)

where:

  STORCLAS - specifies a storage class for a new data set.
             The name can have up to eight characters.

  UNIT     - specifies the generic unit for a new data set.
```

```
              The name can have up to eight characters.

 SPACE     - specifies how much disk space to provide for
             a new data set being allocated.



INCRCKPT
--------

This statement is optional.  Specify this to override default
data set allocation parameters for the incremental update
checkpoint data set:

INCRCKPT      data_set_allocation_parameters

Note:  INCRCKPT is ignored when you specify INCRUPDATE NO.


The incremental update checkpoint data set tracks incremental
update job status and the data that has been processed during
the current daily update cycle.  The incremental update
checkpoint is used to detect and block the input of duplicate
data during incremental update processing.  This data set
will be exactly the same size as prefix.MICS.CHECKPT.DATA
(the unit checkpoint data set), usually 20K to 200K depending
on the prefix.MICS.PARMS(SITE) CKPTCNT parameter (100-9999).

Your INCRCKPT parameter specifications are used in generating
the cccIUALC job (where ccc is the product ID).

o  You will execute the cccIUALC job to allocate and
   initialize the incremental update checkpoint file.  If you
   specified INCRDB PERM, then the cccIUALC job will also
   allocate the incremental update DETAIL and DAYS database
   files.

o  By default the incremental update checkpoint data set is
   allocated as SPACE=(TRK,(5,2)) using the value you
   specified for the prefix.MICS.PARMS(JCLDEF) DASDUNIT
   parameter.

o  Omit the INCRCKPT parameter if you prefer to override
   data set allocation parameters directly in the generated
   prefix.MICS.CNTL(cccIUALC) job.

Specify data set allocation parameters, separated by blanks,
according to SAS LIBNAME statement syntax.  If you need
multiple lines, repeat the INCRCKPT keyword on the
```

```
continuation line.

INCRCKPT accepts the engine/host options documented in the
SAS Companion for the MVS Environment, including STORCLAS,
UNIT, SPACE, BLKSIZE, DATACLAS, MGMTCLAS, and VOLSER.

Important!  DO NOT SPECIFY THE DISP PARAMETER.


Example 1:

INCRCKPT     STORCLAS=MICSTEMP SPACE=(xxxx,(pp,ss),,,ROUND)


where:

  STORCLAS - specifies a storage class for a new data set.
             The name can have up to eight characters.

  SPACE    - specifies how much disk space to provide for
             a new data set being allocated, where:

             xxxx is TRK, CYL, or blklen
             pp   is the primary allocation
             ss   is the secondary allocation

             and ROUND specifies that the allocated space be
             "rounded" to a cylinder boundary when the unit
             specified was a block length.  ROUND is ignored
             with the TRK or CYL options.


Example 2 (multiple lines):

INCRCKPT     STORCLAS=MICSTEMP UNIT=SYSDA
INCRCKPT     SPACE=(xxxx,(pp,ss),,,ROUND)


where:

  STORCLAS - specifies a storage class for a new data set.
             The name can have up to eight characters.

  UNIT     - specifies the generic unit for a new data set.
             The name can have up to eight characters.

  SPACE    - specifies how much disk space to provide for
             a new data set being allocated.
```

```
INCRSPLIT
---------

This statement is optional and defaults to this:

INCRSPLIT IGNORE

Specify the following if you want the incremental update job
for this product to get input measurement data from the
output of the SPLITSMF job.  The optional
data_set_allocation_parameters are used by the SPLITSMF job
when creating the measurement data file for this product.

INCRSPLIT  USE  data_set_allocation_parameters

Note:  INCRSPLIT is ignored when you specify INCRUPDATE NO.

This option would be used when multiple products in a
single unit database are enabled to incremental update.  The
SPLITSMF job performs the same function for incremental
update jobs as the DAILY job DAYSMF step performs for the
DAYnnn database update steps.

o  The SPLITSMF job dynamically allocates, catalogs, and
   populates prefix.MICS.ccc.IUSPLTDS data sets for each
   product in the unit database for which you specified both
   the INCRUPDATE YES and INCRSPLIT USE parameters.  These
   data sets are then deleted after processing by the
   appropriate INCRccc job.

o  Specify data set allocation parameters, separated by
   blanks, according to SAS LIBNAME statement syntax.  If you
   need multiple lines, repeat the INCRSPLIT keyword on each
   continuation line.

o  INCRSPLIT accepts the engine/host options documented in
   the SAS Companion for the MVS Environment, including
   STORCLAS, UNIT, SPACE, BLKSIZE, DATACLAS, MGMTCLAS, and
   VOLSER.

   Important!  DO NOT SPECIFY THE DISP PARAMETER.


Specify the following or accept the default if you want the
incremental update jobs for this product to get their input
measurement data from the data sets specified in the INPUTccc
(or INPUTSMF) member of prefix.MICS.PARMS:
```

```
INCRSPLIT  IGNORE

When you specify INCRSPLIT IGNORE, this product will NOT
participate in SPLITSMF job processing.


Example 1:

INCRSPLIT USE  STORCLAS=MICSTEMP SPACE=(xxxx,(pp,ss),,,ROUND)


where:

  STORCLAS - specifies a storage class for a new data set.
             The name can have up to eight characters.

  SPACE    - specifies how much disk space to provide for
             a new data set being allocated, where:

             xxxx is TRK, CYL, or blklen
             pp   is the primary allocation
             ss   is the secondary allocation

             and ROUND specifies that the allocated space be
             "rounded" to a cylinder boundary when the unit
             specified was a block length.  ROUND is ignored
             with the TRK or CYL options.


Example 2 (multiple lines):

INCRSPLIT  USE  STORCLAS=MICSTEMP UNIT=SYSDA
INCRSPLIT       SPACE=(xxxx,(pp,ss),,,ROUND)


where:

  STORCLAS - specifies a storage class for a new data set.
             The name can have up to eight characters.

  UNIT     - specifies the generic unit for a new data set.
             The name can have up to eight characters.

  SPACE    - specifies how much disk space to provide for
             a new data set being allocated.
```

```
DYNAMWAIT
---------

This statement is optional.  Specify the following:

     DYNAMWAIT   minutes

to override the default amount of time, in minutes, the DAILY
and/or INCRccc job will wait for an unavailable data set.

Note:  This optional parameter is not normally specified.
       The system default is adequate for most data centers.

Internal Step Restart and Incremental Update facilities use
z/OS dynamic allocation services to create new data sets and
to access existing data sets.  Data set naming conventions
and internal program structure are designed to minimize data
set contention.  However, if data set allocation does fail
because another batch job or online user is already using a
data set, DAILY and/or INCRccc processing will wait 15
seconds and then try the allocation again.  By default, the
allocation will be attempted every 15 seconds for up to 15
minutes.  After 15 minutes, the DAILY or INCRccc job will
abort.

If data set contention in your data center does cause
frequent DAILY or INCRccc job failures, and you are unable to
resolve the contention through scheduling changes, you may
want to use the DYNAMWAIT parameter to increase the maximum
number of minutes the DAILY and/or INCRccc jobs will wait for
the data set to become available.

On the other hand, if your data center standards require
that the DAILY and/or INCRccc jobs fail immediately if
required data sets are unavailable, specify the following:

     DYNAMWAIT 0

Note:  You can override the DYNAMWAIT parameter at
       execution-time using the //PARMOVRD facility.  For
       more information about execution-time override of
       dynamic data set allocation parameters, see the PIOM,
       section 2.3.6.


The following section discusses enabling this option:

    1 - Implement Incremental Update
```

7.2.1.5.1 Implement Incremental Update

To implement incremental update in the CA MICS Application Extension for Tandem System, follow the checklist provided below:

```
   ************************************************
   *                                              *
   *           IMPLEMENT INCREMENTAL UPDATE       *
   *                                              *
   ************************************************
```

___ 1. Edit prefix.MICS.PARMS(cccOPS), where (ccc) is the component identifier:

- Specify the following:

  INCRUPDATE YES

- If you want to store the incremental update database files on tape between incremental updates, specify this:

  INCRDB TAPE #gdgs

- If you want to allocate the incremental update database files during the first incremental update of the day and delete these data sets at the end of the DAILY job step, specify this:

  INCRDB DYNAM

- If you specified INCRDB TAPE or INCRDB DYNAM, then you must also specify this:

  INCRDETAIL data_set_allocation_parameters
  INCRDAYS   data_set_allocation_parameters

- If you want the incremental update job for this product to get input measurement data from the output of the SPLITSMF job, specify this:

  INCRSPLIT USE data_set_allocation_parameters

- For additional information on related topic, review the documentation for this product on INCRCKPT, INCRDETAIL, INCRDAYS, or INCRSPLIT parameters to override default data set allocation parameters.

___ 2. Submit the job in prefix.MICS.CNTL(cccPGEN).

___ 3. Edit prefix.MICS.PARMS(JCLGENU) so that it contains two or more lines reading:

```
DAILY
INCRccc cccIUALC cccIUGDG
```

___ 4. Submit the job in prefix.MICS.CNTL(JCLGENU). Ensure that there are no error messages in MICSLOG or SYSTSPRT, that the MICSLOG contains the normal termination message, BAS10999I, and that the job completes with a condition code of zero.

___ 5. Edit the job in prefix.MICS.CNTL(cccIUALC).

- Inspect and/or specify data set allocation parameters for the incremental update database and checkpoint files. If you specified INCRDB TAPE or INCRDB DYNAM, the cccIUALC job will only allocate the incremental update checkpoint data set.
- Submit the job. Ensure that there are no error messages in MICSLOG or SASLOG, and that the job completes with a condition code of zero.

___ 6. If you specified INCRDB TAPE, submit the job in prefix.MICS.CNTL(cccIUGDG) to define generation group indexes for the incremental update DETAIL and DAYS tape data sets. Examine SASLOG, MICSLOG, and SYSPRINT to verify that the generation group indexes were correctly defined.

Note: You may see error messages for the DLTX (or DELETE) statements. This is not a problem. cccIUGDG deletes each index prior to defining it, and an error message is issued if the index does not yet exist (e.g., if this is the first time you ran the cccIUGDG job).

___ 7. The following operational job(s) have changed:

```
DAILY  INCRccc
```

If your site has implemented the operational CA MICS processes in a scheduling product, the JCL may have to be refreshed in that product. See the scheduling product's administrator for the exact processes

involved in updating that product's representation of the CA MICS jobs.

___ 8. Implement operational procedures for gathering input measurement data and executing incremental updates (INCRccc) during the day.

You may also need to modify operational procedures for the DAILY job to ensure that processing is limited to input measurement data that has not been input to one of the day's incremental update executions.

7.2.2 Input DD Statements (INPUTTDM)

The prefix.MICS.PARMS(INPUTTDM) parameter contains the JCL for the input to be included in the daily update processing. The DD statements required in this member are determined by the values coded in TDMOPS:

- If LOADID is specified, INPUTTDM is the required DDNAME for the input data set.
- If NOLOADID is specified, one DDNAME per entity type is required, using the format TDMxxxx where xxxx is the LOADID value specified in the ENTITIES statement in TDMOPS.

NOTE: The DAILY job must be regenerated for changes to this member to become effective. Regenerate the DAILY JCL by specifying DAILY in prefix.MICS.PARMS(JCLGENU) and submitting prefix.MICS.CNTL(JCLGENU). If a DDNAME is added or deleted, make the corresponding change to the OPTIONS and ENTITIES statements in TDMOPS and submit a TDMPGEN.

Examples of INPUTTDM follow. They correspond, by example number, to the examples for TDMOPS:

Example 1: One system with entity type in the LOADID field.

```
//INPUTTDM DD DSN=TANDEM.SYSTEM1.DATA,DISP=SHR
```

Example 2: One system without the entity type in the LOADID field.

```
//TDMCPU   DD DISP=SHR,DSN=TANDEM.SYSTEM2.CPU.DATA
//TDMDISC  DD DISP=SHR,DSN=TANDEM.SYSTEM2.DISC.DATA
//TDMTMF   DD DISP=SHR,DSN=TANDEM.SYSTEM2.TMF.DATA
//TDMPROC  DD DISP=SHR,DSN=TANDEM.SYSTEM2.PROC.DATA
//TDMNETL  DD DISP=SHR,DSN=TANDEM.SYSTEM2.NETL.DATA
```

Example 3: Three systems, two with LOADID values and one without. The ENTITIES statement must match the entity types for the system without LOADID information.

```
//INPUTTDM DD DISP=SHR,DSN=TANDEM.SYSTEM1.DATA
//         DD DISP=SHR,DSN=TANDEM.SYSTEM3.DATA
//TDMCPU   DD DISP=SHR,DSN=TANDEM.SYSTEM2.CPU.DATA
//TDMDISC  DD DISP=SHR,DSN=TANDEM.SYSTEM2.DISC.DATA
//TDMTMF   DD DISP=SHR,DSN=TANDEM.SYSTEM2.TMF.DATA
```

```
//TDMPROC  DD DISP=SHR,DSN=TANDEM.SYSTEM2.PROC.DATA
//TDMNETL  DD DISP=SHR,DSN=TANDEM.SYSTEM2.NETL.DATA
```

7.2.3 Database Space Modeling (DBMODEL)

This section describes the information required to define the Analyzer to the Database Space Modeling Facility. Specifically, the user must provide values for the cycle (data retention) definitions. The special considerations for defining the FILE input statements to the Database Space Modeling facility are discussed.

Section 2.3.4 of the PIOM, Database Space Modeling Facility, explains how these values are used in estimating the DASD requirements of the database, and how the user can input these values to the modeling facility.

This section covers the following topics:

1 - Data Retention Specifications (FILE Statements)

2 - DBMODEL Input Statements

7.2.3.1 Data Retention Specifications (FILE Statements)

Data retention specifications tell the CA MICS database how many cycles of data, for each file in each supported timespan, to save both online and in archive mode. Figure 1-1 is the worksheet to help you collect this information.

The worksheet is organized by information area. Each file in the area is listed by name. For each file, a line is formatted to allow six definitions in the online database and two in the archive database:

- The online database files quantify the number of cycles of data that is maintained in the DETAIL, DAYS, WEEKS, MONTHS, and YEARS timespans and the TABLES data area.

- The two definitions for the archive database files quantify the number of cycles of data to be retained, up to the cutoff limit defined. The archive definitions have no impact on the size of the database and can be specified whethert the weekly and/or monthly archive history files have actually been activated (see Section 2.3.3, CA MICS JCL Planning and Parameters, of the PIOM).

The worksheet formats provide an underscored area for the user's definition, followed by the recommended value, shown within parentheses. If the underscored area contains a value of 00, the file is not supported for the indicated timespan. To add support, perform database tailoring as described in Section 6.2, Tailoring the Database, of the System Modification Guide (SMG).

When specifying a retention limit, remember that the number may never be zero if the file has been defined to be active in the timespan.

File	DETAIL	DAYS	WEEKS	MONTHS	YEARS	TABLES	Archive Cut-Off WEEKS	Archive Cut-Off MONTHS
FILE TDM TDMCLE	07	10	06	06	02	00	000	000
FILE TDM TDMCPE	07	45	12	18	02	00	053	024
FILE TDM TDMDSC	07	45	12	18	02	00	053	024
FILE TDM TDMDSO	01	01	01	01	01	00	000	000
FILE TDM TDMDVE	03	10	18	12	02	00	000	000
FILE TDM TDMFLE	01	07	18	03	02	00	000	000
FILE TDM TDMLNE	01	07	06	06	01	00	000	000
FILE TDM TDMNTL	07	45	18	12	02	00	053	024
FILE TDM TDMOPD	07	45	12	18	01	00	000	000
FILE TDM TDMPRH	10	00	12	18	02	00	000	000
FILE TDM TDMPRS	08	02	12	18	02	00	053	024
FILE TDM TDMSVN	07	10	06	06	02	00	000	000
FILE TDM TDMSQP	01	00	01	01	01	00	000	000
FILE TDM TDMSQS	10	00	12	18	01	00	000	000
FILE TDM TDMSYE	07	15	18	24	02	00	000	000
FILE TDM TDMSYP	07	45	12	18	02	00	053	024
FILE TDM TDMTME	07	15	18	24	02	00	000	000
FILE TDM TDMTMN	07	15	18	24	01	00	000	000

Figure 1-1. Default File Retention Periods

7.2.3.2 DBMODEL Input Statements

The DBMODEL member of prefix.MICS.PARMS provides the input to the Database Space Modeling Facility.

Using the information collected on the worksheet in Figure 1-1, code the DBMODEL member. To actually perform the space modeling, submit the jobs as described in the CA MICS Planning, Installation, Operation, and Maintenance Guide, Section 2.3.4.2.

Chapter 8: Installation

Be sure to review Chapter 6 of this guide before proceeding to the installation checklists in the PIOM. This material provides helpful information about the data sources and the input parameters you will need to supply for proper operation of this application extension.

We recommend initially installing this application extension in a TEST unit. Select about one hour of representative measurement data for each MEASURE entity type and store the data in the appropriate files as determined by your TDMOPS OPTIONS statement, documented in Chapter 7 of this guide.

The CA MICS Application Extension for Tandem Systems executes as step 076 in the CA MICS operational jobs. There will be a step in the DAILY, WEEKLY, MONTHLY, and YEARLY jobs for this product.

Chapter 9: Processing

This chapter is intentionally left blank for future use.

Chapter 10: Modification

This chapter is intentionally left blank for future use.

Appendix A: Messages

Messages generated by this application extension are listed below. Some messages are generated during the processing of the control statements, while others are caused by various conditions in the data found during the processing. The messages are listed in ascending numerical sequence and include the reason for the messages, user action where appropriate, and applicable references to documentation.

The following type codes are used to categorize the messages:

Information	designates a note that documents an option or potentially important feature in the data.
Warning	designates a condition in either the data or control statements that does not affect the application extension's operation but may lead to unexpected results.
Error	designates a problem has been encountered with a control statement or data that will prevent a successful run of this application extension.

Messages:

TDM00001I	Informational message naming the parameter member that is being processed.
TDM00002E	The ORGSYSID value on the TDMOPS OPTIONS statement must be 4 characters. Modify ORGSYSID and rerun TDMPGEN. Reference: Section 7.2.1 of this guide.
TDM00003E	This option value must be LOADID or NOLOADID. Specify LOADID or NOLOADID as the third option value on the OPTIONS statement in prefix.MICS.PARMS(TDMOPS) and resubmit the TDMPGEN. Reference: Section 7.2.1 of this guide.

```
TDM00005W   The SCP keyword on the SYSID parameter should be
            TANDEM.  Verify this is the correct SYSID value.
            Modify the SYSID and TDMOPS parameters and run
            BASPGEN and TDMPGEN.

            Reference:  Section 7.2.5 of this guide.

TDM00006E   TDMOPS System Name is missing from the OPTIONS
            statement.  The required keywords are ORGSYSID,
            system-name (system name on the MEASURE data),
            and LOADID/NOLOADID.

            Reference:  Section 7.2.1 of this guide.

TDM00007E   Only one OPTIONS statement per ORGSYSID is
            allowed.  Remove or correct the ORGSYSID and
            rerun the TDMPGEN.

            Reference:  Section 7.2.1 of this guide.

TDM00008W   This warning is printed if the same entity type
            is entered twice.  The second entry will be
            ignored.  If it is a duplicate, continue with the
            next step.  If it should be another entity value,
            correct the entity value and rerun the TDMPGEN.

            Reference:  Section 7.2.1 of this guide.

TDM00009E   The entity value is not valid.  Correct the
            entity value and rerun the TDMPGEN.

            Reference:  Section 7.2.1 of this guide.

TDM00010E   A SYSID value can only be specified once.  Remove
            the duplicate SYSID value in the SYSID parameter
            and rerun the BASPGEN and TDMPGEN.

            Reference:  Section 7.2.5 of this guide.

TDM00011E   The maximum SYSID limit has been reached.
            Contact CA MICS Product Support.

TDM00012E   Process Group values must be numeric.  Enter a
            numeric value in prefix.MICS.PARMS(PRCGP) and
            rerun the TDMPGEN.

            Reference:  Section 7.2.3 of this guide.

TDM00013E   Process group name is either not defined or
```

invalid. Review your prefix.MICS.PARMS(PRCGP) and enter a correct value.

TDM00014W Process group name exceeds the maximum of 17 characters. Its value is truncated.

TDM00016E The keyword is not valid. Valid keywords are OPTIONS and ENTITIES. Review your prefix.MICS.PARMS(TDMOPS) and correct the keywords.

Reference: Section 7.2.1 of this guide.

TDM00017I Informational message to indicate the end of a parameter member.

TDM00019E Errors were found in the parameter definitions. Review previous messages to determine the error. The TDMPGEN was not successful. Correct the errors and rerun TDMPGEN.

TDM00020E Duplicate ddnames were found in the INPUTTDM parameter. Remove the duplicate ddname and rerun the TDMPGEN.

Reference: Section 7.2.2 of this guide.

TDM00022E If NOLOADID is specified on the OPTIONS statement in TDMOPS, there must be an ENTITIES statement with at least one valid entity type specified. Add a valid ENTITIES statement in prefix.MICS.PARMS(TDMOPS) and rerun TDMPGEN.

Reference: Section 7.2.1 of this guide.

TDM00023E The parameter member named in the message is empty. Enter valid options and rerun the TDMPGEN.

Reference: Section 7.2.1 of this guide.

TDM00024E Values in TDMOPS are compared to INPUTTDM for expected ddnames. An expected ddname was not found in INPUTTDM. Code the required ddname and rerun the TDMPGEN.

Reference: Sections 7.2.1 and 7.2.2 of this guide.

```
TDM00025W   Ddnames were found in INPUTTDM that were not
            referenced by values in TDMOPS.  Input data will
            not be processed from this ddname.

            Reference:  Sections 7.2.1 and 7.2.2 of this
                        guide.

TDM00026E   Ddname found in INPUTTDM is longer than 8
            characters.  TDMPGEN fails with a code 998.
            Correct the ddname in error and rerun TDMPGEN.

TDM00051I   This is an informational message indicating the
            number of records read from each input ddname.

TDM00055I   This is an informational message indicating the
            start of processing for an input ddname.

TDM00056W   The ORGSYSID cannot be determined because the
            system name in the MEASURE data does not match
            any of the system names in the TDMOPS OPTIONS
            statement.  By default, ORGSYSID will be assigned
            the value of the first four characters in the
            Tandem system name.  If this default ORGSYSID is
            not defined in SYSID, the job will abend with a
            U310 code.

            Reference:  Section 7.2.1 of this guide.

TDM00057W   The DURATION field in the input record was found
            to contain negative value.  This may indicate a
            problem when the MEASURE structured file was
            created.  DURATION is set to missing value.
            Contact CA MICS Product Support with the
            information provided in the listing.

TDM00058W   The device type is not recognized.  TDMDVTP will
            be set to the value of "(UNK)" and processing
            will continue.  Advise CA MICS Product Support if
            new support is desired.

TDM00059W   The device subtype is not recognized.  TDMDVSTP
            will be set to the value of "(UNK)" and
            processing will continue.  Advise CA MICS Product
            Support if new support is desired.

TDM00066W   The entity type is not supported and will not be
            processed.  Processing will continue for valid
            entity types.  If an incorrect LOADID was
            specified when the file was written, you may want
```

to process this file as a NOLOADID file. Contact CA MICS Product Support for assistance.

TDM00067W Only the first ten "entity type not supported" messages will be printed. The "option deleted" count will indicate how many records were not processed. Contact CA MICS Product Support for information on support for this version of MEASURE data.

TDM00070W This warning indicates the data is from an unsupported version of MEASURE. The job will continue processing but new fields will be ignored. The supported version and the version found in the data are printed in the message. Call CA MICS Product Support for more information on supported versions of MEASURE.

Reference: Section 6 of this guide.

TDM00072W MEASURE entity records that are shorter than expected will be ignored. The short records will not be processed but the job will continue to process valid records. You should investigate the source of these records.

TDM00073W Only the first ten "short record" messages will be printed. The "short record" count will include the total number of short records.

TDM00076I Informational message indicating the number of records dropped by a user exit.

TDM00098I Informational message indicating the successful completion of a process.

TDM00099I Informational message to print the line read from the parameter member.

Appendix B: DATA DICTIONARY

The Data Dictionary is only available at your site, where it has been customized to your configuration and your product change level.

To see the Data Dictionary at your site, follow the instructions under Document Browse in the Document Access Guide.