Transaction Processing

Bull DPS 7000

TDS Administrator’s Guide

47 A2 32UT Rev07

Transaction Processing

Bull DPS 7000

TDS Administrator's Guide

Subject: This manual is intended for the site administrator, systems engineers, or systems analyst responsible for setting up and managing a TDS application.

Special Instructions: Revision 07 replaces Revision 06.

Software Supported: GCOS 7-V9 (TS 9866).

Software/Hardware required:

Date: June 2001

Bull S.A.
CEDOC
Atelier de reprographie
357, Avenue Patton BP 20845
49008 ANGERS Cedex 01
FRANCE

Bull HN Information Systems Inc.
Publication Order Entry
FAX: (800) 611-6030
MA30/415
300 Concord Rd.
Billerica, MA 01821
U.S.A.

47 A2 32UT Rev07

Your suggestions and criticisms concerning the form, contents and presentation of this manual are invited.
A form is provided at the end of this manual for this purpose.

47 A2 32UT Rev07

Preface

Scope and Objectives

This manual describes how to prepare, generate, implement, protect and optimize a TDS application.

GCOS 7 Release V7 is ***fully upwards compatible*** with Release V6. The term **V6/V7** groups information common to both Releases. The present document treats differences in the TDS versions of V5 and V6/V7. Where applicable, the term *V3/V5* is used for information common to both V3 and V5, as opposed to information specific only to V5.

Change bars indicate the modifications since the last revision.

Intended Readers

This guide is intended for the administrator, systems engineer and systems analyst responsible for installing and managing a GCOS 7 transactional application. Where applicable, references are provided to GCOS 7 specific information, Network Generation on the DPS 7000, file and database management, COBOL Programming, GCL, DOF 7-PO (Programmed Operator Facility) and IOF. The list of documents below gives their reference numbers and additional topics of interest.

Structure

This guide is a member of the set of TDS manuals:

TDS Concepts *47 A2 26UT*
TDS Administrator's Guide *47 A2 32UT*
TDS COBOL Programmer's Guide *47 A2 33UT*
TDS C Language Programmer's Guide *47 A2 07UT*
TDS Quick Reference Handbook *47 A2 04UT*
High Availability Administrator's Guide *47 A2 23UT*
TDS-IMAGEWorks Link User's Guide *47 A2 25UT*
TDS-DPX/20 IMAGEWorks Link User's Guide *47 A2 31UT*

The summary of the contents of this manual is as follows:

Section 1	summarizes administration of TDS.
Section 2	explains how to prepare the environment of a transactional application.
Section 3	describes the TDS generation program and the options for each section.
Section 4	examines the elements of a TDS application.
Section 5	describes how to implement the application.
Section 6	shows how to to protect the TDS application and its environment.
Section 7	recommends how to design and optimize the TDS application by efficiently using system resources.
Section 8	shows how to modify an existing application.
Section 9	explains the master commands used in operating a TDS session.
Section 10	covers running the application with the Programmed Operator Facility.
Section 11	examines dump handling and TDS errors.
Appendix A	gives a list of codes returned by TDS when a file is opened.
Appendix B	lists MU and MV messages indicating TDS errors.
Appendix C	treats the structure of J-type records in the User Journal.
Appendix D	explains how to read the TDS Statistics Report.
Appendix E	gives the programmatic interface of TDS accounting records.
Appendix F	lists error messages sent to the Master Operator.
Appendix G	gives a list of codes returned on MODIFY_TDS issued to an SM Library.
Appendix H	treats a sample TDS Generation Source Program.
Appendix I	lists the messages returned when TDS Master Commands are issued.

Appendix J	lists the messages returned when XCP2 Master Commands are issued.
Appendix K	gives the main TDS disconnection codes at the close of the TDS session.
Appendix L	lists system-reserved keywords for the TDS generation.
Appendix M	lists the errors and responses sent during TP7GEN execution.
Appendix N	gives the main TDS connection rejected codes on initiating the TDS session.
Appendix O	gives the hexadecimal codes sent by the TDS for rejected connections.
Appendix P	gives the messages produced by Default H_XAEVT Transaction.

Bibliography

The following publications give information on topics related to running a TDS application.

For generating the DPS 7000 network:
Network Overview and Concepts *47 A2 92UC*
Network Generation *47 A2 93UC*

For generating the DSA network:
DNS-V4 System Generation *39 A2 22DN*
DNS-V4 NGL Reference Manual *39 A2 23DN*
CNS 7 A1 NOI Operator Guide *39 A2 41DM*
CNS 7 A2 NOI Operator Guide *39 A2 34DN*

For cataloging users, correspondents, and TDS authority codes using the MAINTAIN_CATALOG utility:
GCOS 7 System Administrator's Manual *47 A2 41US*
Catalog Management User's Guide *47 A2 35UF*

For creating and managing forms using the MAINTAIN_FORM utility:
FORMS User's Guide *47 A2 15UJ*

For producing reports:
Generalized Terminal Writer User's Guide *47 A2 55UU*

For COBOL syntax and use:
COBOL 85 Reference Manual *47 A2 05UL*
COBOL 85 User's Guide *47 A2 06UL*

For defining XCP2 correspondents:
CPI-C/XCP2 User's Guide *47 A2 14UT*

For defining XCP1 correspondents:
Transactional Intercommunication using the XCP1 Protocol User's Guide *47 A2 11UT*

For BOS/TP applications and TDS operations in UNIX-BOS/TP environment:
Setting up and Using /HOST7 User's Manual *47 A2 30UT*

For file access and data management:
Full IDS/II Reference Manual 1 ..... *47 A2 05UD*
Full IDS/II Reference Manual 2 ..... *47 A2 06UD*
Full IDS/II User's Guide *47 A2 07UD*
IDS/II Reference Manual *47 A2 11UD*
IDS/II Administrator's Manual *47 A2 13UD*
Database Reorganization (DBREORG) User's Guide *47 A2 15UD*
UFAS-EXTENDED User's Guide *47 A2 04UF*
Data Security Facilities User's Guide *47 A2 09UF*
SECUR'ACCESS Administrator's Manual *47 A3 01BD*

For main console operator commands:
Network User's Guide *47 A2 94UC*
GCOS 7-V7 System Operator's Guide *47 A2 47US*
GCOS 7-V8/V9 System Operator's Guide *47 A2 53US*

For information on installing and optimizing the system:
System Behavior Reporter User's Guide *47 A2 03US*
TILS User's Guide *47 A2 04US*
GCOS 7/V8 System Administrator's Manual *47 A2 54US*
System Installation Configuration and Updating Guide (V8) *47 A2 19US*
System Installation Configuration and Updating Guide (V9) *47 A2 23US*

For concurrency control:
General Access Control (GAC-EXTENDED) User's Guide *47 A2 12UF*

For the MNJAS utility, file recovery procedures and journal usage:
File Recovery Facilities User's Guide *47 A2 37UF*

For GCL commands:
IOF Terminal User's Reference Manual
Part 1 ..... *47 A2 38UJ*
Part 2 ..... *47 A2 39UJ*
Part 3 ..... *47 A2 40UJ*

For Programming in GCL:
GCL Programmer's Manual *47 A2 36UJ*

For JCL statements:
JCL Reference Manual *47 A2 11UJ*
JCL User's Guide *47 A2 12UJ*

For status values and return codes:
Messages and Return Codes Directory *47 A2 10UJ*

For using IQS under TDS:
IQS-V4/TDS User's Guide *47 A2 81UR*

For using ORACLE under TDS:
ORACLE7/TDS User's Guide *47 A2 14UR*

For TDS-HA applications:
High Availability Concepts Guide *47 A2 22UT*
High Availability Administrator's Guide *47 A2 23UT*

For handling documents through IMAGEWorks:
TDS-IMAGEWorks User's Guide *47 A2 25UT*
TDS-DPX/20 IMAGEWorks Link User's Guide *47 A2 31UT*

For using SQL 7 facilities:
SQL 7 Administrator's Guide *47 A2 51UR*
SQL 7 SQL Supplement *47 A2 52UR*

For sorting work files used with SQL 7:
Sort/Merge User's Guide *47 A2 08UF*

For gateway operations in accessing the SQL 7 Database:
DDA SQL 7 Gateway User's and Administrator's Guide *86 A2 62FC*

For using CMA services:
SQL CMA User's Guide *47 A2 22UR*

For migrating between releases:
GCOS 7 Evolution Guide *47 A2 20UG*

For using the Programmed Operator facility:
DOF 7-PO User's Guide *47 A2 80UC*

For using the VOLSET facility:
Administering the Storage Manager *47 A2 36UF*

For using TDS-TCP/IP services:
TDS-TCP/IP User's Guide *47 A2 37UT*

Syntax Notation

Conventions for entering TP7PREP Syntax

UPPERCASE	indicates a keyword introducing a parameter whose value must be supplied by the user.
UPPERCASE	indicates an absolute value that the user may specify. If underlined, it is the default value.
item	indicates a value applicable to the current keyword.
item	italics indicates a term for which the user supplies a value.
{item|item|item}	a list of character strings within braces means that only one is to be selected if the introducing parameter is specified. The item heading the list and underlined designates the default parameter.

Conventions for entering TP7GEN Syntax

UPPERCASE	indicates that this underlined item is a reserved keyword which must be specified if the clause in which it appears is required.
UPPERCASE	indicates a reserved keyword that must be coded as shown. It may be omitted.
item	italics indicates a term for which the user supplies a value.
[item]	optional entry.
{**item**|item|item}	a list of items within braces means that only one is to be selected if the introducing parameter is specified. The item heading the list and appearing in bold is the default value if the introducing parameter is not specified. This list of parameters can also be vertical: {item} {item} {item}

`item . . .`	Three dots following an item indicate that more items having the same form may appear.

`Clause`
`  .`
`  .`
`  .`
	Three vertical dots mean that part of a clause has been intentionally omitted.

`||A||`
`||B||`
`||C||`
	Clauses enclosed in double vertical bars indicate only one occurrence of each.

Conventions for Entering the GCL Syntax

`ITEM`	An item in capitals is a keyword to be specified as shown. Capitals is merely a convention; in practice the item can be specified in either upper or lowercase.

`item`	An item in lower case indicates that a user-supplied value. In most cases it gives the type and maximum length of the value:
	`char46` a string of up to 46 alphanumeric characters
	`dec5` a decimal integer value of up to 5 digits
	`name12` a name of up to 12 characters
	`lib78` a library name of up to 78 characters
	`file78` a file name of up to 78 characters
	`star31` a star name of up to 31 characters.
	
	In some cases, it gives the format of the value:
	`a` a single alphabetic character
	`nnn` a 3-digit number
	`hh-mm` a time in hours and minutes.
	
	In other cases, it is simply descriptive of the value:
	`device-class`
	`condition`
	`any-characters`
	`tp7-name`

`ITEM`	An underlined item is a default value. It is the value assumed if none is specified.

`bool`	A boolean value is either 1 or 0. A boolean parameter can be specified by its keyword alone, optionally prefixed by "N". When specifying the keyword alone, the value is always set to 1. Prefixing the keyword with "N" always sets it to 0.

`{ }`	Braces indicate a choice of values. Only one can be selected. The list can be presented horizontally, with each item separated by a vertical bar:
	`{ item | item | item }`
	or vertically:
	`{item}`
	`{item}`
	`{item}`

`[ ]`	Square brackets indicate that the enclosed item is optional. An item not enclosed in square brackets is mandatory.
`( )`	Parentheses indicate that a single value or a list of values can be specified. A list of values must be enclosed by parentheses, with each value separated by a comma or a space.
`...`	Ellipses indicate that the item concerned can be specified more than once.
`+ = $ * / - .`	These are special non-alphabetic characters to be specified as shown.

If a dashed line appears before a parameter (for example, the TRACE and RESTART parameters in the MODIFY_TDS command), these parameters are to be used at the request of the Service Center only.

EXAMPLE 1:

```
[        { WARM  } ]
[ MODE = {       } ]
[        { COLD  } ]
```

The choice is as follows:

1. Nothing at all (MODE=WARM applies).
2. MODE=WARM (the same as nothing at all).
3. MODE=COLD (COLD applies).

❑

EXAMPLE 2:

```
AUTHORITY_CODE = (dec2 [-dec2] [,dec2]...)
```

Indicates that Authority Codes must be specified. Valid entries are a single value, or a list of values enclosed in parentheses. The list can consist of single values separated by a comma, a range of values separated by a hyphen, or a combination of both. For example:

```
AUTHORITY_CODE=(2-4,28,29,31)
```

❑

EXAMPLE 3:

```
[ IMMED ={ bool | 0 } ]
```

This is a boolean parameter whose default value is zero. The choice is as follows:

1. Nothing at all (IMMED=0 applies)
2. IMMED=0 or simply NIMMED
3. IMMED=1 or simply IMMED

❑

Conventions for Entering the Master Command Syntax

Use this syntax:

- either if logged on under a master mailbox
- or if the command is:
  - stored in a subfile for execution by the M EXEC_TDS command
  - to be executed via the Batch Interface
  - to be executed via the spawning mechanism.

Command names and parameters follow the same naming convention. Separate:

- the command name and the first parameter by at least one blank
- parameters from each other by blanks or commas.

Each master command is followed by various positional parameters and keywords. A parameter can be specified as:

- as a keyword introducing an argument
- as a positional parameter.

Keywords can be specified in any order. Both keywords and positional parameters can be mixed in which case, positional parameters are supposed to denote the values for parameters whose rank is determined by the preceding keyword, if any.

Parameters must be constants. Use the following data types:

- character
- decimal
- boolean
- name
- star-name
- file.

EXAMPLE:

```
M CLOSE_TDS_FILE IFN=T1,DEASSIGN=1
```

Because full GCL facilities are not available, do not:

- use builtin or GCL variables
- mix quoted and unquoted strings
- nest parentheses.

❑

Table of Contents

3. Generating a TDS Application with TP7GEN

4. Elements of TDS

7. Optimizing a TDS Application

8. Modifying a TDS Application

9. Master Commands

Table of Graphics

Figures

1. Introduction to TDS Administration

1.1 Role of the Administrator

The term *Administrator* refers to the Site or System Administrator as opposed to the *TDS Administrator*.

The Administrator has the responsibility of ensuring that the environment in which applications at the site execute is the optimum for their performance.
This involves:

- installing applications and integrating modifications
- defining and allocating the necessary resources
- determining whether applications meet user requirements
- maintaining procedures for the security and integrity of data
- advising personnel on the use of applications
- and managing day-to-day operations.

The Administrator often has the additional responsibility of:

- generating the network through the NETGEN facility
- and defining the necessary Projects, Users, and Authorities in the Catalog.

TDS is delivered as a general application package and is customized for each individual site. Everything specific to the TDS application is the responsibility of the *TDS Administrator* who:

- builds the necessary files using the TP7PREP utility
- writes the source TDS (STDS)
- creates and generates forms using the MAINTAIN_FORM utility
- prepares the runtime JCL/GCL procedures
- collects the TPRs and prepare the Sharable Module libraries
- generates the TDS application using the TP7GEN utility
- modifies and adjusts the application files according to the needs
- collects and prepares any additional runtime modules
- regenerates the application following any adjustments and modifications
- and tests and optimizes the TDS application.

1.2 The Elements of a TDS Application

The elements of a TDS application fit together as shown in Figure 1-1:

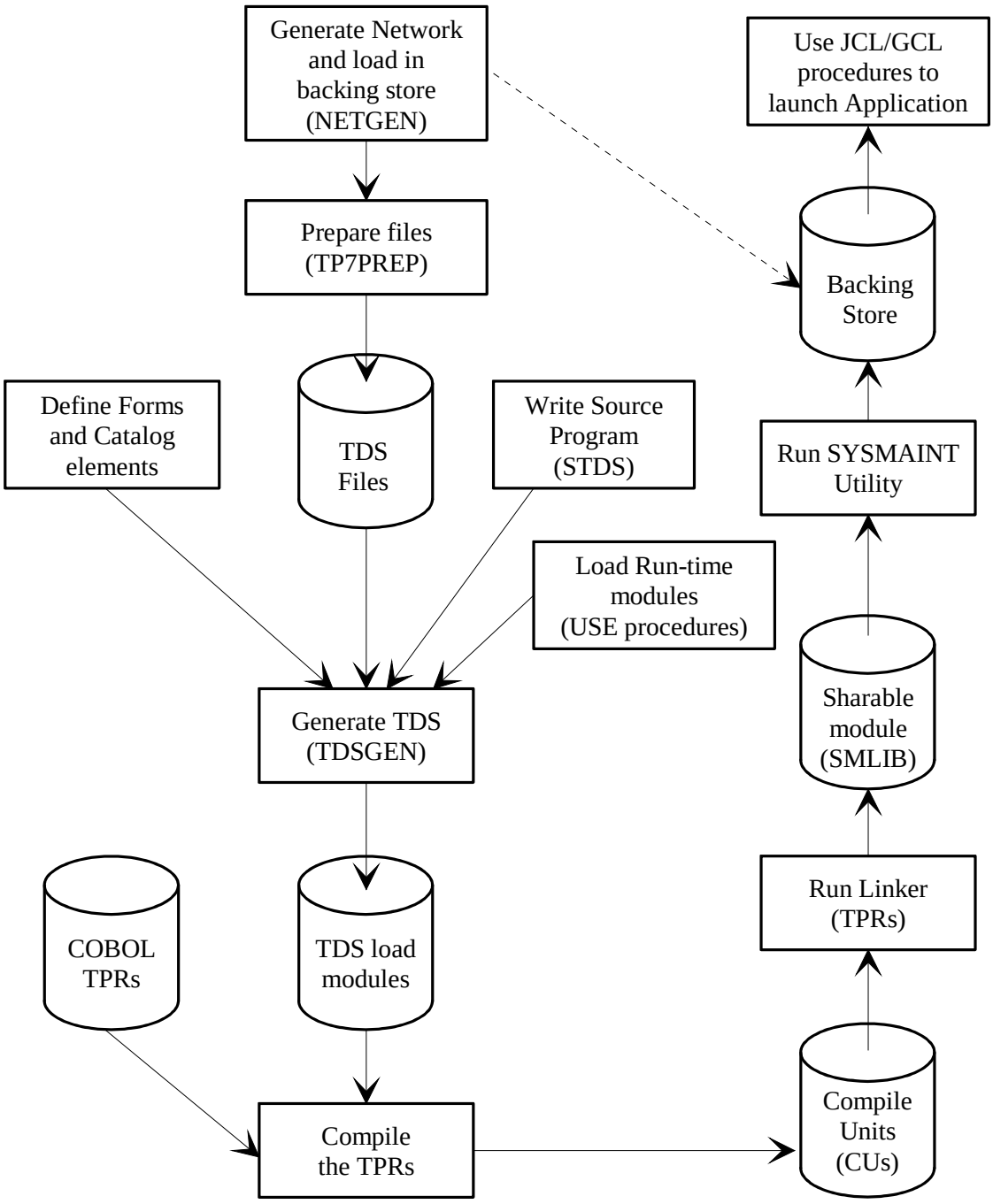


Figure 1-1. Elements of a TDS Application

1.3 Compatibility Between TDS-V5 and TDS-V6/V7

TDS-V6 is compatible with previous releases provided that certain names are changed and functions are modified. See Paragraph 1.3.2 *TDS-V6 Modifications*. In addition, several functions have been added. See Paragraph 1.3.1 *TDS-V6 Enhancements*.

TDS-V7 is compatible with TDS-V6 for all components of TDS excepting TP7PREP for which obsolete parameters have been removed.

Users migrating directly from TDS-V5 to TDS-V7, must take into account enhancements and modifications concerning TDS-V6 as described here, before implementing TDS-V7.

1.3.1 TDS-V6 Enhancements

TDS-V6 has these new features:

- a new set of master commands
  See Section 9 *Master Commands*.
- a new tool, TDS Real Time Statistics, for dynamically supervising and debugging a TDS application
  See Paragraph 5.4.3 *Using TDS Real-Time Statistics*.
- an interface that provides terminal handling, the Terminal Adapter
  See Section 4 *Elements of TDS*.
- an enhanced swap file buffer
  See Paragraph 7.6.2.1 *Computing the Size of the Swap File Buffer*.
- three new spawn statements, DELSPAWN, NBSPAWN and SPAWNTX.
  See the *TDS COBOL Programmer's Guide*.
- an option that provides additional information and allows editing aborted TPRs.
  See Paragraph 3.4.55 *The USE EDIT-EXCEPTION Clause*.
- from TS6152, a new version of the TP7GEN utility that provides additional information in error reporting.
- from TS6152, the new /HOST7 function enables co-operation between BOS-TP and TDS in Client-Server mode. Refer to the *Setting Up and Using /HOST7 User's Manual*.

1.3.2 TDS-V6 Modifications

The modifications to TDS preparation and generation in V6 are:

- the names of the utilities and the functions they provide
- how TPRs are linked and loaded.

1.3.2.1 Name Changes in the STDS

An STDS written for a TDS application in a previous release can still be used to generate TDS. However, the following V5 names must be changed:

V5	V6
MTREP	TP7PREP
MTGEN	TP7GEN
MTLINKxxx	TP7LINKxxx

where: *xxx* is TPR, TPRn or TDS

Clauses obsolete in V6 are ignored and a message informing the user of the action to take, where appropriate, is displayed.

1.3.2.2 TP7PREP

To migrate from V5 to V6, perform the TDS preparation with the parameter DEAL=Y (default is N). The TP7PREP utility functions in the same way as MTPREP in previous releases. There is no need to make any changes in the procedures for its use.

To improve performance when using FBO disks, it is **strongly recommended** to allocate a swap file **block size** of **4096 bytes** when preparing the TDS application.

1.3.2.3 TP7GEN

TDS generation for V6 involves the following changes:

- the MAINTAIN_TDSGEN utility is no longer used
- the XCP1 mailbox must be declared in NETGEN and suppressed in the STDS
- STDS which contains the TDS generation
  - is improved with new clauses
  - and must be edited to remove obsolete clauses:
    NUMBER OF BATCH ENTRIES
    RESERVE MEMORY
    all clauses in the Correspondent Section.

Run the TP7GEN utility after the TP7PREP utility.

The TP7GEN utility provided with TS6152 has the following improvements:

- Errors with severity 3 and greater must be corrected as with previous versions. No TDS LM is produced.
- Errors with severity 2 should be corrected. They are no longer considered warnings even if the TDS LM is linked and produced.
- Message labels and error reports of the generation have been modified. See Appendix N *TP7GEN Errors and Responses*.

1.3.2.4 Linking TPRs

To invoke the linker with command files, use TP7LINKTPR [n], not MTLINKTPR [n]. Any LKUs produced by MTLINKTPR in previous versions can be executed by TDS-V6. However, the converse is not true. LKUs produced by TP7LINKTPR cannot be executed by TDS-V5. Save old LKUs and relink the CU (compile unit) under MTLINKTPR [n].

1.3.2.5 Loading TPRs

The VMAINT utility used to load TPRs in previous versions is replaced by the SYSMAINT utility.

1.3.3 TDS-V6 Compatibility at Program Level

All user sharable modules linked in release V3 or V5 can be executed in TDS-V6.

All compile units compiled in release V3 or V5, including those of the "USE procedure", can be relinked for execution in TDS-V6.

1.3.4 TDS-V6 Modifications at the Program Level

- The size of the swap file buffer is extended up to 128 Kbytes.
- The size of the TRANSACTION-STORAGE can be extended up to 64,000 bytes.
- Three new SPAWN statements are introduced.
- Separate commitment units can be taken for PRIVATE-STORAGE and a part (or the entire) TRANSACTION-STORAGE.
- Options are provided for changing the service-message headers and trailers. TDS service messages no longer need redefining to be sent to the master terminal operator.
- Options are provided for protecting transactions against breaks.
- A new primitive used to call the new output format, TDS Real-time Statistics. This primitive allows the operator to recover statistics of an application that were previously only available by using master commands.

See the *TDS COBOL Programmer's Guide* for information on TRANSACTION-STORAGE, the three new spawn statements, and the new GET-TDS-STAT primitive.

1.3.5 Effect of NETGEN since TDS-V6

If a TDS workstation is declared at the network generation, the following are ignored:

- NUMBER OF TERMINALS Clause
- MASTER MAILBOX Clause
- XCP1 MAILBOX Clause.

If one of these clauses is encountered, a warning is issued in the generation report.

1.3.6 Changes to JCL to Execute TDS-V6

Since the JCL to execute the TDS job is user-dependent, it is not delivered with the TDS package. However, an example is provided in Paragraph 5.3.1 *JCL for Running TDS Job*.

To use the User Journal function, modify the JCL as follows:

- verify that the USER-JOURNAL Clause is specified in the generation
- add the ASSIGN USERJRNL, DUMMY; Statement to the JCL
- remove the DEFINE H_SWAP, JOURNAL=AFTER; Statement from the JCL.

If user files are not protected by the After Journal:

add the DEFINE USERJRNL, JOURNAL=AFTER Statement to the JCL.

1.3.7 TDS-V6 Changes to Master Commands and Operator Activity

The improved list of master commands gives the master operator increased control over TDS applications. The master commands are now converted to GCL and are consistent with GCL naming conventions. Full GCL facilities are available only when the master operator logs onto a TDS application under IOF.

Old master commands can coexist with new V6 commands. The following example shows the difference between the previous and current ways of introducing commands.

Previous Versions	TDS-V6
`ESCAPE tdsname xxxxxxx`	`SNDT 'xxxxxxx' TDS=tds name`

Minor changes have been made to the warm restart of master commands. In general, the master commands remain in effect until the next cold restart. Files closed during a TDS session remain closed after the next warm restart and remain so until the TDS Administrator re-opens them. See Section 9 *Master Commands*.

A TDS-V6 application logged via IOF uses:

- the new GCL syntax with menus for all master commands
- a new set of master commands is partially compatible with earlier versions.

A TDS-V6 application logged via the Master Mailbox uses:

- a new syntax fully compatible with the old syntax
- new parameters and commands.

Several master commands have changed, been replaced, or become obsolete. These three master commands are obsolete in TDS-V6:

- M MEMORY
- M DMM
- M STAT DMM.

When entering one of these commands, no action is taken and the message `COMMAND NO LONGER SUPPORTED` is displayed.

Section 9 gives the correspondence between V5 and V6 commands with an explanation of how to find V6 equivalents for V5 commands.

Master Commands and Warm Restart

Certain master operations are modified, in particular the warm restart function. See Paragraph 9.3.2 *Warm Restart*.

1.3.8 TDS-V6 Output Formats

New output formats for the master commands provide more detailed error reporting. Syntax errors explained in Appendix F are separate from operation messages listed in Appendix I.

Outputs can be stored in the TDS file, *tdsname*.DEBUG.

Summary reports and statistical information can now be requested. TDS Real Time Statistics allows requesting information from any terminal that was previously accessible only by master commands. See Paragraph 5.4 *Output of a TDS Session*.

1.3.9 TDS-V6 Introduction of a Terminal Handling Interface

The Terminal Adapter allows:
- sending mail to the last line (line 25) of the screen
- using foreign alphabets
- making user programs transparent to the terminal type.

The interface is explained in the Paragraph 4.6 *Terminal Adapter* and instructions on its use are in Paragraph 3.4.40 *USE TERMINAL-ADAPTER Clause*.

1.3.10 TDS-V6 Improved Accounting and Security

A TDS-V6 system is protected by:

- NETGEN
- and the Security Access Interface

Depending on how GCOS 7 is configured, security can be automatically provided for a TDS application. For more information, refer to the *SECUR'ACCESS Administrator's Manual*. For NETGEN, see *GCOS 7 Network Generation Manual*.

1.3.11 TDS-V6 Changes to Clauses in TDS Generation

The following Clauses concern the TDS Section.

- New Clauses:
  - NUMBER OF IQS JPPC SEGMENTS replaces V5 USE IQSJPPC*nbseg*.
  - USE STARTUP...REPORT has been added to select the level of messages for display when the TDS application is started.
  - USE-LARGE-STACK-SEGMENT replaces the TP7GEN JCL keyword LSTK3.
- Obsolete Clauses:
  - MAXIMUM NUMBER OF LOCKED PAGES
  - XCP1 MAILBOX
  - NUMBER OF XCP2 CONVERSATIONS
  - USE DPF-DOC-MODE.

1.4 Compatibility Between TDS-V6 and TDS-V7

TDS-V6 and TDS-V7 are upwards compatible.

2. Preparing TDS Files with TP7PREP

Preparing the files to create a TDS application entails:

- determining the size and configuration of the TDS files
- preparing and allocating files using the TP7PREP utility
- using a JCL to launch the TP7PREP utility.

2.1 Types of TDS Files

The TDS application requires offline and online files.

Offline files are used to generate the actual TDS application. These files are also used to compile and link TPRs. Offline files are used to generate the TDS and are not required at runtime.

Online files process transactions during a TDS session. These files contain all the information necessary for a TDS application to function: storage handling, spawns, restart and recovery. Online files are required by the TDS Executive at run time.

These files are defined through the TP7PREP utility. Figure 2-1 shows the files to be prepared for generating a TDS application. Figure 3-1 shows the files created by TP7PREP used by the TP7GEN utility.

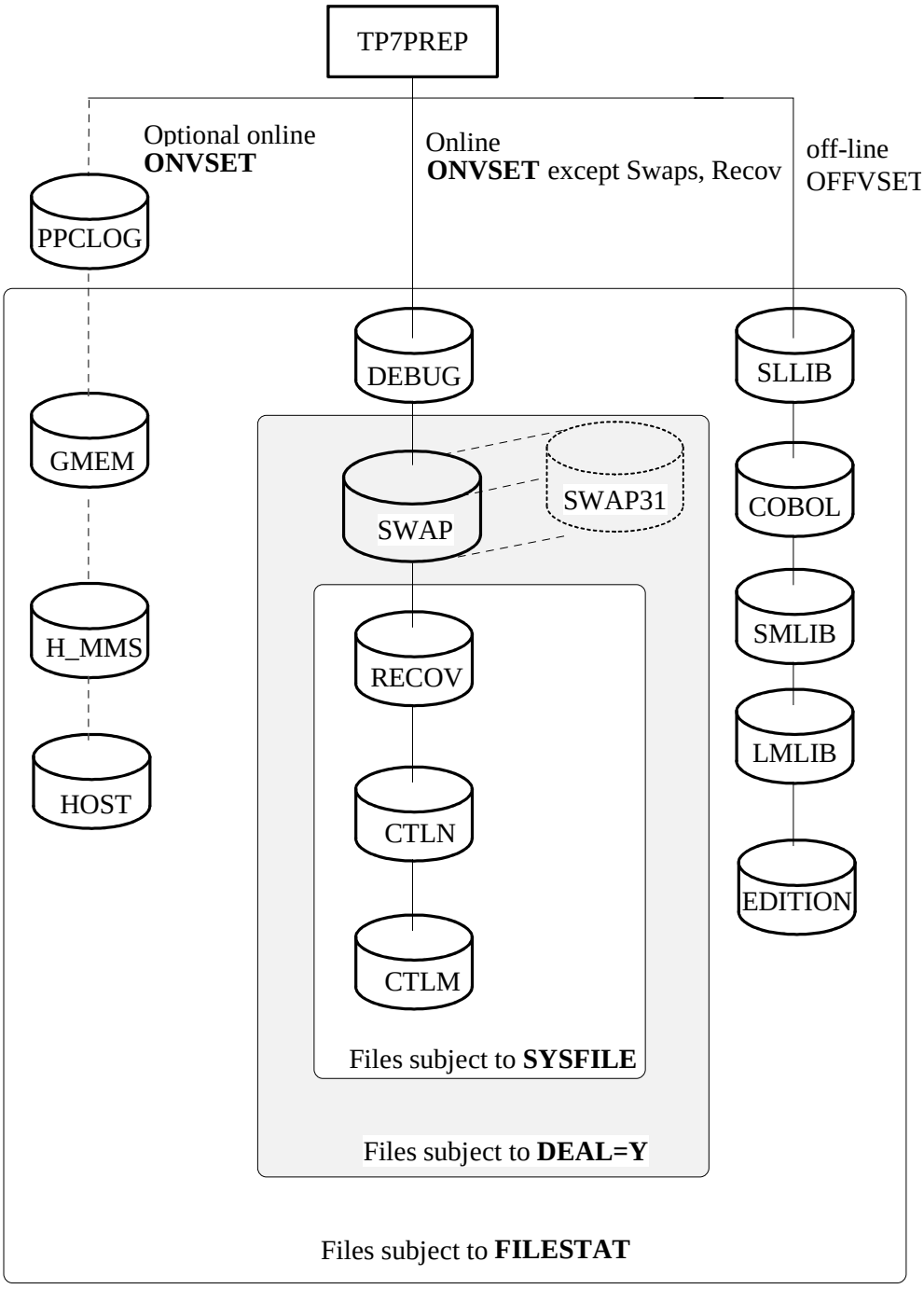


Figure 2-1. Files Created with the TP7PREP Utility

2.2 Analyzing File Requirements

Before preparing the files, determine the requirements of the TDS application:

How many disks are used, how are they configured and how much space is available?	To determine the number of disks that TDS accesses and to evaluate the possibility of dividing up I/Os among them.
How many users are there on the TDS application?	To determine the number and size of swap files.
Will the spawn function be used intensively?	To determine the size of the *tdsname*.CTLM file.
Will High Availability be used?	See the High Availability Concepts and the High Availability Administrator's Guide for further information.
Will online and offline files be cataloged or not?	Cataloging is recommended in the SYSFILE parameter for online files and in the FILESTAT parameter for offline files.
Where will the online and offline files be cataloged?	To determine whether the files are to be put in a private catalog or in the Site Catalog, and then to define declare it in the CATNAME parameter.
What programming language will used?	To determine if the TPRs are written in C or in COBOL.
Which After Journal will be used?	This depends on whether High Availability is used or not.
Will XCP2, IQS, JPPC facilities, or multimedia support be used?	To determine if the optional *tdsname*.PPCLOG, *tdsname*.GMEM, or *tdsname*.MMS files should be declared. See the *CPI-C/XCP2 User's Guide*, the *IQS-V4/TDS User's Guide*.

2.2.1 Offline Files

The *tdsname* corresponds to the name of the TDS application. The offline files are:

tdsname.SLLIB	Name of the Source Language Library containing the source generation program STDS and, after executing TP7GEN, the command files TP7LINKTDS, TP7LINKTPR[*nn*] and TP7GEN *tdsname* invoked to link TPRs.
tdsname.COBOL	Source Language Library containing file descriptions and storage areas defined at TP7GEN, retrieved by TPRs through COPY statements.
tdsname.SMLIB	Sharable Module Library containing the sharable modules in which the Static Linker stores linked units of TPRs.
tdsname.LMLIB	Library containing TP7GEN load modules and user-defined procedures declared by the USE clause in the STDS. The names of these files are determined at TP7PREP: – *tdsname*.LMLIB, if FILESTAT=CAT – TDS.LMLIB, the default name if FILESTAT=UNCAT – *prefix*.LMLIB, if FILESTAT=UNCAT and a prefix is given.
tdsname.EDITION	Workfile used by TP7GEN to contain the report in the subfile GEN*tdsname* produced by TP7GEN. This file can be reprinted by $WRITER (JCL) or DPRINT (GCL).

File Name	Organization	Default values*		Initial N° of Blocks (Directory)*
		Initial File Size in Cylinders	Maximum File Size in Cylinders	
<tdsname>.SLLIB	Library (SL Format)	1	10	5
<tdsname>.COBOL	Library (SL Format)	3	10	5
<tdsname>.SMLIB	Library (SM Format)	3	10	5
<tdsname>.LMLIB*	Library (LM Format)	3	10	5
<tdsname>.EDITION	Library (SL Format)	1	3	5

Figure 2-2. Characteristics of Offline Files

NOTE:

***** If *tdsname*.LMLIB is not cataloged (FILESTAT=UNCAT), its name:

- is either *prefix*.LMLIB if a prefix has been specified

- or the default TDS.LMLIB.

$ Initial default values are allocated by TP7PREP. If the file space is insufficient, TP7GEN aborts. For information on how to specify file size values, see Paragraph 2.4.1.

2.2.2 Online Files

Online files are either resident or cataloged in an auto-attachable catalog. The catalog can be either the Site Catalog or a private catalog. These files are dynamically assigned by the TDS applications and allow:

- the source generation program to be transported to different sites without requiring modification of the configuration parameters
- files to be cataloged with the necessary security and assignment options.

The *tdsname* corresponds to the name of the TDS application. Online files are:

tdsname.DEBUG — A file that contains the result of trace options as specified in the TRACE command.

tdsname.SWAP[nn] — Internal TDS journals containing restart information for each user context; where nn either has no value or ranges from 1 through 31 (maximum).

To estimate of swap file space, multiply the number of users by 128 Kbytes; then add 10% to allow for breaks and interrupted transactions. The number of users is the sum total of the values specified in the NETGEN parameters: TMSESS, XCP1SESS and XCP2SESS.

The size of the user context depends on the application. To optimize the swap filesize, TDS can be forced to send MU67 or MU89 error messages (depending if the system has VBO or FBO disks) by selecting a size that is too small. These messages indicate the number of missing cylinders or blocks.

The number of swap files also depends on the activity of the application. Swap files should be allocated on different volumes to maximize simultaneous I/O access. Up to 32 swap files can be allocated on several volumes of different devices. See the NBSW parameter in Paragraph 2.4.2 and Section 7 *Optimizing a TDS Application*.

Swap files of a TDS application are either FBO or VBO files. A TDS application cannot simultaneously support both FBO and VBO swap files.

However, on a single GCOS 7 system, it is possible to simultaneously run a TDS application with FBO swap files, and another TDS application with VBO swap files.

tdsname.CTLN

A file that contains control information and user defined tables which do not require monitoring by GAC-EXTENDED or journalization. This file is dynamically assigned by TDS with:
- SHARE=NORMAL
- ACCESS=SPWRITE (exclusive assignment in update).

tdsname.CTLM

A file that contains control information requiring monitoring by GAC-EXTENDED and journalization. This file:
- is TDS-controlled and uses the concurrency mechanism
- is dynamically assigned to the TDS job with SHARE=NORMAL and ACCESS=SPWRITE (exclusive assignment in update mode)
- is protected by the Before Journal.

The JCL for starting the TDS job must contain at least one DEFINE statement:

DEFINE *ifn*,JOURNAL=BEFORE

If this statement is not specified, the TDS application or the transaction aborts with the code ITMNAV returned by the Before Journal. When a transaction aborts with ITMNAV, it automatically restarts with the Before Journal option.

If the file is resident and is not cataloged, this statement must be included in the JCL of the TDS job:

DEFINE H_CTLM,JOURNAL=BEFORE

The file *tdsname*.CTLM may be journalized through the use of the After Journal.

If the default file sizes shown in Table 2-2, Characteristics of Online Files, are not suitable for the application, use this formula to calculate the approximate size of the CTLM file:

CIs = `10+10(m)+s*((t/13)+1)+0.1(f)`
where:
- *m* is the number of CONTROLLED COMMON-STORAGEs.
- *s* is the number of sessions with spawns waiting.
- *t* is the number of waiting spawns per session.
- *f* is the number of files declared.

	The size of the CTLN file must be at least 50 tracks. With a CISIZE of 6,146 bytes, there are 5 CIs per track on a MS/B10 disk. Thus for 50 tracks there are 250 CIs.
tdsname.RECOV	A file that contains information needed for the dynamic rollback and rollforward of TDS-controlled files. This file is managed in the same way as a swap file. The *tdsname*.RECOV file must be allocated on the same disk type (FBO or VBO) as the Swap files. If online files are allocated on resident disks of mixed type (FBO, VBO), you can use TP7PREP with the RDSN parameter to place *tdsname*.RECOV on the same type of disk.
tdsname.GMEM	A file that contains IQS objects to be executed by a TDS application. For further details, see the *IQS-V4/TDS User's Guide*.
tdsname.PPCLOG	A file used by the XCP2 service. This information is used for the TDS warm restart and contains the current state of the pools.
tdsname.MMS	A file that contains information necessary to run the MULTIMEDIA SERVICES PHASE 1 utility. This utility allows displaying documents through IMAGEWorks in a second window, while using a TDS application. Refer to *TDS-IMAGEWorks Link User's Guide*.
tdsname.HOST	A file used to record the use of the TDS gateway. It logs information about each service request received from BOS/TP. See *Setting up and Using /HOST7 User's Manual*.

File Name	Organization	Size, in bytes		Record form	Initial default file size
		Block	Record		
<tdsname>.DEBUG	Library (SL Format)	-	-	-	Initial file size is 5 cylinders (Maximum 10) 1 directory block
<tdsname>.SWAP[nn]	Internal	-	-	-	4 cylinders
<tdsname>.CTLN	UFAS indexed	6146	3055	VB	50 tracks (dynamic file extension)
<tdsname>.CTLM	UFAS indexed	6146	3055	VB	100 tracks (dynamic file extension)
<tdsname>.RECOV	Internal	1024	-	-	1 cylinder
<tdsname>.GMEM	UFAS indexed	-	-	-	4 cylinders (dynamic file extension in increment of 4 cylinders). Maximum number of extensions is 15.
<tdsname>.PPCLOG	UFAS indexed	2048	256	FB	20 tracks on VBO disk volume (equivalent to 2 blocks on a FBO disk volume)
<tdsname>.MMS	UFAS indexed	2048	1006	FB	2 cylinders
<tdsname>.HOST	UFAS indexed	4096	200	V	1 cylinder

Figure 2-3. Characteristics of Online Files

2.3 Allocating Files

To allocate the files for a TDS application, use the TP7PREP utility. To allocate a TDS file on a disk of a different device class or to change the file size, use the TP7PREP utility; otherwise errors may occur during TP7GEN or TDS execution. The files to be allocated are divided into two types: system files and the remaining files.

2.3.1 Allocating System Files

System files are online, namely:
- *tdsname*.CTLM
- *tdsname*.CTLN
- *tdsname*.RECOV

The SYSFILE parameter of TP7PREP determines how these system files are allocated.

The *tdsname*.RECOV file must be allocated on the same disk type (FBO or VBO) as the Swap files. If this file is to be resident, you must use the RDSN parameter (of TP7PREP) to place *tdsname*.RECOV on the same type of disk.

Parameter:

```
SYSFILE={ RSD | RSDN | CAT }
```

Arguments:

RSD — *Mutually exclusive with FILESTAT=CAT:* Files are resident and uncataloged. If RSD is selected and *tdsname*.RECOV is allocated on several volumes, TDS aborts at generation time with the message:

```
UNABLE TO INITIALIZE SWAP FILE
RC=XXXXxxxx IS TP7 13, EXTERR
```

To avoid this, ensure that after TP7PREP *tdsname*.RECOV is allocated on only one volume.

RSDN Resident disk volumes in which the three system files are located. These files are located as follows:

CTLMDVC={ondvc|*ctlmdvc*}	Device class of *tdsname*.CTLM
CTLMMD={onmd|*ctlmmd*}	Media of *tdsname*.CTLM.
CTLNDVC={ondvc|*ctlndvc*}	Device class of *tdsname*.CTLN
CTLNMD={onmd|*ctlnmd*}	Media of *tdsname*.CTLN.
RCVDVC={ondvc|*rcvdvc*}	Device class of *tdsname*.RECOV
RCVMD={onmd|*rcvmd*}	Media of *tdsname*.RECOV.

CAT Files are cataloged in the catalog specified by CATNAME (default TDS_APPL):

- if a directory object with the name of the catalog is declared in the Site Catalog, a catalog is built by the job
- if the directory object is not named in the Site Catalog, the catalog must be auto-attachable and the job must be executed under the SYSADMIN project.

Parameters associated with SYSFILE=CAT are:

CATNAME={TDS_APPL|*catname*}	Prefix of the site or private catalog in format *catname*.CATALOG.
CATDVC={ondvc|*catdvc*}	Device class of catalog.
CATMD={onmd|*catmd*}	Media of catalog.
IMPORT={YES|NO}	If *catname*.CATALOG is to be cataloged in the Site Catalog. If "SYS" or "SITE" is explicitly specified in CATNAME, use IMPORT=NO.
LIST={Y|N}	If catalog information is to be printed.
NBOBJ=200	Number of objects to be created in private catalog.
OWNER={SYSADMIN|*owner*}	Project having the right to modify or delete *catname*.CATALOG.

2.3.2 Allocating Non-System Files

Non-system files are:

- online files:
  - *tdsname*.DEBUG
  - *tdsname*.SWAP[nn] (number depends on NBSW parameter)
- optional online files:
  - *tdsname*.PPCLOG
  - *tdsname*.GMEM
  - *tdsname*.MMS
  - *tdsname*.HOST
- offline files:
  - *tdsname*.SLLIB
  - *tdsname*.COBOL
  - *tdsname*.SMLIB
  - *tdsname*.LMLIB (TDS.LMLIB or *prefix*.LMLIB if FILESTAT=UNCAT)
  - *tdsname*.EDITION

The FILESTAT parameter of TP7PREP determines how these non-system files are cataloged.

Parameter:

```
FILESTAT={ CAT | UNCAT }
```

Arguments: Files refer to online and offline files listed above.

CAT	Files are cataloged in the catalog specified by CATNAME.
UNCAT	Files are not cataloged and the LM library is either TDS.LMLIB or *prefix*.LMLIB (see LMUNCAT in Paragraph 2.4.2).

If the FILESTAT parameter value is UNCAT, all the tdsname.SWAP [nn] files must have the same device class, i.e. SW [nn] DVC keyword value.

2.4 Running the TP7PREP Utility

TP7PREP utility is a batch job invoked through JCL statements.

TDS-V7 has an updated version of this utility offering:

- new TP7PREP messages, see Appendix M
- and additional features offered by new parameters of TP7PREP:
  - FILEP to determine which file(s) are processed, one in particular or all. For example, to modify the size of SWAP1 file, specify FILEP=SWAP1.

    By default, FILEP=ALL processes all TDS files.
  - PPRTLIB determines whether the SYSOUT file (JOB_OUT) or a SSF member of a private library is allocated for the complete report containing the list of parameters and UTIL messages concerning allocation phases. An example of report is given in Paragraph 2.4.3.
  - WAIT to define a wait-time during which a file allocation is retired every 5 minutes. It is used if the disk volume is currently busy when the allocation is requested.

The number of steps of TP7PREP V7 JCL is reduced. The steps for preparing the CATALOG and listing the used CATALOG are retained but others are replaced by H_TP7PREP Load Module processing.

To run the TP7PREP utility, use the following JCL:

```
$JOB job-name, USER=userid [,PROJECT=project [,BILLING=billing ]];

INVOKE TP7PREP, SYS.HSLLIB, VALUES=(parameters), LIST=ALL;

$ENDJOB;
```

2.4.1 TP7PREP JCL Positional Parameters

Positional parameters allow arguments to be entered without their introducing parameters. The position of the argument determines the parameter itself. There are 6 positional parameters and are useful only for a small TDS application which only needs to specify the locations of its online and offline files.

The only mandatory positional parameter in VALUES is the first for naming the TDS application *tdsname*. *Avoid using positional-2 to positional-6 parameters. Instead, define them using the corresponding keyword syntax.*

positional-1	*tdsname:* Mandatory and identical to PROGRAM-ID name of TDS SECTION, see Paragraph 3.4.1 PROGRAM-ID Clause.
positional-2	*ondvc:* device class of the online files. The default ("MS/M452") should not be used.
positional-3	*onmd:* media of the online files. The default ("NONE") should not be used.
positional-4	*offdvc:* device class of the offline files. Default is positional-2.
positional-5	*offmd:* media of the offline files. Default is positional-3.
positional-6	*lang:* Programming language is COBOL by default. For TPRs written in C, this parameter is ignored.

2.4.2 TP7PREP JCL Parameters

BLKSZ={1024|*nnnn*}	Blocksize in multiples of 1024 bytes for formatting swap files. For FBO volumes, preallocate a block size of 4096 bytes.
CATDVC={ondvc|*catdvc*}	Applies if SYSFILE=CAT, see 2.3.1.
CATMD={onmd|*catmd*}	Applies if SYSFILE=CAT, see 2.3.1.
CATNAME={TDS_APPL|*catname*}	Applies if SYSFILE=CAT, see 2.3.1.
CBLDVC={offdvc|*cbldvc*}	Device class of *tdsname*.COBOL.
CBLMD={offmd|*cblmd*}	Media of *tdsname*.COBOL.
CBLSZ={3|*cblsz*}	Filesize in cylinders of *tdsname*.COBOL. If cblsz is less than 3, then default is assumed.
CTLMDVC={ondvc|*ctlmdvc*}	Applies if SYSFILE=RSDN, see 2.3.1.
CTLMMD={onmd|*ctlmmd*}	Applies if SYSFILE=RSDN, see 2.3.1.
CTLMSZ={100|*ctlmsz*}	Filesize in tracks of *tdsname*.CTLM. If ctlmsz is less than 1, then default is assumed.
CTLNDVC={ondvc|*ctlndvc*}	Applies if SYSFILE=RSDN, see 2.3.1.
CTLNMD={onmd|*ctlnmd*}	Applies if SYSFILE=RSDN, see 2.3.1.
CTLNSZ={50|*ctlnz*}	Filesize in tracks of *tdsname*.CTLN. If ctlnsz is less than 1, then default is assumed.
DBGDVC={ondvc|*dbgdvc*}	Device class of *tdsname*.DEBUG.
DBGMD={onmd|*dbgmd*}	Media of *tdsname*.DEBUG.
DBGSZ={5|*dbgsz*}	Filesize in cylinders of *tdsname*.DEBUG. If dbgsz is less than 1, then default is assumed.
DEAL={N|Y}	If new system and swap files can be deallocated and reallocated. Y to allocate new files and modify swap file size.
EDTDVC={offdvc|*edtdvc*}	Device class of *tdsname*.EDITION.
EDTMD={offmd|*edtmd*}	Media of *tdsname*.EDITION.
EDTSZ={1|*edtsz*}	Filesize in cylinders of *tdsname*.EDITION. If edtsz is less than 1, then default is assumed.

ESSTEINI={2A|*essteini*}	*Reserved for Service Center:* Value of first STE in SMLIB. INIT command is repetitively executed starting from *essteini* increasing by 1, to initialize SMs within SMLIB.
FILEP={ALL|COBOL|CTLM|CTLN|EDITION|GMEM|HOST|MMS|RECOV|SLLIB|SMLIB|SWAP[*nn*]|LMLIB|PPCLOG}	File(s) to process. *nn* in SWAP either has no value or ranges from 1 through NBSW.
FILESTAT={UNCAT|CAT}	Applies to Non-system files, see 2.3.2.
GMEM={NO|YES}	If TDS uses IQS without JPPC segments. See 3.3.10.
GMEMDVC={ondvc|*gmemdvc*}	Device class of *tdsname*.GMEM.
GMEMMD={onmd|*gmemmd*}	Media of *tdsname*.GMEM.
GMEMSZ={4|*gmemsz*}	Filesize of *tdsname*.GMEM. If gmemsz is less than 1, then default is assumed.
HA={NO|YES}	If the TDS is an HA application.
H_MMS={NO|YES}	If TDS uses MMS.
H_MMSDVC={ondvc|*h_mmsdvc*}	Device class of *tdsname*.MMS.
H_MMSMD={onmd|*h_mmsmd*}	Media of *tdsname*.MMS.
H_MMSSZ={2|*h_mmssz*}	Filesize of *tdsname*.MMS. If h_mmssz is less than 1, then default is assumed.
HOST={N|Y}	If TDS uses /HOST7.
HOSTDVC={ondvc|hostdvc}	Device class of *tdsname*.HOST.
HOSTMD={onmd|hostmd}	Media of *tdsname*.HOST.
HOSTSZ={1|hostsz}	Filesize of *tdsname*.HOST. If hostsz is less than 1, then default is assumed.
IMPORT={YES|NO}	Applies if SYSFILE=CAT, see 2.3.1.

JAS={SYS|BLUE|GREEN}	Type of JAS to which the TDS is linked by the MNJAS LINK command: - SYS for non-HA - BLUE and GREEN for HA. See *File Recovery Facilities User's Guide* and *High Availability* Documentation.
LANG=COBOL	Default programming language. Compatible with TPRs written in C.
LIST={Y|N}	Applies if SYSFILE=CAT, see 2.3.1.
LMDVC={offdvc|*lmdvc*}	Device class of *tdsname*.LMLIB.
LMMD={offmd|*lmmd*}	Media of *tdsname*.LMLIB.
LMSZ={3|*lmsz*}	Filesize in cylinders of *tdsname*.LMLIB. If lmsz is less than 3, then default is assumed.
LMUNCAT={TDS|*prefix*}	Prefix of the load module library (*tdsname*.LMLIB) when FILESTAT=UNCAT.
MAXCBL={10|*maxcbl*}	Maximum filesize in cylinders of *tdsname*.COBOL. If maxcbl is less than 10, then default is assumed.
MAXDBG={10|*maxdbg*}	Maximum filesize in cylinders of *tdsname*.DEBUG. If maxdbg is less than 1, then default is assumed.
MAXED={3|*maxed*}	Maximum filesize in cylinders of *tdsname*.EDITION. If maxed is less than 1, then default is assumed.
MAXLM={10|*maxlm*}	Maximum filesize in cylinders of *tdsname*.LMLIB. If maxlm is less than 10, then default is assumed.
MAXSL={10|*maxsl*}	Maximum filesize in cylinders of *tdsname*.SLLIB. If maxsl is less than 10, then default is assumed.
MAXSM={10|*maxsm*}	Maximum filesize in cylinders of *tdsname*.SMLIB. If maxsm is less than 10, then default is assumed.

MSG={SEND|{RPT|REPORT}|BOTH}	Destination of the TP7PREP informative messages: - SEND to the user's terminal - RPT (REPORT) to the JOR - BOTH to the JOR and the user's terminal. Error messages from JCL statements are always sent to the user's terminal.
NBOBJ=200	Applies if SYSFILE=CAT, see 2.3.1.
NBSW={1|*nn*}	Number of requested SWAP files from 1 through 32. More SWAP files on several devices give better performance than a single large file on a single device.
OFFDVC={positional-4|*offdvc*}	Device class of offline files.
OFFMD={positional-5|*offmd*}	Media of offline files.
OFFVSET=*offvset*	Volume Set for offline files. When omitted, OFFMD and OFFDVC are used to allocate files. (Ignored when FILESTAT = UNCAT).
ONDVC={positional-2|*ondvc*}	Device class of online files.
ONMD={positional-3|*onmd*}	Media of online files (system files, swap files, *tdsname*.DEBUG).
ONVSET=*onvset*	Volume Set for online files except SWAP files and RECOV. When omitted, ONMD and ONDVC are used to allocate files. (Ignored when SYSFILE=RSD[N] and/or FILESTAT=UNCAT).
OWNER={SYSADMIN|*owner*}	Applies if SYSFILE=CAT, see 2.3.1.
PPRTLIB={SYS.OUT|*pprtlib*}	Library of the TP7PREP print file. If PPRTLIB is not SYS.OUT, PPRTSUBF must be specified.
PPRTSUBF=*pprtsubf*	Corresponding subfile of PPRTLIB if PPRTLIB is not SYS.OUT (default).
RCVDVC={ondvc|*rcvdvc*}	Applies if SYSFILE=RSDN, see 2.3.1.
RCVMD={onmd|*rcvmd*}	Applies if SYSFILE=RSDN, see 2.3.1.
RCVSZ={1|*rcvsz*}	Filesize in cylinders of *tdsname*.RECOV. If rcvsz is less than 1, then default is assumed.

SLDVC={offdvc|*sldvc*}	Device class of *tdsname*.SLLIB.
SLMD={offmd|*slmd*}	Media of *tdsname*.SLLIB.
SLSZ={1|*slsz*}	Filesize in cylinders of *tdsname*.SLLIB. If slsz is less than 1, then default is assumed.
SMDVC={offdvc|*smdvc*}	Device class of *tdsname*.SMLIB.
SMMD={offmd|*smmd*}	Media of *tdsname*.SMLIB.
SMSZ={3|*smsz*}	Filesize in cylinders of *tdsname*.SMLIB. If smsz is less than 3, then default is assumed.
SW[*nn*]DVC={ondvc|*sw[nn]dvc*}	Device class of swap file, *nn* from 1 through NBSW: - SW**1**DVC refers to *tdsname*.SWAP - SW**2**DVC refers to *tdsname*.SWAP01 - SW**32**DVC refers to *tdsname*.SWAP31.
SW[*nn*]MD={onmd|*sw[nn]md*}	Media of the swap file, *nn* from 1 through NBSW: - SW**1**MD refers to *tdsname*.SWAP - SW**2**MD refers to *tdsname*.SWAP01 - SW**32**MD refers *tdsname*.SWAP31.
SW[*nn*]SZ={4|*sw[nn]sz*}	Size in cylinders of swap file, *nn* from 1 through NBSW: - SW**1**SZ refers to *tdsname*.SWAP - SW**2**SZ refers to *tdsname*.SWAP01 - SW**32**SZ refers to *tdsname*.SWAP31. If sw[nn]sz is less than 1, then default is assumed.
SWCACHE={BPIOC|NBPIOC}	If I/O cache is to be bypassed during I/Os on swap files. NBPIOC on FBO swap files can increase performance if the system is not CPU or memory-bound.
SYSFILE={RSD|RSDN|CAT}	Applies to System files, see 2.3.1.
WAIT={60|*nnn*}	Wait-time in minutes. If a volume is not currently accessible (BUSY) TP7PREP retries every 5 minutes to access it until timed out.
XCP2={N|Y}	If XCP2 is used. If YES, the optional online file *tdsname*.PPCLOG is built and cataloged even if FILESTAT=UNCAT is specified. A catalog for TDS must exist.

XCP2DVC={ondvc|*xcp2dvc*}	Device class of *tdsname*.PPCLOG.
XCP2MD={onmd|*xcp2md*}	Media of *tdsname*.PPCLOG.
XCP2SZ={20|*xcp2sz*}	Filesize in cylinders of *tdsname*.PPCLOG on a VBO disk volume. For a FBO disk volume, the equivalent size is 2 blocks. If xcp2sz is less than 1, then default is assumed.

2.4.3 TP7PREP Reporting

EXAMPLE of TP7PREP Report:

```
  TP7PREP   10.00  X2015.9   TP7PREP  STDS1    TDS1     TDS1--V3
12:09:50  APR 14, 1994    PAGE    1

****************************************************************
****************************************************************
** GCOS7          ****
** T P 7 P R E P  ****
** VERSION: 10.00 DATED: MAR 03, 1993 ****
****************************************************************
****************************************************************

Parameters are:

   TDSNAME=TDS1
   SYSFILE=CAT
   FILESTAT=CAT
   CATNAME=TDS1
   LMINCAT=TDS
   HA=NO
   DEAL=Y
   FILEP=ALL
   MSG=SEND
   WAIT=60
   NBSW=1
   BLKSZ=1024
   SWCACHE=BPIOC
   H_MMS=NO
   GMEM=NO
   XCP2=N
   HOST=N
   SLSZ=1
   SLMD=BCC024
   SLDVC=MS/B10
   MAXSL=10
   SMSZ=3
   SMMD=BCC024
   SMDVC=MS/B10
   MAXSM=10
   LMSZ=3
   LMMD=BCC024
   LMDVC=MS/B10
   MAXLM=10
   EDTSZ=1
   EDTMD=BCC024
   EDTDVC=MS/B10
   MAXED=3
```

```
CBLSZ=3
CBLMD=BCC024
CBLDVC=MS/B10
MAXCBL=10
CTLMSZ=100
CTLMMD=BCC024
CTLMDVC=MS/B10
CTLNSZ=50
CTLNMD=BCC024
CTLNDVC=MS/B10
RCVSZ=1
RCVMD=BCC024
RCVDVC=MS/B10
DBGSZ=5
DBGMD=BCC024
DBGDVC=MS/B10
MAXDBG=10
H_MMSSZ=2
H_MMSMD=BCC024
H_MMSDVC=MS/B10
GMEMSZ=4
GMEMMD=BCC024
GMEMDVC=MS/B10
XCP2SZ=20
XCP2MD=BCC024
XCP2DVC=MS/B10
HOSTSZ=2
HOSTMD=BCC024
HOSTDVC=MS/B10
SW1SZ=4
SW1MD=BCC024
SW1DVC=MS/B10
.
.
.
SWnSZ=4        }
SWnMD=BCC024   } where n ranges from 1 through 32
SWnDVC=MS/B10  }
.
.
.
JAS=SYS
ESSTEINI=2A
```

```
File processing started:

TV24 TP7GENTDS1 CREATED INTO TDS1.SLLIB
TV56 Successful DEallocation for TDS1.RECOV
TV56 Successful allocation for TDS1.RECOV
TV56 Successful DEallocation for TDS1.SWAP
TV56 Successful allocation for TDS1.SWAP
TV56 Successful DEallocation for TDS1.CTLM
TV56 Successful allocation for TDS1.CTLM
TV56 Successful DEallocation for TDS1.CTLN
TV56 Successful allocation for TDS1.CTLN
```

❑

3. Generating a TDS Application with TP7GEN

The TDS generation describes the environment in which transactions execute and involves:

- defining the source of the TDS Generation (STDS file) using the appropriate syntax
- launching the TP7GEN utility for compilation.

As from Technical Status 6152, TP7GEN message reporting has been improved as follows:

- messages in general have been reclassified
- messages conform to the PMOS interface except for those messages involved in JCL
- severity levels are more clearly reported, namely:
  - `*` `level 1:` warning
  - `**` `level 2:` does not interrupt TP7GEN but no supported generation results
  - `***` `level 3:` fatal abort at the end of the STDS analysis phase
  - `****` `level 4:` fatal or immediate abort of TP7GEN
- *obsolete clauses which cause Severity 2 errors instead of warnings (Severity 1) must be removed.* See Appendix N for *TP7GEN Errors and Responses*.

3.1 Structure of STDS File

The set of statements composing the TDS source generation is built up from clauses of the sections in the following order:

1. TDS Section

The TDS Section supplies global parameters for describing the environment in which transactions execute, creating the links to the prepared TDS files, and defining the operating limits on the application. These limits include the number of processes available, message buffer sizes and defaults.

The syntax of the TDS Section is given in Paragraph 3.4. The order of clauses must be respected as given except for the various USE clauses which can be interchanged.

2. INPUT-OUTPUT Section

The INPUT-OUTPUT Section describes the files and IDS/II database processed, and file-management techniques used. It contains information on user data files for the file and record descriptions used by the TDS application. In addition, it contains details of the Terminal Writer and the User-Journal files.

The information in this section is present in COBOL SELECT statements and FD clauses to be used by the Executive process and any TPRs referencing them. The TP7GEN utility copies these clauses into the *tdsname*.COBOL library and can then be used to write TPRs.

The three clauses in this section are used for declaring data structures that can be later copied to TPRs, namely:

WORKING-STORAGE
SHARED-STORAGE.

The syntax of the INPUT-OUTPUT Section is given in Paragraph 3.5.

3. Transaction Section

The Transaction Section specifies all details of transactions that can be activated by a user. This includes message identifiers and parameters associated with each transaction, and defines the information contained in the region shared by TPRs processing the same transaction.

Each transaction is defined in a single MESSAGE Statement made up of clauses to describe various functions of the transaction.

The following special transactions with TDS-reserved message-identifiers are executed automatically when a given event occurs:

BREAK
DISCNCT
LOGON
LOGOUT
RESTART
TARTUP
SHUTDOWN
SYNCPEVT.

Each transaction requires a separate MESSAGE statement of the format:

```
TRANSACTION SECTION.
message
    [ message ... ]
    [ ...         ]
    [ message ... ]
```

The syntax of the MESSAGE statement of the Transaction Section is given in Paragraph 3.5.

3.2 Preparing TDS Source Generation

The TDS Source Generation is prepared in IOF and is stored in the subfile STDS of the *tdsname*.SLLIB library. The STDS member is also created under IOF through the LIBMAINT utility. See the *Library Maintenance Reference Manual*.

An example of a TDS source generation is given in Appendix H.

3.3 Running TP7GEN Utility

3.3.1 Functions

The TP7GEN utility is the TDS generation program which compiles the source in STDS. It performs the following functions:

- inputs the TDS generation source from the source library *tdsname*.SLLIB. The subfile name is STDS.
- outputs an object version of the generated TDS application to the object library *tdsname*.LMLIB, the load module name being *tdsname*.
- outputs LINKER command files to the source library *tdsname*.SLLIB so that TPRs can be linked. Subfile names are TP7LINKTPR, TP7LINKTPR1, TP7LINKTPR2 through TP7LINKTPR99.
- outputs source COBOL code to a COBOL source library *tdsname*.COBOL for later reference by COPY statements in TPRs.
- initializes all swap files (this implies that all swap file contexts are lost).
- initializes the *tdsname*.RECOV file.
- stores TP7GEN parameters in the *tdsname*.CTLM and *tdsname*.CTLN files.
- outputs a report used for subsequent printing, in *tdsname*.EDITION. The subfile name is GEN*tdsname*.

Figure 3-1 summarizes TP7GEN.

TP7GEN can be run only after all the required files are allocated by TP7PREP and when the TP7GEN source has been stored in the STDS subfile. Execute TP7GEN with the following JCL:

```
$JOB job-name, USER=userid [,PROJECT=project [,BILLING=billing ]];

INVOKE TP7GEN, SYS.HSLLIB

      ,VALUES=(parameters)
```

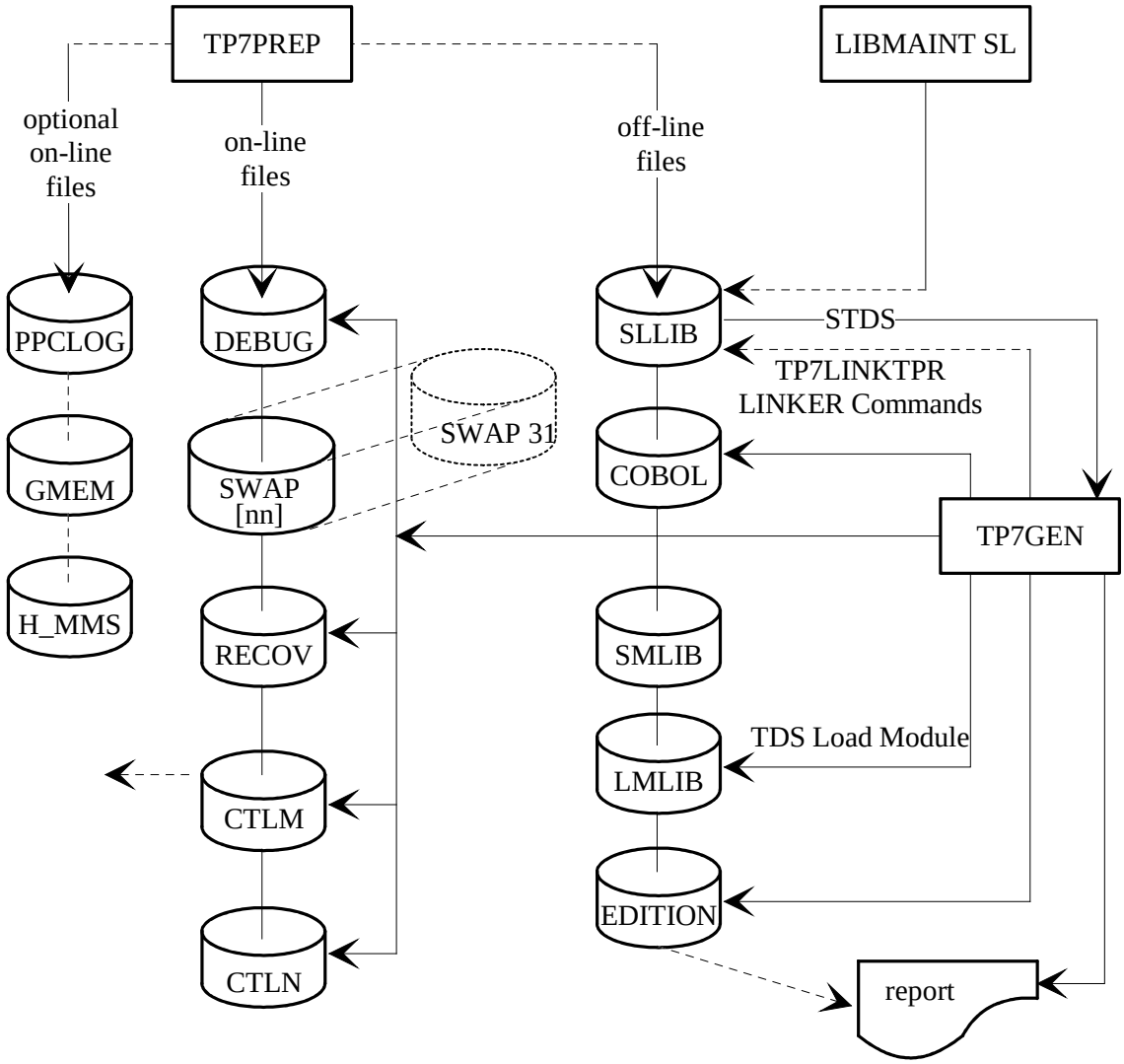


Figure 3-1. Functions of the TP7GEN Utility

3.3.2 TP7GEN JCL Parameters

TP7GEN JCL is a member delivered with GCOS 7 software in the SYS.HSLLIB library.

It presents 6 positional parameters (arguments without introducing parameters) corresponding to the 6 positional parameters of TP7PREP JCL.
See Paragraph 2.4.1.

The only mandatory positional parameter in VALUES is the first being *tdsname* for naming the TDS application.

positional-1	*tdsname:* Mandatory and identical to PROGRAM-ID name of TDS SECTION, see Paragraph 3.4.1 PROGRAM-ID Clause.
positional-2	*ondvc:* device class of the online files. No default is provided. Should be specified if any online file is not cataloged nor resident.
positional-3	*onmd:* media of the online files. Same restriction as positional-2.
positional-4	*offdvc:* device class of the offline files. Default is positional-2. Should be specified if any offline file is not cataloged.
positional-5	*offmd:* media of the offline files. Default is positional-3. Same restriction as positional-4.
positional-6	*lang:* Parameter not used.

Other TP7GEN parameters are introduced by keywords, namely:

CBLDVC={cbldvc|offdvc}	Device class of *tdsname*.COBOL. *Default:* offline files dvc.
CBLMD={cblmd|offmd}	Media of *tdsname*.COBOL. *Default:* offline files media.
CHECKSTX={NO|YES}	Check syntax of STDS subfile without affecting the current TDS generation. Only *tdsname*.EDITION is modified. TP7PREP must be run at least once with FILEP=ALL to generate the JCL TP7GEN*tdsname* in *tdsname.sllib* used for TP7GEN.

SPAWN={NOKEEP|KEEP}	This keyword allows to keep the spawnings requested in a previous TDS session after the TDS generation. When SPAWN=KEEP is issued, all the deferred or immediate spawned transactions are still available after the generation. See also the JCL for Running TDS job to keep the spawning at restart cold. To benefit from this functionality, a TP7PREP with parameters DEAL=Y and FILEP=ALL must be issued (first and once) because the TP7-GENTDSname subfile must be refreshed in the TDSname.SLLIB file. The spawning requests are lost if the generation aborts. So, if STDS subfile has been changed since the last successful generation, the generation must be launched with parameter CHECKSTX=YES. Once the generation syntax is correct, it must be launched with parameters CHECKSTX=NO and SPAWN=KEEP. If the TDS generation is killed by Cancel_Job command, the spawning requests will be lost. In this case, the generation can only be launched with the parameter SPAWN=NOKEEP.
CPRTLIB={TP7FILDSC_C|*cprtlib*}	Print file of the COBOL step. *Default:* automatically set by COBOL to TP7FILDSC_C.
EDTDVC={edtdvc|offdvc}	Device class of *tdsname*.EDITION. *Default:* offline files dvc.
EDTMD={edtmd|offmd}	Media of *tdsname*.EDITION. *Default:* offline files media.
GMEM={NO|YES}	YES if IQS is used.

GPRTLIB={SYS.OUT|*gprt-lib-desc*}	Library of the TP7GEN print file. *Default:* SYS.OUT. If the default is not used, a subfile must be explicitly specified for GPRTSUBF.
GPRTSUBF=gprt-sfile-name	Corresponding subfile and must be explicitly specified if the default GPRTLIB=SYS.OUT is not specified.
H_MMS={NO|YES}	YES if MMS is used.
LM={TDS.LMLIB|*tdsname*.LMLIB}	Library where the generateed TDS load module is stored.
LMDVC={lmdvc|offdvc}	Device class of TDS.LMLIB. *Default:* offline files dvc.
LMMD={lmmd|offmd}	Media of TDS.LMLIB. *Default:* offline files media.
ORASTLIB=*orastlib*	SMLIB which contains the H_ORATDS sharable module.
CMASTLIB=*cmastlib*	SMLIB which contains the H_SM_CMA sharable module.
SA7={NO|YES}	Secures TDS if the site is protected by SECUR'ACCESS. If YES, TDS is secured and the STDS member is automatically completed by MTGENSCL.
SA7IDS={YES|NO}	Modifies the STDS environment. If YES, full IDS environment is used. Used with SA7CULIB.
=YES *(default)* =NO	SA7CULIB=SA7.LIV.CUFIDS SA7CULIB=SA7.LIV.CU_USER
SA7CUDVC={NIL|*cudvc*}	Device class of SA7CULIB if uncataloged. *Default:* NIL.
SA7CULIB={SA7.LIV.CUFIDS } {SA7.LIV.CU_USER}	Library to be searched by TP7GEN for the SECUR'ACCESS compile unit, namely:
=SA7.LIV.CUFIDS =SA7.LIV.CU_USER	if SA7IDS=YES if SA7IDS=NO

SA7CUMD={NIL|*cumd*}	Media of SA7CULIB if uncataloged. *Default:* NIL.
SLDVC={sldvc|offdvc}	Device class of *tdsname*.SLLIB. *Default:* offline files dvc.
SLMD={slmd|offmd}	Media of *tdsname*.SLLIB. *Default:* offline files media.

When TP7GEN includes either the USE procedure clause or the USE procedure FOR TRANSACTION INITIALIZATION clause, specify the following JCL to compile these procedures:

```
$JOB jobname;

LIB SL,INLIB1=( sllibname ,DVC=dvc ,MD=md );

COBOL SOURCE=procedure-name,...,CULIB=TEMP;

INVOKE TP7GEN ...
```

Note that the CUs (compile units) of the above procedure are stored in the temporary library TEMP.

To prevent compiling the procedure at each TDS generation:

- first store the CU in a CU library
- then at the start of each job invoking TP7GEN, move the CU to a TEMP library to be used by TP7GEN.

3.4 Syntax of Section

```
TDS SECTION.

PROGRAM-ID. programid [ WATCHED BY CMSC ].

[ NUMBER OF DUMMY CORRESPONDENT IS { 1 | ndummy }
                          MAXIMUM IS { 1 | maxdummy }.]
[ NUMBER OF VIRTUAL correspondent-familyname IS nbvirt.]

SIMULTANEITY IS nsimu.
[ RPC SIMULTANEITY IS nrpcsimu.]
[ CD-IN DATE FORMAT IS { YYMMDD | DDMMYY | MMDDYY }.]

[ MAXIMUM NUMBER OF BEFORE JOURNAL COMMITMENT UNITS IS nbbjcu.]

RESERVE rsvarea AREAS.
[ATTACH SHARABLE MODULE sm.]

[ MAXIMUM NUMBER OF DYNAMICALLY ADDED TRANSACTIONS IS {10|maxtx}.]
[ NUMBER OF TPR SHARABLE MODULES { 1 | numshare }.]
[ NUMBER OF IQS JPPC SEGMENTS IS nbseg.]

[ DYNAMIC-SUPERVISION FOR SESSION [ WITH LEVEL { 4 | dec1 } ]
  [ ABORT-LEVEL IS { 5 | dec1 } ] [ DELAY IS { 20 | dec5 } ].]
[ DYNAMIC-SUPERVISION FOR PROCESS [ WITH LEVEL { 4 | dec1 } ]
  [ ABORT-LEVEL IS { 5 | dec1 } ] [ DELAY IS { 20 | dec5 } ].]

[ COMMON-STORAGE SIZE IS { 0 | commonsize }.]
[ DEFAULT TRANSACTION-STORAGE SIZE IS { 0 | deftxstorage }.]
[ MAXIMUM TRANSACTION-STORAGE SIZE IS maxtssz.]
[ PRIVATE-STORAGE SIZE IS { 0 | privatesize }.]
[ MAXIMUM PRIVATE-STORAGE SIZE IS maxpssz.

MESSAGE-LENGTH IS { 1024 | msglength } MAXIMUM.

[ FORMATTED MESSAGE-LENGTH IS { 10033 | formsglength } MAXIMUM.]
[ USER-JOURNAL [ WITH [ INPUT ] MESSAGE LOGGING ].]

TPR-TIME-LIMIT IS timelimit MSEC.
[MAXIMUM NUMBER OF TPR WITH NO UNMAPPING IS maxunmap.]

[ DEFAULT ABORT TPR-NAME [IS] tprname.]


[ MAXIMUM IDLE-TIME IS { 0 | idletime }
[ MAXIMUM WAIT-TIME UPON SYNCHRONOUS SEND is nmin.]

[ MASTER MAILBOX IS mbx-name [TYPE IS XCP1 ].]
```

```
[ NUMBER OF XCP1 TRANSACTION AUXILIARY SESSIONS IS { numaux }.]

[ SELECT XCP2-SYNCPEVT eventid [,eventid...].]
[ MAXIMUM XCP2-WAITTIME IS { 0 | maxwait }.]

[ MAXIMUM DTP-WAITTIME IS maxdtpwait.]

[ WATCHTIME[IS]watchtime[EXCEPT [FOR]{TERMINAL|TCP-IP}].]


[ HOST GATEWAY IS gatewayname.]

[ XA-RESYNC-DELAY IS {dec4 | 300 }.]

[ TCP-IP PROTOCOL [ USED ] [ WITH { OPEN7 | GXTI } ].]

[ USE { "?" | "lastmsg" } FOR LAST MESSAGE.]
[ USE { "/" | "menu" } FOR TRANSACTION-MENU.]
[ USE NOPWCHK-ALLOWED.]
[ USE FREE-ACCESS-TDS.]
[ USE procedure FOR TRANSACTION INITIALIZATION.]
[ USE FORMS [ NO IMPLICIT RELEASE ].]
[ USE TERMINAL-ADAPTER.]
[ USE ORACLE.]

[     { STARTUP-DETAILED-REPORT } ]
[ USE { STARTUP-STANDARD-REPORT }.]
[     { STARTUP-SUMMARY-REPORT  } ]

[ USE SHORT-STATISTICS.]
[ USE CONVENTIONAL-ABORT-TPR.]
[ USE M-MASTER-CONVENTION.]

[ USE SPAWNED-TX-MESSAGE-LOGGING.]
[ USE PASSIVE-SPAWN-CHECK.]

[ USE LARGE-STACK-SEGMENT.]
[ USE TYPE3-LARGE-SEGMENT.]
[ USE DEVICE-HEADER-LOGGING.]
[ USE IDS-SUB-SCHEMA.]
[ USE CONNECTION-LOGGING.]

[ USE TILS-SIMULATION.]
[ USE NO_SCROLLING_REQUEST.]
[ USE EDIT-EXCEPTION.]
[ USE WAIT_VIRTUAL_FOR_SHUTDOWN.]
[ USE DISPLAY_IN_JOR.]
[ USE TWRITER-NUMSF-xxx.]
[ USE XCP1-CLOSEPOOL-STRONG.]
[ USE XCP1-ALNTC-PRIMARY-ONLY.]
[ USE JOR-STD-FORMAT.]
```

```
[ USE NO-PROP-MSG-IN-JOR.]
[ USE M-TX-ABORT-ON-BREAK. ]
[ USE LINK-WITH-AUTOBIND.]
[ USE TDS-SPECIALIZED-WEB7.]
[ USE WATCHDOC-ON-ASYNCHRONOUS-SEND-EMI.][ USE procedure.]

[ SERVICE-MESSAGE [ HEADER IS "servheader" ]
                  [ TRAILER IS { "0D25" | "servtrailer" }]
                  [ IMMEDIATE DELIVERY ].]

[ SERVICE-MESSAGE msgnumber IS "servicemsg".]

[ TDSTX-MESSAGE [ HEADER IS { "servheader" | "txheader" }]
                [ TRAILER IS { "servtrailer" | "txtrailer" }].]

[ SPECIAL-CHAR mnemonic IS "code".]...

[ CANCELCTX AT RECONNECTION .]

[ FILE-OPENING OPTION IS option. ]
```

3.4.1 PROGRAM-ID Clause

Syntax:

PROGRAM-ID. *programid* [WATCHED BY CMSC].

Description:

Specifies the name of the TDS application and enables the TDS files prepared with TP7PREP to be linked to the application. This name must be cataloged in the Site Catalog.

Usage:

- *programid* is up to 4 alphanumeric characters used in INVOKE TP7PREP and INVOKE TP7GEN statements.
- *programid* must not be: CVTF, IORP, JCSJ, KJOB, OWF, OWQ, OWQB, OWSM, OWT, PLMT, SLRM, SSMT, SYSN, TWS or XEPT, or an operator command.
- WATCHED BY CMSC option refers to the HA (High Availability) option which is described in *High Availability Concepts* and *High Availability Administrator's Guide*.

NOTE:

programid must be different from the name of the user starting the TDS application.

For a given machine, the *programid* of the TDS applications must be different.

Although the names of TDS applications running on different machines can be the same, the names of TDS applications co-operating through the XCP1, or XCP2 protocol must be unique.

3.4.2 NUMBER OF DUMMY CORRESPONDENT Clause

Syntax:

```
[ NUMBER OF DUMMY CORRESPONDENT IS { 1 | ndummy }
                         MAXIMUM IS { 1 | maxdummy }.]
```

Description:

Creates the number of dummy sessions specified in *ndummy*, when the TDS application starts. Other dummy sessions can be created through [M] MODIFY_COR_POOL command until the limit defined in maxdummy is reached.

A dummy correspondent is a virtual correspondent that can only be spawned to. It serves to execute transactions in which no exchanges occur. The term DUMMY itself is generic; there is a pool of dummy correspondents in TDS.

Usage:

- *ndummy* determines the number of dummy sessions at TDS startup. *ndummy* is greater that zero and must not exceed *maxdummy* (0 < *ndummy* < *maxdummy*)
- *maxdummy* defines the total number of dummy sessions allowable in the TDS application. It must be less than the number of terminals defined and not exceed 100.
- Defaults for *ndummy* and *maxdummy* is 1.

3.4.3 NUMBER OF VIRTUAL Clause

Syntax:

```
[ NUMBER OF VIRTUAL correspondent-familyname IS nbvirt.]
```

Description:

Declares families of virtual correspondents as dummies but without pool.

The correspondent-familyname is chosen by the user and must not exceed ten characters (because a two digits rank is added by TDS).

nbvirt specifies the number of correspondents in the family that TDS will connect at STARTUP, its value ranges from 1 through 100.

This clause can be used up to five times, with different correspondent family names and, if necessary, a different value of *nbvirt*.

Usage:

- Consider the following Example 1:

```
NUMBER OF VIRTUAL BADGER IS 2.
NUMBER OF VIRTUAL MONKEY IS 4.
NUMBER OF VIRTUAL WALRUS IS 1.
NUMBER OF VIRTUAL WEASEL IS 3.
NUMBER OF VIRTUAL KANGAROO IS 100.
```

  In this example the virtual correspondents are:

  BADGER00, BADGER01
  MONKEY00 to MONKEY03
  WALRUS00
  WEASEL00 to WEASEL02
  KANGAROO00 to KANGAROO99

  For TDS before TS6152, if NUMBER OF TERMINALS is specified in the SDTS subfile, the number of terminals specified in this clause must be greater than the sum of the virtual correspondents in all the families and the maximum number of dummies.

The user may spawn by specifying a particular virtual correspondent name, namely:

```
CALL SPAWN BADGER01 txname status
```

Since the correspondent is uniquely identified, private-storage can be used and the transaction can be traced using the TRACE PRINT mechanism.

- Consider the following Example 2:

  ```
  CALL SPAWN BADGER txname status:
  ```

  In this example the user did not specify a particular virtual correspondent, so TDS will choose the most efficient correspondent (typically an idle one) in the BADGER family to run the transaction.

  The CTC master command with the STRONG option can be used on a particular virtual correspondent to abort a running transaction. The correspondent is not disconnected.

  The master terminal command TTDS does not wait for the end of transaction on virtual correspondents to stop TDS.

3.4.4 SIMULTANEITY Clause

Syntax:

SIMULTANEITY IS *nsimu*.

Description:

Defines the number of subordinate processes which the TDS application can use simultaneously. These processes are the operating system resources that must be allocated to the transactions as needed.

Usage:

nsimu is the maximum number of TPRs that can be processed concurrently. This value cannot exceed 250.

NOTE:
The maximum simultaneity level of 250 does not mean that only 250 transactions can execute at the same time. It means that if the number of transactions exceeds 250, some may await activation.

3.4.4.1 Multitasking

SIMULTANEITY is the multitasking level which defines the number of TPRs that can execute simultaneously. If SIMULTANEITY=*n*, the TDS load module contains n + 1 executive processes from P0 to Pn which function as follows:

P0:

- initializes the TDS run occurrence
- starts processes P1...Pn
- opens swap files and loads the TDS tables
- then terminates the TDS run occurrence.

P1 through Pn:

- wait for messages coming from OMH, VCAM, GAC-EXTENDED, timer and TDS itself and treat these messages on a FIFO basis
- and are activated by an incoming message.

When the incoming message involves a TPR:

- the TPR is initialized and control is passed to it
- control is returned to TDS at the end of the TPR.
- TDS decides if a commitment is to be taken and if the TPR needs unmapping.

When the TPR terminates, the actions which follow depend on whether the transaction keeps the process for the next TPR or not:

- If the transaction remains mapped, the process is reused immediately for the next TPR of the same transaction.
- If the transaction is unmapped, the process waits to be reactivated for another TPR of any transaction, not necessarily the next TPR of the current transaction.

CPU time is given to the process according to its dispatching priority. Processes with the same priority are dealt with in FIFO order. A process, functioning as a server to a TPR, can be set to ***wait*** when issuing either I/Os for data management or communications verbs. When this happens, the process loses its allocated CPU time which is then given to another process or to a TDS task.

The time taken for a TPR to execute and the I/O time are far less than operator think-time. During operator think-time, the TPR is unmapped from the process so that the process can then be used for other TPRs. For example, if the operator think-time is 30 seconds and the TPR elapse time is 1 second, then the TDS task can serve 30 users in the meantime.

If at least one transaction works with XCP2, the XCP2 protocol is used and the TDS load module contains $n + 2$ processes from P0 to Pn + 1. The extra process is used by the PPC component of GCOS 7 for processing XCP2 verbs and is not visible to the master operator. Supervision requires one process in the process pool.

Declare at least a simultaneity level of 2. Otherwise TP7GEN forces it to 2.

In Figure 3-2, process P1 and process P2 run 3 transactions: Tx1, Tx2, and Tx3. Initially process P1 works for TPR1 which belongs to transaction Tx1. At the same time, process P2 is working for other transactions. The two processes run in parallel.

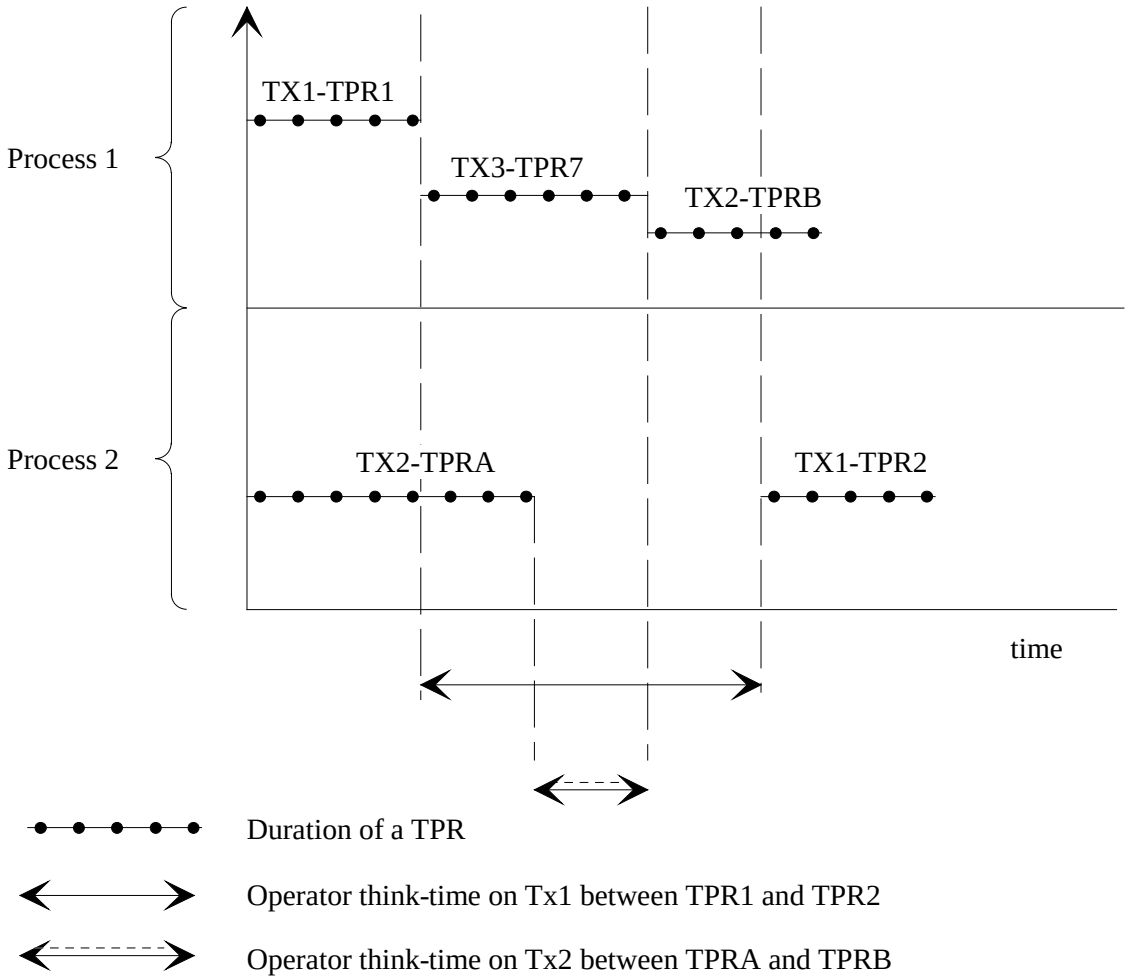


Figure 3-2. Multitasking

3.4.4.2 Segments of an Executive Process

An executive process comprises private and sharable segments. Figure 3-3 shows the different tables and storages used by a TDS application. See *TDS COBOL Programmer's Guide*.

The code of a TPR is sharable. If several processes access the same TPR, only one copy of the TPR is in memory.

The data of a TPR is private and is loaded in its initial state (refreshed) at each new TPR activation.

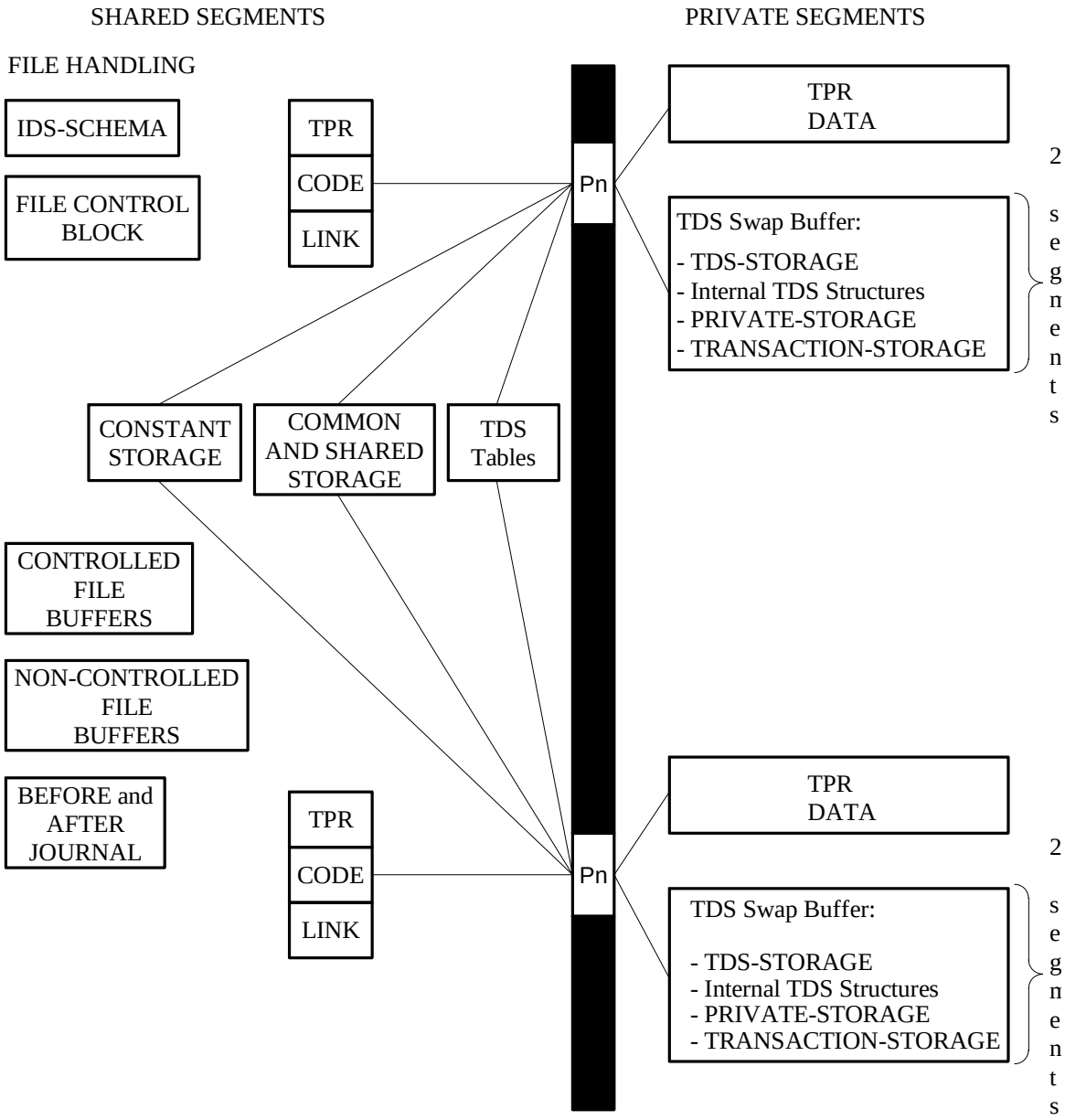


Figure 3-3. Executive Processes

3.4.5 RPC SIMULTANEITY Clause

Syntax:

```
[ RPC SIMULTANEITY IS nrpcsimu.]
```

Description:

Fixes the number of executive processes allowed to execute TPRs using RPC concurrently and allows using RPC simultaneity regulation services.

Usage:

- *nrpcsimu* is the number of TPRs using RPC that can be processed concurrently, *nrpcsimu* must be less than *nsimu*.
- If *nrpcsimu* is greater than or equal to *nsimu*, the severity 3 error message is written to the JOR:

```
TG26 *** ERRONEOUS PARAMETER: RPC SIMULTANEITY NUMBER.
```

3.4.6 CD-IN DATE FORMAT Clause

Syntax:

```
[ CD-IN DATE FORMAT IS { yymmdd | ddmmyy | mmddyy }.]
```

Usage:

Returns the date in the format specified. Default is *yymmdd*.

3.4.7 MAX NUMBER OF BEFORE JOURNAL COMMITMENT UNITS Clause

Syntax:

[MAXIMUM NUMBER OF BEFORE JOURNAL COMMITMENT UNITS IS *nbbjcu*.]

Description:

Specifies the maximum number of commitment units which need the Before Journal that can be processed concurrently.

Usage:

- Allows reducing the size of the TDS Before Journal. If *nbbjcu* is less than the number of terminals declared at NETGEN, *nbbjcu* is taken into account for computing the size of the Before Journal.
- Should only be specified when the Deferred Update mechanism is used for file access. *nbbjcu* must include the number of simultaneous transactions which need the Before Journal, in particular, for *Suppress Deferred Updates* request or CI splitting.
- **If *nbbjcu* is too low for the TDS application, transactions may abort, in which case, the abort code is ENTRYOV.**

3.4.8 RESERVE AREAS Clause

Syntax:

RESERVE *rsvarea* AREAS.

Description:

Specifies the maximum number of file buffers that TDS can reserve for TDS-controlled files and IDS/II realms when buffer pools are to be defined through DEFINE statements (BUFPOOL and NBBUF parameters). Buffers for TDS non-controlled files are selected from these reserved pools of the JCL launching the TDS execution and are referred to as the DEFT pool.

Usage:

- Creates a buffer pool named *tdsname* with NBBUF equal to the value of rsvarea.
- The buffer pool named *tdsname* is shared by all TDS-controlled files and IDS/II realms for which BUFPOOL has been omitted, and no NBBUF parameter values appear in the JCL statement DEFINE in the TDS step.

  The file *tdsname*.CTLM always uses the buffer pool named *tdsname*. For a TDS application, the number of buffers including the default buffer pool *tdsname* must not exceed 20,000.
- The value of *rsvarea* like NBBUF values specified in the JCL statement DEFINE depends on buffer usage. With deferred updates, ensure that the number of buffers required is large enough. See Paragraph 7.6.3 *Allocating Buffer Space*.

  Because the buffer pool named *tdsname* is used at least by the file *tdsname*.CTLM, the minimum value for rsvarea is 10. If a lower value is specified, the TDS application will force this minimum to 10.
- A buffer that is specified but not created is flagged in memory by 8 bytes; several areas can then be declared without affecting memory occupancy.

NOTE:

Several buffer pools can be created in UFAS-EXTENDED. When the DEFINE statement includes the BUFPOOL parameter, the default for NBBUF is *rsvarea*.

Buffers are dynamically created and deleted to meet the needs of executing TPRs. TDS creates as many buffers as possible below the limit fixed by the JCL statement SIZE. If this limit has been reached, TDS will create additional buffers in swappable memory.

One or two file buffers, depending on the file organization, are allocated for each non-controlled file. These buffers are not part of the buffer pool.

3.4.9 ATTACH SHARABLE MODULE Clause

Syntax:

[ATTACH SHARABLE MODULE *sm* .]

Description:

Allows to specify a Sharable Module that is referenced during the execution of the TDS application. The Sharable Module is included in the LINKER command file for the link step of TP7GEN.

Usage:

Available value for *sm*:

H_SM_DCM (refer to *DCE RPC on GCOS 7 Administration Guide*, 40 A2 62CF for details on this Sharable Module).

3.4.10 MAXIMUM NUMBER OF DYNAMICALLY ADDED TRANSACTIONS Clause

Syntax:

MAXIMUM NUMBER OF DYNAMICALLY ADDED TRANSACTIONS IS {**10**|*maxtx* }.

Description:

Specifies the maximum number of new transactions which can be added during the current TDS session through use of the MODIFY_TX command.

Usage:

maxtx ranges from 0 to 50. *Default:* 10.

3.4.11 NUMBER OF SHARABLE MODULES Clause

Syntax:

```
[ NUMBER OF TPR SHARABLE MODULES { 1 | numshare }.]
```

Description:

Specifies the number of TPR SMs (sharable module) to be initialized in the SM library.

TPRs are linked to SMs and placed in the SM library. They are loaded by the SYSMAINT utility into the backing store system files for execution.

TP7GEN creates a command file used by the LINKER utility for up to 100 SMs named TP7LINKTPR, TP7LINKTPR1, TP7LINKTPR2 through TP7LINKTPR99.

Usage:

- *numshare* is the number of TPR SMs whereby, 1 *(default)* <= *numshare* <= 100.
- Up to 252 link, code, and data segments of TPRs can be contained in one sharable module.
- If several TDS applications are running on the same machine, the total number of loaded sharable modules including system sharable modules, must not exceed 65.

3.4.12 NUMBER OF IQS JPPC SEGMENTS Clause

Syntax:

```
[ NUMBER OF IQS JPPC SEGMENTS IS nbseg.]
```

Description:

Specifies the number of segments in JPPC used by IQS. When this clause is used, the GMEM file is no longer used.

Usage:

nbseg determines the number of segments for IQS. Normally a user needs 4 segments. If a very large request is submitted, a maximum of 16 segments could be required. This clause allows selecting the appropriate amount of segments.

3.4.13 DYNAMIC-SUPERVISION Clause

Syntax:

```
[ DYNAMIC-SUPERVISION FOR SESSION [ WITH LEVEL { 4 | dec1 } ]
  [ ABORT-LEVEL IS { 5 | dec1 } ] [ DELAY IS { 20 | dec5 } ].]
[ DYNAMIC-SUPERVISION FOR PROCESS [ WITH LEVEL { 4 | dec1 } ]
  [ ABORT-LEVEL IS { 5 | dec1 } ] [ DELAY IS { 20 | dec5 } ].]
```

Description:

Automatically supervises the activity of a TDS application while it is running and detects errors which reduce the performance of the TDS application. These statements do not identify and eliminate programming problems which cause transactions to abort.

Supervision may be applied to session and/or to process activity from and above a level of error gravity specified.

Usage:

Used at the request of the Service Center. The dynamic supervision of the TDS application may cause the application to abort.

Supervision of session activity tracks sessions that keep data management resources while:

- the sessions are disconnected or interrupted (error level 3)
- the sessions are waiting for a system event (level 2)
- the resources are kept for a long time(level 1).

Process errors are either program or TPR errors. With supervision of process activity:

- all processes are blocked (error level 5)
- a process is blocked in the system code (level 4)
- a process is blocked in the user's code (level 3)
- a process is kept by a transaction for a long time (level 2).

Another way of supervising a TDS application is to use the [M] SUPERVISE_TDS command.

Parameters:

SESSION	Determines whether the TDS session is supervised (=1), or not supervised. *Default:* 0.
PROCESS	Determines whether the process activity is supervised. *Default:* 0. PROCESS overrides any previous value.
LEVEL	Indicates the level at which the system reacts: 1, 2, 3, 4 or 5. Each error level includes itself and all levels with a higher number. Level 3 includes Levels 3, 4 and 5. *Default:* 4 means that all errors of level 4 and 5 are checked and does not apply to Session activity.
Level 1	Each time a minor error is detected, a message (TX78) informs the master terminal operator. No dump is taken.
Level 2	Each time an error is detected, either a message informs the master terminal operator, or the transaction aborts with the return code RESVIOL.
Level 3	Each time a minor error is detected, either a message informs the master terminal operator, or the transaction aborts with the return code RESVIOL. Each time a major error is detected, TDS aborts and is restarted dynamically. A selective dump is taken.
Level 4	Each time a system error is detected, either a message informs the master terminal operator, or TDS aborts and is restarted dynamically. A selective dump is taken.
Level 5	Each time a serious system error is detected, TDS aborts and is restarted dynamically. A selective dump is taken.
ABORT-LEVEL	Indicates the level at which the system aborts. It accepts one of the following values: 2, 3, 4 or 5 and must be => *level*. The type of abort is automatically determined. *Default:* 5 and does not apply to Session activity.
DELAY	Defines the time in seconds between two supervision sessions. DELAY is a number ranging from 0 to 99999. The default value is 20 and the minimum value is 10.

Example:

To activate supervision both for sessions and for processes, specify the DYNAMIC-SUPERVISION clause twice, for example:

```
DYNAMIC-SUPERVISION FOR SESSION WITH LEVEL 2 ABORT-LEVEL IS 2
        DELAY IS 50.

DYNAMIC-SUPERVISION FOR PROCESS WITH LEVEL 2 ABORT-LEVEL IS 2
        DELAY IS 50.
```

NOTE:

Session and process supervision share the same LEVEL, ABORT-LEVEL, and DELAY values. If the DYNAMIC-SUPERVISION clause is specified twice, the LEVEL, ABORT-LEVEL, and DELAY values are those specified in the last DYNAMIC-SUPERVISION clause.

3.4.14 COMMON-STORAGE Clause

Syntax:

```
[ COMMON-STORAGE SIZE IS { 0 | commonsize }. ]
```

Description:

Specifies the size of the COMMON-STORAGE area shared by all transactions when no valid size is provided in the appropriate call procedure (size omitted or greater than the real common area size).

Usage:

- *commonsize* is the size in bytes of COMMON-STORAGE.
  commonsize < 65504 bytes (64 Kbytes - 32 bytes).
- The structure of COMMON-STORAGE can be defined in the WORKING-STORAGE clause of the INPUT-OUTPUT SECTION.
  In this case, the length of the structure describing the COMMON-STORAGE should be less than or equal to *commonsize*.
- COMMON-STORAGE is set to zero when the TDS session is started either for the first time since TP7GEN, or after a cold restart.
- When a TDS application is restarted after a shutdown or an abort, COMMON-STORAGE is set to the state at its last save in the previous session.

3.4.15 DEFAULT TRANSACTION-STORAGE Clause

Syntax:

```
[ DEFAULT TRANSACTION-STORAGE SIZE IS { 0 | deftxstorage }.]
```

Description:

Specifies the default size of TRANSACTION-STORAGE when no size is explicitly specified in the TRANSACTION SECTION.

Usage:

- *deftxstorage* specifies the size in bytes of the TRANSACTION-STORAGE to be assigned to transactions for which the TRANSACTION-STORAGE clause in the MESSAGE statement of the TRANSACTION SECTION has been omitted.

  Default: 0. TRANSACTION-STORAGE must be less than 65527 bytes.

- *deftxstorage* includes the size of the PRIVATE-STORAGE.

3.4.16 MAXIMUM TRANSACTION-STORAGE Clause

Syntax:

```
[MAXIMUM TRANSACTION-STORAGE SIZE IS maxtssz.]
```

Description:

Specifies the maximum size of TRANSACTION-STORAGE.

This size includes the space that can be allocated by the GETSP-U-CNTXT procedure.

Usage:

- This clause must be specified when the GETSP-U-CNTXT procedure is used by transactions.

  maxtssz must be greater than or equal to the greatest TRANSACTION-STORAGE size defined for a transaction.
- If the MAXIMUM PRIVATE-STORAGE clause is specified, *maxtssz* includes the size of the MAXIMUM PRIVATE-STORAGE else, it includes the size of the PRIVATE-STORAGE.
- The space that is defined by the TRANSACTION-STORAGE size is named the STATIC subset. The space subsets that are dynamically allocated by GETSP-U-CNTXT calls are named the DYNAMIC subsets.
- If *maxtssz* is lower than the Transaction Storage size needed by the system transactions, this value is increased.

3.4.17 PRIVATE-STORAGE Clause

Syntax:

```
[ PRIVATE-STORAGE SIZE IS { 0 | privatesize }.]
```

Description:

Defines the size of PRIVATE-STORAGE exclusive to each TDS user.

PRIVATE-STORAGE can be used to store statistics and control information for a given user. There is one copy of PRIVATE-STORAGE per user (correspondent). PRIVATE-STORAGE structure is defined at the beginning of the TRANSACTION-STORAGE, see Paragraph 3.6.22.

PRIVATE-STORAGE can be declared separately from TRANSACTION-STORAGE. For separate access, see *TDS COBOL Programmer's Guide*.

Usage:

- *privatesize* specifies the size in bytes of PRIVATE-STORAGE.

  privatesize is less than deftxstorage and is also less than the minimum size of the TRANSACTION-STORAGE specified in the TRANSACTION SECTION.

- PRIVATE-STORAGE is set to zero at the start of the user session.

  As long as a context is kept for a user (that is, until the user logs off normally), PRIVATE-STORAGE is passed between each transaction of this particular user without being modified by TDS.

- With a FOR INQUIRY transaction, the PRIVATE-STORAGE area is:
  - set to zero at the start of the transaction
  - and at the end of the transaction, restored to its previous contents before the transaction started.

3.4.18 MAXIMUM PRIVATE-STORAGE Clause

Syntax:

```
[ MAXIMUM PRIVATE-STORAGE SIZE IS maxpssz.]
```

Description:

Specifies the maximum size of the PRIVATE-STORAGE.

This size includes the space that can be allocated by the GETSP-U-CNTXT procedure.

Usage:

- This clause must be specified when the GETSP-U-CNTXT procedure is used by transactions for the PRIVATE-STORAGE.
- *maxpssz* must be greater than the PRIVATE-STORAGE size.
- The space that is defined by the PRIVATE-STORAGE size is named the STATIC subset. The space subsets that are dynamically allocated by the GETSP-U-CNTXT calls are named the DYNAMIC subsets.

3.4.19 MESSAGE-LENGTH Clause

Syntax:

MESSAGE-LENGTH IS { **1024** | *msglength* } MAXIMUM.

Description:

Specifies the maximum message length.

Usage:

- *msglength* expresses the size of the largest received message or the largest output message quarantine unit. This area is saved in the swap file and thus has an impact on the swap file size. See Paragraph 7.6.2.1 *Computing the Size of the Swap File Buffer*.

  1024 <= *msglength* <= 32767 bytes. *Default:* 1,024 bytes. For values less than 1024 or greater than 32767, the default is taken.

- If the ADVANCING phrase is used in the SEND statements, msglength must include 3 characters for page throw and 2 for each new line feed.

  If the ADVANCING phrase is absent, msglength need include only 2 characters for an implicit new line feed.

- If the PASS-THROUGH Facility is used, the value specified in the MESSAGE-LENGTH clause must be at least 4,096 bytes.

3.4.20 FORMATTED MESSAGE-LENGTH Clause

Syntax:

```
[ FORMATTED MESSAGE-LENGTH IS { 10033 | formsglength } MAXIMUM.]
```

Description:

Defines the maximum size in bytes for saving the FORMS control structures in the TDS swap file. It is applicable only if the USE FORMS clause is specified.

Usage:

- *formsglength* is calculated as follows:

```
      107 * (Max number of simultaneously active forms per user)
       28 * (Max number of variable fields)
  + 2116
```

- *Default:* 10033 bytes being 7 simultaneously active forms of 256 variable fields.
- 32767 bytes => *formsglength* => 512 bytes. If the maximum is exceeded, the default is taken.
- If FORMS uses WINDOWING, add 512 bytes.
- If TDS is secured (SECUR'ACCESS is used), add 3 Kbytes.

3.4.21 USER-JOURNAL Clause

Syntax:

```
[ USER-JOURNAL [ WITH [ INPUT ] MESSAGE LOGGING ].]
```

Description:

Allows User Journal to be used and defines what messages are to be logged.

Usage:

- MESSAGE LOGGING logs both input and output messages.
- INPUT LOGGING logs input messages but only lengths of output messages.
- The User Journal is part of the After Journal. To retrieve records from the User Journal, execute the DUMPJRNL utility.
- User journalization has no effect on TDS restart/recovery capabilities using the Before and After Journals.
- Service messages (sent by TDS) are not all recorded. See Appendix C for a description of the records logged in the User Journal.

3.4.22 TPR-TIME-LIMIT Clause

Syntax:

TPR-TIME-LIMIT IS *timelimit* MSEC.

Description:

Defines the maximum CPU time in milliseconds that a TPR may take to execute.

Usage:

- *timelimit* prevents endless looping within a TPR.
- *timelimit* is up to 8 decimal digits. 64 <= *timelimit* <= 99999999 milliseconds. *Default:* 64. For values less than 64, the default is taken.
- If *timelimit* is exceeded, the transaction aborts with RC=TIMELIM and a message is sent to the user concerned.
- *timelimit* can be modified dynamically during the TDS session by the [M] MODIFY_TDS MAXCPU command.

3.4.23 MAXIMUM NUMBER OF TPR WITH NO UNMAPPING Clause

Syntax:

[MAXIMUM NUMBER OF TPR WITH NO UNMAPPING IS *maxunmap*.]

Description:

Places an upper limit on the number of TPRs executed consecutively without unmapping.

Usage:

maxunmap is the maximum number of TPRs executed consecutively without unmapping (keeping the process without any commitment). Its value must be between 1 and 32767.

When maxunmap is reached, the corresponding transaction is aborted with the return code TP7,COUNTLIM (if there is an ON-ABORT-TPR, it is executed).

3.4.24 DEFAULT ABORT TPR-NAME Clause

Syntax:

[DEFAULT ABORT TPR-NAME [IS] *tprname*.]

Description:

Specifies a default abort tpr to be executed in case of transaction abort.

Usage:

When a transaction aborts, the following processes occur:

- If ON-ABORT-TPR field of TDS-STORAGE is not blank, its contents are taken as NEXT-TPR
- If ON-ABORT-TPR field is blank, TDS looks if DEFAULT ABORT TPR-NAME has been specified:
  - if yes, *default abort tpr-name* is taken as NEXT-TPR
  - if no, TDS looks if USE CONVENTIONAL-ABORT-TPR has been specified:
    if yes, ABT-*tdsname* is taken as NEXT-TPR, see Paragraph 3.4.37
    if no, transaction terminates with abort message.

tprname is up to 12 alphanumeric characters identifying the TPR loaded at TDS execution.

3.4.25 SYNCHRONOUS SEND WAIT-TIME Clause

Syntax:

```
[MAXIMUM WAIT-TIME UPON SYNCHRONOUS SEND IS nmin.]
```

Description:

This clause, which relates to synchronous SENDs and synchronous outward CONNECTIONS, defines a waiting time in order to avoid TPR blocking within a TDS process.

Recalls:

When, in a single TPR, a SEND (or an output FORMS verb) with a level of EMI is followed by another SEND, the following occurs:

- the first message is transmitted to the network.
- the TPR is blocked within a TDS process until TDS receives acknowledgement from the network for this message.

NOTE:
This method of programming SEND verbs should not be used because it generates poor TDS performance and may provoke widespread TDS deadlock.

Synchronous outward CONNECTIONs take place when:

- a SPAWNing verb requests connection of a correspondent.
- an OCPOOL command related to an XCP1 correspondent is executed.
- an explicit SEND requests connection of a correspondent.

The WAIT-TIME *nmin* is given in minutes from 1 to 15.

There is no default value: if the value is wrong, the clause is ignored and a severity 2 is set at TDS generation time and written in the jor.

When the WAIT-TIME *nmin* value is exhausted during a SEND, a warning message MV38 is sent to the TDS master and the GCOS operator. The terminal is then disconnected and the user is in frozen state.

When the WAIT-TIME *mnin* value is exhausted during an outward CONNECTION, a warning message MV42 is sent to the TDS master and the GCOS operator. The connection fails and an abormal status is returned:

- status 3 upon a "SPAWN" call, status 51 (connection) or 52 (reconnection) upon a call "SPAWNTX".
- status 20 upon an explicit SEND
- status 24 (connection) or 25 (reconnection) upon an OCPOOL command related to an XCP1 correspondent.

3.4.26 IDLE-TIME Clause

Syntax:

```
[ MAXIMUM IDLE-TIME IS { 0 | idletime }.]
```

Description:

Maintains system security by automatically logging off a user if after a transaction is completed, another transaction is not begun before the specified time period elapses.

Usage:

- *idletime*, expressed in seconds, is a limit on elapsed inactive time, that is the time elapsed between 2 transactions.
- Idle-time can be modified dynamically during the TDS session by the [M] MODIFY_TDS MAXIDLETIME command.
- If *idletime* is exceeded, the system logs off the user.

  Default: 0 meaning no limit to *idletime*.

NOTE:

Do not confuse IDLE-TIME with WAIT-TIME which is the maximum think time between TPRs of the *same* transaction. See WAIT-TIME in the *TDS COBOL Programmer's Guide*.

3.4.27 MASTER MAILBOX Clause

Syntax:

[MASTER MAILBOX IS *mbxname* [TYPE IS XCP1].]

Description:

Declares the terminal connecting to the specified mailbox as being the master terminal. This mailbox is created at TDS startup.

If *master mailbox* is omitted in TP7GEN and in NETGEN, the submitter of the TDS job is the master operator. If a *master mailbox* is defined, no activity is possible until the master operator connects to this mailbox. The master operator must logon so that the startup is executed. Then the master operator can logoff without affecting the current session.

master mailbox should be declared in NETGEN. If MASTER MAILBOX is specified in STDS, a message indicating that the clause is obsolete appears in the TP7GEN report.

The master operator must belong to a project with the authority code zero set to 1.

Usage:

- *mbxname* is up to 8 alphanumeric characters and must be cataloged as an application under APPLIST in the site catalog with at least the authority code zero.
- Only one master operator can exist.
- Only XCP1 exchanges occur between TDS and an XCP1 master mailbox. In this case, the master terminal cannot be reconfigured on a non-XCP1 terminal.
- When a TDS application is started with a master mailbox, the TX50 message appears on the submitter's console:
  TDS: *tds* STARTED WITH MASTER MAILBOX: *mbx*
- When a TDS application is started with a master mailbox, messages appear on the submitter's console without their prefixes (see example with message TX50 above).

3.4.28 NUMBER OF XCP1 TRANSACTION AUXILIARY SESSIONS Clause

Syntax:

```
[ NUMBER OF XCP1 TRANSACTION AUXILIARY SESSIONS IS { numaux }.]
```

Description:

Defines the maximum XCP1 auxiliary sessions that a transaction may allocate.

Usage:

0 <= *numaux* <= 10. Mandatory if the TDS workstation is declared at NETGEN with a XCP1 mailbox.

3.4.29 SELECT XCP2-SYNCPEVT Clause

Syntax:

```
[ SELECT XCP2-SYNCPEVT eventid [,eventid...].]
```

Description:

Specifies events which start the SYNCPEVT transaction. This clause is useful only when the XCP2 distributed commitment service is used.

Usage:

eventid ranges from 1 through 10. *Default: no event*. See *CPI-C/XCP2 User's Guide*.

3.4.30 MAXIMUM XCP2-WAITTIME Clause

Syntax:

```
[ MAXIMUM XCP2-WAITTIME IS  { 0 | maxwait }.]
```

Description:

Specifies the maximum wait-time in seconds, for the completion of a XCP2 call statement. This clause is useful only when the XCP2 service is used.

Usage:

0 <= *maxwait* <= 999. *Default:* 0 meaning no time limit.

3.4.31 MAXIMUM DTP-WAITTIME Clause

Syntax:

```
[ MAXIMUM DTP-WAITTIME IS maxdtpwait.]
```

Description:

Specifies the maximum wait-time for Distributed Transaction Processing. Pertains to DTP / XA TDS function. See *XA TDS User's Guide*.

3.4.32 WATCHTIME Clause

Syntax:

```
[ WATCHTIME[IS]watchtime[EXCEPT[FOR]{TERMINAL|TCP-IP}].]
```

Description:

Specifies the time (in seconds) for the network to answer to TDS before the session is abnormally disconnected.

Watchtime is in the range 1 to 9999 and is expressed in seconds. If the keyword EXCEPT is omitted, the clause is available for terminal and TCP-IP sessions. The keyword EXCEPT allows to discard either terminal or TCP-IP sessions from the WATCHTIME generation clause (not from the WATCH-TIME field of the TDS-STORAGE set in TPR). The clause is taken into account only when the WATCH-TIME field has not been set in TPR.

On SEND EMI, the clause is available for terminal session (until CREDIT is coming back) but is ignored for TCP-IP session (because nothing is waited). Service message 44 (TX88) is prompted at master terminal.

See WATCH-TIME field of TDS-STORAGE description in TDS Programmer's Guide 47 A2 33UT for more explanation.

Usage:

0 < *watchtime* <= 9999.

3.4.33 HOST GATEWAY Clause

Syntax:

```
[ HOST GATEWAY IS gatewayname.]
```

Description:

Pertains to TDS-/HOST7 function. See *Setting Up and Using /HOST7 User's Manual*.

3.4.34 XA-RESYNC-DELAY Clause

Syntax:

```
[ XA-RESYNC-DELAY IS {dec4 | 300}.]
```

Description:

If a failure disrupts the XA commitment protocol, the commitment unit concerned is said to be desynchronized.

TDS-XA resynchronizes the Commitment protocol asynchronously with the XA Resource Managers through ORACLE7/TDS-XA and completes the commitment unit when the communication is recovered. This may take some time and several attempts may be required.

The XA-RESYNC-DELAY clause specifies the time between two attempts of an XA commitment unit resynchronization.

The time is expressed in seconds. It must be set to a value that corresponds to the duration of recovery and restart of the XA UNIX components.

The default value is 300 seconds, the minimum is 60 seconds, and the maximum is 9999 seconds.

3.4.35 TCP-IP PROTOCOL Clause

Syntax:

```
[ TCP-IP PROTOCOL [ USED ] [ WITH { OPEN7 | GXTI } ].]
```

Description:

Pertains to TDS-TCP/IP function. See *TDS-TCP-IP User's Guide*.

3.4.36 USE LAST MESSAGE Clause

Syntax:

```
[ USE { "?" | "lastmsg" } FOR LAST MESSAGE.]
```

Description:

Specifies the character used to display anew the last message.

Usage:

- *lastmsg* specifies the single character used when the terminal is either in command mode, or in transaction mode to request that TDS display the last output message. When the terminal is in transaction mode, this character must be the only character transmitted. See Section on Terminal Operations in the *TDS COBOL Programmer's Guide*.
- *Default:* "?"

3.4.37 USE TRANSACTION-MENU Clause

Syntax:

```
[ USE { "/" | "menu" } FOR TRANSACTION-MENU.]
```

Description:

Specifies the character used to display the menu of available transactions.

Usage:

- *menu* is up to 8 alphanumeric characters. *Default:* "/"
- To standardize with GCL, declare: USE "?" FOR TRANSACTION-MENU. In this case, another value must then be specified for *lastmsg* in USE LAST MESSAGE.

3.4.38 USE NOPWCHK-ALLOWED Clause

Syntax:

```
[ USE NOPWCHK-ALLOWED.]
```

Description:

Allows inward connections to TDS by **applications** using the terminal protocol and with "**no password control**" mechanism even if the option CHKPW (Check Password) of the SECOPT configuration is set on the site. This clause has no effect if the option CHKPW is not set on the site.

Usage:

This function may be used by **applications** which have their own security mechanism and that attempt to connect to a TDS on a site protected by the CHKPW option.

To limit the "**No password control"** mechanism to a particular TDS:

- Set the option CHKPW on the site to check passwords, even if a remote application requests a connection with "**no password control**".
- To disable password control in a particular TDS, use the NOPWCHK-ALLOWED Clause in its generation. CHKPW will be ignored for this TDS.

NOTE:
This function is limited to **applications** such as PASSWORD7 or private PASSTHRU. It is not available for a user connection from a terminal.

3.4.39 USE FREE-ACCESS-TDS Clause

Syntax:

```
[ USE FREE-ACCESS-TDS.]
```

Description:

Allows all the inward connections to TDS which use the terminal protocol without catalog control checks on USER, PROJECT, BILLING and PASSWORD. When FREE-ACCESS-TDS is used, the following rules are applied:

- The user name is the one which is received from the connection request.
- There is no password checking nor access control to the TDS application.
- The project name is "FREE_ACCESS"
- The billing name is "FREE_ACCESS"
- Authority codes are "7FFFFFFF" (all rights except the master's).

Usage:

This function may be used when a TDS needs no security control or its own security controls are performed at application or at network level.

It avoids numerous unnecessary retries in the catalog, and avoids accessing the catalog at connection time.

NOTE:

User catalog attributes are needed to access another application using the passthrough facility.

3.4.40 USE PROCEDURE FOR TRANSACTION INITIALIZATION Clause

Syntax:

```
[ USE procedure FOR TRANSACTION INITIALIZATION.]
```

Description:

Specifies the name of the procedure to be used before a transaction is started. The procedure decodes the message for starting the transaction and returns the transaction identifier to TDS.

Usage:

- *procedure* identifies a unique subroutine.
- The data segments of these subroutines are not refreshed each time the subroutine is called. Therefore the initial values in the WORKING-STORAGE of a subroutine should not be modified. To modify these values, re-initialize them at each call so that the WORKING-STORAGE can be reused if the TPR aborts.
- A *procedure* containing IDS/II verbs can be only a secondary IDS/II program specifying DB-DESCRIPTIONS in the LINKAGE SECTION.
- These USE procedures are supposed to be compiled in a *culib* CU library. Before generating the TDS, make this library available to TP7GEN. To do this, specify:

  ```
  LMN CU LIB=TEMP, COM='IL1 culib;MV IL1:*;';
  ```

  in the job used to run TP7GEN.
- When using the XCP2 service, this *procedure* is not called before an XCP2 transaction is started, so a message received via an XCP2 session is not passed to this procedure.
- The subroutine must neither address controlled files nor the IDS/II database.

3.4.41 USE FORMS Clause

Syntax:

```
[ USE FORMS [ NO IMPLICIT RELEASE ].]
```

Description:

Creates the control structures used by FORMS. This clause must be specified if FORMS is to be used in the TDS application.

Usage:

- NO IMPLICIT RELEASE allows a form to remain active between two transactions instead of re-activating it at each new transaction. NO IMPLICIT RELEASE does not apply to transactions for which FORM IS RESETFRM is specified.
- If NO IMPLICIT RELEASE clause is omitted, active forms are released implicitly by the system at the end of each transaction.

3.4.42 USE TERMINAL-ADAPTER Clause

Syntax:

```
[ USE TERMINAL-ADAPTER.]
```

Description:

Requests that the Terminal Adapter be used. Transactions, including the LOGON transaction, can issue specific call statements for managing the user's profile.

Usage:

- This clause creates a user profile, adapts messages to terminals supported by TDS, and controls structures whenever a user logs onto the TDS application. These structures remain present even if the Terminal Adapter is temporarily turned off for a user's terminal during the TDS session.
- These structures require 28 additional bytes per user in the swap file.

Effects on Other Clauses

Control Characters: if a TDS application works with a range of different terminal types, do not use control characters (particularly those defined in the SPECIAL-CHAR clause and the CONSTANT STORAGE) because they can be misinterpreted by some terminals.

If control characters are used, they are either replaced by another character, or sent to the terminal, according to the value of the user-profile variable INVCHAR.

Message Length: to use the TERMINAL ADAPTER, increase the maximum size defined in the MESSAGE LENGTH clause by 256 bytes as a provision for the Terminal Adapter.

Formatted Message Length: the TERMINAL ADAPTER occupies 512 bytes of the area whose size is defined in the FORMATTED MESSAGE LENGTH clause.

If this clause is not present, a default value of 512 bytes will be given (which is overlaid by a higher default value if the USE FORMS clause has been specified).

Master Mailbox Clause: it is strongly recommended that the master terminal not be in Terminal Adapter presentation because incoming messages overwrite the last message received on the 25th line.

Service and TDSTX Messages: Default 0D25 for TRAILER in SERVICE MESSAGE is ignored when messages are sent to a user with the Terminal Adapter presentation.

For a description of the CALL "RDPROF" and CALL "MDPROF" statements used with the Terminal Adapter, see the *TDS COBOL Programmer's Guide*.

3.4.43 USE ORACLE Clause

Syntax 1:

```
[ USE ORACLE.]
```

Syntax 2:

```
[ USE ORACLE-BASE.]
```

Syntax 3:

```
[ USE ORACLE-BASE-databasename.]
```

Description:

Identifies the database used by TDS.

Usage:

See *ORACLE-V6/TDS or ORACLE-V7/TDS User's Guide* depending on the Oracle version.

For Syntax 1 and Syntax 2:

The ORACLE database to be connected to is that specified in the database connection string in the TPR or, by default, is the same as the TDS name.

For Syntax 3:

The ORACLE database to be connected to is that specified in the database connection string in the TPR or, by default, is the one specified in BASE-databasename.

databasename must be less than or equal to 50 characters.

3.4.44 USE STARTUP Clause

Syntax:

```
      { STARTUP-DETAILED-REPORT }
[ USE { STARTUP-STANDARD-REPORT }.]
      { STARTUP-SUMMARY-REPORT  }
```

Description:

Controls the types of message to be delivered during the TDS application startup. These clauses are mutually exclusive.

Usage:

- DETAILED *(default)* displays all files, opened or not opened.
- STANDARD displays a message for each file that cannot be opened.
- SUMMARY prevents any message at file opening to be displayed.

The USE STARTUP clause does not apply to the TDS JOR. Whatever value specified, all messages will be put in the JOR.

3.4.45 USE SHORT-STATISTICS Clause

Syntax:

```
[ USE SHORT-STATISTICS.]
```

Description:

When this clause is present, statistics concerning users are not provided in the TDS General Statistics produced at shutdown time. See Paragraph 5.4.2.1 *Sample TDS Statistics Report* and Appendix D.

3.4.46 USE CONVENTIONAL-ABORT-TPR Clause

Syntax:

```
[ USE CONVENTIONAL-ABORT-TPR.]
```

Description:

Specifies a conventional abort TPR called ABT-*tdsname* to be executed in case of transaction abort.

Usage:

When a transaction aborts, the specified *conventional-abort-tpr* routine named ABT-*tdsname* is run when:

- no other TPR name is specified in the ON-ABORT-TPR field of the TDS-STORAGE
- no DEFAULT ABORT TPR-NAME Clause is defined in STDS. See Paragraph 3.4.21.

If these conditions are satisfied but the ABT-*tdsname* does not exist or is not loaded, the transaction just aborts. Otherwise the ABT-*tdsname* is started and TDS loads its name in the ON-ABORT-TPR field of the TDS-STORAGE.

3.4.47 USE M-MASTER-CONVENTION Clause

Syntax:

```
[ USE M-MASTER-CONVENTION.]
```

Description:

Names the master terminal. Ineffective when MASTER MAILBOX is specified.

Usage:

- If specified, the TDS master terminal name is: `M_____`.
- If omitted, the name of the master terminal is the terminal submitting the TDS job.

3.4.48 USE SPAWNED-TX-MESSAGE-LOGGING Clause

Syntax:

```
[ USE SPAWNED-TX-MESSAGE-LOGGING.]
```

Description:

Allows spawned transactions to produce 01 records in the User Journal.

3.4.49 USE PASSIVE-SPAWN-CHECK Clause

Syntax:

```
[ USE PASSIVE-SPAWN-CHECK.]
```

Description:

Allows spawning only towards passive terminal correspondents.

Usage:

If this clause is present, no spawning is possible towards an active correspondent.

3.4.50 USE LARGE-STACK-SEGMENT Clause

Syntax:

```
[ USE LARGE-STACK-SEGMENT.]
```

Description:

Reserves one of the type 3 large segments for ring 3 stacks.

NOTE:
Mandatory when compiling TPRs with COBOL 85 for using a stack > 64Kbytes.

3.4.51 USE TYPE3-LARGE-SEGMENT Clause

Syntax:

```
[ USE TYPE3-LARGE-SEGMENT.]
```

Description:

To define eight type 3 large segments and four type 2 large segments.

Twelve large segments are usable in a process group. The default for a TDS load module is that eight of the large segments are defined as type 2 and the other four are defined as type 3. If the clause USE LARGE-STACK-SEGMENT (see previous paragraph) is used, one of the four type 3 segments is reserved for ring 3 stacks, leaving only three type 3 large segments available. The clause USE TYPE3-LARGE-SEGMENT ensures that sufficient type 3 segments are available by increasing the total number to eight.

NOTE:

When the clause USE TYPE3-LARGE-SEGMENT is used, it is at the cost of having only four of the segments defined as type 2. Some products such as ORACLE or XCP2, may lack type 2 segments. If the user's procedures contain an object code size greater than 64 KB and are linked with the TDS load module via the USE PROCEDURE clause, TDS itself may lack type 2 large segments and abort, giving a MU02 message at initialization phase.

3.4.52 USE DEVICE-HEADER-LOGGING Clause

Syntax:

```
[ USE DEVICE-HEADER-LOGGING.]
```

Description:

Logs a Device Header in the type 10 record of the User Journal for every reply message occurrence.

Usage:

This clause applies only if the USER-JOURNAL clause is specified. See Appendix C.

3.4.53 USE IDS-SUB-SCHEMA Clause

Syntax:

```
[ USE IDS-SUB-SCHEMA.]
```

Usage:

Specify this clause to use the full IDS/II.

3.4.54 USE CONNECTION-LOGGING Clause

Syntax:

```
[ USE CONNECTION-LOGGING.]
```

Description:

Logs connection/disconnection activity in the User Journal records (types 12, 13, 14, and 15 for connections, and types 16 and 17 for disconnections).

Usage:

This clause applies only if the USER-JOURNAL clause is specified. See Appendix C.

3.4.55 USE TILS-SIMULATION Clause

Syntax:

```
[ USE TILS-SIMULATION.]
```

Description:

Duplicates record types 1, 2, 4, 9, and 10 in a new record, 11, in the User Journal.

Usage:

Applies only if USER-JOURNAL is specified. See Appendix C.

3.4.56 USE NO_SCROLLING_REQUEST Clause

Syntax:

```
[ USE NO_SCROLLING_REQUEST.]
```

Description:

Suppresses the Scrolling request on when information from a list command cannot fit on a screen.

Usage:

To have automatic scrolling without operator intervention.

3.4.57 USE EDIT-EXCEPTION Clause

Syntax:

```
[ USE EDIT-EXCEPTION.]
```

Description:

Prints a warning message in the JOR when an exception occurs in TPR processing.

Usage:

For debugging and modifying TPRs written in both COBOL and C language.

Output:

On a TPR exception, the following information is displayed:

```
WARNING EXCEPTION cc-tt: message-text IN TASK name.nnn
        AT ADDRESS stn.ste.sra OF TPR tprname IN SM smname
        FROM LIB smlibname
```

where:

cc	class of the exception, in hexadecimal
tt	type of the exception, in hexadecimal
message-text	explains the error
name	task name from the $LINKER listing
nnn	task occurrence number, in decimal
stn	segment table number
ste	segment table entry
sra	segment relative address
tprname	the TPR in which the error occurred
smname	the sharable module containing the TPR
smlibname	*the name of the library from which the SM is taken.*

If an anomaly occurs outside a TPR such as in a USE procedure, the message displayed is:

```
WARNING EXCEPTION cc-tt: message-text IN TASK name.nnn
        AT ADDRESS stn.ste.sra OF USE PROCEDURE.
```

NOTE:

Because *edit exception* consumes CPU time, only use it to debug TPRs.

List of exceptions processed by the clause:

Class	Type	Name
0F	0	Decimal data overflow
0F	1	Decimal divide
09	1	Illegal decimal data
10	0	Floating point exponent overflow
10	1	Floating point exponent underflow
10	2	Floating point divide
11	11	Fixed point data overflow
11	11	Fixed point divide
11	11	Out of array range
11	11	Fixed point conversation overflow

Except for the exceptions 09-1 (Class-Type) and 11-2, using the EDIT-EXCEPTION clause may modify the behavior of the system.

When the exceptions 09-01 (ill dec data) and 11-02 (out of array range) lead to the abort of the TPR, the new warning message WARNING EXCEPTION.... is printed in the jor in addition to the usual fatal abort message FATAL EX01.....

3.4.58 USE WAIT_VIRTUAL_FOR_SHUTDOWN Clause

Syntax:

```
[ USE WAIT_VIRTUAL_FOR_SHUTDOWN.]
```

Description:

Requests TDS to wait for the end of running transactions on virtual correspondents (DUMMY and virtual family) before executing a weak shutdown (TTDS WEAK).

Usage:

- When you want TDS SHUTDOWN to wait for transactions running on virtual correspondents to complete.
- **Warning:** If the WAIT_VIRTUAL_FOR_SHUTDOWN is present, and a transaction on a dummy correspondent nevers stops, then the SHUTDOWN (without STRONG) does not take place (since such a transaction is not cancelled).
- CTC STRONG on a virtual correspondent (not DUMMY) cancels the running transaction but does not logout the correspondent from TDS.
- TTDS STRONG does not wait for anybody.

3.4.59 USE DISPLAY_IN_JOR Clause

Syntax:

```
[ USE DISPLAY_IN_JOR.]
```

Description:

Allows messages generated by COBOL "DISPLAY" verbs:

- DISPLAY "xx", and
- DISPLAY "xx" UPON SYSOUT,

to appear in the JOR.

The messages gives the following information:

- TIME,
- the word "DISPLAY",
- the TPR name.

Usage:

If the session is in TRACE mode, messages are still written in *tdsname*.DEBUG.

3.4.60 USE TWRITER-NUMSF-xxx Clause

Syntax:

[USE TWRITER-NUMSF-*xxx*.]

Description:

Provides the maximum number of simultaneous reports to be created by GTWriter.

Usage:

- *xxx* is a numeric value where 1 < *xxx* < 200. *Default: nsimu* parameter of the SIMULTANEITY Clause, see Paragraph 3.4.4
- *xxx* determines the maximum number of NUMSF subfiles in the SYS.TW.OUT queued file that can be simultaneously opened
- For conversations without commitment (unmapped commitments), NUMSF should be greater than the number of TDS simultaneities
- A report is created inside a commitment unit by a call to "H_TW_USTART".

3.4.61 USE XCP1-CLOSEPOOL-STRONG Clause

Syntax:

```
[ USE XCP1-CLOSEPOOL-STRONG.]
```

Description:

From TS7356, this optional clause modifies the behavior of the "CLCPOOL" or "CTC" commands executed with the STRONG option applied to XCP1 correspondents.

Usage:

- If the USE XCP1-CLOSEPOOL-STRONG clause is not specified, then the CLCPOOL STRONG and CTC STRONG commands behave as before TS7356, that is:
  - if all sessions are free (not allocated to a transaction), the pool is closed immediately,
  - if some sessions of the pool are allocated to transactions, the pool is closed at the end of the last transaction.
- If the USE XCP1-CLOSEPOOL-STRONG clause is specified, then the CLCPOOL STRONG and CTC STRONG commands behave as follows:
  - if all sessions are free (not allocated to a transaction), the pool is closed immediately (same behavior as above),
  - if some sessions of the pool are allocated to transactions, these transactions will be aborted with the return code "OPERATOR". The pool will actually be closed at the end of the last transaction.

3.4.62 USE XCP1-ALNTC-PRIMARY-ONLY Clause

Syntax:

```
[ USE XCP1-ALNTC-PRIMARY-ONLY.]
```

Description:

Since TS 7356 this optional clause modifies the behavior of the ALNTC master command ALLOW_NEW_TDS_COR towards XCP1 correspondents.

Usage:

- If the USE XCP1-ALNTC-PRIMARY-ONLY clause is not specified, then the ALNTC command behaves as before TS 7356, that is:

  the XCP1 sessions related to both primary and secondary correspondents are reconnected.

- If the USE XCP1-ALNTC-PRIMARY-ONLY clause is specified, then the ALNTC command behaves as follows:

  only the XCP1 sessions related to primary correspondents are reconnected.

3.4.63 USE JOR-STD-FORMAT Clause

Syntax:

```
[ USE JOR-STD-FORMAT. ]
```

Description:

This clause allows you to obtain additional information from the jor concerning warning and fatal messages sent during execution of a TPR or during the initialization routine. Time, task name and its occurrence, process indent, correspondent and TPR name are added to a line preceeding the message.

Example of a message in jor with the JOR-STD-FORMAT clause:

```
08:31:26 TASK=TDSEXEC OCCURRENCE=2 CORRESPONDENT=my_corresp TPR=my_tpr
WARNING EX02.DECIMAL DIVIDE IN TASK TDSEXEC.2 AT ADDRESS...
```

3.4.64 USE NO-PROP-MSG-IN-JOR Clause

Syntax:

```
[ USE NO-PROP-MSG-IN-JOR. ]
```

Description:

This clause avoids the reporting in TDS JOR of Master commands issued from a programmed operator. The Master commands issued from the Master terminal are still reported.

3.4.65 USE M-TX-ABORT-ON-BREAK Clause

Syntax:

```
[ USE M-TX-ABORT-ON-BREAK. ]
```

Description:

This clause enables a break message, coming from the TDS master operator, to abort the Master transaction launched by the current master command.

Some TDS commands such as CLTF or MDTDS SMLIB=... may be waiting a long time for their execution; CLTF command waits for the termination of the last commitment unit using the file, MDTDS SMLIB=... waits for the termination of all the commitment units in progress.

If this clause is present, the Master transaction launched by the current TDS master command, which is not completed, can be aborted entering a break (return code = break). The break transaction, if exists, is not activated.

If the master operator is in command mode (no transaction running) or processing a user transaction the break transaction, if exist, is activated.

3.4.66 USE LINK-WITH-AUTOBIND Clause

Syntax:

```
[USE LINK-WITH-AUTOBIND.]
```

Description:

This optional clause enables the TDS generation step to create a command AUTOBIND=(FULL=90), in the TP7LINKTDS commands.

This LINKER command makes a binding operation during the linkage.

When several USE procedures are used, the result is the decreasing of the number of type 2 segments because several compile units may be linked in a minimum number of small segments (each output segment is filled with 90% of the 64 K size).

3.4.67 USE TDS-SPECIALIZED-WEB7 Clause

Syntax:

```
[USE TDS-SPECIALIZED-WEB7.]
```

Description:

This clause is available from TS 9662.

If this clause is present, the TDS load module created by the TDS generation, will be executed on a Customer Dedicated Processor (CDP), or on a PSP or an X-HRP (if the machine has such processors).

Such a TDS only accepts connections from the Web via MainWay.

Such a TDS cannot use the XCP1, XCP2, or TCP/IP functionality.

Connections of TM correspondents are not allowed, except for the master terminal, when the TDS is a TDS with a master mailbox. Other TM connections are closed with a code 003D in TS 9662 or a code 003C in the subsequent TSs.

Without this new clause, the TDS can also accept connections from the Web, but:

- It can also accept connections from other types of terminal.
- It cannot run on the specialized processors.

3.4.68 USE WATCHDOG-ON-ASYNCHRONOUS-SEND-EMI Clause

Syntax:

```
[USE WATCHDOG-ON-ASYNCHRONOUS-SEND-EMI.]
```

Description:

This clause is available since TS 9968.

This clause enables TDS to detect a non-response from the network on asynchronous SEND EMI.

For each TPR ending with a SEND EMI, TDS sets a TIMER if the field TD-HWATCH is zero (i.e., if the watchdog was not specified at TPR level). The value of this timer is the TD-WAIT-TIME value increased by 2 minutes.

If the network doesn't answer before the timer is exhausted, a disconnection occurs with reason 45 and the message TX88 (TDS Service Message 44) is sent to the master terminal (if it is not a master mailbox).

In fact, the WATCHDOG acts as the WATCH-TIME for SEND EMI.

See description of the field TD-WATCH-TIME of the TDS-STORAGE in the TDS Programmer's Guide (47 A2 33UT) for more explanation.

3.4.69 USE Procedure Clause

Syntax:

[USE *procedure*.]

Description:

Names a subroutine that can be called in any transaction. These subroutines can be linked with the TDS rather than stored in the SM libraries.

Usage:

- Data segments of such a subroutine are not refreshed each time it is called. Values in its WORKING-STORAGE if modified, must be reset on each call. To do so, use COBOL-85.
- A *procedure* containing IDS/II verbs can be only a secondary IDS/II program specifying DB-DESCRIPTIONS in the LINKAGE SECTION.
- Resubmit the JCL to link the procedure to the TDS load module before running TP7GEN.

Limits:

- The maximum length of the *procedure* name is 12 characters.
- The maximum number USE *procedure* clauses is 255.

Usage of binded CUs:

When CUs are binded, there are two ways to define them in STDS, either:

- declare only one "USE procedure" for the resulting binded CU and make sure that CUs used as input for the binding do not exist in or are suppressed from the CULIB used to link TPRs; or
- declare one "USE procedure" for each CU used as input for the binding and one "USE procedure" for the resulting binded CU. In this case, make sure that the CUs used as input for the binding do not exist in or are suppressed from the CULIB used to link the TDS.
- From TS 9662, an automatic bind can be requested using the clause “USE LINK-WITH-AUTOBIND”. Please refer to the description of this clause.

3.4.70 SERVICE-MESSAGE Clause

Syntax 1:

```
[ SERVICE-MESSAGE [ HEADER IS "servheader" ]
[ TRAILER IS { "0D25" | "servtrailer" }][ IMMEDIATE DELIVERY ].]
```

Syntax 2:

```
[ SERVICE-MESSAGE msgnumber IS "servicemsg".]
```

Description:

Customizes standard output TDS messages to user specifications, namely:

- Service Messages independent of transactions executed for a terminal operator
- Messages sent by system transactions such as BYE, TRACE, or M.
  - Syntax 1 is used to place prefixes before and suffixes after TDS service messages.
  - Syntax 2 is used to modify service messages.

Usage:

- *servheader* prefixes service messages.
- *servtrailer* is up to 8 hexadecimal characters long and suffixes TDS service messages.

  Default is "0D25" (CR/LF).
- IMMEDIATE DELIVERY option sends TDS service messages to the terminal at the end of the current TPR.

  Conversely if this option is omitted, messages transmitted by the [M] SEND_TDS_USER and [M] MODIFY_TDS_MOT are sent to the terminal only when it leaves transaction mode. See *Immediate Delivery and Services Messages* on the following page. Service message 6 is suppressed.
- *msgnumber* is a 2-digit decimal value identifying the TDS service message to be modified. See *TDS Service Messages* on the following page.

- *servicemsg* is up to 46 alphanumeric characters giving the new text of the TDS service message consisting of:
  – user text or parts of a text
  – TDS-supplied variable fields that are dynamically filled at fixed places.

  The length of the user text must not be greater than the default. It must not exceed the TDS-supplied text, except for the text which terminates the messages.

- If *servicemsg* is a blank character, no service message is sent.

NOTE:
Messages of master commands (listed in Appendix F) sent to the master terminal operator are *standardized with GCL and cannot be redefined*.

TDS Service Messages			
Id	Service & Trans	Message Text	Can be redefined?
		Reserved for system use	
0	both	[M] MODIFY_TDS_MOT	Yes
1	both	[M] SEND_TDS_USER	Yes
4		TDS STARTUP IN PROGRESS. PLEASE WAIT	Yes
5	both	READY	Yes
6		WAIT FOR RESOURCE ALLOCATION	Yes
9		TDS STARTED. YOU ARE MASTER TERMINAL OPERATOR	No
15		tdsname SHUTDOWN	No
18		TRANSACTION COUNT:xxxx TPR COUNT:xxxxxxxxxxxxx	Yes
19	both	UNRECIO ON CI NUMBER xxxxxx FOR xxxx	No
20		xxxxxUSERS MAXIMUM	No
24		DISCONNECTED FROM xxxxxxxxxxxx	No
27		TAKEOVER IN PROGRESS. PLEASE WAIT	Yes
31		TRANSACTION xxxxxxx TRACED ON TERMINAL xxxxxxx	No
32	both	TRACE MANAGED BY ANOTHER USER	Yes
34		TRANSACTION REFUSED-CONVERSATION TYPE NOT SUPPORTED	Yes
35		ILLEGAL TRANSACTION	Yes
36		TRANSACTION REFUSED-LOGOUT REQUESTED	Yes
37		TRANSACTION REFUSED-ERROR IN TRACE	Yes
38		TRANSACTION REFUSED-SERVICE XCP2 NOT AVAILABLE	Yes
39		CONFLICT INQUIRY-CONVERSATION TYPE	Yes
40		TRANSACTION NOT AVAILABLE-RETRY	Yes
41		INVALID TRANSACTION	Yes
42		NO RPC PROCESS TO RUN THIS RPC TRANSACTION	Yes
43		CONNECTED FROM xxxxxxxxxxxx	No
44		xxxxxxxxxxxx DISCONNECTED BY WATCHDOG	Yes

NOTE:

Variable fields identified by x's are supplied dynamically.

Service message 6 appears when conflicts for resources delay the execution of a commitment unit. This message is sent either:

- when the NON-CONCURRENT clause does not allow the current commitment unit to execute simultaneously with another commitment unit
- when GAC-EXTENDED detects a conflict on access to monitored files or IDS/II schemas. This conflict is solved by aborting the current commitment unit. The terminal operator must wait until the particular resource is made available.

Immediate Delivery and Services Messages		
Service Message	IMMEDIATE DELIVERY	
	Not Specified	Specified
0 - MODIFY_TDS_MOT	Deferred until end of current TX	Deferred until end of current TPR
1 - SEND_TDS_USER	Deferred until end of current TX	Deferred until end of current TPR
2 - WAIT FOR RESOURCE	Suppressed	Operative

3.4.71 TDSTX-MESSAGE Clause

Syntax:

```
[ TDSTX-MESSAGE [ HEADER IS  { "servheader"  | "txheader"  } ]
                [ TRAILER IS { "servtrailer" | "txtrailer" } ].]
```

Description:

Defines header and trailer for transaction messages. See *TDS Transaction Messages*.

Usage:

- *txheader* is up to 8 hexadecimal characters that prefix TDS transaction messages. The default value is *servheader* of the SERVICE MESSAGE clause.
- *txtrailer* is up to 8 hexadecimal characters that suffix TDS transaction messages. The default value is *servtrailer* of the SERVICE MESSAGE clause.

TDS Transaction Messages			
Ident	Service and Transaction	Message Text	Can be redefined?
5	both	READY	Yes
8		ABORT (x.xx.xxxx) xxxxxxxxxx xxxxxxxxxxxxxxxxxxx	Yes
11		DEVICE NOT AVAILABLE FOR TRACES	Yes
13		ERROR IN TRANSACTION PARAMETERS	Yes
16		xxxxxxxx OPENED	No
17		xxxxxxxx NOT OPENED xxxxxxxxxxxxxxxxxxx	No
19	both	UNRECIO ON CI NUMBER xxxxxxx FOR xxxx	No
21		TERMINAL IN TRACE MODE OUTPUT ON xxxxxxxxxxxx	Yes
22		xxxxx CLOSED	No
23		xxxxxxxx NOT CLOSED xxxxxxxxxxxxxxxxxxx	No
25		xxxx CONNECTED	No
26		xxxx NOT CONNECTED REASON: xx	No
28		INSUFFICIENT AUTHORITY CODES	No
29		TRACE ALREADY ACTIVE	No
30		TRACE NOT ACTIVE	No
32	both	TRACE MANAGED BY ANOTHER USER	No
33		PCF COMMANDS ARE IN xxxxxxxxxx xxxxxxxxxxxxxxx_INXX	No

3.4.72 SPECIAL-CHAR Clause

Syntax:

```
[ SPECIAL-CHAR mnemonic IS "code".]...
```

Description:

Defines user-specific control codes used by transactions in addition to those predefined in CONSTANT-STORAGE. The maximum number of additional codes which can be defined in CONSTANT-STORAGE is 256.

The COPY CONSTANT-STORAGE statement allows the TPR to retrieve the contents of the control codes.

```
01 CONSTANT-STORAGE.                    Hex Value
   02  CR   PIC  X.          (Carriage Return)          0D
   02  LF   PIC  X.          (Line Feed)                25
   02  FF   PIC  X.          (Form Feed - Clear)        0C
   02  PR   PIC  X.          (Page Return - Home)       3C
   02  BLK  PIC  X.          (Blink)                    5F
   02  BEL  PIC  X.          (Bell)                     2F
   02  EM   PIC  X.          (End of Medium)            19
   02  FS   PIC  X.          (Forward Space)            12
```

Usage:

- *mnemonic* is the abbreviation of the control code such as CR for carriage return.
- *code* is a 2 hexadecimal-digit representing the control code such as "0D" for CR.

3.4.73 CANCELCTX AT RECONNECTION Clause

Syntax:

```
[ CANCELCTX AT RECONNECTION .]
```

Description:

Requests to automatically execute a CANCELCTX function at user reconnection.

Usage:

- When this clause is specified, and when a user transaction is interrupted by a disconnection or a TDS stop, this transaction will not be resumed at reconnection time.

  The LOGON transaction and the RESTART transaction are executed at reconnection time.
- When the NOCANCELCTX procedure is called inside the LOGON transaction, the clause is ineffective.

3.4.74 FILE-OPENING OPTION Clause

Syntax:

[FILE-OPENING OPTION IS *option*].

Description:

The option specified in the clause is used to open TDS user files, at TDS launching or during a TDS session thanks to a TPR or the master command OTF.

Usage:

Available values for option are:

BPIOC	When the user option is BPIOC, i.e. "bypass I/O cache", I/O cache is not used for TDS user files, controlled or not controlled, even if IOC is started on site. But, if a shared file has been previously opened by a batch asking usage of the cache, it will continue to use IOC, even if it is concurrently opened by a TDS.
FREEBUF	When the option is FREEBUF, UFAS buffers are released at each end of primitives for all TDS user files, controlled or not controlled. When the clause is not used, the buffers are kept for TDS controlled files, but they are freed for TDS non-controlled files.

3.5 Syntax of Input-Output Section

```
INPUT-OUTPUT SECTION.

FILE-CONTROL. file-control-entry ...
[ FILE-DEFINITION. fd-entry ...]
[ TDS-FILE-DEFINITION. fd-entry ...]

[ IDS-DEFINITION.                                                    ]
[ db-statement                                                       ]
[[ db-statement [ db-statement...]]                                  ]
[[ MAXIMUM NUMBER OF SCHEMA IS {1 | maxschema} PER COMMITMENT ].]

[ IQS-DEFINITION.                  ]
[ ||  VIEWS ARE view [,view ]... || ]
[ ||  FILES ARE file [,file ]... || ]
[ ||  AREAS ARE area [,area ]... || ]

[ PROCESSING-CONTROL.                                                 ]
[[PROCESSING-MODE OF file IS {INPUT-OUTPUT|INPUT|OUTPUT|EXTEND}.]]
[[FILE-INTEGRITY FOR file IS { HIGH | MEDIUM | NONE }.          ]]

[ WORKING-STORAGE.                ]
[        data-description-entries ]
[          .                      ]
[          .                      ]
[          .                      ]
[ *END                            ]

[ SHARED-STORAGE name.            ]
[        data-description-entries ]
[          .                      ]
[          .                      ]
[          .                      ]
[ *END                            ]

[ CONTROLLED COMMON-STORAGE name SIZE IS size. ]
[      [ data-description entry ]              ]
[          .                                   ]
[          .                                   ]
[          .                                   ]
[ *END                                         ]

[ ORACLE-DEF
 ...
 ORACLE-ENDDEF ]
```

3.5.1 FILE-CONTROL Clause

Syntax:

```
SELECT EXTERNAL filename ASSIGN TO ifn [ RESERVE nn AREA[S] ]
ORGANIZATION IS organization [ ACCESS MODE IS accessmode ]
[ FILE STATUS IS filestatus ] [WITH length ]
[ *END ].
```

Description:

Identifies the files to be used by TPRs. Each file-control entry or COBOL SELECT clause is stored in the file *tdsname*.COBOL. These entries are retrieved through COPY statements in the ENVIRONMENT DIVISION of the TPR.

See *COBOL 85 Reference Manual*.

Usage:

- FILE-CONTROL entries are not required for IDS/II areas.
- Specify the EXTERNAL keyword to indicate that the associated internal-file (ifn) be visible to TPRs.
- The first character of the *ifn* (internal file-name) must be a letter, suffixes to the *ifn* must not be used.
- *filename* is limited to 24 characters and must be different from TPR names and *USE-procedure* names.
- The maximum number of files allowed is 500.
- Do not define the *ifn* USERJRNL in any FILE CONTROL entry. It is reserved for the user journal and must not be specified.
- Using TDS-reserved keywords for *filename* and *ifn* may cause the generation to abort. See Appendix L.
- Each file-control entry must be terminated by its own *END.

3.5.2 FILE-DEFINITION Clause

Syntax:

```
FILE-DEFINITION.
FD ...
   01 description-entry.
    02 description-entry.
     03 description-entry.
       .
       .
       .
   01
    .
    .
    .
*END
[FD ...]
[  .   ]
[  .   ]
[  .   ]
[*END  ]
```

NOTE:
A separate *END is required to terminate each FD entry for TDS.

Description:

Declares an FD entry for each non-controlled file. After TP7GEN is run, the FD entries are stored in the file *tdsname*.COBOL. Then they are available and can be retrieved by the user through COPY statements in the DATA DIVISION of the TPR.

For non-controlled files, see Paragraph 4.4 *User Data Files* and *COBOL 85 Reference Manual*.

3.5.3 TDS-FILE-DEFINITION Clause

Description:

Declares an FD entry for each TDS-controlled file. After TP7GEN is run, the FD entries are stored in the file *tdsname*.COBOL. Then they are available and can be retrieved by users through COPY statements in the DATA DIVISION of the TPR.

For TDS-controlled files, see Paragraph 4.4 and *COBOL 85 Reference Manual*.

3.5.4 IDS-DEFINITION Clause

Syntax:

```
IDS-DEFINITION.

db-statement
[ db-statement [ db-statement...]]
[ MAXIMUM NUMBER OF SCHEMA IS { 1 |maxschema } PER COMMITMENT ].
```

Description:

Defines all database schema, subschema, and area (realm) details.

3.5.4.1 REALMS and DB Statements

For each db-statement:

Syntax:

```
DB schemaname.
REALMS ARE ifn-1 [,ifn-2...].
```

Description:

Defines the IDS/II database to be accessed under the TDS application. Several databases can be accessed within the TDS application in different TPRs.

With IDS/II, a COBOL procedure can address only one schema but several procedures of a TPR, or several TPRs of the same commitment unit can address several schemas.

With Full-IDS/II, a COBOL procedure can address up to 32 schemas.

All schemas are loaded in the TDS restart routines. If the load fails, the database cannot be accessed. In this case, TDS must be shut down and then restarted.

Usage:

- *schemaname* identifies a schema which must have been processed by the DDL Processor in the object schema file assigned to TP7GEN.
- The list of *ifn*'s (internal file-name) active in the schemas identify those realms to be accessed by the generated TDS application. *ifn*'s can be overwritten by the run-time IDS options.
- As many db-statements can be declared as there are schemas to define. The maximum number of schemas declared per TDS application is 32.

3.5.4.2 Maximum Number of Schema Statement

Syntax:

```
[ MAXIMUM NUMBER OF SCHEMA IS { 1 | maxschema} PER COMMITMENT ].
```

Description:

Defines the maximum number of schemas accessible in 1 commitment unit.

Usage:

maxschema is the maximum number of schemas for a transaction to access within a single commitment unit. *maxschema* cannot exceed the number of DBs declared.

The clause allows the optimization of the size of the entry used by IDS/ll in the swap context of a user:

- When the clause is omitted , the default is 1.

  If a commitment unit accesses more than 1 schema, the swap area used to save IDS/ll currencies at cleanpoint time may be too small. If this area is too small, from technical status 7356, a fatal abort MV40 with DATALIM return code will stop the TDS execution.
- When the clause is specified, the size of the swap area reserved for IDS/ll is computed using maxschema. This value must represent the maximum number of schemas which can be accessed by a commitment unit.

  If the value is too small, the swap area used to saved IDS/II currencies at cleanpoint time may be too small. If this area is too small, from technical status 7356, a fatal abort MV40 with DATALIM return code will stop the TDS execution.

3.5.5 IQS-DEFINITION Clause

Syntax:

```
IQS-DEFINITION.
   ||  VIEWS ARE view-name [,view-name]... ||
   ||  FILES ARE file-name [,file-name]... ||
   ||  AREAS ARE area-name [,area-name]... ||
```

Description:

Defines the view, file or area to be accessed by IQS queries.

Usage:

- *view-name* identifies either a view or a schema for an IDS/II database or a UFAS-EXTENDED file. IDS/II schemas must also be declared under IDS-DEFINITION.
- *file-name* identifies the UFAS-EXTENDED sequential file or a linked-queued subfile or a library member. Such a file is an IQS permanent file used in IQS Queries: READ, WRITE, SORT and PRINT.

  Files declared permanent must be defined as non-controlled files. A SELECT statement must be specified under FILE-CONTROL and all items of the records must be defined through an FD under FILE-DEFINITION.
- *area-name* identifies the UFAS-EXTENDED file against which a schema has been defined. The area name must be defined in the IDS/II schema description AREA NAME Clause.

  For each UFAS-EXTENDED area used by a view, a SELECT statement must be specified under FILE-CONTROL. All items of the records must be defined through an FD under TDS-FILE-DEFINITION.

NOTE:

If the IQS-DEFINITION clause is present, the VIEWS ARE ... clause must also be present.

Named areas must belong to the views or schemas defined in VIEWS ARE.

For area files and TDS-controlled files, the default processing mode is INPUT-OUTPUT, and the default FILE-INTEGRITY is HIGH.

A maximum of 32 views can be declared.

3.5.6 PROCESSING-CONTROL Clause

Syntax:

```
PROCESSING-CONTROL.
   PROCESSING-MODE
   FILE-INTEGRITY
```

Description:

Lists all Processing Mode and File Integrity clauses.

3.5.6.1 PROCESSING-MODE Clause

Syntax:

```
PROCESSING-MODE OF filename IS
              { INPUT-OUTPUT |INPUT | OUTPUT | EXTEND }.
```

Usage:

- Only one PROCESSING-MODE per file or realm is allowed. *Default:* INPUT-OUTPUT.
- TDS-controlled files and IDS/II realms can only be processed in INPUT or INPUT-OUTPUT.
- To override the processing mode, use [M] OPEN_TDS_FILE command.

3.5.6.2 FILE-INTEGRITY Clause

Syntax:

```
FILE-INTEGRITY FOR filename IS { HIGH | MEDIUM | NONE }.
```

Description:

Selects the minimum security level that must be provided for any file.
At execution time, TDS checks that the journals assigned to the file by the JCL statement DEFINE or by the catalog option are sufficient to obtain the selected security level. If the security level is not sufficient, TDS denies any access to that file.

Usage:

- HIGH applies to a file protected against software failures (TPR, TDS or system aborts) and hardware failures. This protection is invoked by using either Deferred Updates and the After Journal, or the Before and After Journals.
- MEDIUM applies to a file which at least must be protected against software failures. This protection is invoked by using the Before Journal.
- NONE applies to a file whose protection is defined in the JCL and in the catalog.

  When the FILE-INTEGRITY clause is absent, *default* is:

  – HIGH for TDS-controlled files

  – NONE for non-controlled files.

NOTE:

Whenever the journals assigned to a file allow a higher level of security than the one specified at TP7GEN, the higher security applies.

3.5.7 WORKING-STORAGE Clause

Syntax:

```
WORKING-STORAGE.
77  description-entry.
 .
 .
 .
77
01  description-entry.
 02  description-entry.
  03  description-entry.
    .
    .
    .
01
    .
    .
    .
*END
```

Description:

Allows data items retrieved in the TPRs through COPY statements to be defined if they are not in the INPUT-OUTPUT SECTION. These data items are used in TPRs and include Relative Key and File Status.

Usage:

- The data structure must be terminated by a separate entry: *END.
- The contents of the WORKING-STORAGE are stored in the file *tdsname*s.COBOL and retrieved by COPY statements in the DATA DIVISION of the TPR.

NOTE:
If modifying this storage area, ***recompile*** the TPRs and regenerate the TDS.

3.5.8 SHARED-STORAGE Clause

Syntax:

```
SHARED-STORAGE name-1.
   02 description-entry.
    03 description-entry.
     .
     .
     .
*END
[ .                    ]
[ .                    ]
[ .                    ]
[SHARED-STORAGE name-n.]
[   02                 ]
[   .                  ]
[   .                  ]
[   .                  ]
[ *END                 ]
```

Description:

Defines the layout of the SHARED-STORAGE area.

Usage:

- SHARED-STORAGE is user-defined and is shared among transactions of the same TDS application.
- SHARED-STORAGE when specified, must appear after WORKING-STORAGE.
- The maximum number of SHARED-STORAGEs is 63. The maximum size of one SHARED-STORAGE is 64 Kbytes.
- Copy an area to the program through the statement: COPY *shared-name*. Access the Shared Storage with the MOVE or ADD commands.

NOTE:

When naming a SHARED-STORAGE, do not use TDS-reserved terms, names of Call Procedures such as EXITS and SIMBREAKS, or TPR names.

Do not include an 01 level in the declaration.

3.5.9 CONTROLLED COMMON-STORAGE Clause

Syntax:

```
CONTROLLED COMMON-STORAGE name-1 SIZE IS size-1.
[  01 description-entry          ]
[   02 description-entry.        ]
[    03 description-entry.       ]
[    ...                         ]
[   02                           ]
[    ...                         ]
[  *END                          ]
[   ...                                                    ]
[ CONTROLLED COMMON-STORAGE name-64 SIZE IS size-64. ]
[  [01  ]                                                  ]
[  [ 02 ]                                                  ]
[  [ ...]                                                  ]
[  [*END]                                                  ]
```

Description:

Defines the layout of CONTROLLED COMMON-STORAGE, an area shared among transactions of the same TDS application.

Usage:

- When naming CONTROLLED COMMON-STORAGE, do not use TDS-reserved terms, names of Call Procedures such as EXITS and SIMBREAKS, or TPR names.
- CONTROLLED COMMON-STORAGE is initially set to zero when the TDS application is started either for the first time with TP7GEN, or after a cold restart.
- It is protected by the Before Journal and rolled back in the case of an abort.
- CONTROLLED COMMON-STORAGEs are held in the *tdsname*.CTLM file.
- CONTROLLED COMMON-STORAGE can be repeated up to a maximum of 64.
- the name of the 01 record description entry is up to 12 alphanumeric characters.
- size must be an integer less than or equal to 65,536 (64 Kbytes).

NOTE:
If modifying this storage area, ***recompile*** the TPRs and regenerate the TDS.

3.5.10 ORACLE-DEF...ORACLE-ENDDEF Paragraph

Syntax:

```
[ ORACLE-DEF
  ...
  ORACLE-ENDDEF ]
```

Description:

With ORACLE V7, a set of clauses may be specified to configure ORACLE parameters. The internal contents between ORACLE-DEF and ORACLE-ENDDEF is documented in *ORACLE/TDS User's Guide.*

Using ORACLE-DEF...ORACLE-ENDDEF Paragraph implies that:

- the "JOBLIB SM, &ORASTLIB;" is present at the beginning of the TP7GENtdsname subfile of tdsname.SLLIB. If not, run TP7PREP with the DEAL=Y option.
- the ORASTLIB keyword of the TP7GEN JCL is specified with the name SMLIB containing the H_ORATDS SM.

3.6 Syntax of TRANSACTION SECTION

```
TRANSACTION SECTION.

-----Each MESSAGE Statement represents a separate Transaction-----

 MESSAGE "messageid" ASSIGN TO tprname

  [ MAXIMUM NUMBER OF TPR IS maxtpr ][ FOR DEBUG ][ FOR INQUIRY ]
  [ IMPLICIT COMMITMENT]
  [ XCP2 SERVICE USED ]
  [ HEURISTIC { BACKOUT | COMMIT }]
  [ NO-DEFER-RESYNC ]
  [ NO-RESTART ]
  [ RPC SERVICE USED ]
  [ XA SERVICE USED]
  [ CMA SERVICE USED]
  [{ SUPPRESS { BEFORE JOURNAL | DEFERRED UPDATES }            }]
  [{ USE DEFERRED UPDATES [ EXCEPT FOR filenm-1 [,filenm-2...] }]

  [ SUPPRESS CONCURRENT ACCESS CONTROL FOR filenm-1[,filenm-2...]
  [ SHARED READ FOR filenm-a [,filenm-b...]]
  [[MANUALLY] NON-CONCURRENT
              [ WITH "messageid-1" [,"messageid-2"...]]]
  [ CLASS IS { "Z" | "class" }]
  [ PRIORITY IS { 0 | priority }]
  [ FORM IS { formname | RESETFRM }]
  [[ NO ] AUTOMATIC UNMAPPING ]
  [[ TPR ] ACCOUNTING ]
  [{ROLL-BACK|WAIT} COMMITMENT ]

  AUTHORITY-CODES ARE tdscode [,tdscode2...]

  [ HIDDEN ]
  [ PROMPT IS "text"]
  [ THINK-TIME IS think-time ]

  [ TRANSACTION-STORAGE [ name ]
                      SIZE IS { deftxstorage | txstorage }].
     [ 01 TRANSACTION-STORAGE.    ]
     [ 02 data-description-entry. ]
     [ *END                       ]

-----------------------------------------------------------------

[ MESSAGE ... ]
[  .          ]
[  .          ]
[  .          ]
[ MESSAGE ... ]
```

3.6.1 Message ASSIGN DEBUG INQUIRY Statement

Syntax:

```
MESSAGE "message-id" ASSIGN TO tprname
[ MAXIMUM NUMBER OF TPR IS maxtpr ] [FOR DEBUG] [FOR INQUIRY]
```

Description:

A MESSAGE statement is required for each transaction to be processed.

Usage:

- *messageid* is up to 8 alphanumeric characters to identify the transaction. TDS uses *messageid* to load and activate the first TPR of the transaction specified by ASSIGN.
- BREAK, CLOSE, DISCNCT, LOGON, LOGOUT, OPEN, PT, RESTART and STARTUP are TDS-reserved *messageid*'s that start transactions for processing special events such as a break signal.
- *tprname* up to 12 alphanumeric characters corresponds to the name specified in PROGRAM-ID of the TPR IDENTIFICATION DIVISION; this TPR is the first to be started when the *messageid* is keyed in.
- MAXIMUM NUMBER OF TPR prevents a transaction from looping uncontrolled chains of TPR's. If more than maxtpr transactions are submitted, COUNTLIM is returned to flag an abort even if the ON-ABORT-TPR mechanism is set. Maximum maxtpr is 32767. *Default: No control*.
- DEBUG prevents TDS-controlled files from being modified. WRITE, REWRITE and DELETE verbs will be performed but the file will be rolled back to its initial state when the commitment unit terminates.
- FOR INQUIRY applies to transactions with only one commitment unit involving no interaction with the terminal. The transaction receives the initial message, sends the response and then terminates. No files or database are updated. TDS does not use the swap file. On abort or system crash, the FOR INQUIRY transaction is not restarted.

  The contents of PRIVATE-STORAGE if present, are set to zero after being saved; PRIVATE-STORAGE is restored from the save when the FOR INQUIRY transaction terminates.

FOR INQUIRY transactions have the following restrictions:

– they cannot request FORMS; if they do, they abort with RESVIOL
– they cannot be used for spawning operations
– they cannot be traced, trace requests are ignored
– they cannot use RPC services if RPC simultaneity regulation services are required.
– they cannot use XA services

The TRACE command is described in the *TDS COBOL Programmer's Guide*.

Until TS 9764, the maximum number of transactions that can be declared in one TDS application is 2000.

Since TS 9866, the maximum number of transactions has been increased to 3000.

In fact the actual limit is:

3040 - 7 (reserved by TDS)

- NUMBER OF DYNAMICALLY ADDED TRANSACTIONS (between 0 and 50, default is 10)

- system transactions defined in TP7TXLIB subfile of SYS.HSLLIB (21 in TS 9866)

NOTE:
The special purpose transactions (as LOGON, LOGOUT…) defined both in TP7TXLIB and STDS subfiles are counted once.

3.6.2 IMPLICIT COMMITMENT Clause

Syntax:

```
[ IMPLICIT COMMITMENT ]
```

Usage:

- When specifying the IMPLICIT COMMITMENT clause, a commitment is obtained either by explicit request (CALL "DFCMIT") or implicitly when the TPR terminates with a SEND with EGI statement, or when a WAIT-TIME is set. The implicit commitment rule is by-passed when the CALL "NOCMIT" is called.
- When the IMPLICIT COMMITMENT clause is omitted, all commitments must be explicitly requested by calling the CALL "DFCMIT" statement.

3.6.3 XCP2 SERVICE Clause

Syntax:

```
[ XCP2 SERVICE USED ]
```

Description:

Allows a transaction to use the XCP2 protocol.

Usage:

The XCP2 SERVICE clause is available for transactions if a XCP2WKS structure has been defined for the TDS application at NETGEN. See the manual *Network Generation.*

3.6.4 HEURISTIC Clause

Syntax 1:

```
[ HEURISTIC BACKOUT ]
```

Syntax 2:

```
[ HEURISTIC COMMIT ]
```

Description:

Increases the degree of control over whether to commit, or to rollback when the link with the superior transaction fails while the state of the local transaction is uncertain. If the state is uncertain, the TDS application decides to commit or to backout according to the parameter specified.

Usage:

- This clause is ignored if the XCP2 SERVICE clause is absent. It is used only when the distributed commitment function (a part of XCP2 service) is used.
- The default value is BACKOUT.

Refer to *CPI-C/XCP2 User's Guide*.

3.6.5 NO-DEFER RESYNC Clause

Syntax:

[NO-DEFER-RESYNC]

Description:

Serves when the distributed commitment function (a part of the XCP2 service) is used. This clause increases the degree of control over whether to commit, or to rollback when the link with an inferior transaction fails.

Usage:

When NO-DEFER RESYNC is specified, processing is suspended until the link is re-established.

When NO-DEFER RESYNC is not specified, the commitment continues to be processed. The partner transaction with which the link is cut is temporarily ignored.

In both cases, once the link is re-established, any inconsistencies are reported:

- as part of the commitment processing itself if the NO-DEFER RESYNC clause is present
- after the commitment is processed if the NO-DEFER RESYNC clause is not present.

This clause is ignored if the XCP2 SERVICE clause is absent. Refer to *TDS COBOL Programmer's Guide*.

3.6.6 NO-RESTART Clause

Syntax:

```
[ NO-RESTART ]
```

Description:

Defines the value of the NO-RESTART field in the TDS-STORAGE at the beginning of each commitment unit.

Usage:

- The NO-RESTART field in the TDS-STORAGE specifies whether the commitment unit may be restarted or not after a rollback.
- When the clause is included, the NO-RESTART field is set to 1. When the clause is omitted, the NO-RESTART field is set to 0.

Refer to *TDS COBOL Programmer's Guide*.

3.6.7 RPC SERVICE Clause

Syntax:

```
[ RPC SERVICE USED ]
```

Description:

Allows a transaction to use RPC calls.

This clause is ignored if the RPC SIMULTANEITY clause is missing.

Usage:

Mandatory for transactions using RPC calls if the RPC Simultaneity Regulation is used (see paragraph 3.4.5). Otherwise, such transactions may abort with the PMDVIOL return code.

FOR INQUIRY (see Paragraph 3.6.1) and RPC SERVICE are mutually exclusive.

3.6.8 XA SERVICE Clause

Syntax:

[XA SERVICE USED]

Description:

The XA SERVICE USED clause allows the transaction to use XA services. This clause is mutually exclusive with the CMA SERVICE clause.

A TDS-XA is a transactional application in which some transactions call on services offered by XA Resource Managers (RMs) whereas others do not. XA is defined at TDS level if at least one transaction has been declared <<XA SERVICE USED>> during the last TDS generation.

Without the XA interface, access to RMs such as ORACLE or IDS/II databases and UFAS files in a single commitment unit is authorized. The user is, however, advised against this as the TDS and ORACLE commitments are not synchronized.

With TDS-XA you can, in the same CU, simultaneously update ORACLE and IDS/II databases and UFAS files on the one hand, and, on the other hand, simultaneously update several ORACLE databases.

ORACLE7 is the only Resource Manager that provides an XA interface known to TDS. One of the <<USE ORACLE>> or <<USE ORACLE-BASE[-database name]>> clauses must be in the TDS SECTION of the STDS.

3.6.9 CMA SERVICE Clause

Syntax:

```
[ CMA SERVICE USED]
```

Description:

The CMA (Client Mode Access) product is built on a client-server model, with the server on a UNIX system and the client on a GCOS7 machine.

The CMA SERVICE USED clause will provide (as soon as the H_SM_CMA module is available) a transaction for accessing RDBMs such INFORMIX located on a UNIX system.

This transaction is made up of TPRs, which are programs generated by CMA*GEN tool located on UNIX, and transferred to GCOS7 in order to be compiled and linked like a normal COBOL or C TPR. Such programs act as CMA client. The CMA server part is located on UNIX.

The CMA client (TPR) and CMA server communicate via CMA*NET based on TCP-IP protocol.

This clause is mutually exclusive with the XA SERVICE clause.

For further details, refer to the *SQL CMA User's Guide* (47 A2 22 UR).

3.6.10 File Security Options

Syntax:

```
[ SUPPRESS {BEFORE JOURNAL|DEFERRED UPDATES}                    ]
[ USE DEFERRED UPDATES [ EXCEPT FOR filename-1 [,filename-2...]]]
```

Description:

For optimum performance, use these options to modify the security mechanisms that TDS applies by default. If this results in reduced file protection, the clause is ignored and the transaction may abort.

3.6.10.1 SUPPRESS BEFORE JOURNAL Clause

Syntax:

```
[ SUPPRESS BEFORE JOURNAL ]
```

Description:

Inactivates the Before Journal. This clause optimizes performance because accesses to the Before Journal are avoided.

Usage:

- SUPPRESS BEFORE JOURNAL is mutually exclusive with SUPPRESS DEFERRED UPDATES and USE DEFERRED UPDATES.
- SUPPRESS BEFORE JOURNAL may be ignored by a TDS application when *Deferred Updates* cannot be used, as when a commitment is reexecuted after an abort caused by a lack of buffers.
- When SUPPRESS BEFORE JOURNAL is specified for a UFAS-EXTENDED shared controlled file with the Journal option BOTH or AFTER, the first time the file is accessed in update mode, the commitment unit aborts with "XAC 15, ITMNAV" and is restarted with the Before Journal.

  Use either the Before Journal, or Deferred Updates to update a file. Specify USE DEFERRED UPDATES to work on a shared file in *Deferred Update* mode. See *File Security Options and Status of TDS Controlled and Non-Controlled Files* below.

3.6.10.2 SUPPRESS DEFERRED UPDATES Clause

Syntax:

```
[ SUPPRESS DEFERRED UPDATES ]
```

Description:

Prevents transactions from using Deferred Updates to protect TDS-controlled files.

Usage:

SUPPRESS DEFERRED UPDATES is mutually exclusive with SUPPRESS BEFORE JOURNAL and USE DEFERRED UPDATES. See *File Security Options and Status of TDS Controlled and Non-Controlled Files* below.

3.6.10.3 USE DEFERRED UPDATES Clause

Syntax:

```
[ USE DEFERRED UPDATES [ EXCEPT FOR filename-1 [,filename-2...] ]
```

Description:

Forces *Deferred Updates* to ensure file security thus using the After Journal.

Usage:

- USE DEFERRED UPDATES forces using *Deferred Updates* for all files protected by the Before and After Journals, except for those explicitly listed in EXCEPT FOR.
- USE DEFERRED UPDATES applies to TDS-controlled files and IDS/II areas. It is intended for shared UFAS-EXTENDED files under a TDS application, but not for UFAS-EXTENDED files allocated with PREVIOUS parameter.
- USE DEFERRED UPDATES is mutually exclusive with SUPPRESS BEFORE JOURNAL and SUPPRESS DEFERRED UPDATES. USE DEFERRED UPDATES overrides system defaults by using *Deferred Updates* for both shared and non-shared controlled files.

- Files declared after EXCEPT can be shared only in *Exclusive READ* and must not appear in SHARED READ or SUPPRESS CONCURRENT ACCESS CONTROL
- When EXCEPT is used, the files are protected by the Before Journal. EXCEPT limits using UFAS-EXTENDED buffers in *Deferred Updates* mode. If a file is involved in a commitment unit of several TPRs which include a conversation, use SUPPRESS DEFERRED UPDATES or USE DEFERRED UPDATES EXCEPT.
- If a commitment unit aborts due to lack of buffers or is incompatible with *Deferred Updates* such as CI splitting, TDS reexecutes the commitment unit using the Before Journal and ignores the request for Deferred Updates. See *File Security Options and Status of TDS Controlled and Non-Controlled Files* below.

File Security Options and Status of TDS-Controlled and Non-Controlled Files					
Journal Optional	File Sharing	Clause Specified in TP7GEN			
		none	suppress BEF Journal	suppress DEF Updates	use DEF UPD [EXCEPT] ***1***
TDS-Controlled Files					
AFTER	SHARED	BEF + AFT Journal	AFT ***3*** Journal	BEF + AFT Journal	DEF Updates + AFT Journal
	non-SHARED	DEF Updates + ***2*** AFT Journal	DEF Updates + ***2*** AFT Journal	BEF + AFT Journal	DEF Updates + AFT Journal
BOTH	SHARED	BEF + AFT Journal	AFT Journal	BEF + AFT Journal	DEF Updates + AFT Journal
	non-SHARED	DEF Updates + ***2*** AFT Journal	DEF Updates + ***2*** AFT Journal	BEF + AFT Journal	DEF Updates + AFT Journal
BEFORE	SHARED *and* non-SHARED	BEF Journal	BEF Journal	BEF Journal	BEF Journal
Non-Controlled Files					
BEFORE	SHARED *and* non-SHARED	BEF Journal	BEF Journal	BEF Journal	BEF Journal
none	SHARED *and* non-SHARED	No File Integrety	No File Integrety	No File Integrety	No File Integrety

NOTES:

1. Security status for files specified in [EXCEPT] is Before + After Journals. USE DEFERRED UPDATES *can only be used with TDS-controlled files*.
2. File security status automatically switches to Before + After Journals after an abort caused by lack of buffers.
3. File security status automatically switches to Before + After Journals after an abort caused by a file update.

3.6.11 SUPPRESS CONCURRENT ACCESS CONTROL Clause

Syntax:

```
[ SUPPRESS CONCURRENT ACCESS CONTROL FOR filename-1
                                        [,filename-2...] ]
```

Description:

Allows TPRs to access TDS-controlled files or IDS/II realms without the control of GAC-EXTENDED. All locks are ignored and any record can be read.

Usage:

- Files can only be read (Statistical Read) even if declared INPUT-OUTPUT.

  Because CIs or pages are not locked, inconsistencies can occur. A transaction declared SUPPRESS CONCURRENT ACCESS CONTROL may read inconsistent data if another TPR simultaneously modifies the same record. If a transaction attempts to write to an unlocked CI or page, the transaction aborts.
- SUPPRESS CONCURRENT ACCESS CONTROL is implicitly assigned to transactions declared FOR INQUIRY. It is not needed for any transactions using files assigned in SHARE=ONEWRITE and ACCESS=READ.
- Files must not appear in either SHARED READ or USE DEFERRED UPDATES.

3.6.12 SHARED READ Clause

Syntax:

SHARED READ FOR *filename-a* [,*filename-b*...]

Description:

Specifies TDS-controlled files and IDS/II areas to be used in the *Shared Read* mode under the control of GAC-EXTENDED rather than in the *Exclusive Read* mode.

Usage:

- SHARED READ allows a CI or page to be shared with other readers. When specifying SHARED READ, updating a file is always done exclusively. This means that to rewrite a CI, all other commitments must complete to apply an *exclusive lock* before making modifications.
- If SHARED READ is omitted, *Exclusive Read* mode applies. This locks the CI being read by the CU. All other transactions must wait until the CI is released.
- If SHARED READ is specified, Statistical Read mode applies for the files assigned with SHARE=ONEWRITE and ACCESS=READ that the transaction is to use.
- A file specified in SHARED READ Clause must not appear in:
  - SUPPRESS CONCURRENT ACCESS CONTROL Clause
  - and USE DEFERRED UPDATES EXCEPT Clause.

3.6.13 MANUALLY NON-CONCURRENT Clause

Syntax:

```
[ MANUALLY ] NON-CONCURRENT
                    [ WITH "messageid-1" [,"messageid-2"...] ]
```

Description:

Prevents resource conflicts, ensures that TPRs do not execute simultaneously, and reduces deadlocks.
Non-concurrency is applicable at commitment level, that is, when 2 transactions are non-concurrent, a commitment unit of the first transaction is not executed at the same time as a commitment unit of the second transaction.

Usage:

- *messageid* identifies the transactions that are not to execute concurrently with the specified transaction.
  As many transactions as necessary can be declared non-concurrent.

  An empty list of *messageid*'s means that the transaction is non-concurrent with all transactions including another occurrence of the transaction itself.
- If MANUALLY is present, non-concurrency is activated only on the commitment unit after calling SET-NON-CONCURRENT. See the *TDS COBOL Programmer's Guide*.
- If MANUALLY is omitted, non-concurrency is activated from the start of the transaction.
- Non-concurrency must be symmetric. If transaction A is non-concurrent with B, then B must be non-concurrent with A. If the two transactions are not symmetric, an error occurs and its corresponding message is listed in the TP7GEN report regardless of whether MANUALLY was specified or not.
- This clause is not applicable for a FOR INQUIRY transaction.

3.6.14 CLASS Clause

Syntax:

[CLASS IS { **"Z"** | "*class*" }]

Description:

Allows a transaction to be placed in a given class defined by an alphabetic character.

Usage:

- *class* is a single alphabetic character.

  All transactions not specifically classified are grouped in the default class Z.

- The master terminal can prevent and restore the execution of a class of transactions by issuing [M] MODIFY_TX commands specifying the class.

  TDS-supplied transactions belong to class S (system) which cannot be locked by the [M] MODIFY_TX LOCK=1 command; class S therefore should not be specified for user transactions.

3.6.15 PRIORITY Clause

Syntax:

```
[ PRIORITY IS { 0 | priority }]
```

Description:

Assigns a dispatching priority to all TPRs of a transaction, with respect to other transactions and competing batch jobs.

Usage:

priority ranges from 0 through 3 and specifies the relative dispatching priority of the TPR. This *priority* is added to the TDS job dispatching priority to give the final dispatching priority of the TPR. *Default:* 0 (highest).

Examples:

- The following priorities are declared for a site:
  - Batch job with dispatching priority=3.
  - TDS job with dispatching priority=1.
  - Transaction A with PRIORITY of 0.
  - Transaction B with PRIORITY of 1.
  - Transaction C with PRIORITY of 3.
- The final dispatching priorities will be:
  - Batch=3.
  - A = 1 + 0 = 1.
  - B = 1 + 1 = 2.
  - C = 1 + 3 = 4.

  In the above example, transaction A may terminate before B or C because it gets more CPU time than B or C.

3.6.16 FORM Clause

Syntax 1:

[FORM IS *formname*]

Syntax 2:

[FORM IS RESETFRM]

Description:

Allows the FORMS facility to be used in the current transaction.

Usage:

Syntax 1

- Use this clause only for a terminal with a diskette. It is assumed that the terminal operator manually loads the FORMS from a terminal with the diskette option.
- *formname* is the name of the initial active form used by the transaction. The first field in the form must be the name of the transaction to be activated.

Syntax 2

- If the TDS application is generated with USE FORMS NO IMPLICIT RELEASE, FORM IS RESETFRM must be used for every transaction which does not call FORMS.
- After such a transaction is started, the terminal screen is no longer in format mode even if the screen was not cleared or line mode was not entered at the keyboard. The terminal is not reset to format mode until a CALL "CDGET" verb is issued.

NOTE:
A form cannot be named RESETFRM. See Paragraph 4.3 and *IOF Programmer's Manual*.

3.6.17 AUTOMATIC UNMAPPING Clause

Syntax:

```
[ NO ] AUTOMATIC UNMAPPING
```

Description:

Instructs TDS not to automatically release a process at the end of a TPR.

Mapping means allocating a TDS control task (also called a process) to a TPR in execution. At the end of the TPR, the control task is either *unmapped* or kept for the next TPR in the transaction.

Usage:

- Automatic unmapping is performed:
  - following a BREAK
  - following a CANCEL
  - following a GAC ROLLBACK
  - when a value is specified in the WAIT-TIME field
  - after a TPR abort
  - after a SEND WITH EGI verb
  - every 100 commitments (to prevent any deadlock due to a loop)
  - when commitments have been blocked by another due to non-concurrence.
- Unmapping is useful for a TDS session with few simultaneities and several printers. It allows simultaneous printing and asynchronous terminal operations.
- Unmapping involves two additional I/O operations, one for unmapping, another for mapping the next TPR, on the swap file.
- NO AUTOMATIC UNMAPPING prevents a TDS application from using the swap file when a TPR terminates after issuing a SEND WITH EMI, without commitment.

3.6.18 ACCOUNTING Clause

Syntax:

```
[ WITH [ TPR ] ACCOUNTING ]
```

Description:

Enables the transaction and, optionally, TPR accounting records, to be written to the accounting file.

Usage:

The accounting function can be disabled for the session through the M MODIFY_TDS command. See Appendix E for the structure of accounting records.

3.6.19 COMMITMENT Clause

Syntax:

```
[{ ROLL-BACK | WAIT } COMMITMENT ]
```

Description:

Determines how an interrupted commitment unit is handled.

Usage:

- COMMITMENT can be used in any transaction to determine the action to be taken ***before*** activating the BREAK transaction:
  - ROLL-BACK to the previous commitment
  - WAIT for the current commitment unit to complete normally.
- If COMMITMENT is omitted, the resulting *commitment* is:
  - either that defined for the BREAK transaction
  - or if the BREAK transaction has no *commitment*, ROLL-BACK is the default.

3.6.20 AUTHORITY-CODES Clause

Syntax:

AUTHORITY-CODES ARE *tdscode1* [,*tdscode2*...]

Description:

Specifies the authority codes needed by a user to access the given transaction.

Usage:

- *tdscode* ranges from 0 through 31. Authorized users of the transaction must have at least one *tdscode*. See Paragraph 4.1.3 *Specifying Authority Codes*.
- A user who does not have the authority to initiate a transaction, is denied access and will receive the message UNKNOWN TRANSACTION.

3.6.21 HIDDEN Clause

Syntax:

HIDDEN

Description:

Prevents the transaction name from appearing on the menu.

3.6.22 THINK-TIME Clause

Syntax:

[THINK-TIME IS *think-time*]

Usage:

Sets a WAIT-TIME in seconds on SEND EGI in the transaction.
Maximum *think-time* is 32767. *Default* is "Nolimits".

NOTE:

A TPR started by an elapsed *think-time* must include in its starting RECEIVE verb a NO DATA Clause; otherwise an F0 (STCK_VERB) will be returned in the status key for this verb.

3.6.23 PROMPT Clause

Syntax:

PROMPT IS "*text*"

Usage:

text is a maximum of 40 characters displayed in the MENU system transaction for describing the purpose of the transaction.

3.6.24 TRANSACTION-STORAGE Clause

Syntax:

```
[ TRANSACTION-STORAGE [ name ] SIZE
  IS { deftxstorage | txstorage }.]

    [ 01 TRANSACTION-STORAGE.       ]
    [ 02 private-storage-name.      ]
    [ 03 data-description-entry.    ]
    [   .                           ]
    [   .                           ]
    [   .                           ]
    [ 02 transaction-storage-name1. ]
    [  03 data-description-entry.   ]
    [   .                           ]
    [   .                           ]
    [   .                           ]
    [ 02 transaction-storage-name2. ]
    [   .                           ]
    [   .                           ]
    [   .                           ]
    [ *END                          ]
```

Description:

Stores the COBOL or C language data accessible to all TPRs that is necessary to carry out a transaction.

Usage:

- *name* identifies TRANSACTION-STORAGE common to all TPRs processing this transaction
- *txstorage* specifies the size of TRANSACTION-STORAGE in bytes and remains constant for the duration of the transaction. *txstorage* must be less than 65527 bytes.
- When TRANSACTION-STORAGE is omitted, its size is the deftxstorage of the TDS SECTION.
- This storage is set to zero when the transaction is initialized.

NOTE:

When naming TRANSACTION-STORAGE, do not use TDS-reserved terms, names of Call Procedures such as EXITS and SIMBREAKS, or TPR names.

TRANSACTION-STORAGE can be optionally described by an 01 level entry called TRANSACTION-STORAGE followed by standard data description entries and terminated by *END. In this case, name is mandatory.

The data description entries of TRANSACTION-STORAGE can be retrieved through COPY statements from the file *tdsname*.COBOL. The description length must be less than or equal to the declared size of TRANSACTION-STORAGE.

PRIVATE-STORAGE, if specified, should be at the start of the TRANSACTION-STORAGE.

As an alternative to copying the TRANSACTION-STORAGE common to all TPRs, a TPR can define transaction storage in its own LINKAGE SECTION of its DATA DIVISION. The benefit of defining TRANSACTION-STORAGE in the TPRs is that TRANSACTION-STORAGE and PRIVATE-STORAGE can be accessed separately.

See the *TDS COBOL Programmer's Guide*.

4. Elements of TDS

These elements provide TDS the necessary interfaces with:

- the site catalog and the MAINTAIN_CATALOG utility for defining projects, applications and users
- the network and the NETGEN utility for declaring the communications configuration
- FORMS and the MAINTAIN_FORM utility for defining how forms are handled and displayed
- User Data Files for storing data specific to a given TDS application
- the Terminal Adapter Interface for modifying the way messages are sent and received
- GTWRITER, the printing facility
- and the Master Mailbox and its effect on user activity.

4.1 TDS Catalog

A TDS application *tdsname*, like any other application is defined in the GCOS 7 catalog. Each user of TDS must be declared as a USER of a PROJECT in the SITE CATALOG and satisfy the minimum access rights to run a given TDS transaction as described in the AUTHORITIES-CODES Clause, see Paragraph 3.5.18.
See Section 6 *Ensuring Data Security, Integrity and Recovery*.

To enter this information in the SITE.CATALOG, use the MAINTAIN_CATALOG utility to:

- create one or more projects
- enter the TDS project on the application list
- specify the authorities for each application
- enter a list of users for each project.

See *System Administrator's Guide*.

4.1.1 Creating Project

To add a project XYZ to the Site Catalog use the CREATE_PROJECT directive:

```
[CRP.....................XYZ;]
```

To modify a project, use the MODIFY_PROJECT directive:

```
[MDP.....................XYZ;]
```

4.1.2 Entering an Application List

To create the application list for a project, use APPLIST of MODIFY_PROJECT directive:

```
[ APPLIST..............,tdsname/tdscode; ]
```

where *tdsname* is the name of the TDS, and *tdscode* is an 8-character hexadecimal string defining the authority codes for that application.

If an application list exists, it can be modified with ADDAPPL parameter:

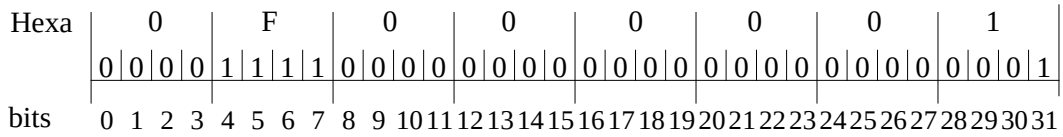
```
[ ADDAPPL tdsname/tdscode; ]
```

4.1.3 Specifying Authority Codes

Each *tdscode* associated with a project is represented internally by 32-bits (numbered from 0 to 31) coded in hexadecimal. For each bit:

- 1 indicates the bit is an authority code
- 0 indicates the bit is not an authority code.

An application with authorities 4, 5, 6, 7, and 31 is represented:



and would be entered as *tdsname*/0F000001.

Each transaction must be generated with a list of authority codes. Only projects whose authority code in the Site Catalog matches at least one of the codes specified in TDSGEN have the right to execute the transaction. So if a transaction is declared in TDSGEN:

```
AUTHORITY-CODES ARE 7, 31, 9
```

any user of project *tdsname*/0F000001 can execute the transaction because the authority code 31 is declared in both. Authority code 0 is reserved for master transactions.

4.1.4 Adding Users to Project

To add users to a project, use the CREATE_USER directive:

```
[ CRU.....................project userid; ]
```

where *project* is the name under which the user is allowed to connect.

4.1.5 Example of MAINTAIN_CATALOG

```
MNCAT;
[CRP............XYZ  APPLIST=TDS1/0F000001;]
[MDP............XYZ  ADDLIST=TDS2/0F000001;]
[CRU............XYZ.USER1;]
```

The first line creates the XYZ project and the TDS1 application. The second line modifies the existing XYZ project by adding the TDS2 application. The third line creates a user named USER1 under the XYZ project. See *GCOS 7 System Administrator's Manual* and *Catalog Management User's Guide*.

4.2 Network Generation

The network used to run TDS must be configured and defined to allow communication between terminals and transactions. The NETGEN (Network Generation) utility declares and describes the communications configuration of the system, relating logical names to physical addresses. A local application can address a partner application by its logical name without needing to know its physical address.

Declare TDS workstations including the master mailbox and the XCP1 mailbox, through NETGEN rather than defining them in individual clauses.

4.2.1 Correspondents

Correspondents refer to entities which dialog. A correspondent can be an application or a user logged on at a terminal located on the same site (local) or on another site (remote). Each correspondent is an object identified by name and address, with static characteristics:

- TM (Terminal Manager): representing a terminal
- XCP1: an application which operates according to the XCP1 protocol
- XCP2: an application which operates according to XCP2 protocol
- *Dummy:* Used by TDS as a virtual entity and need not be defined at NETGEN.

See the following publications:
- *Networks: Overview and Generation Manual*
- *CPI-C/XCP2 User's Guide*
- *Transactional Intercommunication using the XCP1 Protocol User's Guide*.

4.2.2 Site Configuration

The CONFIG statement SITEOPT, defines the site capabilities. The fourth byte of the site options is reserved for TDS applications. To choose milliseconds as the time unit for TDS accounting records, set the first bit of the fourth byte in the site options parameter. If this byte is not set, the default is milliminutes. The remaining bits must be set to zero.

See *System Installation Configuration and Updating Guide*.

4.3 FORMS

FORMS describes the procedures for creating and managing single or multiple FORMS, and windowing. Multiple FORMS can be modified or overlaid. To enable forms, declare USE FORMS Clause in TDS Section. Then use the MAINTAIN_FORM utility to declare the following families of terminals for use with forms:

- DKU7005/DKU7007
- DKU7107
- DKU7211
- VIP7801/VIP7804/VIP7814
- VIP8800/TWS2255 (8 BITS terminals)
- IBM 3270
- IBM 3278/3279
- MINITEL
- PC7800.

To adapt messages to terminals supported by the application, specify USE TERMINAL-ADAPTER Clause in TDS SECTION.

See *IOF Programmer's Manual*.

4.3.1 Declaring FORMS in STDS

Access to FORMS is given through procedures using SDPI (Standard Device Programmatic Interface). See *TDS COBOL Programmer's Guide*. These procedures are called in the TPR for:

- handling and displaying the various FORMS
- sending and receiving messages.

When using FORMS, specify the following statement in TDS SECTION:

```
[ USE FORMS [NO IMPLICIT RELEASE ].
```

NO IMPLICIT RELEASE Clause keeps FORMS active between transactions. If NO IMPLICIT RELEASE Clause is omitted, FORMS are implicitly released at the end of each transaction. NO IMPLICIT RELEASE must be specified if the transaction is to receive data using a FORM mounted by a preceding transaction.

Assign the FORM object file and the FORM object libraries. It is not necessary to assign the system file (SYS.HBINLIB) because it is automatically assigned when TDS is executed through the use of a special internal file. The following ASSIGN statements can be included in the JCL for executing the TDS job:

```
[ ASSIGN H_FORM, UFAS-file-description, SHARE=ONEWRITE,
                                        ACCESS=READ; ]
[ ASSIGN BLIB,    binary-library, SHARE=DIR [,ACCESS=READ];]
[ ASSIGN BINLIB1, binary-library, SHARE=DIR [,ACCESS=READ];]
[ ASSIGN BINLIB2, binary-library, SHARE=DIR [,ACCESS=READ];]
[ ASSIGN BINLIB3, binary-library, SHARE=DIR [,ACCESS=READ];]
```

FORMS can be mounted before starting the transaction. If a FORM is mounted by the operator from the local diskette, specify FORM IS Clause in MESSAGE statement of TRANSACTION SECTION. When a FORM is mounted either by another transaction or locally, TDS expects the transaction name in the first 8 characters of the input message.

On a line failure, TDS abort or crash, the previous FORMS is restored on each terminal with its contents.

Figure 4-1 shows the components of the FORMS utility.

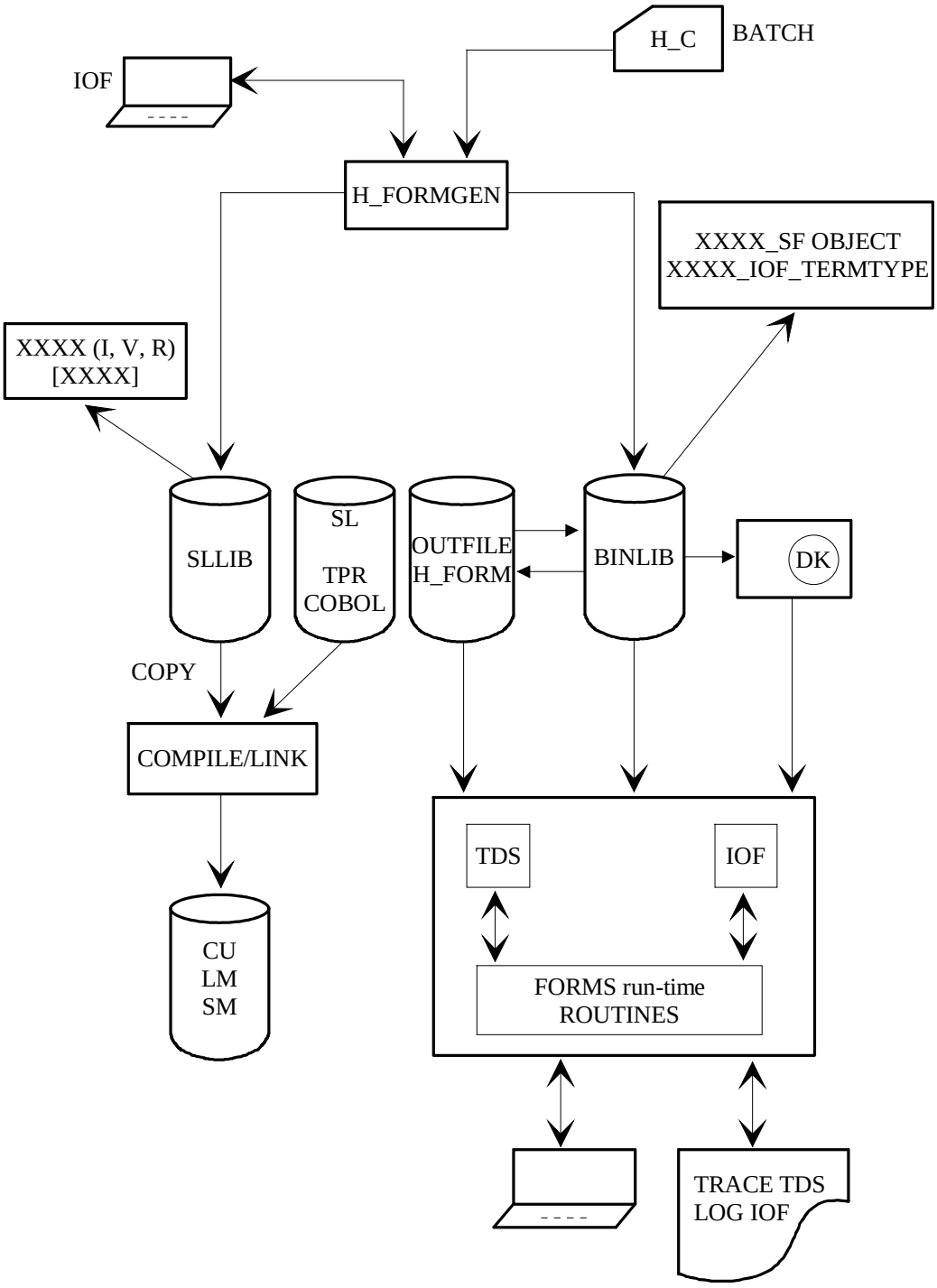


Figure 4-1. The Components of the FORMS Utility

4.3.2 Tracing FORMS

The FORMS runtime package provides diagnostic traces for debugging user applications. Traces are activated through IOF log commands and through the TRACE transaction.

The information obtained consists of:

- a trace of each procedure using FORMS which contains the parameters and the return code of the procedure after its execution
- a trace of messages INBOUND (received) and OUTBOUND (sent) when processed by FORMS.

See TRACE command in *TDS COBOL Programmmer's Guide*.

4.4 User Data Files

Data used by TPRs to process transactions must be stored in offline user data files. The characteristics of these files are:

- File names are up to 24 characters and must be different from the names of TPRs or USE-procedures
- The maximum number of files must not exceed 500
- Specifying the EXTERNAL keyword in the generation FILE CONTROL Clause denotes that the associated *ifn* (internal file name) is visible to TPRs
- The first character of the *ifn* must be a letter, suffixes must not be used.
- The USERJRNL *ifn* is reserved for the user journal and must not be specified in any FILE CONTROL Clause.
- Each FILE CONTROL Clause must be terminated by its own *END.
- Files may be either TDS controlled or non-controlled.

Declare user data files TDS-controlled unless there is a system-specific reason to declare them otherwise. TDS-controlled files appear as MONITORED in the output of the [M] LIST_TDS_FILE master command.

TDS-Controlled Files:

- can be organized only as a UFAS-EXTENDED indexed sequential or relative file, or IDS/II database area
- must have DYNAMIC access mode and can be processed only in INPUT or INPUT-OUTPUT.
- can be protected by Before and After Journals and Deferred Updates
- uses buffers handled as a pool and treated as a system resource for scheduling TPRs
- can be accessed simultaneously by transactions of different commitment units, each of which has its own set of pointers or file currencies

TDS Non-controlled Files:

- can only be protected by Before Journal
- does not use GAC-EXTENDED and the main buffer pool
- cannot be accessed simultaneously by several transactions because TDS maintains only one set of pointers or currencies per file for the whole TDS application
- cannot use the DEBUG option.

4.5 GTWriter

GTWriter is a facility which manages output deliveries and associated terminals. It allows any hardcopy terminal on the local or remote site declared in the network to be used as a printer. The user can request a file or subfile to be printed at any terminal known to GTWriter. Printing is spooled asynchronously, so transaction processing is not affected.

Paper format can be defined per output and/or per terminal. GTWriter uses the same command set for handling outputs as the Output Writer. In addition, there are other commands to handle terminals and to control the operation of GTWriter.

GTWriter prints reports from three files:

- two system SL libraries, SYS.TW.OUT that is mandatory and SITEOUT that is optional
- and from any monovolume private SYSOUT file.

All SSF control records can be used since SSF report formats are fully supported.

See *Generalized Terminal Writer User's Guide*.

4.5.1 Output Requests

Output requests are executed at the commitment time for each output created in SYS.TW.OUT during the last commitment even if the command handler is not running. SYS.TW.OUT is used by the auto-mechanism for TDS. TDS users can create outputs dedicated to GTWriter in SYS.TW.OUT by calling a GTWriter procedure.

SITEOUT is the default library for GTWriter reports if the *efn* (external file name) is omitted from the output request. This file may be used as a general public SYSOUT. Its *efn* is defined when GTWriter is generated.

4.5.2 Composition of GTWriter

GTWriter is composed of:

- the TWGEN load module
- a command handler
- a set of drivers.

GTWriter generation declares various characteristics and defaults for handling reports, terminals and users. The TWGEN module reads the input file which stores this description to create a VMF file to be used by the command handler to initialize GTWriter structures.

The command handler dynamically creates GTWriter commands only when it is started. Until then, these commands are not available to the user.

The drivers manage the communication and printing with terminals in the network. The number of drivers and their dedicated terminals are declared when GTWriter is generated. The drivers are started without operator intervention by the command handler unless NSTART is specified in the generation. In this case, issue the STW command to start the drivers.

4.5.3 Declaring GTWriter at TP7GEN

For a TDS to access GTWriter, a non TDS-controlled file must be declared in the TP7GEN of the TDS. This file:

- must be defined in the INPUT-OUTPUT SECTION as follows:
  - in FILE-CONTROL Clause:

```
SELECT EXTERNAL file-name ASSIGN TO ifn
ORGANIZATION IS SEQUENTIAL WITH SSF VLR
ACCESS MODE IS SEQUENTIAL.
```

  - in FILE-DEFINITION Clause:

```
FD file-name
   LABEL RECORDS ARE STANDARD
   BLOCK CONTAINS 256 CHARACTERS.
01 record-name            PIC X (256)
*END
```

- must be assigned to SYS.TW.OUT in the JCL.

NOTE:

PROCESSING MODE and/or FILE-INTEGRITY need not be specified for this file. If specified, they must be set to INPUT-OUTPUT and NONE respectively.

4.6 Terminal Adapter

The Terminal Adapter is declared in USE TERMINAL-ADAPTER Clause in TDS SECTION. It is a facility which adapts the display of unformatted messages to terminals connected to a TDS application.

It creates and lies on a user profile that defines page settings and ensures that external messages do not interfere with transaction messages. A user profile contains default variables allowing data presentation to be adapted to transactions run by a specific user. Calls to RDPROF (Read Profile) and MDPROF (Modify Profile) allows modifying the user profile. See *TDS COBOL Programmer's Guide*.

As a result, the user terminal receives messages in the same way as a master terminal does when connected to IOF.

This interface handles extended character sets and manages control structures whenever a user logs onto the TDS application. These structures remain present even if the Terminal Adapter is temporarily disabled on a user terminal during the TDS session.

4.6.1 Types of Messages

Messages received by users are either formatted or unformatted:

- Formatted messages are those sent or received through the FORMS utility
- Unformatted messages are any other type of messages including External Messages. External messages are any messages received by the user which were not sent by a transaction, namely:
  - messages from [M] SEND_TDS_USER or [M] MODIFY_TDS_MOT
  - TDS Service Messages
  - SEND messages from other correspondents specifically addressed to a user.

The Terminal Adapter handles the display of unformatted messages either in the context using FORMS or otherwise. When an active form is on, unformatted messages are sent as attention messages to the terminal status line, excepting TDS Service and Transaction messages which may be displayed from the cursor position.

4.6.2 Mode of Message Presentation

The Presentation Mode can be:

- either *Normal* (default) when the user connects to TDS without explicitly specifying that messages are managed by the Terminal Adapter
- or *Format* which must be explicitly declared to enable the user to receive formatted messages (FORMS) while still receiving unformatted (External) messages.

4.6.3 Presentation Type

Presentation Type is only relevant when USE TERMINAL ADAPTER Clause is specified and applies specifically to unformatted messages. It allows the user to determine if the Terminal Adapter is active or not for the terminal.

Presentation Type is independent of Presentation Mode since unformatted messages can be sent to active FORMS and/or terminals in Normal Mode. When the Terminal Adapter is declared in TP7GEN, the interface applies to the entire TDS application. However, a user can override this declaration by specifying the Presentation Type as follows:

- *Free:* when the user suppresses the Terminal Adapter, in which case messages are no longer adapted for the terminal
- *TA (Terminal Adapter):* the user keeps the message adaptation for the given terminal.

The choice of the Presentation Type should follow the rules in the following Paragraph.

4.6.4 Using Terminal Adapter

When the Terminal Adapter is used and an unformatted message is sent, the way it is displayed depends on the type of presentation.

With *Free Presentation*, messages appear at the cursor position and are advanced by the options or control characters specified. Furthermore:

- if the message is longer than the physical screen, the terminal may be disconnected
- if the physical screen can contain the message, but the cursor is not at the top of the screen, the screen is cleared before the message is displayed.

With *TA Presentation*, the ROLL variable determines the type of display:

- if ROLL is specified, the message appears at the cursor position and each new line at the bottom of the page pushes the screen up by one line, erasing the uppermost line
- if ROLL is not specified, message display depends on the type of message and/or where it comes from, whether the transaction is currently running, and the Presentation Mode:

Case 1	The message appears at the cursor position and advances according to the control characters or options specified. If the message surpasses the end of page, the screen is cleared before the message is displayed. If the message is longer that the logical page length, the message is cancelled.
Case 2	Messages up to 62 characters are sent to the terminal status line. These messages are either overwritten by a new message or cleared by TDS when a RECEIVE statement is achieved. Messages exceeding 62 characters may be truncated.

For Cases 1 and 2, if the Terminal Adapter is not supported by the addressed terminal, the message will not be sent and will be cancelled.

The *final* Presentation Type can be determined by reading the following table. The display depends on the origin or type of message and the Presentation Mode. The type of display is also effected by interrupts.

Summary of Presentation Type

Origin/Type of Message	Presentation Mode	
	Format	**Normal**
SEND (from user's TX)	Free Presentation (of a FORM)	Case 1
SEND (from another TX)	Case 2	Case 1 (if no TX running) Case 2 (if TX running)
M SEND M BROADCAST Service Messages	Case 2	Case 1 (if no TX running) Case 2 (if TX running)

Effect of ROLLBACK on Display of Messages

Presentation Type	Presentation Mode	
	Format	**Normal**
TA Presentation	FORM is re-displayed	Screen is cleared at cursor position at commitment point
Free Presentation	FORM is re-displayed	Screen is cleared

Constraints:

- The continuation character at the end of a line, synchronization characters at the end of a page, and function keys are not supported by the Terminal Adapter because these characters could have different meanings on different terminal types.
- The Terminal Adapter is supported only by TM correspondents.

4.7 Master Mailbox

The master mailbox is created at TDS startup. Connecting to the master mailbox allows the TDS operator to use a set of privileged master commands and to receive TDS messages. Only one master operator can logon to this mailbox. The terminal connecting to this mailbox is the master terminal.

The master mailbox can be declared at either NETGEN or TP7GEN. If a master mailbox is defined, no activity is possible until the master operator connects to this mailbox. If a master mailbox is not declared, the submitter of the TDS job is the master operator. The master operator must logon for startup to execute and can subsequently logoff without affecting the current session.

Constraints:

- the user logging on to the master mailbox becomes the master operator
- since only one terminal operator can be connected to any given mailbox, there can be only one master operator
- the master operator must belong to a project with the Authority Code 0 in the Site Catalog
- if the master mailbox is XCP1, only XCP1 exchanges occur between TDS and the master terminal which cannot be reconfigured on a non-XCP1 terminal
- when connected through the master mailbox, the master operator cannot use IOF commands or have an IOF view of application
- GCL commands cannot be entered at the master terminal since TDS does not support GCL; prompts, helps and menus cannot be used and commands submitted from the master terminal must be entered in line mode
- although the master mailbox can receive messages in format mode, it is not recommended because their displays occupy the screen.

5. Implementing the TDS Application

After preparing the TDS files and generating the TDS application, implementing the TDS application involves:

- loading and duplicating sharable modules
- preparing the JCL of the TDS application
- starting and stopping the application
- analyzing output.

5.1 Compatibility between TDS-V3/V5 and TDS-V6/V7

Before loading the sharable modules or launching the JCL, first update TPRs from previous releases. TPRs that run under Release V5 can run under V6/V7. However, TPRs running under Release V3A, must be migrated to Release V6/V7. To do this:

- specify OLDVERS to load an SM containing any LKUs linked under release V3A
- recompile and relink any TPRs containing floating-point data items.

Migrating TPRs from Earlier Releases to V6/V7

ACTION	TPR Compiled and Linked in			
	V3A		V3B/V5	
	WITH FLOATING POINT	WITHOUT FLOATING POINT	WITH FLOATING POINT	WITHOUT FLOATING POINT
Recompile	YES	NO	NO	NO
Relink	YES	NO	NO	NO
Use OLDVERS parameter	YES	YES	YES	YES*

* OLDVERS parameter in the LOAD command of SYSMAINT if one or more LKUs in the sharable module are from Release V3A

5.2 Loading Sharable Modules into Backing Store

Loading SMs (sharable modules) to backing store allows pages to be swapped in and out of main memory and shared by all active transactions. Before starting a TDS session, first load the required modules from the appropriate library into backing store.

5.2.1 Running SYSMAINT Utility

Use the SYSMAINT utility to load the SMs. The JCL for SYSMAINT is as follows:

```
$JOB job-name, USER=userid [,PROJECT=project [,BILLING=billing]];

SYSMAINT COMFILE=*ien;

$INPUT ien;

  { SM | MNSYSSM };

  [ INLIB1 tdsname.SMLIB [ :offline-md:offline-dvc ]
    INLIB2 tdsname.SMLIB [ :offline-md:offline-dvc ]
    INLIB3 tdsname.SMLIB [ :offline-md:offline-dvc ]];

  LOAD MODULE={ * | smname } ,INPUT={ INLIB1 | INLIB2 | INLIB3 }
       [ REPLACE ] [ OLDVERS ];
  [ LOAD...;]
  [ LOAD...;]

  QUIT;
  QUIT;

$ENDINPUT;
$ENDJOB;
```

NOTE:

smname is TPR, TPR1...TPR99 and identifies the SM containing the linked units.

INPUT=INLIB1, INLIB2, INLIB3 are symbolic names of previously assigned libraries from which modules are loaded.

The star convention can be used.

During execution of SYSMAINT, do not issue CANCEL_JOB (CJ).

Since errors detected in the SYSMAINT utility do not always lead to the abort of the SYSMAINT step, correct processing must be checked in its output before launching the TDS step.

SMs cannot be loaded during a TDS session unless they has been deactivated by using another library with [M] MODIFY_TDS SMLIB. If an SM already exists inside the backing store, first cancel it using REPLACE. If REPLACE is omitted, reloading will not occur.

Use OLDVERS to update TPRs compiled and linked prior to Release V3B.

SMs must be reloaded on clean restart because the backing store is reformatted. An SM must be reloaded each time a program is linked to it. However, when TP7GEN is reexecuted not all sharable modules need reloading. Reloading depends on the statement that is modified or inserted in STDS. See Paragraph 5.2.2 *Reloading TPR Sharable Modules into Backing Store*.

EXAMPLE of loading a TPR:

```
R: p
JOBID=MNSYSEAS   USER=BULL2  PROJECT=FLA   BILLING=RES   RON=X0244
----------------------------------------------------------------
10:22:41                                        JOB INTRODUCED FROM
                       DEC 25, 1994
                                               MNSYSEAST EP.V3 RES504
----------------------------------------------------------------
10:22:48   START OF TRANSLATION
          $JOB MNSYSEAST, USER=BULL2, HOLDOUT;
WARNING   126 JOB IDENTIFICATION HAS BEEN TRUNCATED.
          SYSMAINT COMFILE=*MNSYS;
          $INPUT MNSYS PRINT;
          SM;
          INLIB1 EAST.SMLIB:RES504:MS/M500;
          LOAD TPR INLIB1 REPLACE;
          QUIT;
          QUIT;
          $ENDINPUT;
          $ENDJOB;
          RECORD COUNT: 10
10:22:48  END OF TRANSLATION
R:
CANCEL(0) ,HOLD(1) ,RELEASE(2) ,LOCAL PRINT(3) ,SCAN(4): 1
C: scan x244:2
   X244:2:1
   CD:DEC 25, 1993  10:22:48
   LINES:       18  PAGES:     1
```

```
R: p
MNSYS    15.00   X244.1      MNSYSEAS     BULL2     FLA
  10:22:52   DEC 25, 1993      PAGE        1
****************************************************************
****************************************************************
**** GCOS 7                                                    ****
****                                                      MNSYS  ****
****                                                           ****
****                                                           ****
VERSION: 15.00  DATED: APR 01, 1992                           ****
*****MNSYS*****************************************************
****************************************************************
C: SM;
+++
M: INLIB1 EAST.SMLIB:RES504:MS/M500;
M: LOAD TPR INLIB1 REPLACE;
>>> LOADING MODULE TPR FROM EAST.SMLIB/RES504 SUCCESSFUL
                      ONTO SYS.LIBX/
  CR : 06/01/93 11:20  MD : 07/31/93 16:12  MV :   /  /     :
M: QUIT;
C: QUIT;
***LAST LINE
R:
```

❑

5.2.2 Reloading TPR Sharable Modules into Backing Store

Clauses modified or inserted in STDS determine if the SMs will be loaded or not. There are three groups of clauses. In the first type, SMs are not reloaded. Other clauses reload only the TPR SMs that use the modified statement. If modifying *length* statements, reload all SMs. A third type reloads all the TPR SMs.

Reloading TPR Sharable Modules into Backing Store

Clause modified or inserted in STDS	Action
NUMBER OF TERMINALS	Do not reload SMs
NUMBER OF DUMMY CORRESPONDENTS	Do not reload SMs
NUMBER OF VIRTUAL	Do not reload SMs
SIMULTANEITY	Do not reload SMs
RPC SIMULTANEITY	Do not reload SMs
CD-IN DATE FORMAT	Reload only SMs using modified statement
RESERVE AREAS	Do not reload SMs
ATTACH SHARABLE MODULE	Reload only SMs using modified statement
MAXIMUM NUMBER OF DYNAMICALLY ADDED TRANSACTION	Do not reload SMs
NUMBER OF SHARABLE MODULES	Do not reload SMs
NUMBER OF IQS JPPC SEGMENTS	Do not reload SMs
DYNAMIC-SUPERVISION	Reload only SMs using modified statement
COMMON-STORAGE	Do not reload SMs
DEFAULT TRANSACTION-STORAGE	Reload only SMs using modified statement
MAXIMUM TRANSACTION-STORAGE	Reload only SMs using modified statement
PRIVATE-STORAGE	Reload only SMs using modified statement
MAXIMUM PRIVATE-STORAGE	Reload only SMs using modified statement
MESSAGE-LENGTH	Reload only SMs using modified statement
FORMATTED MESSAGE-LENGTH	Reload only SMs using modified statement
USER-JOURNAL	Reload only SMs using modified statement
TPR-TIME-LIMIT	Do not reload SMs
MAXIMUM NUMBER OF TPR WITH NO UNMAPPING	Do not reload SMs
DEFAULT ABORT TPR NAME	Reload only SMs using modified statement
IDLE-TIME	Do not reload SMs
MASTER MAILBOX	Do not reload SMs
XCP1 MAILBOX	Reload only SMs using modified statement
NUMBER OF XCP1 TRANSACTIONS AUXILIARY SESSIONS	Reload only SMs using modified statement
NUMBER OF XCP2 CONVERSATIONS	Do not reload
SELECT XCP2-SYNCPEVT	Do not reload
MAXIMUM XCP2-WAIT TIME	Do not reload
HOST GATEWAY	Do not reload

Clause modified or inserted in STDS	Action
XA-RESYNC-DELAY	Do not reload
TCP-IP PROTOCOL	Do not reload
USE LAST MESSAGE	Do not reload
USE MENU	Do not reload
USE NOPWCHK-ALLOWED	Do not reload
USE FREE-ACCESS-TDS	Do not reload
USE PROCEDURE FOR TRANSACTION INITIALIZATION	Do not reload
USE FORMS	Reload all SMs
USE TERMINAL-ADAPTER	Do not reload
USE ORACLE	Do not reload
USE STARTUP	Do not reload
USE SHORT-STATISTICS	Do not reload SMs
USE CONVENTIONAL-ABORT-TPR	Do not reload SMs
USE M-MASTER-CONVENTION	Do not reload SMs
USE SPAWNED-TX-MESSAGE-LOGGING	Do not reload SMs
USE PASSIVE-SPAWN-CHECK	Do not reload SMs
USE LARGE-STACK-SEGMENT	Rload all SMs
USE DEVICE-HEADER-LOGGING	Do not reload SMs
USE IDS-SUB-SCHEMA	Do not reload SMs
USE CONNECTION-LOGGING	Do not reload SMs
USE TILS-SIMULATION	Do not reload SMs
USE NO SCROLLING REQUEST	Do not reload SMs
USE EDIT-EXCEPTION	Do not reload SMs
USE WAIT_VIRTUAL_FOR_SHUTDOWN	Do not reload SMs
USE DISPLAY IN JOR	Do not reload SMs
USE TWRITER-NUMSF-xxx	Do not reload SMs
USE XCP1-CLOSEPOOL-STRONG	Do not reload SMs
USE XCP1-ALNTC-PRIMARY-ONLY	Do not reload SMs
USE Procedure	Reload all SMs
TDSTX-MESSAGE	Do not reload SMs
SPECIAL-CHAR	Reload all SMs
CANCELCTX AT RECONNECTION	Do not reload SMs
FILE OPENING OPTION	Do not reload SMs
FILE-CONTROL	Reload all SMs
FILE DEFINITION	Reload all SMs
TDS-FILE-DEFINITION	Reload all SMs
IDS-DEFINITION	Reload only SMs using modified statement
DB and REALMS	Reload only SMs using modified statement
Maximum Number of Schema	Reload only SMs using modified statement
IQS-DEFINITION	Reload only SMs using modified statement

Clause modified or inserted in STDS	Action
PROCESSING-CONTROL	Reload only SMs using modified statement
WORKING-STORAGE	Reload only SMs using modified statement
SHARED-STORAGE	Reload only SMs using modified statement
CONTROLLED COMMON-STORAGE	Reload only SMs using modified statement
MESSAGE ASSIGN DEBUG INQUIRY	Do not reload SMs
IMPLICIT COMMITMENT	Do not reload SMs
XCP2 SERVICE	Do not reload SMs
HEURISTIC	Do not reload SMs
NO-DEFER-RESYNC	Do not reload SMs
NO-RESTART	Do not reload SMs
RPC SERVICE USED	Do not reload SMs
XA SERVICE USED	Do not reload SMs
CMA SERVICE USED	Do not reload SMs
SUPPRESS BEFORE JOURNAL	Do not reload SMs
SUPPRESS DEFERRED UPDATES	Do not reload SMs
USE DEFERRED UPDATES	Do not reload SMs
SUPPRESS CONCURRENT ACCESS CONTROL	Reload only SMs using modified statement
SHARED READ	Reload only SMs using modified statement
MANUALLY NON-CONCURRENT	Reload only SMs using modified statement
CLASS	Do not reload SMs
PRIORITY	Do not reload SMs
FORM	Do not reload SMs
AUTOMATIC UNMAPPING	Do not reload SMs
ACCOUNTING	Do not reload SMs
COMMITMENT	Do not reload SMs
AUTHORITY	Do not reload SMs
HIDDEN	Do not reload SMs
PROMPT	Do not reload SMs
TRANSACTION-STORAGE	Reload only SMs using modified statement

The maximum size allowed for loading of private data segments is 65,504 bytes. Beyond this limit, loading aborts with the return code SNDARERR and the message:

```
*** UNABLE TO CREATE WORKING SEGMENT ..."
```

5.2.3 Unloading Sharable Modules

SMs can be unloaded from backing store by this JCL command sequence:

```
SYSMAINT COMFILE=*ien;
$INPUT ien;
  { SM | MNSYSSM };
  UNLOAD MODULE=smname, EFN=efn;
  [ UNLOAD...;]
  QUIT;
  QUIT;

$ENDINPUT;
```

NOTE:
efn is the name of the file from which the module has been previously loaded.

EXAMPLE of unloading a TPR:

```
 10  $JOB T4UNLO ,HOLDOUT ,JOR=ABORT;
 20
 30  SYSMAINT COMFILE=*UNL1 ,PRTFILE=DUMMY;
 40  JUMP ERR,SEV GE 3;
 50  SEND 'T4UNLO COMPLETED';
 60  JUMP FIN;
 70  ERR:
 80  SEND 'T4UNLO ABORTED';
 90
100  FIN:
110
120   $INPUT    UNL1 ,JVALUES ,PRINT;
130   MNSYSSM;
140      UNLOAD MODULE=TPR    ,EFN=T4.&1;
150      STATUS RESET;
160      UNLOAD MODULE=TPR1   ,EFN=T4.&1;
170      STATUS RESET;
180      UNLOAD MODULE=TPR2   ,EFN=T4.&1;
190      STATUS RESET;
200      UNLOAD MODULE=TPR10  ,EFN=T4.&1;
210      STATUS RESET;
220      QUIT;
230      QUIT;
240   $ENDINPUT;
250
260   $ENDJOB;
```

❑

5.2.4 Duplicating Sharable Module Libraries

To modify SMs online, first create other SM libraries. See Paragraph 9.5.19 *MODIFY_TDS SMLIB*.

Duplicating a library involves first allocating a new library, then duplicating the SM.

To allocate a library issue:

```
LIBALLOC SM (smlibname,dvc,md,SIZE=(3,1)),DIRSIZE=5;
```

To duplicate a sharable module:

```
$JOB DUPLISM, HOLDOUT;

  LIB SM,INLIB1=( tdsname.SMLIB,...);

  LIBMAINT SM LIB=( smlibname ,dvc ,md),

  COMMAND='MOVE INLIB1:TPR[nn],REPLACE;';

$ENDJOB;
```

where TPR[*nn*] = TPR, TPR1 ... TPR99.

In the above example, the sharable module is moved from the SM library allocated by the TP7PREP utility to a user-defined SM library.

5.2.5 Using Multiple Sharable Module Libraries

Multiple libraries allow adding or modifying TPRs while a TDS session is running. This feature is useful for debugging while online because it substitutes one version of a TPR for another. For this reason, at least two SM libraries on the TDS application are useful.

TDS has access only to the images in the backing store in use; it cannot access multiple libraries simultaneously. Images can be updated in one library then the TDS search rules can be modified using [M] MODIFY TDS_SMLIB.

First define the SM libraries in the JOBLIB statement, for example:

```
JOBLIB SM,TDS1.SMLIB,DEBUG1.SMLIB,DEBUG2.SMLIB;
```

TDS1.SMLIB is the SM library in use, and DEBUG1.SMLIB and DEBUG2.SMLIB contain later versions of the TPRs.

Ensure that any updated TPRs are linked to each library. Then modify the search path:

```
[M] MODIFY_TDS SMLIB old-path-name new-path-name.
```

5.3 Starting a TDS Session

The TDS session is started like any other GCOS 7 job either through the Input Reader, or if the job description is stored in a JCL library, through EJR.

The same master operator in the previous TDS session can end a TDS session either normally or forcibly, and then perform a warm restart. At warm restart, an ordinary user (that is, not a master operator) known by the TDS application at the previous shutdown time (even in frozen state), cannot become the master operator. If this is attempted, the TDS application is aborted with the MU02 error message with the DUPNAME return code.

Restrictions

1. TDS Submitted by a Station Operator whose IOF session is started in class "U".
   - If the TDS is started by the "first" user connected under the OPERATOR project with the Station attribute and whose name is a site name (that is, not MAIN), the TDS may abort with the MU02 message "UNSUCCESSFUL TDS TABLE ENTRY CREATION with the USERUNKN return code. The IOF session of such a user is started in class "U". There is no abort if the value of the Station attribute is MAIN.
   - There is no abort if the TDS is started by a user other than the "first" user (with the above attributes). However, there may be problems receiving messages and/or sending TDS commands if this other user (not the "first" user) is disconnected and then reconnected. This other user may become the "first" at reconnection if the previous "first" user has also been disconnected.

   Since the above results are difficult to predict (and are undesirable), you are recommended that the TDS is **never started** by such a user.
2. TDS Submitted by a user under a project that is not the user's default project.
   - Unexpected results may occur if a TDS is started by a user under a project that is not the user's default project.
   - If the TDS application is not "known" to the user's default project, authority-codes are set to 'FFFFFFFF'. Otherwise, authority-codes are taken from the default project instead of from the project used for the connection.

5.3.1 JCL for Running TDS Job

The JCL syntax for starting the TDS job is:

```
$JOB job-name, USER=userid [,PROJECT=project [,BILLING=billing] ]
               [,CLASS=J ] [,HOLDOUT ]...;
JOBLIB SM,tdsname.SMLIB [,smlibname-2 [,smlibname-3]];
[ATTACH.............;]
STEP tdsname,
  FILE=({ TDS.LMLIB | libname } [,DVC=offline-dvc,MD=offline-md] )
  [,REPEAT ] [,DUMP={ DATA | ( DATA PRIVATE ) } ]
  ,OPTIONS='[COLD][CN=NONE][SPAWN=KEEP][HA='&HA'][,MINMEM] ';
SIZE declared-working-set, POOLSIZE=poolsize, NBBUG=nbbuf;
[ ASSIGN H_FORM, ufas-file-description [,SHARE=ONEWRITE]
                                       [,ACCESS=READ];]
[ ASSIGN BLIB, binary-library, SHARE=DIR [,ACCESS=READ ];]
[ ASSIGN BINLIB1, binary-library, SHARE=DIR [,ACCESS=READ ];]
[ ASSIGN BINLIB2, binary-library, SHARE=DIR [,ACCESS=READ ];]
[ ASSIGN BINLIB3, binary-library, SHARE=DIR [,ACCESS=READ ];]
[ ASSIGN H_BJRNL, DVC=dvc, MEDIA= (md-1[,md-2[,md-3[,md-4]]]),
                           FILESTAT=TEMPRY, NEXT, POOL; ]
[ ASSIGN H_BJRNL1, DVC=dvc, MEDIA=(md-a[,md-b[,md-c[,md-d]]]),
                           FILESTAT=TEMPRY, NEXT, POOL; ]
[ ASSIGN DBUGFILE,tdsname.DEBUG [,DVC=online-dvc, MD=online-md]
                           SHARE=DIR; ]
[ ASSIGN ifn [,SHARE=sharing-type ] [,ACCESS=access-type ]...; ]
[ ASSIGN H_GM,tdsname.GMEM [,DVC=dvc, MD=md], ACCESS=SPWRITE
                           SHARE=MONITOR [,assign-parameters];]
[ ASSIGN USERJRNL, DUMMY; ]
[ DEFINE USERJRNL, JOURNAL=AFTER; ]
DEFINE H_CTLM, JOURNAL=BEFORE;
[ DEFINE ifn, JOURNAL={ BEFORE | AFTER | BOTH } ;]
[ DEFINE H_GM, JOURNAL={ BEFORE | BOTH } ; ]
[ $SYSOUT H_DPPR, WHEN=IMMED; ]
ENDSTEP;
$ENDJOB;
```

NOTE:

Mandatory statements are in **bold**.

There are no TDS-specific limitations on using the JCL.

Enclosures may be assigned to non-controlled files or for some specific files as the IDS/II run-time options or the Program Checkout Facility command file.

Multi-volume files must be cataloged.

5.3.2 Mandatory Statements in TDS Job

The three mandatory statements are:

- JOBLIB
- STEP
- SIZE.

5.3.2.1 JOBLIB Statement

JOBLIB specifies the SM libraries used during the TDS session. When an SM is not found in the first library, the other two are searched in the order specified.

[M] MODIFY_TDS SMLIB is used to dynamically change the order of libraries in JOBLIB.

5.3.2.2 STEP Statement

tdsname identifies the TDS load module specified in PROGRAM-ID of TDS SECTION.

libname specifies the library containing the TDS load module which can be preinitialized.

- REPEAT repeats the TDS step after a TDS or system failure so that all files remain available to the TDS application and will not be assigned to another job.
- DEBUG must not be used even if the "TRACE PRINT PCF", "TRACE PRINT XPCF" commands are used.
- Using the CPTIME parameter is ***not recommended*** since certain of its values may block the TDS step. If the remaining CPTIME is less than TPR-TIME-LIMIT declared in STDS, neither TPRs nor Master commands can execute.
- DUMP takes values:

  DATA = segments are type 0, 1, 2, and 3.

  DATA PRIVATE = segments are type 3 and some type 2 for debugging.

  If TDS aborts, all types of segments will be found in the dump of process P=0, regardless of the dump option specified.

- OPTIONS takes values:

COLD	cold restart of TDS application, overriding WARM in [M] MODIFY_TDS_RESTART_OPTION or [M] TERMINATE_TDS for previous TDS session.
CN=NONE	no correspondent is reconnected or allowed to connect until [M] ALLOW_NEW_TDS_COR is issued.
SPAWN=KEEP	spawned transactions are kept at restart COLD instead of being erased. All deferred or immediate spawned transactions are still available after the restart cold. Note: if the SPAWN=KEEP functionality has been requested at generation time, the first start of TDS after generation will keep the spawned transactions even if the SPAWN=KEEP string has not been set in the step option chain.
HA='&HA'	system configured with High Availability. See *High Availability Administrator's Guide*.
MINMEM	see Paragraph 7.4.3.

5.3.2.3 SIZE Statement

SIZE defines memory allocated to the step. *dws* (declared-working-set) specifies the amount of memory in Kbytes. POOLSIZE optimizes the TDS buffer pool.

5.3.3 Optional Statements

The three optional statements are:

- ATTACH
- ASSIGN
- DEFINE.

5.3.3.1 ATTACH Statement

ATTACH specifies the catalogs associated with user files. Cataloged files are referenced through ASSIGNs or by [M] OPEN_TDS_FILE.

5.3.3.2 ASSIGN Statement

ASSIGNs specify all files used during a TDS session. ASSIGN statically assigns a file to the TDS step. TDS startup opens only user files defined in TP7GEN. For TDS restart, the effect of *assignment* may be different.

- On WARM restart of a TDS application:
  - files assigned at the end of the TDS session remain assigned
  - files de-assigned at the end of the TDS session, are assigned as in the ASSIGNs.
- On COLD restart of a TDS application all files are assigned as defined in the ASSIGNs. Files can be dynamically assigned and de-assigned during the session through [M] OPEN_TDS_FILE and [M] CLOSE_TDS_FILE.
- Note the following ASSIGN parameter values:

SHARE=ONEWRITE	one job (TDS or batch) is writer and several jobs are readers.
SHARE=MONITOR	for TDS-controlled files and IDS/II areas if shared by several applications.
ACCESS=SPREAD ACCESS=SPWRITE	if the file is used exclusively by the TDS step. Several internal file names cannot be assigned to the same external file name for files in exclusive mode.

DUMMY	means that the file to be assigned has no actual existence and that no volume supporting the file is required. This parameter may be used to allow checking and testing the transactions and TPR file accesses or to simulate some files needed by a package, while the files are not yet allocated. Files assigned with DUMMY must obey the rules specified by access methods. They must follow the integrity rules such as defined at 5.3.3.3 (DEFINE statement). If any inconsistency is detected by TDS at file opening time (mainly vs. FILE-INTEGRITY defined at generation time), a TX64 error message will be issued. Refer to *JCL Reference Manual* for more details about the DUMMY parameter.

- ASSIGN H_FORM declares the FORM description is used in the UFAS FORMS file.
- ASSIGN H_GM must be specified if using IQS under TDS.
- ASSIGN USERJRNL, DUMMY is mandatory if the User Journal is used.
- ASSIGN H_BJRNL and ASSIGN H_BJRNL1 are optional:
  - ASSIGN H_BJRNL specifies all the volumes used by the Before Journal.
  - ASSIGN H_BJRNL1 lists Before Journal backups if an extension on H_BJRNL fails.

Before Journals can be explicitly assigned for sites to run in unattended mode once the TDS session is started. If these files are not explicitly assigned in the JCL, the operator replies to the JL07 and JL09 messages as in previous releases.

For the first session TP7GEN:

```
JL07    ron SPACE NEEDED FOR BEFORE JOURNAL FOR tdsname.
        "WHERE?".
```

For subsequent sessions:

```
JL09    ron BEFORE JOURNAL FOR tdsname ON media-list ?
```

5.3.3.3 DEFINE Statement

DEFINE defines the journals used by a TDS application for a given file.

To use a User Journal, ensure that the JCL:

- omits DEFINE H_SWAP, JOURNAL=AFTER
- and includes DEFINE USERJRNL, JOURNAL=AFTER if no files are protected by the After Journal.

No dynamic command can modify journal allocation. Files which are not open because of incorrect or missing DEFINEs are unavailable for the duration of the TDS session.

At least one DEFINE ifn, JOURNAL=BEFORE must be included unless an assigned user file is cataloged with JOURNAL=BEFORE.

Files are protected by the file security level defined in the JOURNAL parameter:

BEFORE	TDS uses the Before Journal as a MEDIUM security for TDS-controlled files.
AFTER	TDS uses the After Journal and Deferred Updates as a HIGH security for TDS-controlled files. Before Journal is mandatory because it is required by the H_CTLM file.
BOTH	TDS uses either the Deferred Updates and the After Journal, or the Before and After Journals as security for TDS-controlled files (level of security is HIGH).

For full file protection and recovery, define the journal options in the catalog.

If DEFINE conflicts with the FILE INTEGRITY Clause of the INPUT-OUTPUT SECTION, the corresponding file will not be opened.

DEFINE H_CTLM JOURNAL=BEFORE is mandatory if *tdsname*.CTLM file is not cataloged.

5.3.4 Simulating a Terminal in TDS Job

The batch utility, H_TDSCTP, simulates a terminal via an input and an output file. This step may be part of any job. See Section on *Debugging Using Batch Programs* in the *TDS COBOL Programmer's Guide*.

5.3.5 Stopping the TDS Application

The TDS session terminates in one of two ways:

- normally with [M] TERMINATE_TDS
- or exceptionally with CANCEL_JOB (CJ) *tdsron*.

5.4 Output of a TDS Session

The JOR gives the results of an executed TDS job. The TDS Statistics report lists the global statistics of TDS. TDS Real-time Statistics displays selected information.

When a private sysout is used for a given TDS:

before TS 7560/8560, the TDS Statistics Report contains the information related to the last TDS session. Previous TDS Statistics Report is not kept.

Since TS 7560/8560:

if the TDS is started after the abort of the previous session, the TDS Statistics Report of the previous TDS session is kept.

If the TDS is started after the normal termination of the previous session, TDS Statistics Report of the previous TDS session is not kept.

5.4.1 Analyzing the JOR

All master commands of the TDS application are written in the JOR. A section of this report explains the statistics that may appear for UFAS-EXTENDED files.

```
>>> EFN=external file-name
    GETCICOUNT=a        HITCOUNT=b        IOCOUNT=c

==> POOL=pool-name
    NBFILES=d           NBBUF=e           GETCICOUNT=f
                                          HITCOUNT=g

>>> XUFAS STEP STATISTICS                 STEP=step-name
    POOLSIZE=h          USED SIZE=i       NBPOOLS=j
    SEGCR=k             SEGDL=1           ADDPATHCR=m
                                          ADDPATHDL=n
    READIOCT=o          WRITEIOCT=p       USEDAP=q
```

Parameters:

GETCICOUNT	total number of accesses to Cls on disks or in the buffer pool.
HITCOUNT	number of CIs accessed without a physical I/O operation.
IOCOUNT	number of physical I/O requests, an I/O request for each CI.

NBFILES	*Meaningful only if buffer pools have been defined:* maximum number of files simultaneously opened in a given pool.
NBBUF	maximum number of buffers declared for the pool.
POOLSIZE	declared amount of memory in bytes dedicated to buffers in the step.
USED SIZE	*poolsize* actually used. *used size* < *poolsize*.
NBPOOLS	maximum number of simultaneously opened pools.
SEGCR	number of buffers created.
SEGDL	number of buffers deleted.
ADDPATHCR	number of address paths to buffers created.
ADDPATHDL	number of address paths to buffers deleted.
USEDAP	maximum number of address paths simultaneously used.
READIOCT	number of read I/O operations.
WRITEIOCT	number of write I/O operations.

NOTE:
READIOCT and WRITEIOCT equal the sum of IOCOUNTs for all pools.

5.4.2 Analyzing TDS Statistics

See Appendix D for an Example of the *TDS Statistics Report* with its description.

FILE_OPENING_REPORT contains the messages sent by TDS when a job starts and gives the responses to file opening messages.

EXECUTION_REPORT provides information about the users:

- LAST_TERM: the terminal name associated with the user at logoff
- TX_COUNT: the number of transactions executed since the last connection
- TPR_COUNT: the number of TPRs executed since the last connection.

A line is printed for each user who before TDS shutdown:

- logged off the TDS application using BYE
- or was cancelled as a result of operator action
- or was abnormally disconnected
- or was disconnected due to normal TDS shutdown on [M] TERMINATE_TDS.

GENERAL STATISTICS gives information on the transactions activated during the session and the TPRs executed.
It is normal for the number of transactions and TPRs as reported in the general statistics, to be greater than the sum of the corresponding values as reported in the JOR (that is, for TX_COUNT and TPR_COUNT). For example, you will see that the following transactions (and associated TPRs) are always taken into account in the general statistics, but not in the JOR:

- system transactions such as M or BYE
- transactions spawned towards a dummy correspondent
- transactions involving FOR INQUIRY.

Similarly, you will find that the overall number of transactions and TPRs as reported in the TDS session accounting record is greater than the number of transactions and sessions as reported in the user session accounting record.

Operations on files or database areas are also provided. The number of conflicts is a count of the pages requested and held between TPRs by a commitment unit.

ABORT STATISTICS provides information concerning:

- ***Job Termination:***
  - TDS aborts
  - the cancellation of the TDS job by CJ
  - the stopping of a TDS application by issuing TERMINATE_TDS STRONG=1.
- ***Buffer Overflows:*** the number of TPRs aborted because of a lack of buffers. Transactions were restarted by TDS with the Before Journal instead of Deferred Updates, serialized after restart, or aborted.
- ***Deadlocks:*** the number of commitment units aborted due to return code DEADLOCK from GAC. The commitment units aborted because DIE-WAIT mechanisms are not taken into account. High values indicate that the responsible transactions should be made non-concurrent with each other or with other transactions.
- ***Locked Page Table:*** the number of aborts resulting from a locked page table of inadequate size. Transactions aborted in this way are restarted with a number of locked page entries exclusively reserved for them.

DEBUG INFO is printed when a TDS aborts or CANCEL_JOB is issued. A line is printed for each user who is not frozen at the time of shutdown, giving:

- USER_NAME the user identification
- TX_NAME the transaction activated by the user
- STATE the state of execution of the user's transaction.

5.4.3 Using TDS Realtime Statistics

TDS Realtime Statistics allows requesting global or specific data about the application, files or correspondents. This report gives the latest detailed data of a running application.

Previously this type of information had to be requested by master commands such as:

[M] DISPLAY_TDS
[M] LIST_TDS_FILE
[M] DISPLAY_TX
[M] LIST_TDS_COR

These commands must be submitted from and returned to the master terminal.

TDS Real-time Statistics may be requested inside a TPR to be displayed on the terminal connected.

To realize this function, use the COBOL call:

`"CALL GET-TDS-STAT" USING data-name1, ....`*`data-namen`*`.`

where *data-name* is the name of a structure to pass input parameters or to get output reports and *n* ranges from 1 through 15.

The first structure **TRS-CONTROL** is dedicated to passing input parameters such as the type of reports to be produced. This structure is the only one that requires most of the fields to be initialized before TRS (TDS Real-time Statistics) is called.

TRS-STATUS Informs the calling TPR of possible errors in the call to TRS:

0: Call to TRS is successful, TDS data requested is returned.

1: Value(s) in at least one of the fields NB-GBLK, NB-UBLK, NB-FBLK, MAX-USERS or MAX-FILES are negative.

2: Value in NB-GBLK in TRS-CONTROL is not 0 or 1.

3: Too many parameters passed to TRS. At least one field NB-GBLK, NB-UBLK or NB-FBLK exceeds the TRS limits.

4: Too few parameters passed to TRS. The number of parameters should be
1 + *nb-gblk* + *nb-ublk* + *nb-fblk*.

5: Either *nb-uent* < 0 when USERINFO is requested - or *nb-fblk * nb-fent* < *max-files*.

6: Either *nb-uent* < 0 when FILEINFO is requested - or *nb-fblk * nb-fent* < *max-files*.

2: Structure passed to TRS is too small to receive all data requested. The *occurs* values for USERINFO and/or FILEINFO differ from those in MAX-USERS and MAX-FILES.

TRS-SUB-STATUS Field indicating control structure where an error has occurred:

1: GENERAL control structure
2: USER control structure
3: FILE control structure.

NB-GBLK If TRS-GENERALINFO structure is passed to TRS. Values 0 and 1.

NB-UBLK Number of TRS-USERINFO structures passed to TRS.

NB-UENT Number of entries in each TRS-USERINFO structure. All TRS-USERINFO structures have same number of entries.

SELECT-USERS Selects users about whom TRS returns data. Only a single trailing star in star-convention is supported: A* and ABC* are valid, but A*B and *B are not. Data returned as follows:

– Spaces same as "*" - all users
– A full *userid* without star - only specified user
– Star-convention name - users matching the convention. Number of users must be compatible with *max-users:* if A* returns 20 users but MAX-USERS sets limit to 10, only data on the first 10 users.

MAX-USERS Calling TPR defines maximum number of users on whom detailed data is required in TRS-USERINFO structure. If no data, set NB-UBLK, NB-UENT and MAX-USERS to 0.

```
nb-ublk * nb-uent => max-users
```

NB-USERS	Actual number of users on whom detailed data is retrieved. Rest of entries up to MAX-USERS are cleared by TRS.
NB-FBLK	Number of TRS-FILEINFO structures passed to TRS.
NB-FENT	Number of entries in each TRS-FILEINFO structure.
SELECT-FILES	Selects files about which TRS returns data. Only a single trailing star in star-convention is supported: A* and ABC* are valid, but A*B and *B are not. Data returned as follows: – Spaces same as "*" - all files – A full *filename* without star - only specified file – Star-convention name - files matching the convention. Number of files must be compatible with *max-files:* If A* returns 20 files but MAX-FILES sets limit at 10, only data on the first 10 files.
MAX-FILES	Calling TPR specifies maximum number of files on which detailed data is required in TRS-FILEINFO structure. If no data, set NB-FBLK, NB-FENT, and MAX-FILES to 0. `nb-fblk * nb-fent => max-files`
NB-FILES	Actual number of files on which detailed data is retrieved. Rest of entries up to MAX-FILES are cleared by TRS.

If specified in TRS-CONTROL, the **TRS-GENERALINFO** structure reports data on the TDS application.

TDS-NAME	Name of TDS application in which the TPR runs.
TPR	Number of TPRs executed.
TPR-ABORT	Number of TPR aborts.
TX	Number of transactions executed.
TX-ABORT	Number of transaction aborts.
EXCHANGE	Number of exchanges (message dialogs).
COMMIT	Number of commitments completed.
CPU-MIN	Total CPU time in minutes, used by all TPRs in the TDS.

CPU-MILSEC	Total CPU time in milliseconds, used by all TPRs in the TDS.
ELAPSED-MIN	Total elapsed time in minutes, of all TPRs up to now.
ELAPSED-MILSEC	Total elapsed time in milliseconds, of all TPRs up to now.
USERS	Total number of users (connected/frozen) currently known to TDS.
FROZEN	Number of users from total in USERS abnormally disconnected from TDS but whose contexts are kept to allow them to resume at logon.
TOTAL-FILES	Total number of files opened and closed in the application.
CUR-MEM-AREAS	Current number of TDS memory-areas.
MAX-MEM-AREAS	Maximum number of configurable TDS memory-areas.
BUFOV	Number of aborts caused by buffer-overflows.
LONGWAIT	Number of aborts by access conflicts due to longwaits.
TABOV	Number of aborts caused by lock table-overflows.
DIRTY-READ	Number of aborts caused by IDS *dirty-reads*.
WDNAV	Number of aborts due to unavailable deferred writes.
DEADLOCK	Number of TPR aborts due to the return code DEADLOCK received from GAC.
SERIAL	Number of times TDS serialized such as due to [M] CLOSE.
NON-CONC	Number of times any user waited for a transaction non-concurrent with those in execution.
TIMELIMIT	Current TPR CPU time-limit at which the TPR aborts.
IDLE-TIME	TDS idle-time at which a user remains at *command level* without entering anything before being logged-off from TDS.
MAX-TERMS	Maximum number of terminals connectable to a running TDS.

CUR-SIMU Current TDS simultaneity level.

FRZ-SIMU Number of simultaneities not used by TDS, namely, those suspended by [M] MODIFY_TDS SIMUL=N.

COMMON-SIZE Size of non-controlled common-storage available in TDS.

PRIVATE-SIZE Size of private-storage in transaction-storage.

MAX-TX-SIZE Size of largest transaction-storage for setting size of swapfile.

HA-TDS High Availability or non-HA TDS:

`"H"`	HA TDS *watched by CMSC* (Complex Management Service)
`" "`	non-HA TDS.

BROADCAST-SIZE Length of current broadcast message. 0 if none exists.

BROADCAST-MESS Broadcast message sent by [M] MODIFY_TDS_MOT.

SMLIB_NM Current sharable module libraries assigned to TDS in the order in which they are searched.

If specified, the **TRS-USERINFO** structure reports details on TDS users. Several TRS-USERINFO structures can be used. Declare MAX-USERS to print the user report, then specify users in SELECT-USERS of TRS-CONTROL to customize the report.

USER-TYPE Type of User:

`" "`	normal user
`"B"`	batch-interface user
`"I"`	master operator as user connected via IOF
`"D"`	dummy correspondent
`"X1"`	XCP1 correspondent
`"X2"`	XCP2 correspondent.

USERID User identification.

STATE Current state of user:

1: Processing - executing a TPR
2: End TPR - between TPRs, next TPR has not been started
3: Command - not running a transaction, READY or IDLE
4: Frozen - abnormal disconnection including $*$DIS
5: Blocked - blocked due to non-concurrency
6: GAC Wait - waiting for GAC after abort on LONGWAIT, DEADLOCK, TABLV or ENQUE
Transient - in a state of flux or between states
7: Unlogged - user unlogged after master command
8: M CANCEL
CMG Wait - waiting commitment unit restart
9: after abort on WCNAV, BUFNBOV, CMWSOV, ITMNAV or ENQUE. ENQUE received at TPR start if another commitment unit requests serialization.

SEND-LEVEL Type of event that TDS expects:

- for EMI: an acknowledgement of the sent message
- for EGI: the next input from the terminal.

1: No physical send - no send or only send ESI
2: Send EMI and no acknowledgement received
3: Send EGI and the next terminal input awaited.

WAITTIME If previous TPR set wait-time in TDS-STORAGE before completing:

`" "` wait-time not set
`"W"` wait-time set (determines start of NEXT-TPR).

SERIALIZE If user executes alone (serializes) or waits for all active commitment units to finish before executing alone.

`" "` not serializing
`"S"` serializing or waiting to serialize.

BLOCKING If specified user blocks other users by running a transaction non-concurrent with itself or with other transactions.

`" "` not blocking other users
`"B"` blocks other users.

MULTI-TPR-COMMIT If user executes a multi-TPR commitment unit by running TPR on SEND WITH EGI without a commitment.

`" "` last TPR ended with a commitment
`"M"` last TPR ended without a commitment

PASSTHROUGH If user uses TDS Pass-Through:

`" "` no
`"M"` yes

TERM-TYPE Type of terminal to which user is connected.

TERM-NAME Name of user's terminal as symbolic source or destination.

TX-NAME Name of transaction that user is running.

TX-STORE-SIZE Size of TRANSACTION-STORAGE of transaction.

TX-NAMED-BLOCKED Transaction on which user is blocked if STATE is BLOCKED.

If specified, the **TRS-FILEINFO** structure reports detailed information about TDS files. More than one TRS-FILEINFO structure can be used. Declare MAX-USERS to print the file information, then specify files in the SELECT-FILES parameter of TRS-CONTROL to customize the report.

FILE-TYPE Type of file:

`"C"` TDS controlled file
`"N"` TDS non-controlled file
`"D"` IDS-II database area

FILE-NAME Internal file name specified at TP7GEN and used by TPRs.

FILE-STATE If file is open or not:

"O" file is open
"C" file is closed or being closed.

AFTER-JOURNAL If file is protected by After Journal:

"." no After Journal on file
"A" file is protected by After Journal.

BEFORE-JOURNAL If file is protected by Before Journal:

"." no BeforeJournal on file
"B" file is protected by Before Journal.

SHARE-LEVEL Current sharability of the file. A file closed and re-opened without the option SHARED on [M] OPEN, is *exclusive*.

"S" file is opened in shared-mode
"E" file is exclusive to TDS.

DEFERRED-UPDATE If Deferred Updates is used for file:

"." Deferred Updates is not used
"D" Deferred Updates is used.

SHARE Sharing mode of file:

"N" normal
"M" monitor
"O" onewrite
"D" directory
"F" free.

ACCESS Access mode of the file:

"R" read
"W" write
"S" spwrite
"P" spread.

INIT-PMD + CUR-PMD Initial processing-mode for file, specified in TDSGEN and retrieved at the first TDS startup or the last cold restart (whichever was last). If file is closed and re-opened, current processing-mode may be different. Current processing-mode is retained for the next session if a TDS warm restart is used. The processing-modes are:

"IN"	input
"UP"	input-output/update
"OU"	output
"AP"	extend/append.

FILE-INTEGRITY File integrity specified in TDSGEN being the minimum level of protection that TDS provides for file:

"H"	high
"M"	medium
"N"	none
"EFN"	External file name.

5.4.4 DISPLAY_TDS_STATISTICS (DTSTAT)

Purpose:

This command allows an IOF user to:

- obtain information concerning a TDS that he has not started
- display statistical information concerning a given TDS
- display information concerning users connected to a given TDS
- display information concerning files of a given TDS.

Syntax:

```
{DTSTAT
{DISPLAY_TDS_STATISTICS}
{RON=name5|TDS=name4}
[{COR|USER}=star12]
[IFN=star8]
[PRTFILE=file78]
```

Parameters:

RON	Run Occurrence Number of the TDS for which information is required used as research criteria. If the research criteria must be done by the name of the TDS, this field must be left blank (the RON keyword mutually exclusive of TDS keywords). For example: RON=X7, RON=X07, RON=X007, or RON=X0007.
TDS	TDS name used as research criteria. This field must be left blank (TDS keyword is mutually exclusive of RON keywords).
COR alias USER	Information concerning users is required. A blank value means no information is required. * means information concerning all users is required. A given userid means information concerning this user is required. The star convention is supported. The default value is no information.

IFN Information about files is required.

A blank value means no information is required.

* means information concerning all files is required.

A given ifn means information concerning this ifn is required.

Star convention is supported. Default value is no information.

PRTFILE The file or library members containing the required information.

A blank value means the information will be displayed on the terminal.

If the command is activated in batch mode (EJR), the PRTFILE keyword is mandatory.

SYS.OUT means that output is displayed in the standard SYS.OUT file.

Examples:

```
DTSTAT RON=X400 PRTFILE=my_file..dtstat1
```
Statistical information of the TDS with a RON of X400 is registered in the member "dtstat1" of the cataloged library "my_file".

```
DSTAT TDS=JDB COR=raoul IFN=()
```
Statistical information about the user called "raoul" of the TDS called "JDB" are displayed.

Output

```
S: DTSTAT TDS=JDB COR=* IFN=*

TDSname           GENERAL ACCOUNTING INFORMATION    Time and Date
TPR               :       11       TPR ABORTS     :         0
TX                :        7       TX ABORTS      :         0
DIALOG            :        5       COMMIT         :         7
USERS             :        8       FROZEN         :         0
CPU               : 0000.016       ELAPSED        :  0000.047

                   ABORT DETAILS
BUFOV             :     0          LONGWAIT       :    0
TABOV             :     0          DIRTY READ     :    0
WDNAV             :     0          DEADLOCK       :    0
SERIAL            :     0          NON-CONC       :    0
```

```
PT T USER-ID  TERMINAL  TERMTYPE   STATE   SEND  TX-NAME  TX-STORAG
   I H___PMS            MASTER     COMMAND
   I BENOIST            MASTER     COMMAND
   D DUMMY                         COMMAND
   V BADGER00                      COMMAND
   V BADGER01                      COMMAND
   V WEASEL00                      COMMAND
   V WEASEL01                      COMMAND
   V WEASEL02                      COMMAND

PT   JDB1    BR0ETTYP4 DKU7105    END TPR       PTJM     256
     JDB2    BR0ETTYP4 DKU7105    END TPR  EGI  TEST     100
   NO COMMIT
   X                   XCP2       PROCESSING    X2RSTX2  35

FILETP    FILENAM   INIT-PMD CUR-PMD  INTEGR  JOURNAL  FSTATE
DATABASE  AREACINE  UPDATE   UPDATE   MEDIUM  BEFORE   OPENED      EXCLUSIVE
TDS FILE  BOUFFI    UPDATE   UPDATE   MEDIUM           CLOSED
TDS FILE  H_CTLM    UPDATE   UPDATE   MEDIUM  BEFORE   OPENED      EXCLUSIVE
TDS FILE  H_FORM    INPUT    UPDATE   NONE             CLOSED
TDS FILE  INDEX1    UPDATE   UPDATE   MEDIUM  BEFORE   OPENED
TDS FILE  INDEX2    UPDATE   UPDATE   NONE             CLOSED
TDS FILE  INDEX4    UPDATE   UPDATE   HIGH    BOTH     OPENED
TDS FILE  T1        UPDATE   UPDATE   MEDIUM           CLOSED
NON CONT  INDEX3    UPDATE   UPDATE   NONE             OPENED
NON CONT  TWRITER   UPDATE   UPDATE   NONE             CLOSED
```

NOTE:
CPU and elapsed are expressed in minutes.milliminutes.

General accounting information

Number of users must be seen as the sum of:

- Master terminal plus its pmos session H_PMS (if no master mailbox is used).
- Terminal users.
- Sessions of the pool of the dummy correspondent.
- Xcp1 sessions of primary correspondent.
- Xcp1 sessions of secondary correspondent on which a call "spawn" or a call "dfrecov" has been issued.
- Xcp2 sessions used for a partner transaction.

Information concerning users connected to a given TDS or information concerning files of a given TDS cannot be obtained if the given TDS is securized using SECUR'ACCESS product. In such case, the message:

```
NO MORE INFORMATION DISPLAYED: SECUR'ACCESS IS ACTIVE
```

is displayed.

User Information

PT	User is in TDS pass-thru mode.
T	Type: "I" IOF "D" DUMMY "V" Virtual "M" Master mailbox "X" XCP1 or XCP2

For Xcp1 correspondents, the following sessions are displayed:

- the sessions of the primary correspondent
- the sessions of the secondary correspondent when a spawning or a 'call "DFRECOV" 'has been issued on them

NOTE:
See "correspondent role" paragraph of *Transactional Intercommunication Using XCP1 Protocol User's Guide* (47 A2 11UT) for the definition of Xcp1 primary and secondary correspondents.

For Xcp2 correspondents, only session running a partner transaction are displayed.

NOTE:
See "Co-operative transaction, Local transaction, Partner transaction" paragraphs of *CPI-C/XCP2 User's Guide* (47 A2 14UT) for the definition of a XCP2 partner transaction.

State

PROCESSING	Executing a TPR.
END TPR	Between TPRs, next TSP has not been started.
COMMAND	Not running a transaction.
FROZEN	Abnormal disconnection, including $*$DIS.
BLOCKED	Blocked due to NON-CONCURRENCY.
GAC WAIT	Waiting for GAC after abort or LONGWAIT, DEADLOCK, TABOV or ENQUE.
TPRTERM	Temporarily in TPRTERM procedure at end of TPR (with or without commit) or at end of transaction (next state will be END TPR or COMMAND).
UNLOGGED	User being unlogged.
CMG WAIT	Waiting commitment unit restart after abort on WDNAV, BUFNBOV, CMWSOV, ITMNAV or ENQUE. ENQUE received at TPR start if another commitment unit requests serialization.
SEQUENCE	User is executing a sequence of actions (logon or logout or break or disconnect phase for example) and is in an action that is not a transaction.

SEND-LEVEL

This indicates the level of the last physical send from the TDS to the terminal (EMI or EGI) or more accurately, the type of event that the TDS is waiting for (for EMI, an acknowledgement of the sent message; for EGI, the next input from the terminal).

" "	No physical send. No send or send ESI.
EMI	Send EMI and no acknowledgement received. Wait for VCAM credit).
EGI	Send EGI and the next terminal input awaited.

EXTRA-FIELD

SERIALIZE	If a user executes alone (serializes) or waits for all active commitment units to finish before executing alone.
NO-COMMIT	Last TPR is not committed.
BLOCKING	User blocks other users due to non-concurrence.
WAIT-TIME	Previous TPR set WAIT-TIME in TDS-STORAGE before completing.

In the previous example:

- user JDB1 has activated the transaction PTJM and is in Pass-thru mode.
- user JDB2 has activated the transaction TEST and is awaited for terminal input after a send EGI without commit.
- an running partner XCP2 transaction "X2RSTX2".

File Information

- TDS File stand for TDS Controlled File.
- NON CONT stands for NON CONTROLLED FILE.
- Extra field EXCLUSIVE means the file is not sharable with other process groups.

Errors

Errors are reported through the STATUS and SUBSTATUS fields.

Status

7	Something has failed during TDS search. See SUBSTATUS for more information.

Substatus

-1	Incorrect RON.
-2	RON contains a non-numeric value (except X part).
-3	J number cannot be found.
-4	Root table cannot be built.
-5	Correspondent table cannot be built.
-6	Session table cannot be built.
-7	Mailbox table cannot be built.
-8	Transaction phase table cannot be built.
-9	Transaction table cannot be built.

Error Messages

There are three kinds of error messages:

- DTSTAT: OPTIONS STRING ERROR
- DTSTAT: CANNOT RETRIEVE TDS INFORMATION

  STATUS=X SUBSTATUS=Y
- FATAL: ERROR FROM TP7STAT (REASON=Z) rc, where rc is the last RETURN CODE.

REASON can have a value of 01 to 07:

01	Cannot open prtfile or terminal access method.
02	Error when analyzing prtfile specifications.
03	Incorrect prtfile specifications.
04	Cannot assign prtfile.
05	Cannot open prtfile.
06	Cannot create "101" record in prtfile.
07	Cannot write in prtfile or message cannot be displayed.

Example:

```
DTSTAT TDS=JDB:
  DTSTAT: CANNOT RETRIEVE TDS INFORMATION
  STATUS= 7 SUBSTATUS= -3
```

Meaning: the TDS JDB is not running on the site.

6. Ensuring Data Security, Integrity and Recovery

Ensuring the security of information, maintaining the integrity of data, and providing the means of recovering from errors or aborts are essential to a TDS application. TDS is protected as follows:

- Information is protected by establishing access rights
- File integrity is maintained by non-concurrency facilities, GAC-EXTENDED, and protected reads
- Recovery mechanisms, Journals, and the rollforward utility provide the means of saving data and restarting after aborts.

6.1 Protecting Information

Access to information is restricted by the access rights. These rights are checked before users logon, when files are assigned. A project must possess all the necessary rights to access files assigned to the TDS application. These access rights are the Authority Codes declared in the Site Catalog. As users are declared as belonging to projects, an authorized user must have permission to read and or write to files.

The project that the master operator belongs to must have at least the RECOVERY access right to recover TDS files after a TDS failure has occurred. See *Data Security User's Guide*.

6.1.1 Restricting Access to TDS

Access to a TDS application can be restricted by defining a password in the Site Catalog. Without this password, the terminal operator is denied access to the application.

Setting the user's codes in AUTHORITY-CODES Clause of TRANSACTION SECTION further restricts the user in initiating only certain transactions.

6.1.2 Controlling Access With Master Commands

The following master commands dynamically control users and their access:

[M] CANCEL_TDS_COR
To logout a user at the end of the current transaction.

[M] CANCEL_TDS_COR STRONG=1
To logout a user immediately.

[M] MODIFY_TX AUTHORITY_CODE
To modify the authority codes associated with a transaction.

6.1.3 Protecting Files

The options for protecting files are declared in MESSAGE Statement of TRANSACTION SECTION. For TDS-controlled files, transactions use DEFERRED UPDATES whether the files are accessed in shared or exclusive mode, or if DEFINE JOURNAL=BOTH is specified in the JCL or in the catalog. Otherwise the Before Journal is used.

If SUPPRESS Clause results in a conflict between access and the file protection, SUPPRESS is ignored. For example, SUPPRESS BEFORE JOURNAL is ignored if a transaction tries to modify a file protected only with BEFORE JOURNAL in the JCL statement DEFINE or in the CATALOG.

These two statements are used as follows:

- to optimize space occupancy use SUPPRESS DEFERRED UPDATES
- to reduce the number of I/O operations use SUPPRESS BEFORE JOURNAL.

6.2 Ensuring Integrity of Data

Data must be verified on entry before being applied to files or used to produce reports. This verification is separate from file updates. Data after being is verified, is processed by the transaction to update files.

When a transaction commits a file update, the update is irrevocable. Updates can only be undone by user intervention.

Integrity of files is ensured by preventing incidents which interrupt sessions and corrupt files or the database, and by controlling concurrent access to information. Preventing incidents combines the facilities provided by TDS itself and GAC-EXTENDED. TDS provides complete facilities for file recovery based on journalization and file saves.

6.2.1 Controlling Access to Files

Problems of file access arise when two transactions attempt to simultaneously update the same record. Assume the following sequence of events:

- Transaction A reads record X
- Transaction B reads record X
- Transaction A updates record X and rewrites it
- Transaction B updates record X and rewrites it.

The file now contains a record X which has only been updated by B; the B-Update overwrote the A-update.

It is essential that once record X is updated by transaction A, any other transaction trying to access record X is temporarily denied access. GAC-EXTENDED ensures this protection. However, if a transaction only reads a file, this control can be inhibited at TP7GEN by specifying:

```
SUPPRESS CONCURRENT ACCESS CONTROL
```

in MESSAGE Statement of TRANSACTION SECTION. This option can cause inconsistent reads because the record could be updated while being read.

The problem of simultaneous access can also be solved by declaring at TP7GEN that certain transactions may not execute at the same time as other transactions at commitment level.

EXAMPLE:

```
MESSAGE "FINDM" ASSIGN TO TPR17

NON-CONCURRENT WITH "FINDM", "FINDX", "UPDY", ...
```

❑

When a commitment of transaction FINDM is processed, commitments of transaction FINDX, UPDY or another FINDM will be suspended until the current commitment unit of FINDM terminates.

Concurrent access to resources can cause deadlocks. When two transactions are waiting for each other to release resources, TDS aborts one to allow the other to continue to terminate normally. The aborted commitment unit then restarts.

6.2.2 Using GAC-EXTENDED Protection

GAC-EXTENDED is a file sharing facility allowing several concurrent users in various processing environments to read and write to the same file. The CONFIG GAC statement defines the maximum number of pages that can be simultaneously locked in the system.

GAC-EXTENDED maintains a system-wide lock list for all files it controls. This list contains details of all CIs currently locked by users together with the type of locking. Each lock-list entry contains the CI number, the type of locking and the identification of the user. At commitment time, all entries in the lock-list associated with the user are freed.

When GAC-EXTENDED is installed, the size of the lock-list is defined by the LOCKSIZE parameter of the CONFIG statement GAC. The size of the lock-list ranges from 5 through 320 Kbytes. For GAC-EXTENDED, the maximum size of the lock-list corresponds to 10,234 entries. The default *locksize* of 20 Kbytes gives about 1,000 entries.

Entries in the lock list are not pre-allocated because a commitment unit is dynamically allocated all the lock-list entries that it requests. If a commitment unit requests more lock list entries than the number specified by *maxlock*, the commitment unit is aborted.

If applications executing at the site need more locked pages than provided for, the CONFIG job must be rerun to specify a larger LOCKSIZE. The defaults are:

locksize=20 Kbytes
maxlock=512 entries

See *GAC-EXTENDED User's Guide*.

6.2.3 Preventing Concurrent Access to Files

Depending on the option chosen at TP7GEN, a given TDS-controlled file may be read by a particular transaction in three different modes:

- Exclusive Read
- Shared Read
- Statistical Read.

6.2.3.1 Exclusive Read

Exclusive Read (default) when no option has been declared.

When the transaction accesses a record, the complete CI is put in EXCLUSIVE lock by GAC-EXTENDED until the end of its commitment unit. No other transactions can access the CI.

Exclusive Read can prevent deadlocks from occurring if several users try to modify records in the same CI at the same time. Although such a method can induce more conflicts, there will be significantly fewer deadlocks.

6.2.3.2 Shared Read

Shared Read is specified by SHARED READ Clause in MESSAGE Statement of TRANSACTION SECTION.

When the transaction reads a record, the complete CI is assigned a NORMAL lock by GAC-EXTENDED. Other commitment units can access the same CI unless they request an EXCLUSIVE lock. SHARED READ is recommended whenever a transaction does not modify records.

6.2.3.3 Statistical Read

Statistical Read is specified by SUPPRESS CONCURRENT ACCESS CONTROL Clause in MESSAGE Statement of TRANSACTION SECTION. Transactions cannot update files.

When the transaction reads a record, GAC-EXTENDED does not consult or update the lock-list and allows a CI to be accessed even if there are other readers or another writer accessing it. This is useful for immediately retrieving statistical information such as the number of occurrences of a given item in a file.

The transaction cannot be sure that the fields it accesses have not been updated by other commitment units.

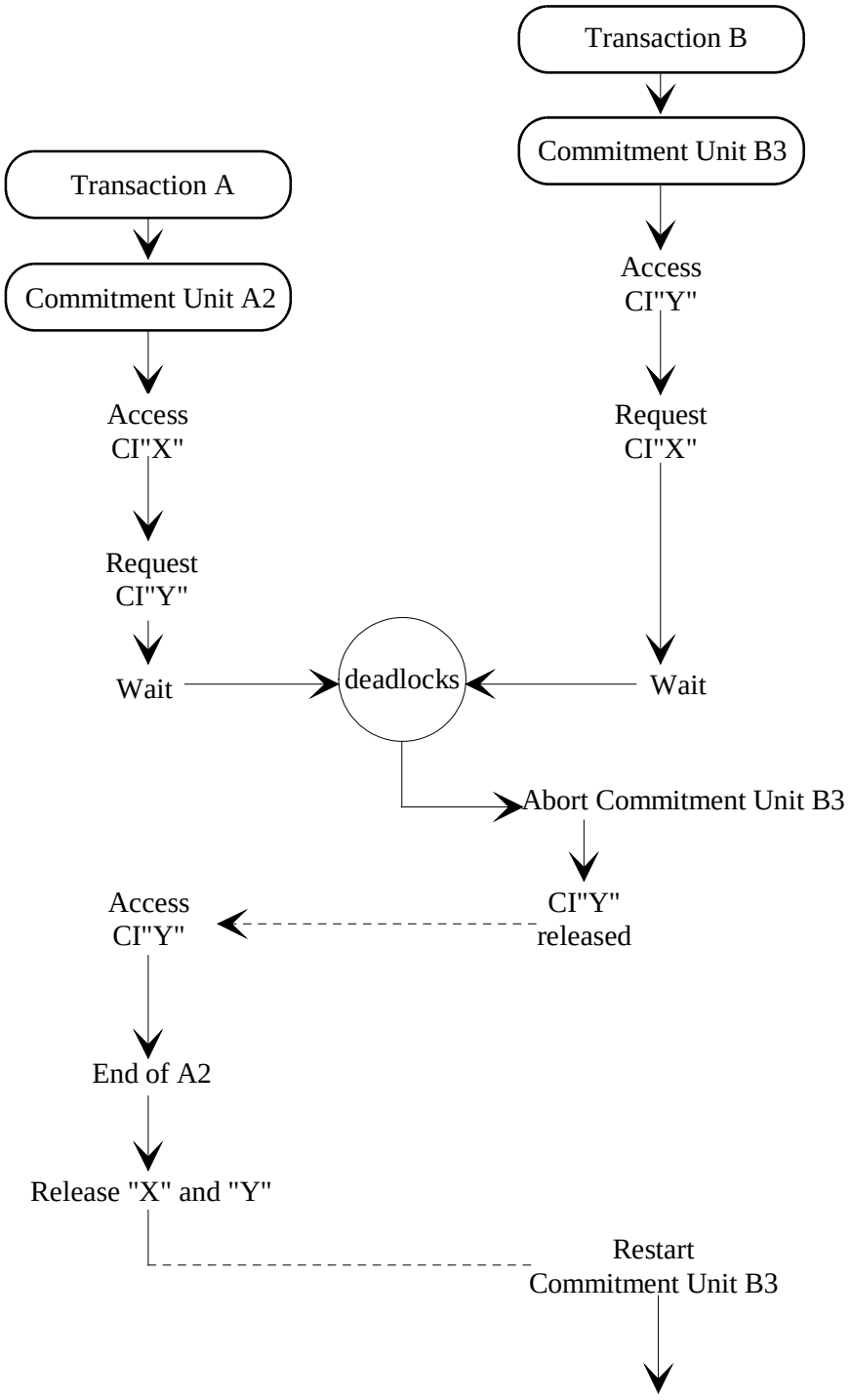


Figure 6-1. Prevention of Deadlocks

6.2.4 Using Master Commands to Ensure Integrity

Ensuring integrity involves controlling transactions and their access to data:

- To control specific transactions or class of transactions:

  `[M] MODIFY_TX LOCK=1`
  To prevent a transaction or class of transaction from being started.

  `[M] MODIFY_TX LOCK=0`
  To cancel the effect of the [M] MODIFY_TX LOCK=1.

- To modify access to files and database areas:

  `[M] OPEN_TDS_FILE and [M] CLOSE_TDS_FILE`
  To control access to files and database areas.

  `[M] MODIFY_TX VALIDATE=0`
  To prevent a transaction or a class of transactions from physically changing a file (sets the specified transaction to DEBUG)

  `[M ] MODIFY_TX VALIDATE=1`
  To allow a file to be modified by suppressing the DEBUG option for a transaction or a class of transactions.

6.2.5 Levels of System Integrity

Integrity levels are determined by the type of protection applied to files. The three levels of integrity for the system are:

NONE	no protection
MEDIUM	protection against software failure
HIGH	protection against both hardware and software failure.

The integrity method is specified by a combination of entries made in the TP7GEN, and the JCL statements associated with the execution of the job. Assigning the Before Journal provides software security; the After Journal ensures hardware security.

For non-controlled files, the coherence of the files is not guaranteed because concurrent access control cannot apply to these files.

These tables summarize different levels of integrity for non-controlled and controlled files.

Non-Controlled Files:

INTEGRITY LEVEL	TYPE OF JOURNAL
No file integrity	NONE
Medium	BEFORE

Controlled Files:

INTEGRITY LEVEL	TYPE OF JOURNAL
High (default)	Unspecified
Medium	BEFORE
High*	AFTER
High**	BOTH

NOTE:

* High integrity is applied assuming enough memory for Deferred Updates.

** High integrity obtained by *Before and* After Journal or *After Journal and Deferred Updates*.

6.2.6 Minimum Integrity Level

To guard against possible omissions in the JCL at runtime, define a minimum level of file integrity by declaring FILE-INTEGRITY Clause in INPUT-OUTPUT SECTION, where the defaults are:

- NONE for non-controlled files
- HIGH for TDS-controlled files.

TDS uses FILE INTEGRITY to resolve conflicts occurring between what is specified in the JCL and what is specified at TP7GEN. When the integrity level defined in the runtime JCL is lower than that specified explicitly or implicitly by FILE-INTEGRITY, TDS refuses access to the file. If the runtime JCL provides for a higher level of protection than that specified for FILE-INTEGRITY, TDS applies the higher level.

6.3 Updating and Recovering Data

A software or hardware failure can occur at any time during the TDS session. When a failure occurs on restarting GCOS 7, the contents of all files and databases are restored to the state they were in when the failure occurred. This can be done because a record of data is kept each time the data is accessed and modified. Such historical information held in a journal enables data to be reconstituted.

The four journals used for updating and recovery during the TDS session are as follows:

- Internal Swap File
- After Journal
- Before Journal
- User Journal*.

NOTE:

* The User Journal is not a standard file protection tool. It must be specified and controlled by the user.

6.3.1 Swap File

The swap file is used by TDS for saving messages, currencies, TRANSACTION-STORAGE, TDS-STORAGE and data shared between TPRs. This file is not visible to the user except when mapping and unmapping involve I/O activity on this file.

6.3.2 After Journal

6.3.2.1 Function

After Journal is a GCOS 7 function and is not specific to TDS. It protects against hardware failure. Cataloging files with After Journal enables writing a copy of the updated record as an *after* image to After Journal. After Journal contains sequences of chronologically updated records to be written to a file or database.

If a commitment unit aborts, all data modified by it must be invalidated. After Journal alone cannot be used to immediately restore data as it was before modification. Either Deferred Updates or Before Journal must also be applied.

In Deferred Updates, a WRITE or REWRITE is not physically executed until the commitment unit has terminated normally. See Figure 6-2.

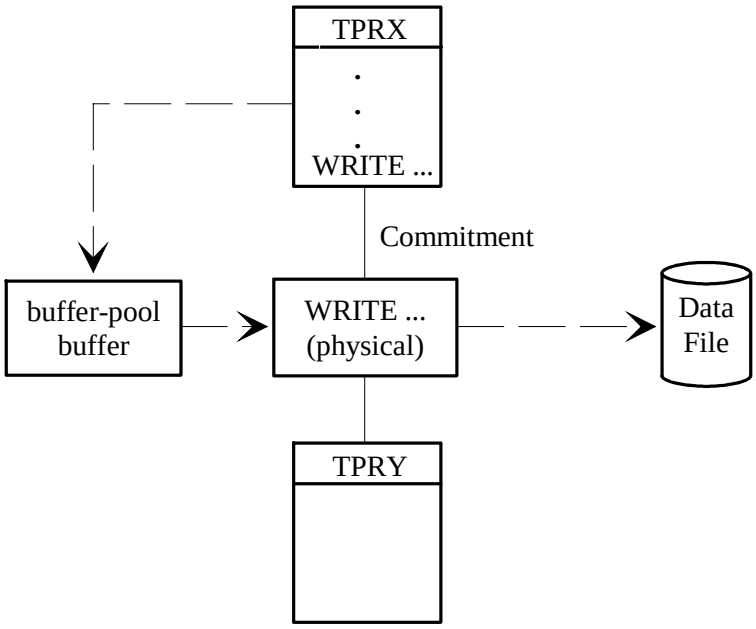


Figure 6-2. Flow of Data with Deferred Updates

The output record is placed in a buffer during the commitment unit. After Journal is used for immediate recovery on failure during physical writing of Deferred Updates. *After* images for all outstanding updates are applied to the file in dynamic rollforward:

- when a TDS application aborts and is then restarted
- when GCOS 7 is WARM restarted after a crash. See Figure 6-3.

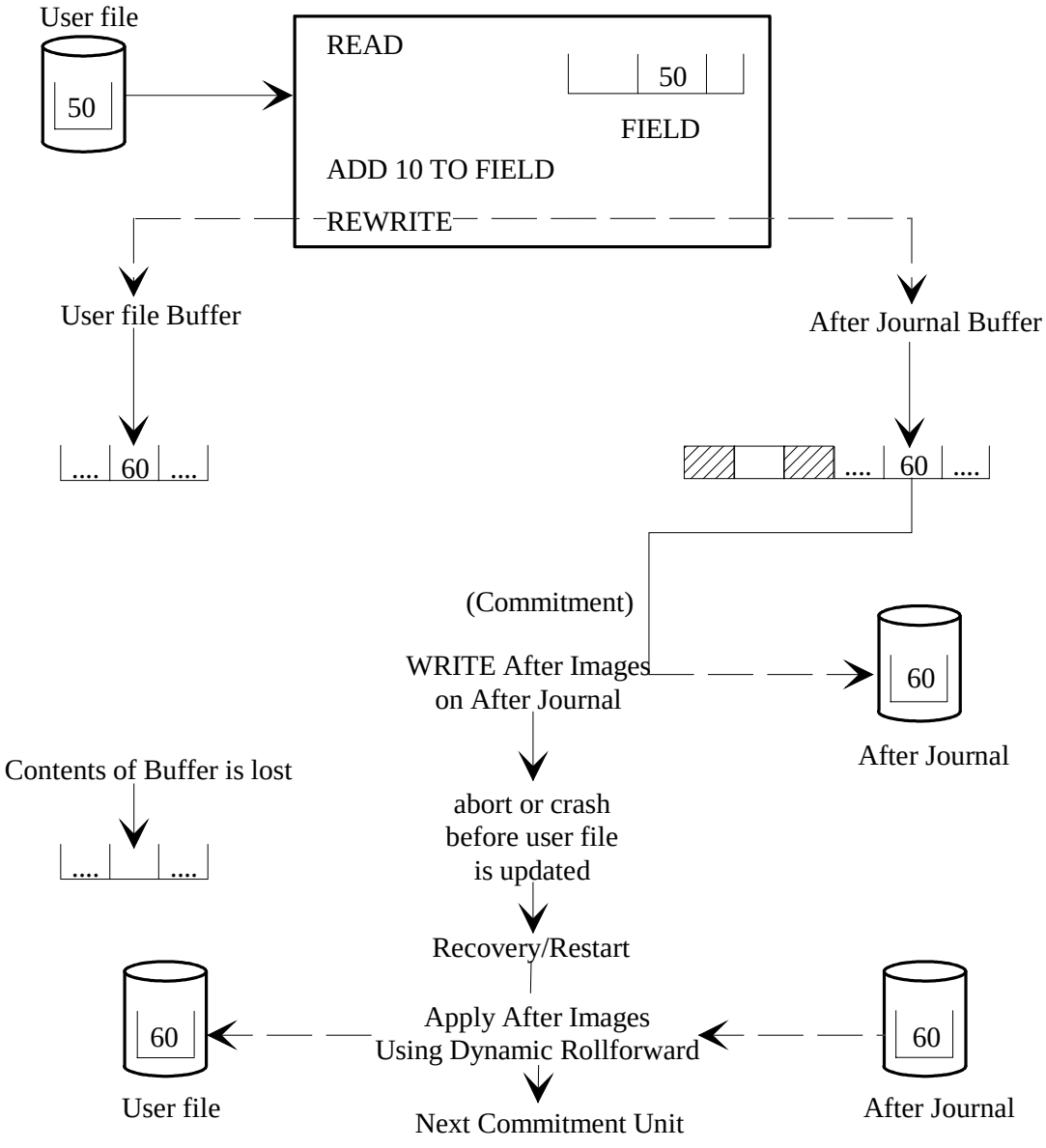


Figure 6-3. Using Deferred Updates for an Abort during Commitment

After Journal may also be used for Deferred Recovery. After a device failure or an incomplete recovery, *after* images are applied to the last saved version of the file in static rollforward using the ROLLFWD utility. Saving files is run in batch and is the user's responsibility. See Figure 6-4.

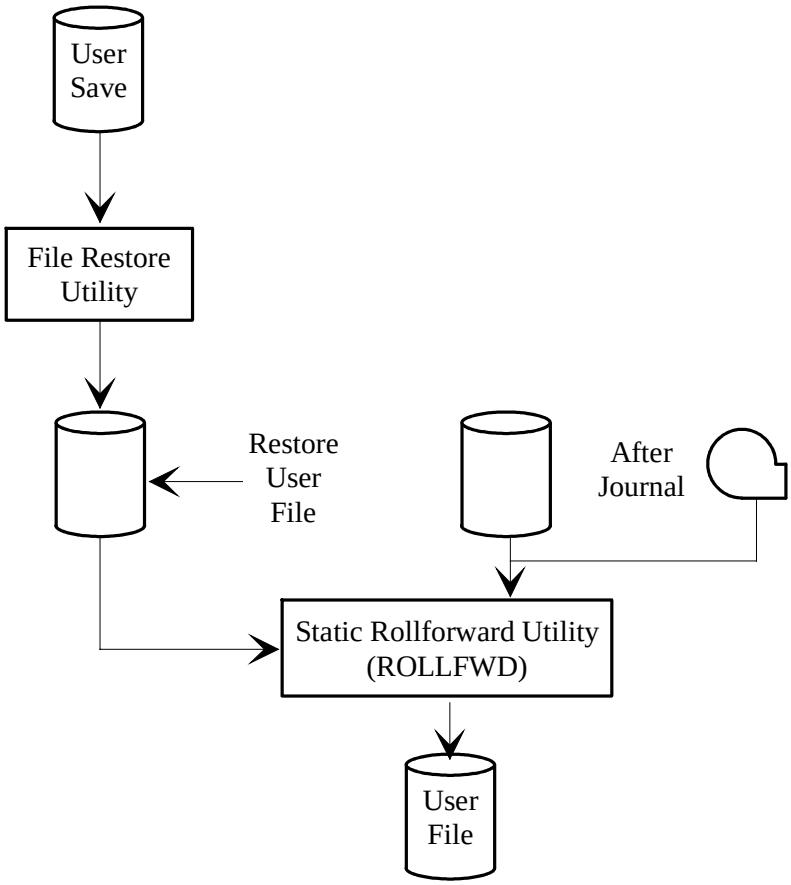


Figure 6-4. After Journal and Static Rollforward

6.3.2.2 The ROLLFWD Utility

Rollforward is used recovering a damaged file on device failure, incomplete rollback or dynamic rollforward. ROLLFWD utility works in conjunction with After Journal to restore the file to a point of restart.

Rollforward is run in batch for each user file to be recovered and is submitted through ROLLFWD as the GCL command or the JCL statement.

See *File Recovery Facilities User's Guide*.

Syntax of the JCL statement:

```
ROLLFWD

 { outfile | OUTFILES=((outfile1)[...(outfile25)]) }

 [ BEGDATE=yy.mm.dd/hh[.mm[.ss[.msmsms]]]]
           [ENDDATE=yy.mm.dd/hh[.mm[.ss[.msmsms]]]]

 [ DUMP = {NO|DATA} ]

[ STEPOPT=(step-parameters) ]

[ SIZEOPT=(size-parameters) ];
```

Parameters:

where *outfilei* is:

```
                [ { RESIDENT                          } ]
outfile[i]=efn  [ { DEVCLASS=dvc, MEDIA=(md [md...]) } ]
                [ { CATALOG=n                         } ]
```

- where *efn* names the file to be rolled forward.
- For the RESIDENT, CATALOG, DEVCLASS and MEDIA, see ASSIGN statement.
- If running an HA application, ROLLFWD can only be run if the JAS is in ACTIVE state. See *File Recovery Facilities User's Guide* and *High Availability Administrator's Guide*.

SIZEOPT

Defaults for SIZE and POOLSIZE allow rollforward on UFAS-EXTENDED and IDS/II files with CISIZE => 4 Kbytes.

– If CISIZE > 4 Kbytes, SIZE and POOLSIZE in Kbytes:
```
SIZE = 120 + ( 6 * (MCI - 4))
POOLSIZE = 60 + ( 6 * (MCI - 4))
```
where *MCI* is maximum CISIZE of UFAS-EXTENDED files and IDS/II areas.

– If memory is available, increment SIZE and POOLSIZE by the same amount to optimize file management:
```
SIZE = 120 + ( 6 * (MCI - 4)) + I
POOLSIZE = 60 + ( 6 * (MCI - 4)) + I
```
where *I* is the increment.

BEGDATE

Start date for ROLLFWD. If BEGDATE is omitted, ROLLFWD takes the latest date when the outfiles were saved. This date is recorded in After Journal Directory by FILSAVE utility. If no save date is known, the date of the first update is taken.

ENDDATE

Date when *after* images were applied. If ENDDATE is omitted, ROLLFWD applies all known *after* images to files. Any images of incomplete updates are skipped. When *after* images are applied, outfiles are set to their last stable state.
Use . or / to separate year, month, day, hours, minutes, seconds and milliseconds.

6.3.3 Before Journal

Before Journal is a GCOS 7 function and is not specific to TDS.

Before Journal protects files in the case of software failure. When information is being processed, a copy of the original record is written as a *before* image to Before Journal. Figure 6-5 shows Before Journal used with the After Journal.

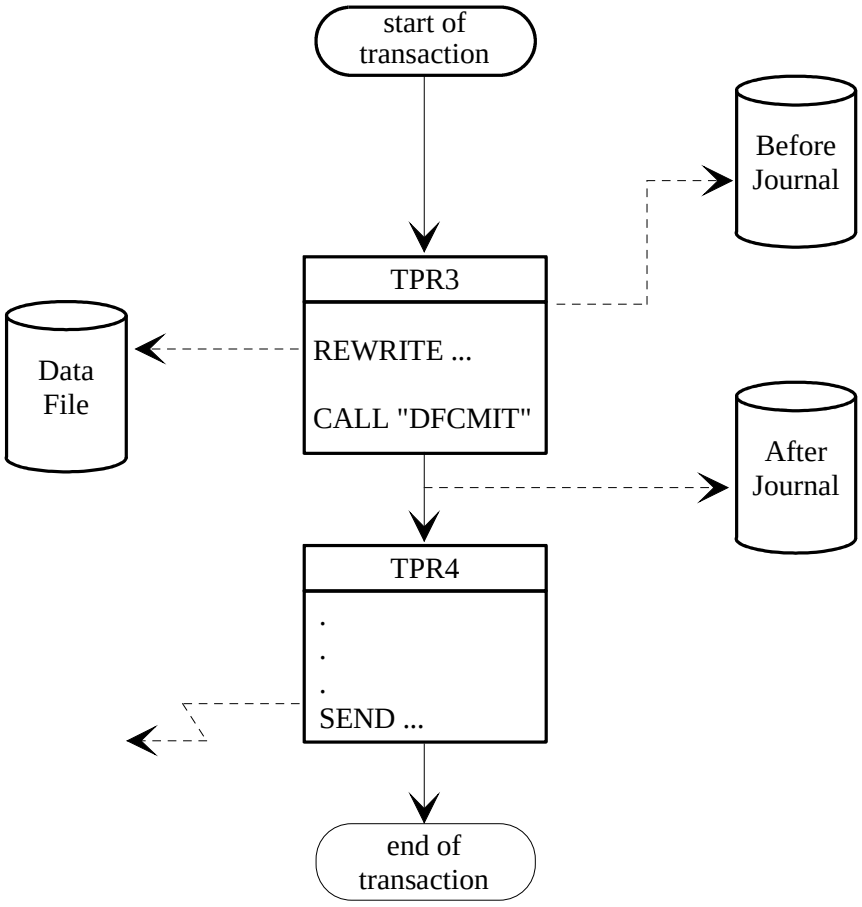


Figure 6-5. Using the Before and After Journals Without Deferred Updates

If a commitment unit terminates normally, the disk space in Before Journal used for the *before* images are eligible to be overwritten by the *before* images of the succeeding commitment unit.

If a commitment unit has to be restarted, a rollback is applied. The *before* images are used to restore files and databases as shown in Figure 6-6.

Before images are applied by the rollback in the case of an abort and subsequent reinitialization of a TDS application or in the case of a restart after a GCOS 7 crash.

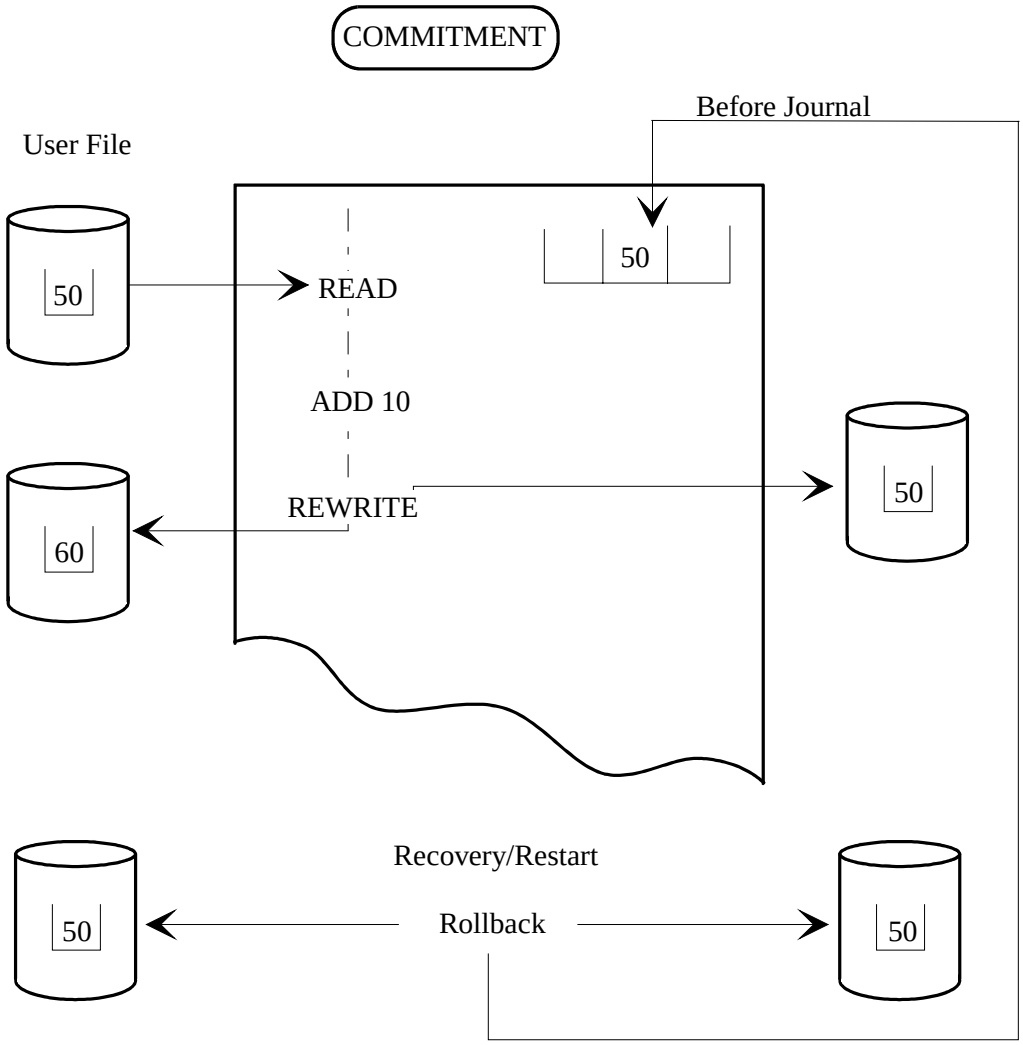


Figure 6-6. The Before Journal and Rollback

6.3.4 User Journal

User Journal stores non-specific information about the transaction. Records can be written to this file by WRITEs. TDS can be instructed to log input and output messages.

User Journal is not a standard file-protection tool for non-controlled files. The user is responsible for journalization and for any recovery which involves this file. The structure of the User Journal is described in Appendix C.

6.3.4.1 Declaring User Journal

User Journal takes effect when:

- declaring USER-JOURNAL Clause in TDS SECTION
- in the JCL which launches the TDS application:
  - removing DEFINE H_SWAP, JOURNAL=AFTER;
  - and adding ASSIGN USERJRNL, DUMMY;

The absence of user journalization does not prevent a TDS application from executing normally. User Journal requires After Journal to be initialized since there must be at least one file protected by After Journal.

Insert DEFINE USERJRNL, JOURNAL=AFTER if no files are protected by After Journal.

6.3.4.2 Using DUMPJRNL Utility

DUMPJRNL utility extracts User Journal data stored in After Journal and appends it to a user file. Execute this utility by entering DUMPJRNL as the JCL statement or the GCL command. GCL and JCL syntax are identical (except for STEPOPT and SIZEOPT which are JCL parameters).

User Journal is a sequential output file:

- assigned with MOUNT
- and opened in APPEND mode by DUMPJRNL.

Syntax:

```
DUMPJRNL   TDS=name4

   OUTFILE=( sequential-output-file )

   [ OUTDEF=( define_parameters )]

   BEGDATE=[yy]yy.mm.dd[.mm[.ss[.msmsms]]]

   [ ENDDATE=[yy]yy.mm.dd[.mm[.ss[.msmsms]]]]

   [ DUMP={ NO | DATA }]

   [ STEPOPT=( step_parameters )]

   [ SIZEOPT=( size_parameters )]
```

6.3.4.3 The XDUMPJRNL Function

The new XDUMPJRNL function makes it possible to extract the "user journal" for 1 to 25 TDSs at the same time. The user must have access rights to the directory TDSname.DUMPJRNL_CTRL for each of the TDSs.

The user can either supply a start date that is the date for all the TDSs, or not supply it. (The latter is recommended.)

The function can be called by starting the XDUMPJRNL job from SYS.HSLLIB using EJR, RUN, or INVOKE.

Syntax:

```
EJR XDUMPJRNL LIB = SYS.HSLLIB VL= (number of TDSs,
TDS 1 name, TDS 1 output file .....,
TDS n name, TDS n output file,
[BEGDATE = yy.mm.dd/hh[mm[.ss[.msmsms]]]],
[ENDDATE = yy.mm.dd/hh[mm[.ss[.msmsms]]]],
[DUMP= {NO|DATA}]
[OUTDEF= 'DEFINE parameters common to all the output files'])
```

Parameter Description:

The following parameters are positional:

NUMBER OF TDSs	This is a decimal number from 1 to 25. It indicates the number of TDSs for which the "user journal" is to be extracted.

(The following two parameters constitute a pair of parameters, repeatable up to 25 pairs. There must be at least one pair of these parameters).

TDS NAME	Name of a TDS for which the "user journal" is to be extracted.
TDS OUTPUT FILE	This is the sequential file that is to receive the "user journal" images of the TDS whose name is given in the previous parameter.

The following parameters are not positional:

BEGDATE	This is the start date for extraction of "user journal" images of the various TDSs. It is recommended not to supply it if you want to start the extraction at the end date of the previous extraction for this TDS (or for the first time, at the date of the first use of the "user journal" of this TDS). Warning messages may be presented to indicate, when this date is supplied, that it does not correspond to the ranges recorded by the Journal in its Directory. Note that the year must be entered in 2-character form.
ENDDATE	This date corresponds to the end of the extraction of the "user journal" images. If it is not supplied, it is set at the date of the start of executing of the command. Note that the year must be entered in 2-character form.
DUMP	This indicator allows a dump to be taken if the step aborts.
OUTDEF	This string between quotes (') describes the parameters with which the output file will be handled; e.g., NBBUF. If there are several TDSs, these parameters apply to all the output files.

6.3.5 Summary of Journalization Techniques

Specify Before and After Journals with cataloged options rather than through the JCL statement DEFINE, as follows:

```
JOURNAL=BEFORE
JOURNAL=AFTER
JOURNAL=BOTH
```

where BOTH is for both Before and After Journals.

The effect of these entries and the use made of the journals depends on what appears at TDSGEN and the transaction:

- TDS treats JOURNAL=BOTH and JOURNAL=AFTER in the same way.
- When incident occurs concerning a file protected by JOURNAL=AFTER and when Deferred Update Mode *does not* function at restart, the file is protected as if it was declared with JOURNAL=BOTH.
- USE DEFERRED UPDATES Clause in TRANSACTION SECTION allows Deferred Updates to be used for files protected by JOURNAL=AFTER or JOURNAL=BOTH. In either case TDS protects files with After Journal and Deferred Updates. Any files specified in EXCEPT FOR are not covered by Deferred Updates.
- USE DEFERRED UPDATES applies whether the file is shared or not. See Paragraph 3.5.8.3 *USE DEFERRED UPDATES Clause*.
- If the commitment unit aborts because of a lack of buffers while Deferred Updates is being used, the commitment unit is restarted with Before and After Journals without Deferred Updates.
- A Security Option in MESSAGE Statement of TRANSACTION SECTION can be used to modify the effect of JOURNAL entry in DEFINE statement for a particular transaction for BEFORE JOURNAL or DEFERRED UPDATES. See Paragraph 3.5.8 *File Security Options*.

Protection from software failures is assured by either Before Journal, or After Journal and Deferred Updates. Files are protected from hardware failure by After Journal and SAVE utility.

The methods of journalization and recovery are summarized below.

Summary of Integrity and Recovery Methods

Type of Failure	Integrity Method	User Action
Software	- After Journal and Deferred Updates or - Before Journal	None
Hardware	After Journal	- Resolve Hardware problem - Use File Restore and Recovery Utility
Hardware and Software	- After Journal and Deferred Updates or - After Journal and Before Journal	- Resolve Hardware problem - Use File Restore and Recovery Utility

6.3.6 Recovery and Restart

On system crash, transaction processing will be in one of three states:

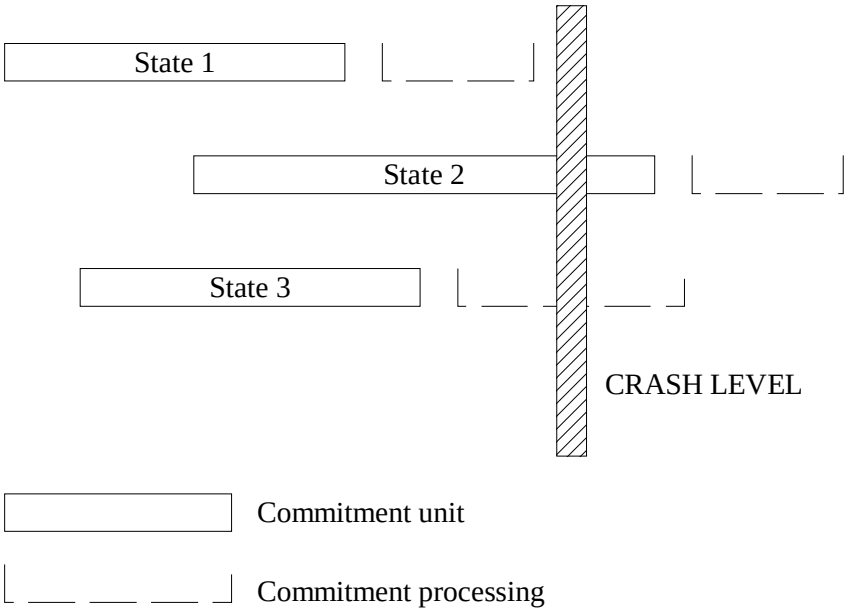


Figure 6-7. Crash During Transaction Processing

- State 1: the commitment unit and its associated commitment processing completes.
- State 2: the abort occurs in the middle of the commitment unit.
- State 3: the abort occurs during the processing of the commitment.

Whatever the state, warm restart is performed automatically at warm GCOS restart which is mandatory for recovery. Dynamic rollforward and rollback occur at GCOS warm restart.

A transaction may be restarted either at the start of the previous commitment unit or the one following. The choice of which commitment unit depends on the state the transaction in terms of the commitment processing level when it aborts. See Figure 6-8 below.

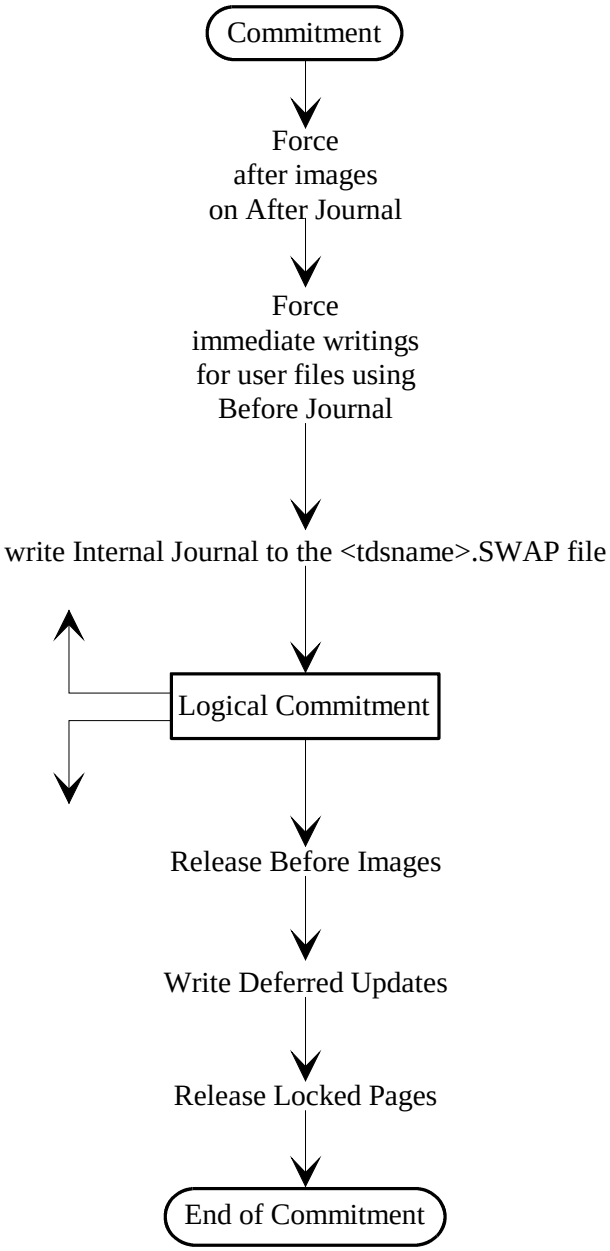


Figure 6-8. Processing a Commitment Unit

Once a logical commitment point is reached, previous results cannot be changed. If the system crashes, the transaction restart point depends on whether the abort occurred before or after a logical commitment. Figure 6-9 shows three possible cases.

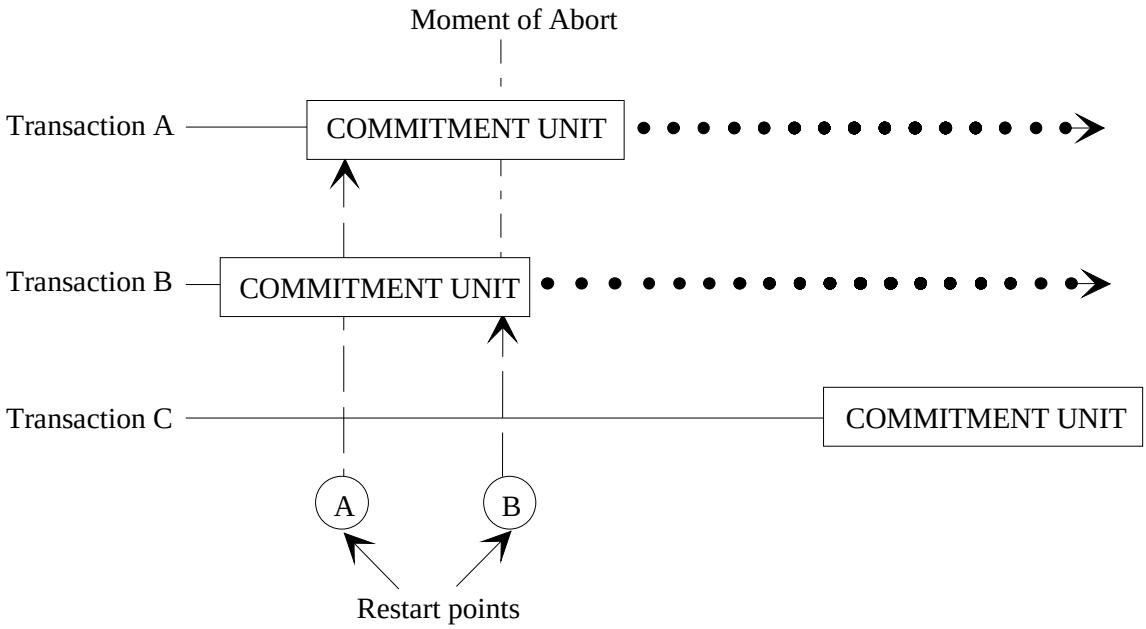


Figure 6-9. Restarting a Commitment Unit after an Abort

If an abort occurs during the processing of a commitment point as shown for Transaction A, the commitment unit is restarted from restart point A. The context of the transaction is restored from the internal swap file.

- If file integrity is provided by the Before Journal, the file is rolled back and uses *before* images for recovery.
- If file integrity is provided by Deferred Updates, file recovery is not required because modifications have not been written to user files.

If an abort occurs after a logical commitment has been processed as shown for Transaction B, the transaction continues from restart point B.

If file integrity is provided by After Journal and Deferred Updates, *after* images are used to rollforward the files.

If an abort occurs during processing when no commitments are involved, as indicated for Transaction C, restart is not necessary and processing continues from the start of the next commitment unit.

6.3.7 Restoring Files in TDS and Batch Processing

If a file or database is updated by both TDS and batch, After Journal must be specified for both applications. Files must therefore be cataloged as being protected by After Journal.
Static rollforward only applies where After Journal contains the updates made by the batch job.

7. Optimizing a TDS Application

Getting optimum performance of a TDS application leads to obtaining rapid response times and maximum processing throughput.

The application is designed and implemented on a set of objectives. An application once developed undergoes testing and where necessary, debugging to meet the criteria then optimizing. Any modification which has an impact on the application and its running environment, leads to other modifications.

Performance considerations should be dealt with from the very beginning when the application is designed to its final optimization. The purpose of this section is to indicate how high performance can be achieved.

7.1 Optimizing Performance

An optimized TDS application makes the most efficient use of resources and continues to ensure throughput even when saturation occurs.

The recommendations are designed for a large system comprising a fast CPU, several disks and terminals. Tuning is most effective on large systems.

Optimization involves measuring what is expected against what is realised. The optimized application improves overall system performance.

Using the Suggestions in this Section

Many of the recommendations that follow vary in importance. Each suggestion is preceded by a word that indicates of its impact on optimizing an application:

ESSENTIAL	Of primary importance for any TDS application.
IMPORTANT	Concerns most TDS applications.
Useful	Applicable to many TDS applications.
Optional	Recommended but not necessary for the application.

7.2 Effect of Load on Performance

Good performance involves servicing all incoming requests within a short response time. The number of users connected to the application and the frequency of requests determine the load on the system and hence the response time.

7.2.1 Small Networks

In a small network, the average number of connected terminals is about 30 and does not exceed 60. The number of requests waiting for service is less than or equal to this low number of terminals.

For small networks, since there are few terminals, the load imposed on the system and therefore throughput, is not heavy.

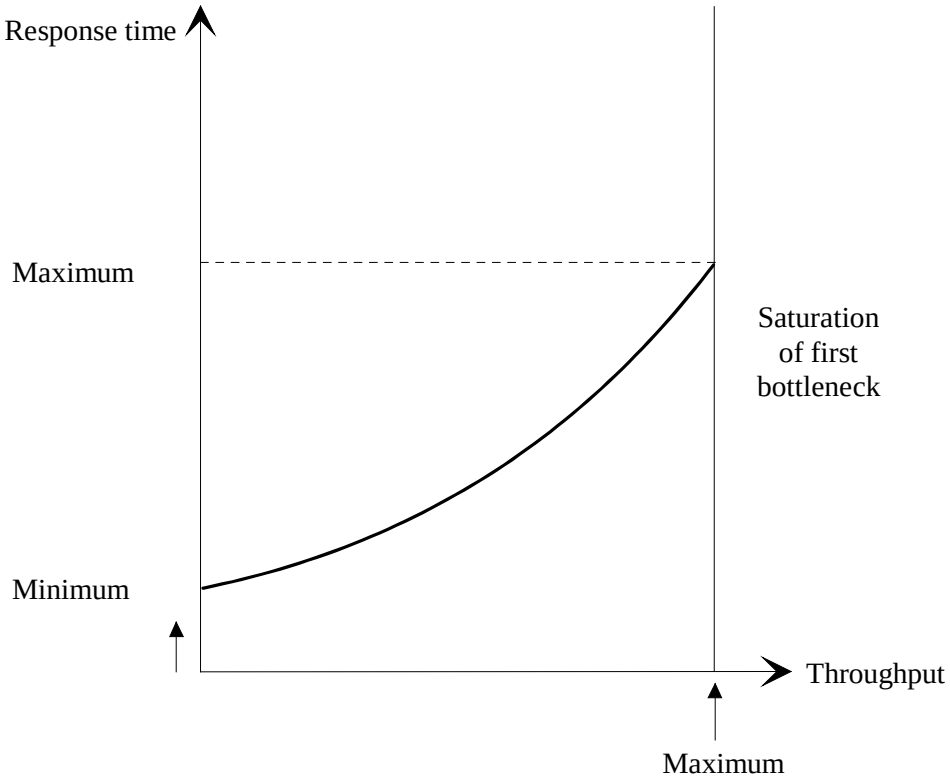


Figure 7-1. Response Times Versus Throughput

7.2.2 Large Networks

The number of terminals in a large network is more than 100 and can reach 3000. There is no limit to response time. When the system is saturated, response time can take more one minute.

A large network data processing installation may rapidly reach saturation. Throughput is regulated more by service capacity than by demand.

Response times become too slow during peak hours due to increased demands on the system. A public database query system freely accessed by all users is prone to saturation.

Response time is a ratio of the workload submitted to the system to the maximum throughput that the system is capable of handling near saturation:

$$\text{Response time} = \frac{\text{workload}}{\text{maximum throughput near saturation}}$$

Maximizing throughput at saturation increases the performance and decreases the response time.

A slow response time is the result of a throughput capacity that is too small at saturation, and hence poor system design.

7.3 Tuning the TDS Application

Distributing the load on the available hardware improves overall efficiency without affecting what is required. Tuning the TDS application improves the ratio of response times to throughput. SBR and TDS statistics help identify the points which should be optimized. Some examples are:

Problem: The system is saturated but the hardware resources are not.
Action: Check GAC-EXTENDED conflicts and TDS non-concurrency conflicts. If these are not responsible, increase the simultaneity level where possible (depending on software limits).

Problem: The system is saturated; the first bottleneck is the system disk.
Action: Use multiple-backing stores and move SYS.LIB to a lightly loaded disk/channel.

Problem: Many TPRs abort with RC=LONGWAIT or DEADLOCK.
Action: Condense the structure of the commitment units and avoid embedded conversation.

Problem: Many TPRs abort with RC=BUFNBOV.
Action: Increase the UFAS buffer pool size and entries concerning DEFERRED UPDATES and/or reduce the simultaneity level.

7.3.1 Optimizing TPRs

A TPR is a sharable compilation unit which is concurrently executed by several transactions of the same TDS application. An exchange (commitment unit) encompasses at least one TPR.

ESSENTIAL: Avoid writing TPRs that are too big and take up a lot of CPU time, main memory, or use many I/Os. Upper bounds per exchange are explained later in this section. Transactional processing should not substitute for batch processing.

ESSENTIAL: Do not use SENDs with EMI within the TPR. Although no exchange takes place, these SENDs are all synchronous except for the ***last*** SEND with EMI or EGI. Similarly, SENDs issued to terminals other than the current one are also synchronous. The result is reduced throughput. See Section on Message Handling without FORMS, *TDS COBOL Programmer's Guide*.

IMPORTANT: Reduce TPR loading overhead. As many I/Os are performed as there are code segments referenced. To have only one I/O per TPR, use the COBOL compile options:

```
CODAPND
PSEGMAX=64K
DSEGMAX=63K (because DSEGMAX must not exceed 65504 bytes)
NDEBUG
```

In COBOL 74, the maximum value of PSEGMAX is 32K; in COBOL 85, the maximum value of PSEGMAX is 64K.

IMPORTANT: After the kernel of TDS is loaded into main memory, all TPR-related I/Os are issued on SYS.LIB. Depending on throughput, to avoid a possible bottleneck, allocate SYS.LIB on a lightly loaded disk and channel using multiple backing store.

Useful: To suppress data refreshing I/O on SYS.LIB at the start of a TPR, apply the M command LOAD_TDS_MEMORY to the most frequently activated TPRs. The code segments and the initial copy of the data segments of the TPRs specified are allocated in memory instead of remaining on disk.

Useful: To minimize TPR memory occupancy whenever possible:
- Share code by using PERFORMs instead of macros
- Suppress unnecessary entries expanded by COPYs
- Avoid needlessly large working areas by using REDEFINEs
- Avoid a needlessly large TRANSACTION STORAGE.

TRANSACTION STORAGE is written to and read from the swap file:
- at each commitment point
- and at the end of each TPR where unmapping occurs.

7.3.2 Optimizing Commitment Units

A commitment unit starts with a consistent state of the database and completes later, leaving the database in a modified consistent state. To prevent inconsistencies in TDS-controlled files, GAC-EXTENDED locks all CIs and pages for the duration of the commitment unit. Once a page is locked in Exclusive Read or for *modification* by GAC-EXTENDED, it cannot be accessed by other commitment units until the commitment unit holding it releases the lock. Any attempt to access a page locked in Exclusive Read or for *modification*, produces a conflict. This conflict can result in the commitment unit being aborted and then restarted.

To detect and unblock apparent deadlocks, use the CHECK_TX_CONVERSATION command described in Section 9 *Master Commands*.

To optimize GAC-EXTENDED efficiency:

ESSENTIAL: Do not extend a commitment unit over the duration of either a conversation or a WAIT-TIME.

Conversation should always be preceded by one of these commitment requests:
- IMPLICIT COMMITMENT clause of the TRANSACTION SECTION
- explicit CALL "DFCMIT".

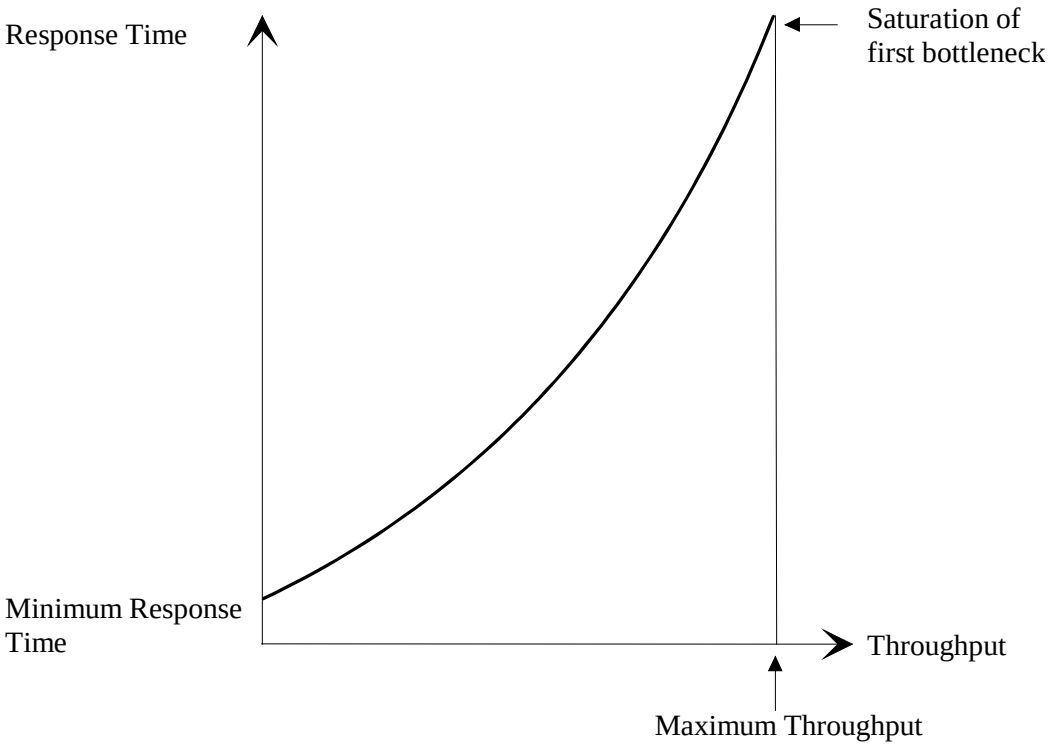
Likewise, whenever WAIT-TIME is specified in TDS-STORAGE, ensure that it is always in the last TPR of the current commitment unit (implicit or explicit commitment). If not, the resources remain locked for so long (typically 10 to 60 seconds) that conflicts will probably occur.

IMPORTANT: If possible, do not share files in update mode between a TDS application and batch while the TDS application is operating under full load. Sharing is often incompatible with good performance of either the TDS or the IOF/batch job and requires special tuning.

IMPORTANT: Do not have several commitments within a single exchange since too many commitments mean increased CPU and I/O overheads. Ideally, there should be only one commitment per exchange (IMPLICIT COMMITMENT), provided the following conditions for the commitment unit are satisfied:

- its elapsed time is about 1 second
- its number of locked pages averages 15.

If these conditions are not satisfied, conflicts can increase so affecting throughput.



7.3.3 Using UFAS-Extended and GAC-EXTENDED Facilities

GAC-EXTENDED allows TPRs to share controlled files in read or update mode while committing data for consistency. Conflicts on shared CIs or pages can arise depending on how two commitments simultaneously access the same object. GAC-EXTENDED conflicts are resolved by thoroughly analyzing the dynamic behavior of the database and subsequently tailoring its structure and physical mapping.

The files of a given TDS application can share buffers within a single buffer pool. These files may be controlled by GAC-EXTENDED. UFAS-EXTENDED supports a large number of buffers and allows specifying several buffer pools.

The higher the number of simultaneities, the larger the buffer pool(s) should be. For a TDS application running under release V6 or V7, the maximum number of buffers is 20000.

ESSENTIAL: With regard to the use of a buffer pool, a single IDS/II database is more efficient than multiple UFAS-EXTENDED files.

ESSENTIAL: A low GAC-EXTENDED conflict rate of less than 0.1 per exchange is a prerequisite to good performance. This can be attained through careful analysis, design, and mapping of the database. For instance, IDS/II user-controlled directories should only be read, and not processed with data to be updated.

CIs and pages are locked by the commitment unit until it terminates. When a conflict occurs, the TPRs needing access to the locked CIs and pages stay blocked for the duration of the conflict, equivalent to the duration of an average commitment. Too many commitments increase overheads, whereas lengthy commitment units cause too many conflicts. A compromise must be found between the two extremes.

Useful: Files assigned with SHARE=MONITOR and ACCESS=READ, should be specified with SHARED READ in the TRANSACTION SECTION to avoid unnecessary GAC-EXTENDED conflicts.

See *UFAS-EXTENDED User's Guide* and *GAC-EXTENDED User's Guide*.

7.3.4 Improving Simultaneity Level

The optimum simultaneity level that corresponds to the hardware configuration is determined by:

- throughput
- response times
- CPU busy time per exchange
- and the number of I/O operations per exchange.

Multitasking must be high enough to ensure simultaneous operation of CPU and I/Os for maximum throughput at the first bottleneck but not too large to decrease the throughput because of:

- thashing for both Virtual Memory and buffer-pool
- and GAC-EXTENDED conflicts.

ESSENTIAL: The simultaneity level needed for simultaneous operation of CPU and I/Os is not possible because of limits such as the Before Journal.

In the case of UFAS-EXTENDED, the most likely limit to increasing the simultaneity level are the linker limits.

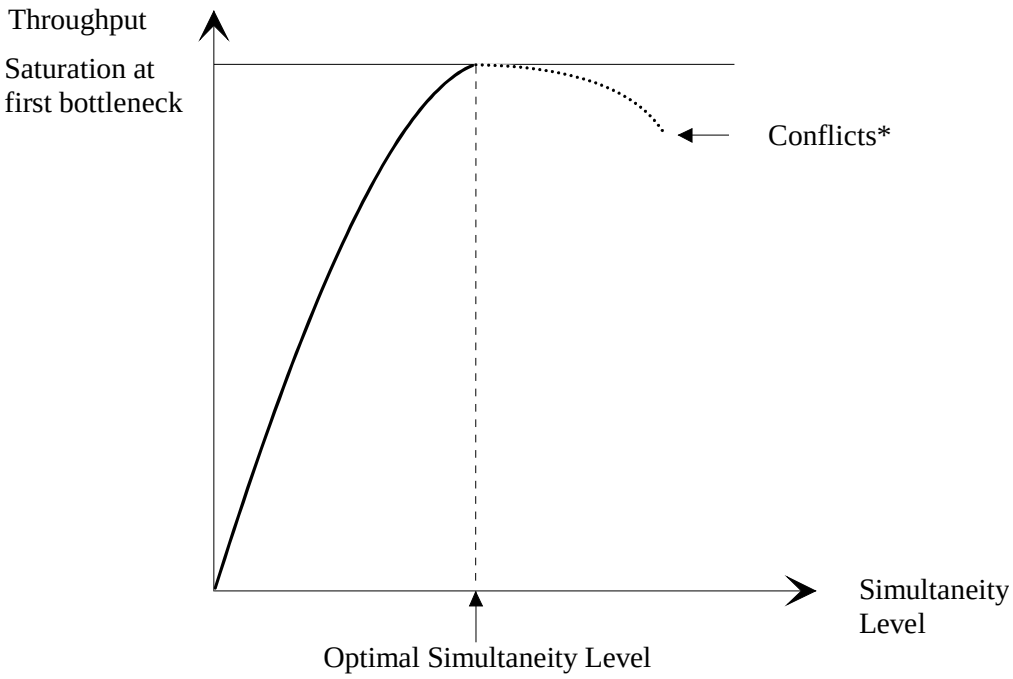


Figure 7-2. Throughput Versus TDS Simultaneity Level

NOTE:
* GAC-EXTENDED Conflicts, VMM thrashing and buffer-pool thrashing.

To determine the optimal simultaneity level:

1. Calculate the optimum level, ignoring the system limits.
2. Calculate the maximum simultaneity level (S), compatible with all possible limitations (UFAS table entries, vacant type 0 entries, available main memory, Before Journal, ...).

   For UFAS-EXTENDED files, the effective TDS simultaneity level is much greater than the TDS theoretical optimum simultaneity level in the absence of system limits.

   In V7, the maximum simultaneity level is:

$$250 - \min\left(2047,\ \text{Number of Terminals declared at NETGEN}\right) - \frac{}{32}$$

   The maximum number of commitment units for a TDS application is 2,047. If the number of terminals declared at NETGEN is a lower, the lower number is taken into account.
3. If the TDS theoretical optimum simultaneity level in the absence of system limits is greater than the effective TDS simultaneity level:
   - Reduce the number of simultaneities until performance starts to decrease.
   - Decrease the number of I/O operations by changing the file organization, for example, changing indexed to random and increasing CI sizes.
   - Reduce the total elapsed time by lowering access times with better localization of files over more disks and controllers.
   - Reduce the number of files accessed by a TDS application to increase the number of type 2 segments available for UFAS-EXTENDED buffers and the maximum simultaneity level correspondingly.
   - Use either a single IDS/II database with few areas instead of multiple UFAS-EXTENDED files, or use several versions of a TDS application, each version working on a subset of the initial file set.

IMPORTANT: If 40% or so of the possible simultaneity level is required, the following approximate algorithm may be used:

$$[1]\ Sopt > P * \left(1 + 90 * \left(Nio * r\right)\right) - \frac{}{Tcpu(\text{ms})}$$

This formula assumes enough peripherals online to make the CPU the first bottleneck of the system. It estimates the minimum simultaneity level needed to overlap I/O activity with CPU processing. A 100% overlap means that the CPU devotes its entire capacity to TDS.

$$[2a]\ Nb > S * (1 + U/Uf)$$ without DEFERRED UPDATES

$$[2b]\ Nb > S * (1 + U *(1/Uf + 0.4))$$ with DEFERRED UPDATES

This formula estimates the minimum number of UFAS-EXTENDED buffers needed to support a simultaneity level of S, where:

Sopt:	the TDS theoretical optimum simultaneity level in the absence of system limits.
S:	the effective TDS simultaneity level.
P:	the number of CPUs in a multi-processor system.
Nio:	the total average number of physical tape and disk I/O operations per exchange for TDS activity.
r:	the maximum portion of total CPU power available for TDS including its induced consumption in J=0 and for the communications server. r=1 when the TDS induced CPU consumption approaches 100%.
Tcpu:	the average CPU busy time per exchange, on the effective CPU model, under full load including induced consumption in J=0, communications server.
Nb:	the number of available entries in RESERVE AREAS clause for UFAS-EXTENDED file buffers.
U:	the average number of user UFAS-EXTENDED physical I/O operations per exchange.
Uf:	the average number of user physical I/Os per UFAS-EXTENDED file accessed within an exchange (commitment unit). These formulas outline the constraints on the simultaneity level and on the number of simultaneously opened TDS files. Depending on initial assumptions, these formulas can be applied in various ways to compute certain effective limits.

IMPORTANT: Many simultaneities are usually better than too few, if sufficient main memory is dedicated to the TDS application and the number of buffer-pool entries is large enough. For additional simultaneity, increase the working set by 70-150 Kbytes.

7.3.5 Avoiding Non-Concurrency and Serialization

Serialization results in only one commitment unit being active at a time. This occurs when an exclusive transaction, not concurrent with any others, is activated. It may also be initiated by a TDS application after certain aborted TPRs are restarted.

ESSENTIAL: Avoid non-concurrency because it degrades performance. When the following commands are executed, they cause serialization:

[M] LOAD_TDS_MEMORY
[M] UNLOAD_TDS_MEMORY
[M] MODIFY_TDS SMLIB.

TPRs tested with PCF commands also cause serialization.

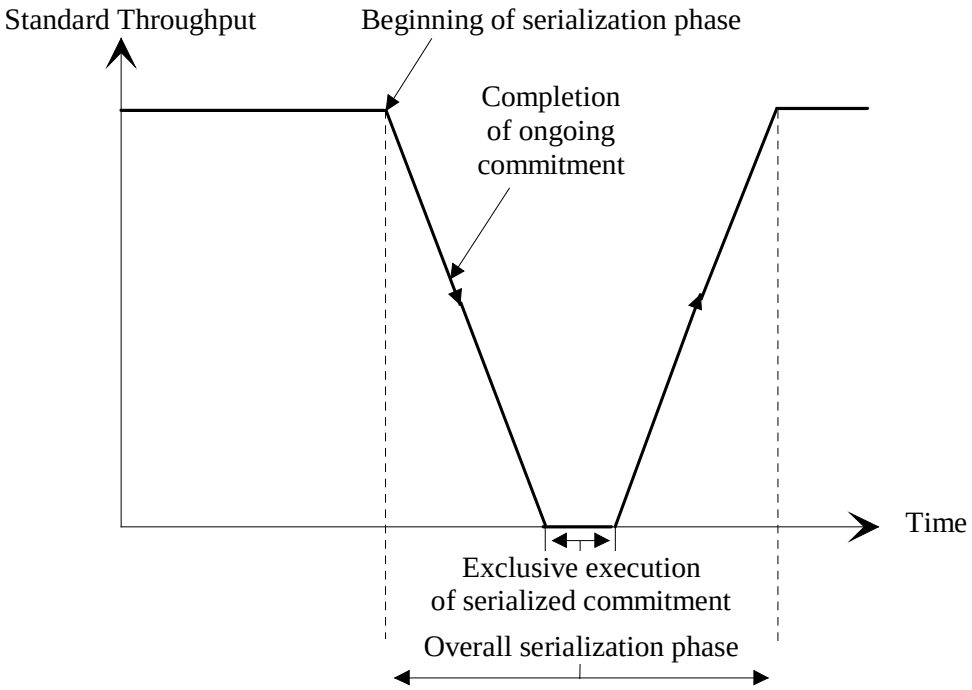


Figure 7-3. Serialized Commitment Units

IMPORTANT: Commitment units of TPRs tested under PCF, are executed in exclusive mode between TRACE...PCF and TRACE OFF commands. TRACE should not be performed on an active TDS application.

Likewise transactions declared FOR DEBUG in the TRANSACTION SECTION should not be executed concurrently with active transactions. All files updated by FOR DEBUG transactions are rolled back and incur more overheads and I/O operations on journals than transactions without FOR DEBUG.

7.3.6 RPC Transaction Management

In the DCE context, TDS transactions can communicate with Unix applications through the RPC mechanism. RPC transactions can use more CPU time and system resources than other transactions. To avoid degrading performance and to control resources used by the RPC mechanism, the TDS administrator may control the number of processes which can concurrently process RPC transactions by:

- fixing the RPC simultaneity level number in the STDS (RPC SIMULTANEITY Clause). This level can be dynamically changed at runtime through the parameter RPC_SIMUL of [M] MODIFY_TDS.
- declaring all the transactions using RPC services:
  - statically by specifying the RPC SERVICE USED Clause in the description of the transaction in the STDS
  - and dynamically at runtime through [M] MODIFY_TX.
- displaying at TDS runtime:
  - the RPC simultaneity level number or RPC statistical information with [M] DISPLAY_TDS
  - and the use of RPC services by a transaction with [M] DISPLAY_TX.

NOTE:

If the RPC simultaneity level number is not specified in STDS, RPC simultaneity regulation can never be activated during the TDS session. ***All*** processes can therefore run RPC services.

If this level is specified in the STDS, it can be dynamically changed at runtime through the RPC_SIMUL parameter of [M] MODIFY_TDS.

7.3.7 Before Journal

Before Journal saves the *before* image of each rewritten CI or page. These images are needed to roll back all database modifications left uncommitted by aborted commitment units. Taking *before* images increases the number of Input/Output operations. However, DEFERRED UPDATES instead requires additional buffers in memory. For the Before Journal, the number of input-output operations per exchange should not exceed 1.5.

IMPORTANT: If enough buffer-pool entries are available, use DEFERRED UPDATES. UFAS-EXTENDED files updated in the DEFERRED UPDATE mode can be shared among several TDS applications ("$ASSIGN..., ACCESS=WRITE, ..."). However, these files cannot be updated in the same way for batch/IOF processing.

IMPORTANT: If the Before Journal must be used due to CI splitting, allocate the TDS extension of the Before Journal file on a lightly loaded disk/channel, to avoid a bottleneck.

Optional: When a given transaction either is read-only or uses DEFERRED UPDATES for all updated files, specify SUPPRESS BEFORE JOURNAL in the TRANSACTION SECTION to reduce CPU overhead.

7.3.8 After Journal

After Journal saves the *after* image of each rewritten logical record. These images are used to restore the database. The After Journal is mandatory when DEFERRED UPDATES is specified. For the After Journal, the number of input-output operations per exchange should not exceed 1.5.

After Images are placed in buffers and written to the After Journal:
- when the current After Journal block is full
- when a commitment unit ends and the current block contains an image belonging to it.

Useful: To use all the After Journal functions, allocate the After Journal files on a disk rather than on tape.

Useful: To avoid multiple I/O operations on the After Journal, define a sufficiently large size for the After-Journal blocks so that only one I/O operation is performed for each commitment. The maximum block size for the After Journal is:
- 32,000 bytes (or 1 track) for a disk
- 12,800 bytes for a tape.

After a TDS session, run MAINTAIN_JAS specifying PRINT. If more than 10% of the After Journal blocks are filled to 90-100%, use MNJAS to increase the blocksize.

Useful: Because the After Journal is shared by all applications and by all journalized files on a given site, avoid bottlenecks by allocating the After Journal files on a fast, lightly loaded disk. For tape, use a 200 ips, 6250 bpi handler.

See *File Recovery Facilities User's Guide*.

7.3.9 TDS Swap File

Two swap file I/Os are performed per commitment unit or exchange. The number of I/Os per exchange should not exceed 2 for the swap file. Two additional I/Os are performed for each intermediate cleanpoint (TPR chaining within the commitment unit), occurring when:

- a WAIT-TIME is specified between two TPRs
- a synchronous SEND with EMI statement is performed in the completing TPR.

ESSENTIAL: To avoid a possible bottleneck, allocate as many swap files as necessary on lightly loaded disks. A swap disk should not serve more than 10-15 I/Os per second to maintain good performance.

Useful: Each swap file entry holds a number of system-defined and user-defined areas, each entry not exceeding 64 Kbytes.

To avoid long swap I/Os, try reducing TRANSACTION STORAGE size.

Optional: Specify FOR INQUIRY for transactions with a single exchange per commitment unit. Since file-inquiry transactions involve no file updating, avoid two swap file I/Os for each such exchange. The transaction is not restarted if an abort or crash occurs.

7.3.10 Using FORMS and LIBRARY

Store forms on a UFAS-EXTENDED file. If not, use only one binary library.

7.3.11 Improving Use of Main Memory

To improve the use of main memory, follow these suggestions.

ESSENTIAL: Performance cannot be improved when memory is constrained. For example, when there is more than 1 missing segment per exchange, loading the TPR does not cause missing segments.

An initial approximation for TDS application on a dedicated system with 50 to 100 terminals needs at least 8 megabytes of main memory to perform satisfactorily on a DPS 7000/300.

Once this application has been used in a real situation, analyze the JOR to find out the amount of memory actually used.

This value does consider the amount of memory occupied by the buffer pool.

16 megabytes is the minimum requirement for main memory. The POOL dimension must be at least 8 megabytes in size in order to avoid memory conflicts.

IMPORTANT: Virtual Memory tuning is meaningless on a system dedicated to a single TDS application. Performance improves when the TDS dimension is not sized.

IMPORTANT: Sizing the TDS dimension requires more physical memory. This should be done only when sufficient main memory is available for the total system workload.

7.3.12 Configuring Communications Network

Note the following points when configuring the communications network:

ESSENTIAL: Optimizing a TDS application is meaningless if transmission over the network is slow, for example, 3 to 5 seconds. On many sites, response times can be improved by adding new lines or by replacing the existing lines by faster ones.

IMPORTANT: Optimizing the network requires:
- sufficient memory for I/O buffer space for frontend processor servers
- and fast communications lines, depending on the message length.

Video terminals may require transmitting complete screens of 2-4 Kbytes. Although FORMS optimizes transfers, the time taken can still be too long. On a 2400-baud line, a 2-Kbyte transfer will keep the line busy for more than 6 seconds.

Useful: A TDS application performs best when all active users remain logged on from the beginning until the end of the TDS session. Logon and logoff while a TDS application is running at full capacity involves multiple access control and housekeeping thereby slowing down TDS throughput.

Optional: When all transactions use a common form, retain the form from one transaction to the next by specifying NO IMPLICIT RELEASE in the TDS SECTION. Specifying this clause saves one I/O.

7.4 Measuring TDS Performance

Optimizing performance requires accurate throughput measurements involving response times, saturation levels, usage percentage, and queue lengths.

SBR takes such measurements with minimal overhead. The SBR file should be saved together with all the reports for the test session.

When a thorough analysis is intended, SBR COLLECT should be performed. Start the SBR with:

```
LEVEL=EXTENDED  TDS=YES
```

For TDS-specific parameters, the command for SBR analysis, is:

ANALYSIS TDS, *tdsname*

For complicated cases, more than one test session may be needed for comparison. Take measurements for all sessions with identical hardware and software configurations using the same workload characteristics, being the number of users and TPRs activated.

In parallel with SBR, simulate the TDS application on the system using TILS. TILS complements SBR.

The profile of an average exchange can be derived from the SBR charts or the TDS JOR report. To be statistically meaningful, these occurrences must take place over a long time period, a session lasting a day.

Where the computed values exceed typical upper/lower bounds, poor performance results.

See *TILS User's Guide* and *System Behavior Reporter User's Guide*.

7.4.1 Improving Response Times

Reasonable demands per exchange results in a reasonable response time of 1 to 5 seconds.

ESSENTIAL: To avoid slow response times, do not exceed the following average upper limits per average exchange for a medium-to-large network:

`No. of TPRs`	1 - 3
`No. of commitments`	1 - 1.5
`No. of locked pages per commitment`	5 - 10
`No. of physical user I/Os`	5 - 15
`No. of GAC -EXTENDED conflict:`	< 0.1
`Total CPU time (DPS7000/40)`	250 millisecs

These values do not guarantee faster response times, but disregarding them leads to slow response times.

For large networks, slow response times are usually a result of insufficient throughput capacity.

A slow response time not related to throughput, does not improve even if the load were reduced.

IMPORTANT: Queries are so complex that the system takes a long time to process them. If these exchanges involve few I/Os but use a lot of CPU time, a faster CPU may be needed. If these are *batch-like* exchanges, the solution is to redesign the application.

Useful: Additional delays are caused when servicing the exchange with slow communications lines for transmitting full screens of data. Delays are avoided by configuring faster lines.

7.4.2 Increasing Throughput to Avoid Saturation

When the load is increased to saturation, performance becomes degraded due to competition for hardware and software resources. This results in queuing and hence, additional delays. To increase the throughput of an application, proceed as follows:

- find the bottleneck and analyze its effects
- share the load for the given configuration.

IMPORTANT: For most systems, 20% of the transactions account for 80% of the workload. For an active application, use the TDS statistics and SBR reports to identify those components which contribute most to the total workload. It is this *core* of the application, that must be optimized.

Eliminate incidents which serialize and thereby decrease throughput over long periods, even if they are infrequent.

7.4.3 Avoiding Interference

Interference occurs when other jobs executing simultaneously with the TDS application degrade TDS response times. Degradation is measured against the response times of the TDS application when it executes alone. Two SBR measurements provide the answer:

- one with TDS alone
- another with the full workload and the same TDS application.

One process interferes with another only when both simultaneously and frequently require the same resource. Interference is unlikely when either:

- resources are not shared
- a resource is shared but is lightly loaded such as a shared CPU running under 30% capacity
- a heavily loaded resource is shared but simultaneous access is unlikely such as online files and batch files on the same volume
- a heavily loaded resource such as CPU time is simultaneously required by both processes but the TDS application has privileged priority.

To solve interference problems, take one or all of the following actions:

- suppress sharing
- allocate more memory for shared resources
- schedule jobs to avoid conflicts on shared resource
- assign privileged priority to the TDS application.

Common interference problems are:

ESSENTIAL: *Virtual memory contention*
Physical memory is insufficient for mapping TDS and all other jobs. Consequently, the missing TDS page rate grows to more than one page per exchange.

Increase the amount of available memory and/or decrease the non-TDS workload by reducing the multi-programming level.

ESSENTIAL: *GAC-EXTENDED conflicts*
Some files are shared in update mode between the TDS and another Batch or IOF job. Sharing on such frequently referenced TDS files degrades TDS performance.

Share the files being updated among the TDS applications only, making sure that both TDS applications use the DEFERRED UPDATES option for these files.

IMPORTANT: *Virtual memory interference*
Although the physical memory available can handle the total workload, interference slows down the TDS. This occurs particularly when the time to load missing pages is high due to overloaded SYS.LIB backing store files.

To investigate the effective usage of memory, use the SNAPMM option of SBR.

Increase the amount of memory:

- and/or decrease the non-TDS workload
- then allocate a private and non-shared portion of this memory to the TDS.

Dedicate the amount of memory to TDS by sizing the TDS dimension which affects all TDS applications, through the MEM TDS command at CONFIG time.

IMPORTANT: *I/O queuing*
TDS accesses files on volumes or through channels shared with other job steps, thus incurring additional queuing delays.

Check the SBR report for the necessary corrective action:

- Use SBR-EXTENDED + VOLLIST to identify the volume, channel or file in conflict.
- Separate the TDS volume, channel or file from those belonging to other job steps.
- Increase the number of disk drives depending on throughput.

Useful: *CPU queuing*
The relative dispatching priorities are not in the right order. They must be as follows:

communications servers > SBR > TDS > IOF > Output Writer > x

When a TDS application shares the CPU with other jobs, 90% or more of the CPU can be used without detrimental effects on TDS response times, provided that:

- relative priorities are correctly ordered
- the TDS application does not use more than 50-70% of the CPU.

If the TDS application uses more than 70% of the CPU, a faster CPU is needed. When CPU contention has been resolved, all other contentions such as I/Os will have to be considered.

Optional: *Interference caused by centralized services*
Services are processed by the nucleus (J=0) with higher priorities than TDS such as:

- Loader
- Scheduler
- Stack overflow handler
- PIO termination handler.

These slow down TDS activity when they are activated frequently.

7.4.4 Analyzing Bottlenecks

Overincreasing throughput saturates the system resulting in at least one bottleneck. At saturation, overall system throughput or the number of exchanges per second can be calculated as follows:

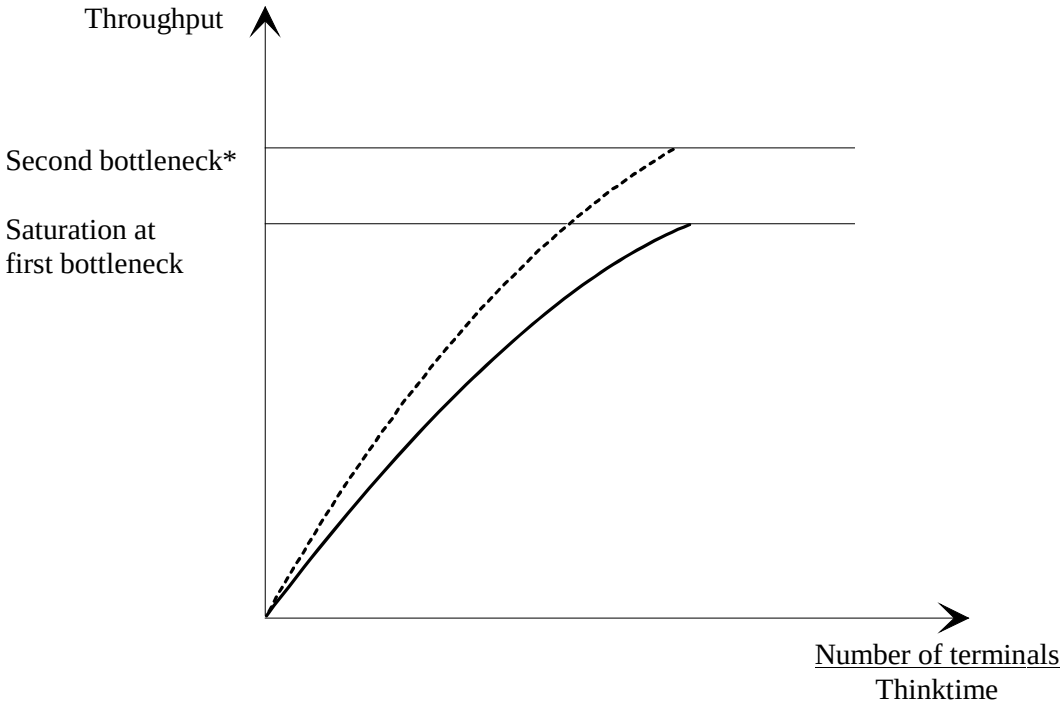


Figure 7-4. Throughput Versus Load

NOTE:
* The second bottleneck occurs after the first bottleneck has been resolved.

To increase the throughput capacity:

- identify the first bottleneck
- increase its throughput so that the next bottleneck is then encountered
- repeat the first two steps until all bottlenecks are resolved and performance improves.

7.5 Managing Data

Managing data is essential for all TDS applications. To obtain optimum throughput and response times:

- Reduce the number of times that files are accessed to limit CPU and elapse times.

  The average I/O has an elapse time of 30-50 milliseconds and takes 5-10 milliseconds of CPU time. The average number of I/Os per TPR is calculated from the TDS statistics report and the JOR by dividing the I/O figure by the number of TPRs.

- Take commitments as soon as possible to reduce concurrent access conflicts.

  Avoid conversations within a commitment unit as these lock CIs for long periods. The CI is locked for the entire commitment unit even if the CI is not updated.

  In IDS/II, set selection involves accessing several pages which are then no longer available to other transactions. To increase file availability, avoid file or database structures with centralized features such as user-controlled directories, or place data in read-only files.

  The initial loading of the CIs should be organized to leave space proportional to the expected rate of record insertions. This reduces physical scattering of related data and thus the number of CIs that must be locked.

  Traces produced by TDS indicate any aborts due to concurrent access conflicts with or without deadlocks. If a deadlock occurs, the transactions concerned can be declared NON-CONCURRENT. It is more efficient to wait and avoid a potential deadlock than to let TDS or GAC-EXTENDED resolve the deadlock when it occurs.

- Use the SHARED READ or SUPPRESS CONCURRENT ACCESS where possible.
- Examine the TDS statistical report for details of deadlock and concurrent access conflicts. Eliminate these anomalies because the resulting aborts:
  - create considerable I/O activity
  - incur further processing
  - increase response times
  - involve operator intervention.
- Assess the methods used for ensuring file integrity. For example, protection by the Before Journal increases I/O activity. Determine if the Before Journal is necessary.

- Limit the dynamic reorganization of indexed files which cause heavy I/O overheads. Provide enough free space when creating the file, see Paragraph 7.5.1 *Using the CIFSP UFAS-EXTENDED Parameter* and the *UFAS-EXTENDED User's Guide* for more information.
- Split records such as inventory records and directories that generate frequent access conflicts as they are accessed by many transactions. When these records are broken up into several smaller records, they can be dispersed throughout different CIs.

7.5.1 Using the CISIZE UFAS-EXTENDED Parameter

In UFAS-EXTENDED, the number of buffers created is given in the JOR as SEGCR. The size is always a multiple of 512 bytes. All CIs (data, index and label) are the same size.

To optimize their use:

- minimize the number of buffers having different CISIZE values
- specify only 2 index levels for files.

7.5.2 Using the CIFSP UFAS-EXTENDED Parameter

The CIFSP parameter is used for file organization. Values for this parameter depend on:

- the number of records per CI
- the rate of file expansion
- and the frequency of file reorganization.

If the file expands frequently, the split rate tends to be high. If the rate occurs for every session, the CIFSP value does not need to be as high as for less frequent reorganizations.

This parameter is inserted in one of two modes, either random or mass.

In random insertion, records are created one at a time anywhere in the file. This requires free space to reduce splits. The CIFSP parameter should be specified in the PREALLOC statements. To determine the values for this parameter:

- run the FILLIST utility periodically to obtain the number of CI splits
- then decide, on the FILLIST results, which values to declare for the CIFSP parameter in the PREALLOC statement and how often to reorganize the file.

In mass insertion, records are created in ordered sequence of contiguous key numbers. All new records go one behind the other in the same CI.

7.5.3 Responding to UFAS-EXTENDED Return Codes

The following return codes may appear if a TDS application using UFAS-EXTENDED files has not been optimized:

- UFAS RC=CMWSOV
- VMM RC=CMWSOV or UFAS RC=BUFNAV
- UFAS RC=BUFNBOV

When UFAS RC=CMWSOV is displayed, the poolsize is not large enough. Increase the poolsize value, and, therefore, the DWS in the $SIZE statement.

When VMM RC=CMWSOV or UFAS RC=BUFNAV are displayed, increase the DWS in the $SIZE statement.

When UFAS RC=BUFNBOV is displayed, the minimum number of UFAS-EXTENDED buffers exceeds the number of buffers specified in the RESERVE rsvarea AREAS clause.

7.6 Allocating Resources

Another part of optimizing an application is ensuring that hardware, memory and buffer resources are efficiently allocated.

7.6.1 Distributing Hardware Resources

A site performs poorly because its configuration only allows a fraction of the hardware/software resources to be used to capacity. This problem not restricted only to TDS applications. Imbalances can frequently be traced to bottlenecks which reduce system throughput.

A configuration is optimized when the frequency of using a resource is directly proportional to its cost.

ESSENTIAL: *Imbalance between the CPU and I/O devices*

A fast CPU cannot be used to its capacity when there is a lack of peripherals.

Add new disks and MSPs (Mass Storage Processors) to obtain more simultaneous I/O accesses.

ESSENTIAL: *Disk Imbalance*

Enough disks are online, but some are overloaded and others are nearly idle. The system is I/O bound, and a queue builds up for volumes mounted on overloaded disk drives.

Remount volumes containing files which are extensively updated on idle drives.

ESSENTIAL: *CPU/Memory Imbalance*

Main memory is too restricted for the multi-programming level configured.

Add more memory.

IMPORTANT: *Disk/MSP Imbalance*

The workload is I/O bound because of long delays in servicing disks. Delays are long due to MSP conflicts and not to disk overload.

- Add new MSPs and distribute the load among them by using the dual LOS (Level Of Simultaneity) Controller (two MSC paths per disk). For increased performance and availability, it is preferable to have two valid access paths to a disk drive rather than a single access path.
- Use the configuration option RPS (on by default) to reduce MSC overloading.

7.6.2 Allocating Memory

When considering the amount of memory to be allocated, define the sizes of:

- the swap file buffer
- the TDS Working Set.

7.6.2.1 Computing the Size of the Swap File Buffer

As from TDS-V6, the swap file buffer is managed as two separate segments, namely system areas as distinct from user areas. The result is an increase in the size of the swap file buffer used to hold the user's context. The swap file buffer now holds up to 130047 bytes for VBO files and 126975 bytes for FBO ones.

The size of the swap file buffer is the amount of memory set aside for moving a user context in and out of swap files. This size applies to all the transactions for a given TDS application. To estimate the size of the swap file buffer, two segments and the length of each area stored in them must be considered.

First buffer segment size is 64512 (VBO) or 61440 (FBO) bytes maximum and is composed of Areas 1 through 9. Second buffer segment size is 65535 and is composed of Areas 10 and 11. The following tables jointly describe the contents of the Areas and their lengths and related information.

Length of Areas in the Swap File Buffer

AREA	*Contents*
1	Message (input or ouput) and its control information
2	Session information
3	Transaction control information
4	XCP2 log information
5	ORACLE log information
6	GAC-EXTENDED information
7	Access Method (UFAS-EXTENDED and IDS/II) information
8	TDS-STORAGE
9	FORMS information
10	PRIVATE-STORAGE
11	TRANSACTION-STORAGE

AREA	*Length of Control Data*	*Length of User-specified-Data*	*Rounded Up to Next Block Size*
1	128	*maximum message length*	Yes
2	312* (3+numaux)	-	No Yes if XCP1 used
3	544	-	No Yes if XCP2 used
4	20 {without XCP2} (105 + 64) *C {if XCP2 is used}	-	Yes {No if ORACLE is not used}
5	0 or 1024	-	Yes
6	84*f + (s*k)	-	No
7	128	-	No
8	704	Formatted Message Length (or 0)	Yes
9	-	PRIVATE and TRANSACTION STORAGE	Yes

See overleaf for Explanation of Mnemonics and Letters.

Explanation of Mnemonics and Letters:

- Some values such as 128, 311 or 607 are those of TS6152 and can fluctuate.
- *maximum message length* is value in MESSAGE-LENGTH Clause of TDS SECTION.
- *numaux* is in NUMBER OF XCP1 TRANSACTION AUXILIARY SESSIONS Clause of TDS SECTION.
- *maxconv (maximum XCP2 conversations)* is in XCP2WKS NETGEN Directive.
- *g* is GAC-EXTENDED area size computed as follows:

  `16 + (4 * f)` where *f* is the number of controlled files declared at TP7GEN including IDS areas.
- *u* is the ACCESS METHOD area size computed as follows:
  - Sum up the sizes of all controlled files excluding IDS areas with the algorithm:
    `(16 + x * ( 28 + k + msk ))`, where:
    . *x* is set to 16 if file is used under IQS or is set to 1 if not
    . *k* is the size of the primary key for indexed sequential files
    . *msk* is the largest size of secondary key (if any) in the file.
  - *u* is the smaller value between the result of the summation and 6144.
- *i* is the IDS area size computed as follows:
  - Compute the size needed for each schema according to the algorithm:
    `(11 * nbas + 10 * nbrs + 10 * nbss + 76)`
    . *nbas* is the number of areas in schema.
    . *nbrs* is the number of records in schema
    . *nbss* is the number of sets in schema.
  - To obtain *i*, multiply the largest schema computed size by *maxschema* in MAXIMUM NUMBER OF SCHEMA PER COMMITMENT Clause of INPUT-OUTPUT SECTION.
- *Formatted Msg Lgth* is specified in the FORMATTED MESSAGE-LENGTH Clause of TDS SECTION when FORMS is used; otherwise it is equal to zero.
- j = 96 [+ 28 *if Terminal Adapter used*].
- *m* is not taken into account if FORMATTED MESSAGE LENGTH Clause is used.

  m = 512 if Terminal Adapter is used without FORMS; otherwise
  m = 10647 [+ 2057 *if SA7 is specified*]. SA7 is specified in VALUES when invoking TP7GEN. SA7=YES means that TDS is protected by SECUR'ACCESS.

- TRANSACTION-STORAGE is the largest value declared for TRANSACTION-STORAGE Clause in the MESSAGE Statement of TRANSACTION SECTION including system transactions such as BYE or M of the TDS application.

  System transactions are automatically supplied and are in either the output file or *tdsname*.EDITION produced when executing TP7GEN.

Usage:

To calculate the length of the swap file buffer for each row 1 through 11:

- add the length (control and user information) and sum the result
- when indicated, round up the result to the next block size after having added 8 bytes

The block size (BLKSZ) of the swap file is 1024 bytes by default. For swap files allocated on FBO volumes, allow for 4096 bytes.

Constraints:

- The cumulated size from Area 1 through Area 9 being the first swap buffer segment, must not exceed 64 Kbytes minus 1 block.

  If the cumulated size exceeds 64 Kbytes minus 1 block, then TDS will reduce Areas 6 and 7 allocated to GAC-EXTENDED and UFAS-EXTENDED.

  If the remaining memory to be allocated to GAC-EXTENDED and UFAS-EXTENDED is large enough (1 Kbyte is required), then TDS initialization is successful and the TDS job may run.

  If the reduction is not enough, TDS initialization is unsuccessful. Reduce the maximum length defined in MESSAGE LENGTH or FORMATTED MESSAGE-LENGTH Clauses so that the size of the first swap buffer segment fits into 64 Kbytes minus 1 block.

  The problem that reduction can cause is that transactions using too many files may abort with return code DATALIM or may cause a TDS application to abort with MU85 error message.

  If a transaction aborts with DATALIM, design the application so that:

  – TPRs using many files without commit are avoided
  – the number of files accessed in the TPR is reduced.

  If TDS aborts with the MU85 error message, reduce the number of CALL "KEEP-CURRENCIES" used in commits.

- The declared length of PRIVATE-STORAGE plus the length of the largest TRANSACTION-STORAGE in STDS subfile must not exceed 65527 bytes.

7.6.2.2 Calculating the TDS Working Set

The DWS for the TDS application is defined in the SIZE statement.

The syntax of the SIZE statement is:

`SIZE` *`size`*`, POOLSIZE` *`poolsize`*`, NBBUF` *`nbbuf`*`;`

where *size* is the size of the DWS in units of 1024 bytes.

Usage:

DWS is calculated from the following:

- The TDS nucleus, produced as a result of the LINKER run during TP7GEN execution, is based upon the sum of the values given in the Linker listing for each of the following:

  MINIMUM CONTROL MEMORY REQUIRED
  MINIMUM USER MEMORY REQUIRED
- The POOLSIZE parameter in the SIZE statement
- The size of buffers. The size of buffers belonging to non-controlled files is determined by allowing 2 buffers for sequential files and 1 buffer for other files and then summing up the sizes of all the files opened simultaneously.
- TDS swap buffers. These buffers are calculated as follows:

  `S * SWSZ + SWDIRSZ + SWCPSZ`, where:

S	level of simultaneity
SWSZ	Swap Buffer Size
SWDIRSZ	Swap Directory Size; 64 Kbytes
SWCPSZ	Swap Channel Program Size:
FBO files only	`(512 * S) + 512`
VBO files only	`(512 * S) *` *`(Nbr of swapfiles)`* `+ 512`

To determine the declared working set of the TPRs, the following formula should be applied:

DWS of TPRs = $S * (d + C + L)$, where:

S level of simultaneity
d size of all the data segments of the average TPR
C size of the largest code segment of the average TPR
L size of the linkage segment

7.6.3 Allocating Buffer Space

The number of buffers and the amount of buffer space is calculated on the basis of a buffer pool. A pool is a set buffers for files defined with the same pool name via the JCL statement DEFINE, or defined implicitly by TDS in a pool named *tdsname*.

Several pools may be specified for the same application. By default, there is only one real buffer pool and a pseudo-pool named DEFT for TDS non-controlled files.

Using a large UFAS-EXTENDED/IDS/II buffer pool (32 megabytes) can reduce the number of physical I/O operations per exchange by up to 40%.

Buffer space is allocated as follows:

- in the JCL with the POOLSIZE value specified in the SIZE statement
- at TP7GEN with the value of the RESERVE AREAS clause in the TDS SECTION
- by the NBBUF parameter of the DEFINE statement to specify the number of buffers.

To choose which parameter(s), it is necessary to know how a buffer pool is handled and how the parameters of the application effect buffer size.

7.6.3.1 Managing the Buffer Pool

UFAS-EXTENDED creates as many buffers as possible, up to the limit of either the NBBUF value or the POOLSIZE value. These buffers can be in one of the following states:

Busy	the buffer is active and contains a CI which can be read or updated.
Write Deferred	the buffer contains a valid CI which is continually updated. This buffer is kept active until the commitment unit which has locked the CI in exclusive mode terminates. The number of Write-Deferred buffers used for a transaction depends on the number of updates made by the transaction on files for which Deferred Updates is requested. See *File Security Options*.
Remember	a buffer that contains a valid CI currently not being accessed. Such a buffer is kept in memory to be subsequently reused.
Empty	see *UFAS-EXTENDED User's Guide*.

In summary, the minimum number of buffers and the amount of buffer space must be greater than the number of buffers and buffer space of the Busy and Write-Deferred buffers. Otherwise, TDS switches from the Deferred Update mode to the Before Journal to decrease the demand for buffers. This degrades the overall TDS performance.

Since the technical status 7458, a generation clause of TDS SECTION, FILE-OPENING OPTION is FREEBUF can be used. This clause allows the UFAS buffer manager to release the buffers used at each primitive for all the TDS user's files, controlled or not controlled. In this case, these UFAS buffers will be freed and kept in remember state. If the total number of buffers is not large enough, the number of physical I/O operations may increase, thus reducing the performances ; so, it should be necessary to increase the value given in the RESERVE AREAS clause of the TDS generation.

7.6.3.2 Effects of Application Parameters

To evaluate the global POOLSIZE and the number of buffers NBBUF, compute the mean average of active commitment units. This number is usually larger than the simultaneity level because a commitment unit may include several chained TPRs whereas the process (simultaneity) can be unmapped at the end of each TPR. This is the case for TPRs that hold a conversation or for TPRs without the NO AUTOMATIC UNMAPPING clause.

Theoretically, the maximum number of active commitment units for a TDS application (with commitment units containing several TPRs) is the minimum number of declared terminals and the number of LOCKID defined at configuration time. Adjust this value according to the type of application and the type of commitment unit.

7.6.3.3 NBBUF Parameter of DEFINE Statement

Defines the number of buffers that may be allocated to a file. Use this parameter and the associated BUFPOOL parameter to specify a buffer pool.

Syntax:

```
DEFINE ifn, BUFPOOL=name4, NBBUF=nbbuf;
```

Usage:

Calculate the number of buffers (NBBUF) using this algorithm:

```
NBBUF = Number of controlled files + k
```

where *k* is the mean average of active commitment units being the number of BUSY buffers added to the number of buffers containing Deferred Updates.

7.6.3.4 POOLSIZE Parameter

Description:

Defines the maximum memory size which may be used for all the UFAS-EXTENDED and IDS/II buffer pools.

Syntax:

`SIZE` *`size`*`, POOLSIZE=`*`poolsize`*`, NBBUF=`*`nbbuf`*`,`
where:

size is the TDS application working set in Kbytes

poolsize is in units of Kbytes

nbbuf is the number of buffers.

The *poolsize* value is the part of the *size* parameter dedicated to UFAS-EXTENDED/IDS/II buffers regardless of the file or pool to which they belong. The poolsize value includes the UFAS-EXTENDED buffers. The buffer manager tries to fit as many buffers as possible up to the limit of the poolsize value, unless the maximum number of buffers for the pool is already reached.

If specifying several buffer pools, the POOLSIZE value indicates the total amount of memory for all the buffer pools combined.

Usage:

POOLSIZE value can be approximately estimated using the following formula:

`POOLSIZE = NBBUF * `*`mean`*` CISIZE`

CISIZE must be rounded up in multiples of 4 Kbytes.

7.6.3.5 RESERVE AREAS Clause

RESERVE AREAS Clause in TDS SECTION specifies the number of UFAS-EXTENDED or IDS/II file buffers to be reserved for all transactions.

8. Modifying a TDS Application

Modifying a TDS application involves:

- ascertaining what needs modifying
- where, how and when the modification is applied
- and how to verify that the modification is effective.

This section also gives complementary information on what to expect and avoid.

8.1 Adding, Removing or Modifying a File

A file may need to be inserted or removed to add new transactions to the application or to modify existing transactions.

Files are included in or removed from the TDS Generation and the JCL ASSIGN statements used to launch the TDS job. The clauses to be modified in the generation depend on whether the file is TDS-controlled or not.

For TDS-controlled files, modify:

- TDS-FILE-DEFINITION Clause
- PROCESSING-MODE and FILE-INTEGRITY Clauses in PROCESSING-CONTROL
- ASSIGN statement of the JCL.

For Non-Controlled files, modify:

- FILE-DEFINITION Clause
- ASSIGN statement of the JCL.

Having once specified the files:

- regenerate the application
- reload all sharable modules, even those with TPRs that do not use the new or modified files
- relaunch the TDS job.

Verify that the file has been assigned and opened after the TDS job is relaunched.

- check to see if the file was created according to specifications
- test this by accessing the file with one or more transactions.

NOTE:

Use `[M] OPEN_TDS_FILE` to open a dynamically assigned file.

For non-controlled files, files can only be assigned statically in the JCL and not declared at generation.

8.1.1 Modifying Number or Placement of Swap Files

Add or move a swap file, as applicable, if:

- response time becomes too long due to an increased number of users
- bottlenecks occur
- the I/O swaps on disks are overloaded
- there are conflicts with other files.

To add or remove swap files, modify the swap file parameters in the TDS Preparation then re-prepare the files and regenerate the application as follows:

- declare or remove these swap file parameters in TP7PREP:
  - NBSW to define the maximum number (from 1 to 32) of swap files.
  - BLKSZ to format the block size of the swap file.
  - SWnnDVC to define the device class of a new swap file.
  - SWnnMD to define the media of the swap file.
  - SWnnSZ to define the size, in cylinders, of the file.
- deallocate the swap file manually when changing the number of or moving the swap files to different volumes
- run the TP7PREP utility
- followed by the TP7GEN utility.

To verify that I/O is evenly distributed, run the SBR facility then use GCL to list the volumes.

NOTE:

All modifications require running TP7PREP and TP7GEN.

Several swap files are better for performance than one large file. However, if the swap files are on the same disk performance is not improved.

8.1.2 Modifying Size of Swap File

Modify the size of a swap file on any of these conditions:

- if the number of users of the application increases
- when modifying:
  - for TP7GEN: NUMBER OF TERMINALS Clause in TDS SECTION
  - for NETGEN: TMSESS and XCP1SESS Parameters of TDS (TDSWKS) Directive, and MAXSESS parameter of X2P (XCP2POOL) directive
- when adding XCP1 or XCP2 correspondents
- if an error message indicating that the swap file is too small, appears
- if MESSAGE-LENGTH Clause of TDS SECTION defines a size too small for message storage and thus the swap file
- if FORMATTED-MESSAGE-LENGTH Clause of TDS SECTION defines a size too small to save the FORMS control structure in the swap file.

To modify the size of the swap file, modify the swap file parameters in the TDS Preparation and in the *message* Causes in TDS SECTION. To increase the size of an existing swap file or create a new swap file on a different media:

- recalculate the value in MESSAGE-LENGTH and/or FORMATTED-MESSAGE-LENGTH
- allocate a new swap file or modify these existing swap file parameters of the TP7PREP utility:
  - BLKSZ to format the block size of the swap file.
  - SWnnDVC to define the device class of a new swap file.
  - SWnnMD to define the media of the swap file.
  - SWnnSZ to define the size, in cylinders, of the file.
- run the TP7PREP utility with DEAL=Y (if the modified swap file is allocated on the same disk)
- run the TP7GEN utility.

NOTE:
All modifications require running TP7PREP and TP7GEN.

8.2 Adding Users to a TDS Application

8.2.1 NETGEN or TDSGEN

Declare new users or correspondents to be added to the TDS application in either NETGEN or TDSGEN (TP7GEN):

1. If using NETGEN:
   - modify the TMSESS Parameter of TDS (TDSWKS) Directive
   - rerun NETGEN.
2. If using TP7GEN:
   - modify NUMBER OF TERMINALS Clause of TDS SECTION
   - rerun TP7GEN utility (a cold restart is automatically performed)
   - declare users belonging to a project in the Site Catalog.

Verify that each user added can access the transactions for which access rights have been declared.

NOTE:
Adding users may require modification to the size of the swap file.

8.2.2 Modifying User Access Rights

To change the access rights for users defined on an application:

- either give users authority to access certain transactions which were previously restricted
- or restrict previously accessible transactions.

To define access rights:

- add the authority code of the projects in question in AUTHORITY-CODES Clause of MESSAGE Statement of TRANSACTION SECTION
- rerun TP7GEN
- execute the JCL for the TDS job.

Verify whether users can or cannot access the transactions for which access rights have been redefined.

NOTE:
A project can also be modified in the Site Catalog to include the authority codes declared in TDSGEN.

8.3 Adding Transactions

Modifications to TRANSACTION SECTION may involve adding transactions to new applications. For each transaction:

- declare all the Clauses in its MESSAGE Statement
- compile and link the SMs (sharable modules) associated with each transaction
- load all SMs
- rerun TP7GEN.

Use the transaction to test that it has been accurately added.

NOTE:
`[M] MODIFY_TX` temporarily generates or modifies a transaction of an active session. This type of transaction is only saved until the next cold restart.

8.4 Adding Network Elements

Adding or removing TM, XCP1 and XCP2 correspondents modifies the communications environment of the system or the TDS application:

- specify the required TMCOR, XCP1COR and XCP2COR NETGEN Directives
- rerun NETGEN
- perform cold restart of TDS.

NOTE:
When running NETGEN, there are certain objects that cannot be modified while TDS is active.

See *Networks: Overview and Generation.*

9. Master Commands

9.1 Introduction to Master Commands

A TDS session is controlled by the master commands. These are statements that perform one of three basic functions: modify existing TDS applications, list or display information, aid the Service Center in debugging applications.

Master commands must be entered by a master operator, a user who has the appropriate authority code declared in the Site Catalog. The master operator is the only user allowed to enter commands preceded by M (TDS master transactions).

The two ways of becoming a master operator are when:

- logging on under the master mailbox defined at TDSGEN or NETGEN
- submitting the TDS job **and a master mailbox is not defined** at TDSGEN nor NETGEN.

TDSGEN is a generic term for **TDS Generation** comprising TP7PREP and TP7GEN.

When a TDS application is executed in the system startup, all users of the project with the MAIN attribute have the right to access the TDS application. The system administrator should ensure that all potential users are mapped onto a single TDS session so that they cannot start a master command at the same time.

TX messages are issued as the result of master commands. See Appendix I.

TDS messages can be filtered when TDS is launched in system startup (See System Operator's Guide).

When TDS is started (or restarted) by a user, filtering is ineffective.

9.2 Compatibility between TDS-V5 and TDS-V6/V7

A new set of master commands was introduced in TDS-V6 using GCL syntax. GCL equivalents of TDS-V5 master commands are listed in Paragraph 9.2.3 under *Correspondence of Master Commands between V5/V6-V7*.

For information on GCL, see *IOF Terminal User's Reference Manual* and *GCL Programmer's Manual*. The conventions, notations, and rules for defining the syntax of the master commands are given in the Preface of this guide. The convention used throughout this manual for denoting master commands is the prefix [M].

TDS-V7 master commands are those of TDS-V6.

9.2.1 Using Old Syntax

Master commands of previous releases can be entered, from the keyboard, from a subfile using [M] EXEC_TDS stored as data for batch programs, or as a message to be spawned. Their syntax is translated into GCL (all the GCL features are not supported). *When issuing old commands, the messages output correspond to the new commands.*

The master operator connected through IOF and submitting a TDS application, issues the old master commands as a function of SEND_TDS (SNDT).

EXAMPLES:

```
SEND_TDS   'M OPEN ifn,efn' TDS=tdsname;
```

The following warning message appears:

```
"PLEASE USE NEW MASTER COMMAND AND SYNTAX".
```

except when entering [M] MDTDS, [M] MDTX and HELP.

Logon through a mailbox allows directly issuing old master commands.
`M OPEN ...`

Modify any programs and files containing master commands of previous releases. When using the old syntax, any syntax errors produce messages that correspond to the new syntax. See Appendix F *Error messages produced by Master Commands*.

❑

9.2.2 Master Commands Obsolete in V6/V7

The following master commands are no longer supported in TDS-V6/V7:

- MEMORY
- DMM
- STAT DMM

REPEAT and FORCE of [M] MODIFY_TDS_RESTART_OPTION are not supported. COMMAND NO LONGER SUPPORTED appears as a result.

9.2.3 Correspondence Table for the Master Commands

Each TDS-V3/V5 master command has its functional equivalent in a GCL command for Release V6. A TDS-V6 master command can group several old commands. This means that some old commands are replaced by a single parameter in a new command.

Correspondence of Master Commands between V5 and V6-V7

V5 Commands	V6-V7 Command	Alias
ANEW	**ALLOW_NEW_TDS_COR**	ALNTC
AUTHORITY	**MODIFY_TX** AUTHORITY_CODE	MDTX
BROADCAST	**MODIFY_TDS_MOT**	MDTMOT
CANCEL	**CANCEL_TDS_COR** STRONG	CTC
CLOSE	**CLOSE_TDS_FILE**	CLTF
CMM	**UNLOAD_TDS_MEMORY**	UNLDTMEM
DEFPOOL	**CLOSE_COR_POOL**	CLCPOOL
DEFPOOL	**MODIFY_COR_POOL**	MDCPOOL
DEFPOOL	**OPEN_COR_POOL**	OCPOOL
DBUG	**CHECK_TX_CONVERSATION**	CKTXCONV
DSPAWN	**CANCEL_TDS_SPAWN**	CTSPW
DUMP	**MODIFY_TDS** DUMP	MDTDS
EXEC	**EXEC_TDS**	EXECT
HELP	**HELP** or **?command-name**	
INVALIDATE	**MODIFY_TX** VALIDATE=0	MDTX
LDIQS	**LOAD_TDS_IQS**	LDTIQS
LOCK	**MODIFY_TX** LOCK=1	MDTX
LSPAWN	**LIST_TDS_SPAWN**	LSTSPW
MDTDS	**MODIFY_TDS**	MDTDS
MDTX	**MODIFY_TX**	MDTX

Correspondence of Master Commands between V5 and V6-V7

V5 Commands	V6-V7 Command	Alias
MODOPT	**MODIFY_TDS_RESTART_OPTION**	MDTRSO
MODTX	**MODIFY_TX**	MDTX
NONEW	**PREVENT_NEW_TDS_COR**	PVNTC
OPEN	**OPEN_TDS_FILE**	OTF
PMM	**LOAD_TDS_MEMORY**	LDTMEM
RELEASE	**MODIFY_TX** LOCK=0	MDTX
REMOVE	**CANCEL_TDS_COR** STRONG=0	CTC
SEND	**SEND_TD_USER**	SNDTU
SIMU	**MODIFY_TDS** SIMUL	MDTDS
STAT COR	**LIST_TDS_COR**	LSTC
STAT FILE[S]	**LIST_TDS_FILE**	LSTF
STAT FROZEN	**LIST_TDS_COR**	LSTC
STAT IFNS	**LIST_TDS_FILE**	LSTF
STAT LOGGED	**LIST_TDS_COR**	LSTC
STAT SESSION	**LIST_COR_POOL**	LSCPOOL
STAT SIMU	**DISPLAY_TDS** SIMUL=1	DTDS
STAT SM	**DISPLAY_TDS** SMLIB=1	DTDS
STAT SM	**LIST_TDS_SPAWN**	LSTSPW
STAT SWAP	**DISPLAY_TDS** SWAP=1	DTDS
STAT TDS	**DISPLAY_TDS** STAT	DTDS
STAT USER[S]	**LIST_TDS_COR**	LSTC
STOP	**TERMINATE_TDS**	TTDS
SWITCH	**MODIFY_TDS** SMLIB	MDTDS
TIMELIM	**MODIFY_TDS** MAXCPU	MDTDS
TRACE	**MODIFY_TDS** TRACE	MDTDS
UNLIQS	**UNLOAD_TDS_IQS**	UMLDTIQS
VALIDATE	**MODIFY_TX** VALIDATE	MDTX

9.3 Details of Master Commands

Like all commands, master commands have abbreviations. Throughout the manual, both the abbreviation and the full name of the command are used.

9.3.1 Scrolling

When the information returned by a listing command cannot fit on the screen, answer the question "*command-name* SCROLLING UP MORE?" by entering:

- SNDT 'Y' or SNDT 'N' if logged on under IOF
- Y or N if logged on under the master mailbox
- Y is assumed if you not answer within 30 seconds.

NOTE:
The Scrolling request can be suppressed by specifying the USE NO-SCROLLING-REQUEST clause in the TDS generation.

9.3.2 Warm Restart

The following master commands retain their effects on warm restart of a TDS application:

- for commands not already started during this TDS session:
  CLOSE_COR_POOL
  CLOSE_TDS_FILE
  EXEC_TDS
- for all parameters:
  MODIFY_COR_POOL
  MODIFY_TDS
  MODIFY_TDS_XA
- for their subsequent effects:
  MODIFY_TDS_RESTART_OPTION
  MODIFY_TX
  OPEN_COR_POOL
  OPEN_TDS_FILE
  OPEN_TDS_SOCKET.

Since the effect of the following commands is immediate, they need not be reentered:

SEND_TDS_USER
UNLOAD_TDS_IQS
UNLOAD_TDS_MEMORY
LOAD_TDS_IQS
LOAD_TDS_MEMORY
MODIFY_TDS_MOT
SUPERVISE_TDS.

9.4 Entering Master Commands

Master commands can be repeatedly issued during a TDS session. However, wait for a response from TDS before entering the next command. A command entered when the user does not have the turn, will be rejected with TX60 message.

TDS processes the commands in the order entered. TDS continues to execute even if no transaction is started as when a spawned transaction is running on the user's behalf.

There are several ways of entering master commands. How these commands are entered depends on whether the master operator is connected to a TDS application via a master mailbox, or directly through IOF.

9.4.1 Entering Master Commands under IOF

Connection is direct under IOF when no master mailbox is declared at NETGEN or TDSGEN. The master operator enters commands using the GCL facilities: in menu mode, or using prompts and helps. The prefix M is not necessary when logged on via IOF.

In line mode, commands are entered after the S: prompt. In menu mode, a typical IOF screen is displayed. When the S: prompt appears, IOF is ready to accept a command. Since TDS commands are GCL directives, they can also be issued at the level of C:, F:, R: or I: prompts. For the R: and I: prompts, enter the commands as directives prefixed by $$.

EXAMPLE:

To close and deassign the file T1 of the TDS application named TEST, enter at [S:]:

```
CLOSE_TDS_FILE  T1 DEASSIGN=1  TEST;
```

Online helps display information on a particular command or parameter.

❑

9.4.1.1 Restrictions Under IOF

- Logon on under IOF is the default, in which case command responses are displayed synchronously. When SYNC=0 is specified in the GCL command MAIL, command responses are displayed asynchronously as in previous releases.
- On using SEND_TDS to start a transaction, or to reply to a transaction already started, command responses are displayed asynchronously whatever the value of SYNC. MAIL SYNC has no effect on SEND_TDS.
- To break a Master Command issued under IOF (for example, if it gives no reply), the TDS Master must do the following:
  - First, issue a $*$BRK to interrupt the IOF session and get the turn (and be able to issue another command),
  - Then, issue the Master Command SNDT with a zero-length message. As indicated in the paragraph 9.5.26 "SEND_TDS", this will make TDS simulate a break.

Therefore, breaking a TDS Master Command issued under IOF, must be done at two levels: first at IOF level, then at TDS level.

9.4.1.2 Using TDS Parameters

If the system has several TDS applications, indicate which application is to execute by specifying the name of the TDS application in the TDS parameter. However this TDS parameter cannot be used if logon is under the TDS master mailbox.

EXAMPLE:

If the name of the TDS application is PKIT, at the S: prompt enter:

```
CLOSE_TDS_FILE  ifn DEASSIGN=1 PKIT;
```

If only one application needs controlling, define the default TDS application through the system variable #WTDS (4 characters long) to avoid having to specify the TDS application each time when issuing a master command. To specify the default application, enter:

```
LET #WTDS=PKIT;
```

If administrating several TDS applications, fill in the TDS field of each command with the name of the TDS application for which the command is intended.

❑

9.4.1.3 Passing a Blank as a Parameter Value

If you are entering a Master Command in line mode, and you want to pass a ' ' (a blank), you must specify VALUES = (”””” ””””) to protect the blank. Here there are 7 single quote marks each side of the blank. These quotes are needed because of the various layers (GCL, PMOS, TDS) involved.

If you are entering a Master Command in prompt mode, and you want to pass a ' ' (a blank), you must specify VALUES = (”” ””) to protect the blank. Here there are **just** 3 single quote marks each side of the blank. In this case, fewer quotes are needed because GCL assures the protection at the other layers (PMOS, TDS).

For a complete description of GCL conventions and syntax, see the *IOF Terminal User's Reference Manual*.

9.4.2 Enter Master Commands under Master Mailbox

The master mailbox must be declared in either TDSGEN or NETGEN. When the master operator is logged on via a master mailbox, only a subset of the GCL facilities is available and menus do not appear.

The master operator enters the commands with the prefix M in line mode as for previous releases.

Only one command at a time is accepted. The TDS response COMMAND COMPLETED allows the next command to be entered. Commands stored in a subfile and started by M EXEC_TDS must also wait for the turn. When logged on via the master mailbox, prompts, help texts and menu mode cannot be used.

EXAMPLE:

To close and deassign the T1 file, enter:

```
M CLOSE_TDS_FILE     T1 DEASSIGN=1;
```

This syntax is also valid for commands stored in a subfile for execution by the M EXEC_TDS command, via the Batch Interface, or via the spawning mechanism.

❑

9.4.3 Entering Master Commands using Administrative Transactions

Another way to control a TDS application is to develop administrative transactions. These transactions give access to control-operator commands through COBOL procedures. Administrative transactions must be declared with AUTHORITY-CODES ARE 0 in the STDS and can only be used by the master operator.

Administrative transactions developed by the user, allow performing tasks similar the following master commands:

[M] ALLOW_NEW_TDS_COR
[M] CLOSE_COR_POOL
[M] DISPLAY_TDS
[M] LIST_COR_POOL
[M] LIST_TDS_COR
[M] LIST_TDS_POOL
[M] MODIFY_COR_POOL
[M] OPEN_COR_POOL.

NOTE:
[M] DISPLAY_TDS provides session information only.

To execute administrative transactions, log on to a TDS application through either a TM session or an XCP1 session. Do not log on via a XCP2 session/conversation.

The TDS programmer must supply the appropriate *messageid* for starting these administrative transactions. Special procedures are described in *TDS COBOL Programmer's Guide*.

9.5 Syntax of the Master Commands

For a brief summary of each command, see *Abbreviated Master Command Catalog* in Paragraph 9.5.11 HELP.

Commands Affecting the TDS Session:
DISPLAY_TDS
DISPLAY_TDS_XA
EXEC_TDS
MODIFY_TDS
MODIFY_TDS_RESTART_OPTION
MODIFY_TDS_XA
TERMINATE_TDS.

Commands Affecting User and Pool Correspondents:
ALLOW_NEW_TDS_COR
CANCEL_TDS_COR
CLOSE_COR_POOL
LIST_COR_POOL
LIST_TDS_COR
MODIFY_COR_POOL
OPEN_COR_POOL
PREVENT_NEW_TDS_COR.

Commands Affecting Files:
CLOSE_TDS_FILE
LIST_TDS_FILE
OPEN_TDS_FILE.

Commands Affecting Transactions:
DISPLAY_TX
MODIFY_TX.

Commands Affecting Spawn Transactions:
CANCEL_TDS_SPAWN
LIST_TDS_SPAWN.

Commands Affecting Messages:
MODIFY_TDS_MOT
SEND_TDS_USER.

Commands Affecting Debugging:
CHECK_TX_CONVERSATION
DUMP_TDS
SUPERVISE_TDS.

Commands Affecting IQS:
LOAD_TDS_IQS
UNLOAD_TDS_IQS.

Miscellaneous Commands:
HELP
LOAD_TDS_MEMORY
SEND_TDS
UNLOAD_TDS_MEMORY.

Commands Affecting TDS-TCP/IP:
CLOSE_TDS_SOCKET
DISPLAY_TDS_SOCKET
OPEN_TDS_SOCKET.

9.5.1 ALLOW_NEW_TDS_COR (ALNTC)

Purpose:

Cancels the previous [M] PREVENT_NEW_TDS_COR entered and reconnects all passive TM, XCP1, XCP2, and TCP correspondents. According to the parameter specified:

- allows new terminal correspondents and TCP correspondents to logon to a TDS application
- establishes new XCP1 and XCP2 session pools
- increases the number of allocated sessions for pools already opened.

Syntax:

```
[M] { ALNTC             }
    {                   }
    { ALLOW_NEW_TDS_COR }

    [{ TMC | TM_COR }=bool ]

    [{ X1C | XCP1_COR }=bool ]

    [{ X2C | XCP2_COR }=bool ]

    [ TDS={ #WTDS | name4 }]

    [{ TCPIPC | TCPIP_COR }=bool ]
```

Parameters:

TM_COR	Terminal correspondents.
XCP1_COR	XCP1 sessions.
XCP2_COR	XCP2 sessions.
TCPIP_COR	TCP correspondents, see *TDS-TCP/IP User's Guide.*

Usage:

Use this command after:

- [M] PREVENT_NEW_TDS_COR (PVNTC) is executed
- a TDS startup with CONNECT=NONE during the previous session in either:

  [M] TERMINATE_TDS (TTDS)
  [M] MODIFY_TDS_RESTART_OPTIONS (MDTRSO)
- a TDS startup with CN=NONE in OPTIONS for the JCL when starting the TDS job.

NOTE:
[M] ALNTC can be issued without a previous
[M] PREVENT_NEW_TDS_COR to reconnect passive terminals and XCP1 sessions.

Constraints:

Specifying no parameter is the equivalent to specifying all the correspondents regardless of their type.
Refer to the clause USE XCP1-ALNTC-PRIMARY-ONLY for more details on XCP1 sessions.

Output:

The TX54 message indicates that the command has been taken into account.

An unsuccessful attempt to connect a passive correspondent causes the following message to appear:

```
userid- UNAVAILABLE TERMINAL
```

Examples:

`[M] ALNTC TMC;`	Allows all terminal correspondents to log onto TDS.
`[M] ALNTC TDS=ELCA;`	Allows all correspondents to log onto the TDS application named ELCA.

9.5.2 CANCEL_TDS_COR (CTC)

Purpose:

Forces the specified correspondent(s) to logoff even if the user is frozen.

See [M] CLOSE_COR_POOL.

Syntax:

```
[M] { CTC            }
    {                }
    { CANCEL_TDS_COR }

    { COR | USER }=star12

    [ STRONG={ 0 | bool }]

    [ FROZEN={ 0 | bool }]

    [ TDS={ #WTDS | name4 }]
```

Parameters:

COR *alias* USER — Correspondent or user.

STRONG — Forces or allows all active transaction(s) to complete:

=1 — Effect depends on the type of correspondent:

- For TM correspondents, if a transaction is in progress, the executing TPR aborts, the ON-ABORT-TPR is executed, and the transaction terminates. The ABORT-CODE field in TDS-STORAGE is set to "OPERATOR". See *TDS COBOL Programmer's Guide*.
- For XCP2 correspondents, conversations abort but transactions continue and are notified of any disconnected conversations via an abnormal status. Equivalent to [M] CLOSE_COR_POOL [STRONG]. See *CPI-C/XCP2 User's Guide*.

– For TCP correspondents, if a transaction is in progress, the executing TPR aborts, the ON-ABORT-TPR is executed, and the transaction terminates. The remote client application is aware of disconnection thanks to the returned status it will obtained on the next API verb, which will attempt to receive data. See *TDS-TCP/IP User's Guide*.

– **Warning:** If a correspondent is executing a TPR which is looping, the CANCEL_TDS_COR command will not take effect until the TPR-TIME-LIMIT (defined at TDS generation time) is reached.

=0 *Default:* the transaction terminates, then TDS before the user is logged off.

[M] CTC *user* [STRONG=0] does not cancel the user from the TDS application if the user is frozen and if a transaction was in progress at the time the user was frozen.

FROZEN Frozen state or otherwise of the correspondent:

=1 Applicable to frozen TM correspondents.

=0 *Default:* All TM correspondents whether frozen or not.

Output:

If the specified user is not known to the TDS application, or if no user matches the star-convention, or, if no user can be canceled, the message TX56 appears:

UNKNOWN OR NON MATCHING OBJECT

Using the star convention for the correspondent name, if at least one correspondent can be cancelled, the message TX54 CTC COMMAND COMPLETED appears.

After the command is successfully executed, the TX51 service message appears on the master console for each correspondent connected when the command began to execute.

cor-name DISCONNECTED FROM *address*

For frozen correspondents, the following message appears:

username DISCONNECTED FROM TDS *tds-name*.

The TX54 CTC COMMAND COMPLETED message means that the cancel request is taken into account. The TX51 message means that the consecutive disconnection is done.

If the state of the correspondent does not immediately allow the processing of the cancel, the TX51 message will not follow the TX54 message.

This case may happen, even if the STRONG option has been requested, when the session related to the correspondent is waiting for an acknowledgment of the network. In such a case the cancel processing is delayed until the waiting situation has disappeared.

Examples:

`CTC USER=PAYROLL;`	Cancels the user named PAYROLL for the TDS application predefined in the #WTDS variable.
`CTC FROZEN TDS=CA;`	Cancels all frozen users from the TDS application named CA.

```
S: ctc *
    TX54 CTC COMMAND COMPLETED
--> TX51 STDS1 DISCONNECTED FROM TDS TEXT

S: ctc * strong
    TX54 CTC COMMAND COMPLETED

S: ctc *
    TX54 CTC COMMAND COMPLETED

S: ctc LAUR
    TX56 CTC UNKNOWN OR NO MATCHING OBJECT : LAUR

S: ctc * frozen
    TX54 CTC COMMAND COMPLETED

S: ctc laur frozen
    TX56 CTC UNKNOWN OR NO MATCHING OBJECT : LAUR

S: CTC MAST (where MAST is the name of the user who is the
    master of TDS)
    TX56 CTC UNKNOWN OR NO MATCHING OBJECT: MAST

S: CTC COR* FROZEN (assuming that a correspondent COR1, whose
    type is XCP1, exists but there is no frozen TM user whose name
    begins with COR)
    TX56 CTC UNKNOWN OR NO MATCHING OBJECT: COR*

S: CTC COR* FROZEN (assuming that a correspondent COR1, whose
    type is XCP1, and a frozen user COR2 whose type is TM exist)
    TX54 CTC COMMAND COMPLETED
```

9.5.3 CANCEL_TDS_SPAWN (CTSPW)

Purpose:

Deletes transactions spawned to a specified user but does not affect transactions already started.

Syntax:

```
[M] { CTSPW            }
    {                  }
    { CANCEL_TDS_SPAWN }

    { COR | USER }=star12

    [ TDS={ #WTDS | name4 }]
```

Parameters:

COR *alias* USER	Userid or XCP1 Correspondent for whom a session pool has been defined.

Output:

If the specified user is not known to TDS, or if no user matches the star-convention, the following message appears:

```
NO SPAWNED TRANSACTIONS ON THE SPECIFIED USER(S)
```

Otherwise the TX54 message appears:

```
CTSPW COMMAND COMPLETED
```

Examples:

`CTSPW COR=CHEQUE;`	Cancels transactions spawned to CHEQUE for the default #WTDS TDS application.
`CTSPW USER=COMP TDS=BILAN;`	Cancels transactions spawned to COMP for BILAN.
`CTSPW COR=*`	Cancels all spawned transactions.

Message TX79 indicates that no user(s) correspond to the request:

```
S: ctspw user=stephan;
   TX79 CANCEL_TDS_SPAWN: NO SPAWNED TRANSACTION ON SPECIFIED USER
S: ctspw *
   TX79 CANCEL_TDS_SPAWN: NO SPAWNED TRANSACTION ON SPECIFIED USER
```

9.5.4 CHECK_TX_CONVERSATION (CKTXCONV)

Purpose:

Detects and optionally unblocks apparent deadlocks caused by TPRs performing SEND WITH EGIs without a commitment. According to the parameters specified, displays messages at the master terminal and aborts the transactions causing the deadlock.

Resources may be tied up by TPRs during the potentially long conversation period.

Syntax:

```
[M] { CKTXCONV              }
    {                       }
    { CHECK_TX_CONVERSATION }

    EVENT={ NCMITWTM | NCMIT | NONE }

    [ ABORT={ 0 | bool }]

    [ TDS={ #WTDS | name4 }]
```

Parameters:

EVENT	Condition under which transactions issue SEND WITH EGI without commitment:
=NCMITWTM	Starts detecting such transactions doing so and without specified WAIT-TIME.
=NCMIT	Starts detecting transactions doing so regardless of WAIT-TIME.
=NONE	*Mutually exclusive with ABORT=1:* Stops detection.
ABORT	*Mutually exclusive with EVENT=NONE:* Aborts the detected transactions with return code RESVIOL.

Constraints:

EVENT=NONE and ABORT=1 are mutually exclusive.

NOTE:

On detecting such an event, the following message appears:

```
CKTXCONV TDS=tds-name, TX=txname, TPR=tprname,
USER=username, WT=wait-time.
```

9.5.5 CLOSE_COR_POOL (CLCPOOL)

Purpose:

Closes one or several session pools established between the local TDS application and a partner application. A session pool is identified by the local name of the partner correspondent and a pool name.

See [M] CANCEL_TDS_COR.

Syntax:

```
[M] { CLCPOOL        }
    {                }
    { CLOSE_COR_POOL }

    { COR | CORRESPONDENT }=name12

    [ POOL={ name8 | * }]

    [{ ATTR | ATTRIBUTE }={ name2 | * }]

    [ STRONG={ 0 | bool }]

    [{ DRNSRC | DRAIN_SOURCE }=bool ]

    [{ DRNTGT | DRAIN_TARGET }=bool ]

    [ TDS={ #WTDS | name4 }]
```

Parameters:

COR	XCP1 primary correspondent or XCP2 correspondent.
POOL	*Applicable only for XCP2 correspondent.*
ATTRIBUTE	*Applicable only for XCP1 correspondent: S*pecifies the local extended address. * closes all sessions of correspondent. *Default:* First two characters of correspondent.

STRONG How the session pool is terminated:

=0 *Normal termination:*

- For XCP2: sessions of the pool are closed immediately if no conversation uses them; otherwise they are closed after the conversations using them end.
- For XCP1: sessions are released when transaction(s) using them end; when all sessions of the pool are released, the pool is deleted.

=1 *Forced termination: Mutually exclusive with DRAIN_SOURCE and DRAIN_TARGET:*

- For XCP1: all sessions are immediately disconnected (any related transactions abort) and the pool is deleted.
- For XCP2: conversations using sessions of the pool are forcibly terminated and the pool is closed after transactions whose conversations abort, abnormal status is sent and these conversations are deallocated.

DRNSRC *or* DRAIN_SOURCE

Mutually exclusive with STRONG=1: How the Local Application handles conversation-requests in the queue.

DRNTGT *or* DRAIN_TARGET

Mutually exclusive with STRONG=1: How the Remote Application handles conversation-requests in the queue.

Applicable only to XCP2 correspondents with STRONG=0 and mandatory with POOL= for closing all pools of specified XCP2 correspondents:*

DRAIN_SOURCE must be set as follows:
- process conversation-requests
- reject conversation-requests.

DRAIN_TARGET must be set as follows:
- process conversation-requests
- reject conversation-requests.

See *CPI-C/XCP2 User's Guide*.

Usage:

- [M] CLOSE_COR_POOL does not apply to DUMMY correspondents.
- DRAIN_SOURCE and DRAIN_TARGET apply only for XCP2 correspondents.
- STRONG is mutually exclusive with the DRAIN_SOURCE and DRAIN_TARGET. If STRONG is specified, TDS forces DRAIN_SOURCE and DRAIN_TARGET to 0.

Output:

Depending on the parameters specified, the messages returned can be TX71, TX72, TX73, TX74, TX75, TX80 or TX56. See Appendix I.

See Appendix J for return codes sent by CLOSE_COR_POOL.

Examples:

```
CLCPOOL COR=APPLXCP2 POOL=POOL1;
```

Closes POOL1 for correspondent named APPLXCP2.

```
M CLCPOOL POOL=TP3TOTP4 COR=TP4;
CLOSE_COR_POOL: COMMAND COMPLETED
```

The local application TP3 closes the pool named TP3TOTP4 on the partner application named TP4.

```
S: clcpool prim;
   CLOSE_COR_POOL :  MANDATORY VALUE MISSING FOR DRAIN VALUES

S: clcpool prim attr=pr;
   TX72 CLOSE_COR_POOL COMMAND NOT PERFORMED FOR PR TDS REASON:D
```

9.5.6 CLOSE_TDS_FILE (CLTF)

Purpose:

Closes and deassigns a file opened for the current TDS session. To reopen the file(s) use [M] OPEN_TDS_FILE. A file that is already closed can be deassigned.

Syntax:

```
[ M ] { CLTF           }
      {                }
      { CLOSE_TDS_FILE }

      IFN={*|star8}

       [{ DASG | DEASSIGN }={ 0 | bool }]

      [ TDS={ #WTDS | name4 }]
```

Parameters:

IFN	File(s) known by TDS (except H_CTLM file). If star convention is used, H_FORM is not taken into account.
DEASSIGN	Deassigns (DEASSIGN=1) the file.

Constraints:

- If the file is TDS-controlled, TDS waits for all commitment units to terminate before closing the file.
- CALL "KEEP-CURRENCIES" executed in the current commitment unit does not prevent the file from being closed at the end of the current commitment.
- If the file is closed after the end of the current commitment unit, the next commitment unit will not be allowed to start and the transaction will be aborted with the IFNERR return code. Please refer to Appendix B of the *TDS COBOL Programmer's Guide.*

Output:

The result of the command is reported as follows:

```
TX63 CLTF: ifn CLOSED
TX64 CLTF: ifn NOT CLOSED, RC=return-code
```

See Appendix J for return codes.

Examples:

```
CLTF PKIT23;
```
Closes the file named PKIT23.

```
CLTF PKIT23 DEASSIGN;
```
Closes and deassigns file PKIT23 at the same time.

```
S: cltf IFN=T1 DEASSIGN=0
TX63 CLTF: T1 CLOSED
```
File named T1 is closed but is not deassigned.

```
S: cltf T1
TX55 CLTF COMMAND NOT PERFORMED RC=0CB20012->TP7   50,ALREADY
```
File named T1 is already closed.

```
S: cltf t2 deassign
TX63 CLTF : T2 CLOSED
```
Closes and deassigns file named T2.

```
S: CLTF *CINE
TX63 TDS:JDB,CLOSE_TDS_FILE:AREACINE CLOSED
```
The matching file AREACINE is closed.

```
S: CLTF INDEX*
TX63 TDS:JDB, CLOSE_TDS_FILE:INDEX1 CLOSED
TX64 TDS:JDB, CLOSE_TDS_FILE:INDEX2 NOT CLOSED
                    RC=0C8F0012->TP7 15, ALREADY
TX64 TDS:JDB, CLOSE_TDS_FILE:INDEX4 NOT CLOSED
RC=0C8F0012->TP7 15, ALREADY
TX63 TDS:JDB, CLOSE_TDS_FILE:INDEX3 CLOSED
```
The matching files INDEX1 and INDEX4 are closed. The matching files INDEX2 and INDEX4 were already closed.

If no files match with the star convention request, the message TX56 is issued.

9.5.7 CLOSE_TDS_SOCKET (CLTS)

Purpose:

Closes the TDS listening socket. See *TDS-TCP/IP User's Guide*.

9.5.8 DISPLAY_TDS (DTDS)

Purpose:

Displays information about the current TDS session according to parameter specified. Additional information is displayed when SIMUL and/or STATUS are required.

Syntax:

```
[ M ] { DTDS        }
      {             }
      { DISPLAY_TDS }
      [{ STATUS | STAT }=bool ]
      [ SIMUL={ 0 | bool }]
      [ SMLIB={ 0 | bool }]
      [ SWAP={ 0 | bool }]
      [ TDS={ #WTDS | name4 }]
      [{ RPC_STAT | RPC_STATUS }=bool ]
```

Parameters:

STATUS	Displays general TDS statistics.
SIMUL	Displays information about simultaneity levels.
SMLIB	Displays the current sharable module libraries assigned to TDS in the order in which they are searched. The names of the sharable module libraries can be obtained from the JCL for starting the TDS application. The search mode of the sharable modules depends on the order of the SM libraries: – either in the JOBLIB statement – or in the previously executed MDTDS SMLIB command.
SWAP	Displays statistics on swap file allocation.
RPC_STATUS	Applicable only if RPC SIMULTANEITY Clause is specified in TDS SECTION: Displays statistics on the use of the RPC service. Otherwise the TX55 message with INVUSE is displayed and the command is not executed.

Constraints:

- When no parameters are specified, general TDS statistics are displayed (STAT=1).
- STATUS, SIMUL, SMLIB and SWAP are mutually exclusive. When specifying several parameters at the same time, only the first parameter according to the syntax is taken into account. For example, if DTDS SWAP=1 SIMUL=1 is issued, only SIMUL is taken.

Examples:

`DTDS;` Displays the general TDS statistics.

`DTDS SMLIB=1 TDS=PKIT;`
Displays the current sharable modules libraries assigned to the TDS application named PKIT.

Output:

1) *Display the General Statistics of an application named PKIT:*

```
S: dtds sTATUS
            ---------------------------------------------
            ------  TDS=PKIT  14:27:42  FEB 14, 1994  ----
            ------       GENERAL TDS STATISTICS       ----
            ---------------------------------------------
          INIT.SIMU.COUNT   =  5     CUR.SIMU.COUNT    = 4
          ACC.SESS.ALLOC    =  0     ACC.SESS.REJECT   = 0
          USED TX COUNT     =  16    TX ABORT. COUNT   = 0
          USED TPR COUNT    =  27    TPR ABORT COUNT   = 0
          COMMIT COUNT      =  16    DIALOG COUNT      = 10
          TPR ELAPSED TIME  =  3     TPR CPU TIME      = 0
          DEADLOCK COUNT    =  0     NON CONCUR WAIT   = 0
          TABOV ABT COUNT   =  0     WDNAV ABT COUNT   = 0
          LGWAITABT COUNT   =  0     DIRTY READ ABORT  = 0
          BUFOVABT COUNT    =  0     SERIALIZATION     = 0
          MAX TM SES        =  10    CUR TM SES        = 0
          MAX XCP1 SES      =  5     CUR XCP1 SES      = 0
          MAX XCP2 SES      =  50    CUR XCP2 SES      = 0
          MAX VIRT SES      =  3     CUR VIRT SES      = 1
          PMOS COR COUNT    =  1     MAX IDLE TIME     = 2000
          POOL USED (KB)    =  160   POOL SIZE (KB)    = 500
          PSEUDO BUFFERS    =  0
          WAITING TPR MEAN  =  0     MAX CPU TIME      = 90000
          CUR TCP SES       =  2
TDS: PKIT, DTDS COMMAND COMPLETED
```

2) *Display search rules of a sharable module library in the application named PKIT:*

```
S: dtds smlib
              ------------------------------------------------
              ------  TDS=PKIT   14:27:23  MAY 2, 1994  ------
              ------    CURRENT SEARCH RULES OF SMLIB   ------
              ------------------------------------------------
          SM LIBRARY 1 : TEXT.SMLIB
          SM LIBRARY 2 : TEXT.SMLIB2
          SM LIBRARY 3 : TEXT.SMLIB3
DTDS COMMAND COMPLETED
```

3a) *Display the swap files of an application named PKIT:*

```
S: DTDS SWAP
              ------------------------------------------------
              ------  TDS=PKIT  14:27:33  JUN 10, 1994  ------
              ------         LIST OF SWAP FILES          ------
              ------------------------------------------------
    SWAP_FILE  ACT  LOGGED_CNTXT  OCCUPANCY  ALLOC_FACTOR
       H_SWAP   Y            8    11.000%      100.%
DTDS COMMAND COMPLETED
```

LOGGED_CNTXT *logged context* is the total number of contexts:

- allocated in the swap file to logged and frozen users
- *and* logically deallocated but still allocated in the swap file for IO gain reasons

ACT *active* means that the swap file(s) are assigned and opened

OCCUPANCY means the % of logged contexts to the maximum number of contexts in this swap file

ALLOC_FACTOR *allocation factor* is the % of the size of this swap file to the total size of the set of swap files.

3b) *This example shows a TDS application (named CASH) with 2 swap files.*

```
S: DTDS SWAP=1
              ------------------------------------------------
              ------  TDS=CASH  14:27:33  MAR 17, 1994  ------
              ------         LIST OF SWAP FILES          ------
              ------------------------------------------------
    SWAP_FILE  ACT  LOGGED_CNTXT  OCCUPANCY  ALLOC_FACTOR
       H_SWAP   Y            3    5.550%      43.000.%
       H_SWAP1  Y            5    6.940%      57.000.%
DTDS COMMAND COMPLETED
S:
```

4a) *When DTDS SIMUL is issued and RPC SIMULTANEITY Clause is specified in STDS of application PKIT:*

```
S: dtds simul
             ------------------------------------------------
             ------  TDS=PKIT  14:27:42  APR 14, 1994  ------
             ------    CURRENT SIMULTANEITY LEVELS     ------
             ------------------------------------------------
         MAX.SIMU.COUNT     = 4        CUR.SIMU.COUNT     = 4
         FROZEN SIMU COUNT  = 0
         MAX RPC SIMU COUNT = 0        CUR RPC SIMU COUNT = 0
TDS:PKIT, DTDS COMMAND COMPLETED
```

MAX RPC SIMU COUNT is the maximum RPC simultaneity level as fixed at TP7GEN or through MDTDS.

CUR RPC SIMU COUNT is the number of current RPC simultaneities.

4b) *When DTDS SIMUL is issued and RPC SIMULTANEITY Clause is not specified in STDS of application PKIT:*

```
S: dtds simul
             ------------------------------------------------
             ------  TDS=PKIT  14:28:10  APR 14, 1994  ------
             ------    CURRENT SIMULTANEITY LEVELS     ------
             ------------------------------------------------
         MAX.SIMU.COUNT     = 4        CUR.SIMU.COUNT     = 4
         FROZEN SIMU COUNT  = 0
TDS:PKIT, DTDS COMMAND COMPLETED
```

5a) *When DTDS RPC_STATUS is issued and RPC SIMULTANEITY Clause is specified in STDS of application PKIT:*

```
S: dtds RPC_STATUS
             ------------------------------------------------
             ------  TDS=PKIT  14:28:50  FEB 14, 1994  ------
             ------          TDS RPC STATISTICS          ------
             ------------------------------------------------
         USED RPC TX COUNT   = 0      USED RPC TPR COUNT  = 0
         MAX RPC TX COUNT    = 0      CUR RPC TX COUNT    = 0
         WAITING TPR MEAN    = 0      RPC TPR ELAPSE TIME = 0
         WAITING RPC TPR MEAN= 0
TDS: PKIT, DTDS COMMAND COMPLETED
```

USED RPC TX COUNT is the total number of RPC transactions since the start of the TDS application.

USED RPC TPR COUNT is the total number of TPRs launched by RPC transactions since the start of the TDS application.

MAX RPC TX COUNT is the maximum number of RPC transactions active at a given time.

CUR RPC TX COUNT is the current number of active RPC transactions at command time (that is, when the DTDS command was executed).

RPC TPR ELAPSE TIME is the total elapsed time of TPRs launched by RPC transactions since the start of the TDS application.

WAITING TPR MEAN is the average number of TPRs waiting for a process at a given time.

WAITING RPC TPR MEAN is the average number of TPRs launched by RPC transactions waiting for a process at a given time.

5b) *When DTDS RPC_STATUS is issued and RPC SIMULTANEITY Clause is not specified in STDS, the command is not performed and the TX55 message with INVUSE return code, is displayed.*

9.5.9 DISPLAY_TDS_SOCKET (DTDSS)

Purpose:

Display the socket parameter of a TDS session. See *TDS-TCP/IP User's Guide*.

9.5.10 DISPLAY_TDS_XA (DTDSXA)

Purpose:

Displays the XA parameters of a TDS session and also the global state of each Resource Manager accessed through XA protocol at that moment.

Syntax:

```
[ M ] { DTDSXA          }
      {                 }
      { DISPLAY_TDS_XA  }

      [{ RM_STATUS | RM_STAT }= { 0 | bool } ]

      [ TDS={ #WTDS | name4 }]
```

Parameters:

RM_STATUS When RM_STATUS = 1, displays the name of the XA Resource Managers (ORACLE databases for instance) accessed by the TDS application at the moment the command is entered together with information about these RMs.

Examples:

`DTDSXA TDS=PL;` Displays the XA parameters of the TDS session PL.

Output:

```
-----------------------------------------------
------  TDS=PL   17:17:52  AUG 08, 1994   ------
------       CURRENT XA PARAMETERS        ------
-----------------------------------------------
  XA_RESYNC_DELAY   =  600
  TDS : PL, DTDSXA COMMAND COMMAND COMPLETED
```

`DISPLAY_TDS_XA RM_STATUS = 1 TDS=PL;`
Displays the XA parameters of the TDS session PL with information about the RMs accessed.

Output:

```
-----------------------------------------------
------  TDS=PL   17:17:52  AUG 08, 1994   ------
------        CURRENT XA PARAMETERS        ------
-----------------------------------------------
  XA_RESYNC_DELAY   =  600

  RM_IDENT          CNCT   SES  RESYNC  START
  D:$ARE2:ORA71       4     0      Y      OK
  T:DPX321:ORA        0     0      N      OK
  OTHER               3     1      Y      OK
  D:CHICAGO           1     1      N      KO
  TDS : PL, DTDSXA COMMAND COMMAND COMPLETED
```

RM_IDENT Physical identifier of the ORACLE database, displayed to a maximum of 44 characters.

CNCT Number of physical ORACLE connections to this RM.

SES Number of active XA sessions for this RM, that is active XA commitment units that have connected to this ORACLE database.

RESYNC Indicates if, at that moment, at least one commitment concerning this RM is being resynchronized. Possible values are Y (yes) or N (no).

START Indicates if the last attempt to start an XA session on this RM (that is, when a CU tried to connect to the ORACLE database) was successful or not. Possible values are OK, or KO if any problem occurred.

If the command fails, the following message is issued:

`TX55 TDS: PL, DTDSXA COMMAND NOT PERFORMED` *`rc`*

9.5.11 DISPLAY_TX (DTX)

Purpose:

Displays the characteristics of a transaction either:

- specified in the MESSAGE statement of the TRANSACTION SECTION
- created through [M] MODIFY_TX.

Syntax:

```
[ M ]{ DTX        }
     {            }
     { DISPLAY_TX }

     TX={ star8 | * }

     [{ FILE_SECURITY | FSEC }={ 0 | bool }]

     [ FSCAC={ 0 | bool }]

     [{ FILE_SHARED_READ | FSHRD }={ 0 | bool }]

     [ TXNCONC={ 0 | bool }]

     [ AUTHCODE={ 0 | bool }]

     [ TDS={ #WTDS | name4 }]
```

Parameters:

TX	Up to 8 alphanumeric characters identifying the transaction. When specifying the star convention, only the list of matching transaction names is given.
FILE_SECURITY	Displays the type of file security option: – suppress before journal – suppress deferred updates – use deferred updates except for.
FSCAC	Displays the files which have been specified in SUPPRESS CONCURRENT ACCESS Clause.
FILE_SHARED_READ	Displays the files specified in the SHARED READ clause.

TXNCONC	Displays the transactions specified in [MANUALLY] NON CONCURRENT Clause.
AUTHCODE	Displays the list of authority codes specified for the transaction.

Constraints:

- The following parameters are mutually exclusive:
  - FILE_SECURITY (FSEC) = 1
  - FSCAC
  - FILE_SHARED_READ (FSHRD)
  - TXNCONC
  - AUTHCODE
- When specifying several parameters at the same time, only the first parameter is taken into account. Respect the order of the parameters as listed above. For example, if DTX FSHRD=1 FSEC=1 is issued, FSEC is taken into account.

Examples:

`DTX *;`	Lists all known transactions.
`DTX TRANS1;`	Displays the global characteristics of transaction TRANS1.
`DTX TX134 AUTHCODE=1;`	Displays the list of authority codes for transaction TX134.
`DTX TX134 FSAC;`	Displays the files specified in SUPPRESS CONCURRENT ACCESS Clause and accessed by transaction TX134.

Output:

Depending on the parameters specified, the messages returned can be TX08, TX11, TX12, TX13, TX14, TX25, TX44, TX52, and TX57. See Appendix I.

See Appendix J for return codes.

When the information returned by the command cannot fit on the screen, answer the question "DTX SCROLLING UP?" by entering:

- SNDT 'Y' or SNDT 'N' if logged on under IOF
- Y or N if logged on under the master mailbox.

1a) *Display Non-concurrency for transaction where NON-CONCURRENT Clause is specified:*

```
S: DTX DEADLCK1 TXNCONC

   DISPLAY _TX  : DEADLCK1
   TX NON-CONCURRENT WITH: PWFORM
   DTX COMMAND COMPLETED
```

1b) *Display Non-concurrency for transaction where NON-CONCURRENT Clause is not specified:*

```
S: JPL TXNCONC

   DISPLAY _TX  : JPL
   NO CLAUSE SPECIFIED
   DTX COMMAND COMPLETED
```

2) *Display the authority codes for the transaction:*

```
S: DTX JPL AUTHCODE
   DISPLAY _TX  : JPL
   LIST OF AUTHORITY CODES
   0  1  2  3  29  30  31
   DTX COMMAND COMPLETED
```

3) *Display all transactions:*

```
S: DTX *
          ------------------------------------------------
          ------  TDS=PKIT  14:28:17  FEB 28, 1994  ------
          ------        LIST  OF TRANSACTIONS        ------
          ------------------------------------------------
   TX7        ANASEND    BREAK     BUDG      TX5       C
   CLASS1     CLOSE      COMPIL    DISCNCT   DOD       DSPAWN
   ECHO       FORJM      FORJ2     FVA       TX9       TX13
   INQUIRY    JMS        JOB       JPL       LOGON     LOGOUT
   TX12       TX15       MANUALNC  MDPF      MDP1
   OPEN2      ORA        PRIVATE              RESTART   RESTFORM
   SECON      SERIALNC   SHUTDOWN  SIMBRK    SORT      SPAWN
   STARTUP    TAMD       TAMQ      TARF      TASI      TCOM
   TOPEN      TOTO       TX3       TSND      TX        TXABT
   DTX COMMAND COMPLETED
```

4) *Display list of all files specified in SUPPRESS CONCURRENT ACCESS CONTROL FOR for transaction:*

```
S: DTX ANCONC1 FSCAC
   DISPLAY _TX  : ANCONC1
   SUPPRESS CONCURRENT ACCESS CONTROL FOR:
   FILEA      FILEB          H_FORM
   DTX COMMAND COMPLETED
```

5a) *Display RPC attributes for the transaction if RPC SIMULTANEITY Clause is in STDS of transaction:*

```
S: DTX TEXC
     ------------------------------------------------
     ------- TDS = PKIT    15:42:44  FEB 28, 1994 ------
     -------  CHARACTERISTICS OF THE TRANSACTION  ------
     ------------------------------------------------
   MESSAGE                  : TEXC
   FIRST ASSIGNED TPR       : TEXCEP
   CLASS                    : Z
   IMPLICIT COMMITMENT      : Y
   AUTOMATIC UNMAPPING      : Y
   RPC SERVICE USED         : Y
   XA SERVICE USED          : N
   XCP2 SERVICE USED        : N
   PRIORITY                 : 0
   TRANSACTION STORAGE SIZE : 1024
   ACCOUNTING               : NO
   FORM                     :
   LOCKED TRANSACTION       : N
   FOR DEBUG TX             : N
   FOR INQUIRY TX           : N
   HIDDEN (IN MENU) TX      : N
   TDS : PKIT, DTX COMMAND COMPLETED
```

5b) *If RPC SIMULTANEITY Clause is not specified in STDS:* RPC SERVICE USED does not appear in output.

6) display XA attributes when XA is defined at TDS level.

```
S: DTX TX = MENU TDS = PL
     ------------------------------------------------
     ------- TDS = PL      15:42:44  FEB 28, 1994 ------
     -------  CHARACTERISTICS OF THE TRANSACTION  ------
     ------------------------------------------------
   MESSAGE                   : MENU
   FIRST ASSIGNED TPR        : MENUAB-1
   CLASS                     : Z
   IMPLICIT COMMITMENT       : Y
   AUTOMATIC UNMAPPING       : Y
   RPC SERVICE USED          : Y
   XA SERVICE USED           : Y
   XCP2 SERVICE USED         : N
   PRIORITY                  : 0
   TRANSACTION STORAGE SIZE  : 16
   ACCOUNTING                : O
   FORM                      :
   LOCKED TRANSACTION        : N
   FOR DEBUG TX              : N
   FOR INQUIRY TX            : N
   HIDDEN (IN MENU) TX       : N
   TDS : PL, DTX COMMAND COMPLETED
```

7) display CMA attribute when CMA is defined at TDS level.

```
S: DTX TX = MENU TDS = PL
     ------------------------------------------------
     ------- TDS = PL      15:42:44  FEB 28, 1994 ------
     -------  CHARACTERISTICS OF THE TRANSACTION  ------
     ------------------------------------------------
   MESSAGE                  : MENU
   FIRST ASSIGNED TPR       : MENUAB-1
   CLASS                    : Z
   IMPLICIT COMMITMENT      : Y
   AUTOMATIC UNMAPPING      : Y
   RPC SERVICE USED         : N
   XA SERVICE USED          : N
   CMA SERVICE USED         : Y
   XCP2 SERVICE USED        : N
   PRIORITY                 : 0
   TRANSACTION STORAGE SIZE : 16
   ACCOUNTING               : O
   FORM                     :
   LOCKED TRANSACTION       : N
   FOR DEBUG TX             : N
   FOR INQUIRY TX           : N
   HIDDEN (IN MENU) TX      : N
   TDS : PL, DTX COMMAND COMPLETED
```

9.5.12 DUMP_TDS (DPT)

Purpose:

Used at the request of Service Center: Dynamically dumps while the TDS application is running, memory areas concerning a specified user or the contents of TDS in a user-defined library member.

Syntax:

```
[ M ] { DPT      }
      {          }
      { DUMP_TDS }

      [{ USER | COR }=star12 ]

      [ TABLE={ ALL | CORRESPONDENT | FSTAT |
                MAILBOX | TRANSACTION | TX-PHASE }]

      [ PRTFILE=file78 ]

      [ TDS={ #WTDS | name4 }]
```

Parameters:

COR *alias* USER	*Mutually exclusive with TABLE:* Correspondent or user.
TABLE	*Mutually exclusive with COR:* TDS tables: – ALL - *all* tables of TDS application – CORRESPONDENT - correspondent table – FSTAT - root table of TDS – MAILBOX - mailbox table – TRANSACTION - transaction table – TX-PHASE - transaction-phase table.
PRTFILE	Library member file to contain the dump. If omitted, the previous value applies.

Constraints:

COR and TABLE are mutually exclusive.

Output:

Dumped data is in EBCDIC. On the left, the address path (STN.STE.SRA) is given.

If the command is not successfully completed, the message TX55 appears.

Examples:

`M DPT USER=BAIL;` Takes a dump of all data belonging to the user named BAIL.

`M DPT TABLE=CORRESPONDENT PRTFILE=PKIT.DEBUG..DUMPCOR;` Writes a dump of the correspondent table to member DUMPCOR of library PKIT.DEBUG.

9.5.13 EXEC_TDS (EXECT)

Purpose:

Requests the execution of master commands, or transactions stored in a source library (SL) member. The previous syntax of this command is supported.

Syntax:

```
[M] { EXECT    }
    {          }
    { EXEC_TDS }

    { MB | MEMBER }=name31

    LIB=lib78

    [{ VL | VALUES }=( values )]

    [ ECHO={ 1 | bool }]

   [ TDS={ #WTDS | name4 }]
```

Parameters:

MEMBER	SL library member whose record cannot exceed 130 characters and is considered an initial transaction message.
LIB	SL library containing the member to be executed.
VALUES	List of up to 16 values that may be passed to the transactions. Values entered in line mode are separated by spaces or commas. Enclose special characters in quotes. Values can be indicated by &*n* where: – & denotes a value to be substituted – *n* denotes the position that each value occupies. Variable information supplied at runtime replaces the appropriate &*n* when the transaction executes.
ECHO	Displays (ECHO=1) the list of transactions to be executed on the master terminal.

Constraints:

- If a transaction starts a dialog, the master terminal operator is expected to answer. TDS sets the master terminal to the passive state where the master terminal cannot start a transaction until all the spawned transactions are processed.
- [M] EXEC_TDS cannot be requested through [M] EXEC_TDS.
- [M] EXEC_TDS aborts if the library containing the member to be executed cannot be accessed. DF02 console message and a return code indicating that the library is unavailable is sent to the main operator. See *Console Messages Manual*.
- If an error occurs when [M] EXEC_TDS is executed such as where substitution of a value fails, transactions immediately following are not taken into account. Preceding transactions are started and the following message appears:
  `COMMAND PARTIALLY COMPLETED, RC=`*`return-code`*
- If one of the transactions aborts, the other transactions stored in the same subfile are not affected and are processed normally.
- The length of each transaction with its validated parameters must not exceed 130 characters. If a transaction exceeding 130 characters is encountered when [M] EXEC_TDS is executed, this transaction and any transactions immediately following are not executed.
- For TDS warm restart, *the spawning priority must not be specified:*
  – if TERMINATE_TDS submitted through EXEC_TDS ended the previous session
  – and if the STARTUP transaction spawns another EXEC_TDS transaction.

Output:

The TX55 message appears when the command cannot be executed.

Examples:

```
M EXEC_TDS MB=STORETX LIB=TEST.SLLIB VL=(T1,WT1,param,0);
```

Contents of the STORETX member:

```
record1:   M OPEN_TDS_FILE &1
record2:   &2  &3
record3:   M CLTF  &1 &4
```

The following transactions are executed:

`M OTF T1`	opens T1
`WT1 param`	starts user transaction: WT1
`M CLTF T1,0`	closes T1, but the file is not deassigned (DEASSIGN=0).

```
S:EXECT EXC2 TEXT.SLLIB VL=(T1)
  TX55 EXECT COMMAND NOT PERFORMED RC=4FDB1008->DQULK 27,SFNUNKN
```

This command is not performed because EXC2 is not a subfile of text.sllib.

```
S:EXECT EXEC2 TEXT;
  TX55 EXECT COMMAND NOT PERFORMED RC=4BC61007->DFPRE  6,EFNUNKN
S:
```

This command is not performed because TEXT is not a library.

Suppose that EXEC2 contains the following commands:

```
M OTF &1
M LSTF &1 DTLD
M CLTF &1
M CLTF &1 1
```

when executing EXEC2, the following output results:

```
S: EXECT EXEC2 TEXT.SLLIB VL=(T1)
         EXECT :  M OTF T1
         EXECT :  M LSTF T1 DTLD
         EXECT :  M CLTF T1
         EXECT :  M CLTF T1 1
S:
--> TX64 OTF : T1 NOT OPENED   RC=0C860909->TP7    6,IFNNASG
-->             ------------------------------------------------
                ------  TDS=PKIT   14:30:48  MAY 4, 1994  ------
                ------            LIST OF FILES           ------
                ------------------------------------------------
-->         IFN    OPEN  MONITORED    PMD
-->          T1      N       Y      UPDATE
-->
-->
--> TX54 LSTF COMMAND COMPLETED
--> TX55 CLTF COMMAND NOT PERFORMED RC=0CB20012->TP7  50,ALREADY
--> TX63 CLTF : T1 CLOSED
-->      READY
S:
```

9.5.14 HELP

Purpose:

Displays the specified command syntax or the list of all master commands. HELP is available only when connected through a master mailbox.

When operating under IOF, enter ? to obtain help. Enter ?*command-name* for help on a master command. For help on an individual parameter, enter *command-name*? to bring up a menu; then enter ? for the required parameter.

Syntax:

```
HELP [ COMMAND=name31 ]
```

Parameters:

COMMAND Command from previous releases are allowed.
Default: Lists all master commands.

Examples:

- To display the help text of [M] PREVENT_NEW_TDS_COR:

```
HELP COMMAND=PREVENT_NEW_TDS_COR;
```

- To show the syntax of [M] TERMINATE_TDS:

```
S: SNDT 'HELP TTDS'
     M { TERMINATE_TDS | TTDS }
       [ STRONG={ bool | 0 }]
       [ MODE={ COLD | WARM }]
       [{ CONNECT | CN }={ ALL | PASSIVE | ACTIVE | NONE }]
       [{ IDLE_TOO | IDLE }={ bool | 0 }]
       [{ EXEC_DISCONNECT_TX | EXDISTX }={ bool | 0 }]
       HELP COMMAND COMPLETED
S:
```

- The HELP of an old command provides the name of the new command:

```
S: SNDT 'HELP STOP'
    STOP [ STRONG [,DISREST]] [,CNT:<LIST_OF_CNT_OPTIONS> ]
                                          [,{WARM|COLD}  ]
<LIST OF CNT_OPTIONS> = {ALL|NONE|ACTIVE [,PASSIVE]
                  [,NOTIDLE]|PASSIVE [,ACTIVE] [,NOTIDLE ]}
 IT IS RECOMMENDED THAT YOU USE THE NEW MASTER COMMAND:
          M { TERMINATE_TDS | TTDS }
 HELP COMMAND COMPLETED
S:
```

- For the list of commands and their abbreviated functions:

```
S: SNDT 'HELP'
           ======================================
           =  ABBREVIATED MASTER COMMAND CATALOG  =
           ======================================
   see overleaf for list of available commands

 HELP COMMAND COMPLETED
```

ALNTC	Reopens all sessions to allow new correspondents
CTC	Forces the specified correspondent to log off
CTSPW	Cancels and deletes spawned transaction(s)
CLTF	Closes a file currently opened for the TDS session
CLTS	Closes the TDS listening socket
CLCPOOL	Closes session pool(s) for a correspondent
DTDS	Displays information about the TDS session
DTDSS	Displays information about TDS socket
DTX	Displays characteristics about a specified transaction
EXECT	Requests execution of message-id's stored in SLLIB
HELP	Displays catalog of commands and their syntax
LSCPOOL	Displays session information about specified pool
LSTC	Lists (one or several) correspondents
LSTF	Lists (one or several) files
LSTSPW	Lists information about spawned transactions

LDTIQS	Loads specified IQS information
LDTMEM	Loads reference data into non-swappable memory
MDCPOOL	Modifies session pool characteristics
MDTDS	Modifies global characteristics of class of transactions
MDTMOT	Sends a message to all logged correspondents
MDTRSO	Modifies TDS restart options
MDTX	Modifies characteristics of a (class of) transaction(s)
OCPOOL	Opens session pool(s) between TDS and a correspondent
OTF	Opens dynamically a previously closed file
OTS	Opens the TDS listening socket
PVNTC	Prevents new correspondents from logging on to TDS
SNDTU	Sends a text message to a specified correspondent
TTDS	Prevents new TDS activity before shutdown
UNLDTIQS	Unloads specified IQS information
UNLDTMEM	Unloads data from non-swappable memory
DPT	Dumps TDS memory areas (RESERVED FOR SERVICE CENTER)
SPRVT	Activates the supervision of a TDS session
CKTXCONV	Checks conversations without commitments
DTGW	Displays HOST GATEWAY information
DTDSXA	Displays XA parameters and RM information of a TDS session
MDTDSXA	Modifies XA parameters of a TDS session

9.5.15 LIST_COR_POOL (LSCPOOL)

Purpose:

Displays static or dynamic characteristics of specified pool or the list of pools running for a XCP2 correspondent. Also lists XCP1 primary correspondents.

Syntax:

```
[M] { LSCPOOL       }
    {               }
    { LIST_COR_POOL }

    { COR | CORRESPONDENT }=name12

    [ POOL={ name8 | * }]

    [{ ATTR | ATTRIBUTE }={ name2 | * }]

    [{ NG | NETGEN }={ 0 | bool }]

    [{ PRTMB | PRINT_MEMBER }=name31 ]

    [ TDS={ #WTDS | name4 }]
```

Parameters:

CORRESPONDENT XCP1 primary or XCP2 correspondent: *DUMMY correspondent cannot be specified*.

POOL *Applicable to XCP2 correspondents and mutually exclusive with ATTRIBUTE:* * lists all pools.

ATTRIBUTE *Applicable only for a XCP1 correspondent and is not used for XCP2 or dummy correspondents. Mutually exclusive with POOL:* Local extended address. * lists all defined attributes for the specified correspondent.

When an attribute is given the following information about all the sessions of the pool having this attribute are returned:

1) Information about all the sessions:

   ACTSESNB means ACTIVE SESSION NUMBER (i.e., LOGGED).

   FRZSESNB is the FROZEN session number.

   TRSTSESNB means the number of sessions in TRANSIENT state (i.e., LOGON, DISC, UNLOGGED).

2) Information for each session:

   a) state of the session (ex: LOGON, LOGGED, DISC, FROZEN, UNLOGGED).

   b) state of the session (ex: LOGON, LOGGED, DISC, FROZEN, UNLOGGED).

      - LOGON means in LOGON phase.
      - LOGGED is the normal case after the logon phase.
      - DISC means session is being disconnected.
      - FROZEN means session has been abnormally disconnected.
      - UNLOGGED means session has been abnormally disconnected.

   c) State in terms of resource of the session (ex: PERMANENT, FREE, ALLOCPRINC, ALLOCAUX, FREE/NALC).

- PERMANENT is the state for a terminal session and should not appear.
- FREE means session is ready to become ALLOCPRINC or ALLOCAUX.
- ALLOCPRINC means session is allocated to a transaction as the PRINCIPAL SESSION. (spawning case).
- ALLOCAUX means session is allocated to a transaction as the AUXILLARY SESSION (CP-ALLOCATE). In this case, CORRESP column indicates the name of the user who has launched the transaction.
- FREE/NALC means session is not allocated to a transaction, but not yet allocatable (waiting for a protocol event).

NETGEN *Applicable to XCP2 correspondents:* State of pools:

=1 Characteristics declared at Network Generation.

=0 Current characteristics.

PRINT_MEMBER Member in *tds-name*.DEBUG file used to store output to be printed. The file is always opened in output and is erased on cold restart of TDS.
Default: Results are displayed on the screen.

Usage:

- NETGEN and POOL relate to XCP2 correspondents.
- ATTRIBUTE and POOL are mutually exclusive.

Output:

Depending on the parameters specified, the messages returned can be TX25, TX30, TX31, TX32, TX35, TX36, TX56 or TX57:

- For NETGEN=0, both TX35 and the TX32 messages are displayed.
- For NETGEN=1, only TX32 message is displayed for the characteristics of pools known at network generation.
  See Appendix I.

Examples:

Lists characteristics of POOL3 for correspondent FUNDS:

```
LSCPOOL COR=FUNDS POOL=POOL3 TDS=BUL1;
```

Lists all pools for correspondent PAYROLL:

```
LSCPOOL COR=PAYROLL POOL=*;
```

Lists current state of pool TP3TOTP4:

```
M LSCPOOL POOL=TP3TOTP4 COR=TP4 NG=0
----------------------------------------
-----    18:05:06 JULY 1, 1994       -----
-----  CHARACTERISTICS OF THE POOL   -----
----------------------------------------
POOL CURSESNB CURWINSC CURWINTG TRSTSSNB DRAINSC DRAINTG MAXSYNC
TP3TOTP4 2        2        0        0       N       N       C

POOL MAXSESNB MINWINSC MINWINTG AUTOACT  DRAINSC DRAINTG MAXSYNC
TP3TOTP4 4        2        2        2       N       N       C

LSCPOOL: COMMAND COMPLETED
```

Lists state of pool TP3TOTP4 as declared at NETGEN:

```
M LSCPOOL TDS=TP4 POOL=TP3TOTP4 NG=1;
----------------------------------------
-----    18:06:20 JULY 1 1994       -----
-----  CHARACTERISTICS OF THE POOL   -----
----------------------------------------
POOL  MAXSESNB MINWINSC MINWINTG AUTOACT DRAINSC DRAINTG MAXSYNC
TP3TOTP4  4       2        2        2       N       N       C

LSCPOOL  : COMMAND COMPLETED
```

The pool is not known to PPC because the pool was not opened:

```
M LSCPOOL COR=TP4 POOL=TP3TOTP5;
LSCPOOL: COMMAND NOT PERFORMED FOR TP3TOTP5,
         PPC REASON:RC=353F0110->PPC   63, NOTDONE / 9, 6
```

The pool is not known to NETGEN:

```
M LSCPOOL COR=TP5 NG;
LSCPOOL : UNKNOWN OR NO MATCHING CORRESPONDENT : TP5


M LSCPOOL COR=TP4 POOL=TP3TOTP5 NG
LSCPOOL : COMMAND NOT PERFORMED FOR TP3TOTP5,
          NETGEN REASON:RC=B8021017->CDH  2,OBJUNKN
```

List of characteristics of an XCP1 correspondent:

1. List of the attributes of the XCP1 correspondent JDX1.

```
LSCPOOL COR=JDX1 ATTRIBUTE=*TDS=JDB

-------------------------------------------------------

--         TDS = JDB      09:00:08       SEP 30, 1994  --

-----              CHARACTERISTICS OF THE POOL       -----

-------------------------------------------------------

JD         JE

TDS: JDB, LSCPOOL COMMAND COMPLETED
```

2. State of sessions with a given attribute.

```
LSCPOOL COR=JDX1 ATTRIBUTE=JD TDS=JDB

-------------------------------------------------------

--         TDS = JDB      09:10:08       SEP 30, 1994  --

-----              CHARACTERISTICS OF THE POOL       -----

-------------------------------------------------------

    ATTRIB.    ACTSESNB    FRZSESNB     TRSTSESNB

    JD                2           1             1

    ATTRIB.    STATE       RESSTATE      CORRESP

    JD00       LOGON       FREE/NALC

    JD01       FROZEN      ALLOCPRINC

    JD02       LOGGED      FREE

    JC03       LOGGED      ALLOCAUX     RALPH

TDS:JDB, LSCPOOL COMMAND COMPLETED
```

9.5.16 LIST_TDS_COR (LSTC)

Purpose:

Displays information about all correspondents or about specified correspondent(s).

Syntax:

```
[M] { LSTC         }
    {              }
    { LIST_TDS_COR }

    { USER | COR }=star12

    [ TYPE={ * | TM | XCP1 | XCP2 | DUMMY  | TCPIP }]

    [{ NG | NETGEN }={ 0 | bool }]

    [{ LOG | LOGGED }=bool ]

    [{ DTLD | DETAILED }={ 0 | bool }]

    [ SORT={ 0 | bool }]

    [{ PRTMB | PRINT_MEMBER }=name31 ]

    [ TDS={ #WTDS | name4 }]
```

Parameters:

COR *alias* USER	Correspondent or user for TM correspondent.
TYPE	Type of correspondent.
NETGEN	State of correspondent characteristics:
=0	Current characteristics of correspondents.
=1	*Mutually exclusive with LOGGED:* Characteristics of correspondents declared at network generation and currently accessible to TDS.

LOGGED	*Mutually exclusive with NETGEN=1:* Status of Correspondents:
=1	Correspondents currently logged on to TDS or in the process of logging on.
=0	Frozen users.
	Default: All users logged ***and*** frozen are listed and UNSPEC appears in the output banner.
DETAILED	Type of information:
=0	*Default:* Only the list of correspondents known to TDS.
=1	Detailed information such as transaction/TPR counters concerning the specified correspondent(s). See Output.
SORT	Order DETAILED entities are displayed:
=0	*Default:* Random order.
=1	Alphabetical order. Ensure enough memory space and CPU resources available for sort.
PRINT_MEMBER	Member in *tds-name*.DEBUG file for storing output to be printed. The file is always opened in output mode and is erased on cold restart of TDS.
	Default: Results are displayed on the screen.

Constraints:

NETGEN=1 and LOGGED are mutually exclusive.

Output:

Depending on the parameters specified, the messages returned can be TX22, TX23, TX24, TX25, TX26, TX27, TX28, TX29, TX33, TX34, TX56 or TX57.
ee Appendix I.

The output of this command is as follows:

```
-----------------------------------------
-----  TDS=PKIT  18:00:23  MAY 5, 1994  -----
-----       LIST OF CORRESPONDENTS      -----
-----------------------------------------
```

`STATE : state / TYPE : cortyp / LIST : list / OPTION : option`

Where:

state	LOGGED or FROZEN or UNSPEC.
type	TM, XCP1, XCP2, DUMMY or TCPIP.
list	DETAILED or SHORT.
option	*dynamic* or STATIC.

NOTE:

PKIT is TDS application; 18:00:23 is time of day when command was issued.

When DETAILED=1, the following return codes are displayed depending on correspondent type or the parameters specified:

NETGEN=0 (dynamic)	TX23 and TX24 for TM correspondents TX29 for XCP1 correspondents TX27 for XCP2 correspondents
NETGEN=1 (static)	TX26 for TM correspondents TX33 for XCP1 correspondents TX34 for XCP2 correspondents

Examples:

```
LSTC USER=JONES LOGGED=1 TDS=BUL3;
```

Displays information about a user called JONES if currently logged to BUL3.

```
LSTC USER=J* LOGGED=1 SORT=1;
```

Displays in alphabetical order all the correspondents known to TDS, whose names begin with J.

For a list of all types of correspondents known to TDS:

```
S: LSTC * *

               ------------------------------------------------
               ------  TDS=PKIT  14:32:21  FEB 28, 1994  ------
               ------       LIST OF CORRESPONDENTS       ------
               ------------------------------------------------

STATE: UNSPEC   / TYPE: TM / LIST: SHORT / OPTION: DYNAMIC
STEPHAN           H___PMS

STATE: UNSPEC   / TYPE: DUMMY / LIST: SHORT / OPTION: DYNAMIC
DUMMY

LSTC COMMAND COMPLETED
```

For detailed list of all Terminal Manager correspondents:

```
S: lstc * TM DTLD

               ------------------------------------------
               -----  TDS=PKIT  14:32:34  FEB 28, 1994  ----
               -----     LIST OF CORRESPONDENTS         ----
               ------------------------------------------

STATE : UNSPEC  / TYPE : TM / LIST : DETAILED / OPTION: DYNAMIC

CORRESPONDENT   ADDRESS      TX_COUNT   TPR_COUNT  STATUS TX_NM

STEPHAN                         51         91         P     M
H___PMS                          0          0         I

      LSTC COMMAND COMPLETED
```

where status codes apply to the correspondent and are:

F	FROZEN.
I	IDLE.
P	Executing TPR or in Sequence Mode.
G	Awaiting GAC event.
T	Awaiting TIMER notification.
V	Awaiting VCAM event: terminal list answer in conversation.
W	Awaiting for any other reason.

NOTE:

H___PMS is the internal correspondent for the programmed operator facility.

For detailed list of type TP4 correspondents declared in the NETGEN:

```
M LSTC COR=TP4 NG DTLD;

               ------------------------------------------------
               ------  TDS=PKIT  15:41:57  JUN 14, 1994  ------
               ------       LIST OF CORRESPONDENTS       ------
               ------------------------------------------------

STATE : UNSPEC/ TYPE : XCP2 / LIST : DETAILED / OPTION : STATIC

CORRESPONDENT   ADDRESS      COR_BACKUP   PRIM  PRL  WIN SYNLVL
TP4             BPCCX2TPS4                 Y    Y    Y    S

LSTC :  COMMAND COMPLETED
```

For detailed list of TP4 correspondents known to TDS but not declared in NETGEN:

```
M LSTC COR=TP4 DTLD;
               ------------------------------------------------
               ------  TDS=PKIT  15:49:06  JUN  2, 1994  ------
               ------       LIST OF CORRESPONDENTS       ------
               ------------------------------------------------

STATE: UNSPEC   / TYPE: XCP2 / LIST: DETAILED / OPTION: DYNAMIC
CORRESPONDENT     ADDRESS        NB_OF_POOLS
   TP4            BPCCX2TPS4         1
LSTC:  COMMAND COMPLETED
```

For a list of all correspondents known to TDS:

```
LSTC *;
               ------------------------------------------------
               ------       16:10:10  JUN 14, 1994       ------
               ------       LIST OF CORRESPONDENTS       ------
               ------------------------------------------------

STATE : UNSPEC   / TYPE : TM / LIST : SHORT / OPTION : DYNAMIC
PAILLUS      ROUDE         H___PMS

STATE : UNSPEC   / TYPE : TM / LIST : SHORT / OPTION : DYNAMIC
DUMMY

LSTC:  COMMAND COMPLETED
```

For a list of all currently logged correspondents:

```
LSTC * LOGGED;
               ------------------------------------------------
               ------  TDS=PKIT  16:10:28  JUN 14, 1994  ------
               ------       LIST OF CORRESPONDENTS       ------
               ------------------------------------------------

STATE : LOGGED / TYPE : TM / LIST : SHORT / OPTION : DYNAMIC
PAILLUS       ROUDE      H___PMS

STATE : LOGGED / TYPE : DUMMY / LIST : SHORT / OPTION : DYNAMIC
DUMMY

LSTC:  COMMAND COMPLETED
```

For a list of all frozen correspondents, whatever their type:

```
LIST_TDS_COR * TYPE=* LOGGED=0;
               ------------------------------------------------
               ------  TDS=PKIT  16:49:06  FEB 28, 1994  ------
               ------       LIST OF CORRESPONDENTS       ------
               ------------------------------------------------

STATE : FROZEN / TYPE : TM / LIST : SHORT / OPTION : DYNAMIC
ROUDE

LSTC:  COMMAND COMPLETED
```

For a list of all types of correspondents known by TDS, when a TCP transaction is being executed:

```
S: LSTC * *
               ----------------------------------------
               --  TDS = PL  10:38:40  NOV 14, 1996  --
               -----   LIST OF CORRESPONDENTS     -----
               ----------------------------------------

STATE : UNSPEC / TYPE : TM / LIST : SHORT / OPTION : DYNAMIC
LEVENEZ-P H___PMS

STATE : UNSPEC / TYPE : DUMMY / LIST : SHORT / OPTION : DYNAMIC
DUMMY

STATE : LOGGED / TYPE : TCPIP / LIST : SHORT / OPTION : DYNAMIC
TOTO
TDS : PL, LSTC COMMAND COMPLETED
```

For detailed list of all TCP correspondents:

```
S: LSTC * TCPIP DTLD
               ----------------------------------------
               --  TDS = PL  10:49:12  NOV 14, 1996  --
               -----   LIST OF CORRESPONDENTS     -----
               ----------------------------------------

STATE : LOGGED / TYPE : TCPIP / LIST : DETAILED / OPTION : DYNAMIC
CORRESPONDENT ADDRESS TX_COUNT TPR_COUNT STATUS TX_NM
TOTO                  3        8          P    ESSAI
TDS : PL, LSTC COMMAND COMPLETED
```

9.5.17 LIST_TDS_FILE (LSTF)

Purpose:

Displays information about the specified file(s).

Syntax:

```
[M]{ LSTF          }
   {               }
   { LIST_TDS_FILE }

    [ IFN={ * | star8 }]

    [{ OPEN | OPENED }=bool ]

    [{ DTLD | DETAILED }={ 0 | bool }]

    [ SORT={ 0 | bool }]

    [{ PRTMB | PRINT_MEMBER }=name31 ]

    [ TDS={ #WTDS | name4 }]
```

Parameters:

IFN	File(s) known to TDS.
OPENED	State of files:
=0	Closed files
=1	Opened files.
	Default: All matching files regardless of their state.
DETAILED	Type of information on files:
=0	*Default:* Only the names of files known to TDS according to the star-convention and the open filters.
=1	Details on files.
SORT	Order of file objects:
=0	*Default:* Random order.
=1	Alphabetical order: Ensure enough memory space and CPU resources are available for sort.

PRINT_MEMBER Member in tds-name.DEBUG file for storing the output to be printed. The file is always opened in output mode and is erased on cold restart of TDS.

Default: Results are displayed on the screen.

Output:

Depending on the parameters specified, the messages returned can be TX17, TX18 and TX57. See Appendix I.

Examples:

`LSTF IFN=CUST1;` Displays information about a file named CUST1.

`LSTF IFN=* OPENED=1 SORT=1;`
Displays information in alphabetical order on all opened files.

`LSTF IFN=ISEG2 DTLD;`
Displays the characteristics of the file named ISEG2

To list all files beginning with T:

```
S: LSTF T*
                  ------------------------------------------
                  -----  TDS=PKIT  14:24:45  APR 2, 1994  -----
                  -----           LIST OF FILES            -----
                  ------------------------------------------
       T1        T2       T3
       LSTF COMMAND COMPLETED
```

To display a detailed list of all files:

```
S: LSTF * DTLD
                 ------------------------------------------
                 -----  TDS=PKIT  16:15:05  APR 08, 1997   ----
                 -----           LIST OF FILES            -----
                 ------------------------------------------
        IFN       OPEN    MONITORED     PMD       EFN
        H_CTLM      Y        Y          UPDATE   PKIT.CTLM
        H_FORM      N        Y          UPDATE
                    RC=0C8E0909->TP7 14,IFNNASG
        T1          Y        Y          UPDATE   PKIT.INDEX1
        T3          Y        Y          INPUT    PKIT.INDEX3
        IFN       OPEN    MONITORED     PMD       EFN
        T2          N        N          OUTPUT  PKIT.INDEX2
                    RC=4B301879->DSMGT 48,CATERR
          TDS : PKIT, LSTF COMMAND COMPLETED
```

9.5.18 LIST_TDS_SPAWN (LSTSPW)

Purpose:

Lists spawned transactions not yet started to a group of users or to specific users.

Syntax:

```
[M] { LSTSPW         }
    {                }
    { LIST_TDS_SPAWN }

    { USER | COR }=star12

    [{ DTLD | DETAILED }={ 0 | bool }]

    [ SORT={ 0 | bool }]

    [{ PRTMB | PRINT_MEMBER }=name31 ]

    [ TDS={ #WTDS | name4 }]
```

Parameters:

COR alias USER	TM or XCP1 correspondent, or DUMMY.
DETAILED	Type of information displayed:
=0	*Default:* Displays only the list of users known to TDS.
=1	Displays detailed information such as number of spawned transactions with priorities for specified user(s).
SORT	Order of DETAILED entities:
=0	*Default:* Random order
=1	Alphabetical order: Ensure that enough memory space and CPU resources are available for sort.
PRINT_MEMBER	Member in *tds-name*.DEBUG file for storing output to be printed. The file is always opened in output mode and is erased on cold restart of TDS.
	Default: Results are displayed on the screen.

Output:

Depending on the parameters specified, the messages returned can be TX25 and TX38. See Appendix I.

Example:

```
LSTSPW USER=CASH DTLD PRTMB=POUT TDS=BILAN;
```

Displays detailed information on spawning for user CASH and stores the results in member POUT for the application BILAN.

```
LSTSPW COR=CREDIT DTLD SORT=1 TDS=BILAN;
```

Displays the number of transactions spawned towards CREDIT and the priorities in spawning queues.

9.5.19 LOAD_TDS_IQS (LDTIQS)

Purpose:

Loads IQS objects into type 2 segments of TDS.

Syntax:

```
[M] { LDTIQS       }
    {              }
    { LOAD_TDS_IQS }

    OBJECT=char30

    [ TDS={ #WTDS | name4 }]
```

See IQS-V4/TDS User's Guide.

Example:

```
M LDTIQS OBJECT='MYIQSSCHEMA';
```

The loaded schemas are listed for default #WTDS TDS. This is issued from the master mailbox.

9.5.20 LOAD_TDS_MEMORY (LDTMEM)

Purpose:

Loads the specified data into the non-swappable memory. This command may be used to improve the response time of a TDS application.

Syntax:

```
[M] { LDTMEM          }
    {                 }
    { LOAD_TDS_MEMORY }

    { TPR=name12                          }
    {                                     }
    { USE_PROC=name12                     }
    {                                     }
    {{ SHRDSTO | SHARED_STORAGE }=name12  }
    {                                     }
    { IFN=name8                           }
    {                                     }
    {{ TDAT | TDS_DATA }=bool             }

    [ TDS={ #WTDS | name4 } ]
```

Parameters:

TPR	Transaction program whose LINKAGE SECTION, data segments and the first code segment are loaded into memory.
USE_PROC	User procedure.
SHARED_STORAGE	SHARED STORAGE area.
IFN	File whose associated control structures are to be loaded, the file must be open.
TDS_DATA	TDS data structures.

Constraints:

TPR, USE_PROC, SHARED_STORAGE, TDS_DATA and IFN are mutually exclusive.

Output:

When the command successfully completes, one of these two messages appears:

```
TDS_DATA loaded into main memory
```

xxxxxxxxxxxx loaded into main memory

An unsuccessful attempt to execute the command results in the following message:

```
LOAD_TDS_MEMORY unsuccessful; rc XXXXXXXX
```

Examples:

```
LDTMEM TPR=LUBLU;
```

Loads the TPR named LUBLU "into the non-swappable memory.

```
S: LDTMEM?

   1/1                    LOAD_TDS_MEMORY                    -->:

                            load memory

     TPR                    tpr name                        TPRJPL

     USE_PROC               procedure name

     SHARED_STORAGE         shared storage

     IFN                    internal file name

     TDS_DATA               TDS internal data structure?

     TDS                    tds name                        TEXT

-----------------------------------------------------------------

S:
S:LOAD_TDS_MEMORY TPR=TPRJPL;
  TX55 LDTMEM COMMAND NOT PERFORMED RC=22011878->DYNAD 1,NOMATCH

S:LDTMEM TDSDATA
  TX55 LDTMEM COMMAND NOT PERFORMED RC=22011878->DYNAD 1,NOMATCH
```

tdsdata is not a known TPR.

```
S:LDTMEM IFN=T1
  TX67 LDTMEM T1 LOADED INTO MAIN MEMORY
```

9.5.21 MODIFY_COR_POOL (MDCPOOL)

Purpose:

Modifies certain characteristics of the specified pool of sessions belonging to the specified correspondent.

Syntax:

```
[M] { MDCPOOL         }
    {                 }
    { MODIFY_COR_POOL }

    { COR | CORRESPONDENT }=name12

    [ POOL=name8 ]                  [{ ATTR | ATTRIBUTE }=name2 ]

    [{ ACTSESS | ACTIVE_SESSION }={{+ | -} { 0 | dec4 }}]

    [ MAXSESSNB=dec4 ]              [ WINSRC=dec4 ]

    [ WINTGT=dec4 ]                 [ AUTO=dec4 ]

    [ TDS={ #WTDS | name4 }]
```

Parameters:

CORRESPONDENT	XCP1 primary correspondent, XCP2 correspondent or a virtual correspondent (DUMMY).
ATTRIBUTE	*Applicable only for XCP1 correspondent:* Local extended address.
	Default: First two characters of correspondent.
ACTIVE_SESSION	*Applicable only to XCP1 and dummy correspondents:* Number of sessions created in or deleted from the pool:
+dec4	Adds the number to the current number of sessions.
-dec4	Subtracts the number from the current number of sessions.

Applicable only for XCP2 correspondent:

POOL	Pool of sessions.
MAXSESSNB	Maximum number of sessions.
WINSRC	Minimum number of winner sessions for source.
WINTGT	Minimum number of winner sessions for target.
AUTO	Maximum number of winner auto-active sessions.

Constraints:

- POOL, MAXSESSNB, WINSRC, WINTGT and AUTO only apply for XCP2 correspondents.
- For the XCP1 and XCP2 correspondents, the total number of sessions must not exceed the maximum declared at the network generation.
- For DUMMY correspondent, the total number of sessions must not exceed the total declared in TDSGEN.

Output:

Depending on the parameters specified, the following messages can appear TX71, TX72, TX73, TX74, TX75, TX80 and TX56. See Appendix I.

For return codes sent by MODIFY_COR_POOL, see Appendix J.

For all error cases except TX80, please refer to the TDS COBOL Programmer's Guide.

If an error occurs, the CALL "OPEN-POOL" procedure (output parameters) give the significance of the TDS REASON.

For an XCP1 pool, the error message TX80 NOT CONNECTED displays a REASON value of 24 or 25. For more details, refer to the Master command MDTDS TRACE = 02.

Example:

```
MDCPOOL COR=CUST POOL=POOL4 MAXSESSNB=2;
```

Modifies the maximum number of sessions for the pool named POOL4 to which the XCP2 correspondent belongs.

9.5.22 MODIFY_TDS (MDTDS)

Purpose:

Modifies the global characteristics of a TDS session.

Syntax:

```
[M] { MDTDS      }
    {            }
    { MODIFY_TDS }

    [{ ACCNT | ACCOUNT }=bool ]

    [ SIMUL=[{ + | - }] dec3 ]

    [{ RPCS | RPC_SIMUL }=[{ + | - }] dec3 ]

    [ OTHER_SIMUL = dec3 ]

    [ MAXCPU = dec6 ]

    [ DUMP = bool]

    [ SMLIB = (smlib44 smlib44 [ smlib44 ]) ]

    [ TDS={ #WTDS | name4 } ]

    [ TRACE = hexa2 ]

    [ RESTART = bool ]

    [ MAXIDLETIME = dec6 ]

    [ XCP2TRC=hexa2 ]
```

Parameters:

ACCOUNT	Accounting for a TDS application at user level (LOGOUT/DISCNCT), transaction or TPR level. *Accounting at TDS session level is not modified.*
=1	Enables accounting
=0	Disables accounting.
SIMUL	Dynamically modifies the simultaneity level for current TDS session: – unsigned *dec3* is the new simultaneity level – signed *dec3* adds to or subtracts from the current simultaneity level.
RPC_SIMUL	Dynamically modifies the rpc simultaneity level *nrpcsimu* for the current TDS session: – unsigned *dec3* is the new number of RPC processes – signed *dec3* adds to or subtracts from the current number of RPC processes. *nrpcsimu* must have been previously defined through RPC SIMULTANEITY Clause in STDS. Otherwise INVUSE is returned and the command is not performed. The new *nrpcsimu* must be less than the current *nsimu*. Otherwise COUNTLIM is returned and the command is not performed.
OTHER_SIMUL	This keyword is reserved for internal purposes and must not be set.
MAXCPU	CPU time limit in milliseconds for processing a TPR defined in TPR-TIME-LIMIT Clause of TDS SECTION. *dec6* => 64 milliseconds. If *dec6* <= 64 milliseconds, *maxcpu* defaults to 64 milliseconds.
DUMP	On TPR abnormal termination:
=1	Dump taken. For PROCEXP abort code, a dump is always produced.
=0	No dump.

SMLIB

Modifies search order of specified SM libraries declared in the JCL statement JOBLIB and loaded in backing store. Declare active libraries before inactive ones.

The following actions can be performed on an *inactive* smlibrary:
– adding or removing a TPR
– substituting any TPR for an existing TPR.

[M] MODIFY_TDS SMLIB is non-concurrent with all transactions. To find out which search order is currently in use, enter:
[M] DISPLAY_TDS SMLIB=1.

The updated search order is stored and retrieved at next warm restart. *Referenced Libraries must contain SMs with the same name.* When starting a TPR in an SM not attached to the current TDS even if the SM is loaded after the TDS is started, the transaction aborts and DYNAD INDOUT is returned.

The command is executed if the new simultaneity level is at least 1 and <= *nsimu* specified in SIMULTANEITY Clause of STDS. Otherwise the command is not performed and COUNTLIM is returned.

TRACE

Used only at the request of the Service Center: TDS monitor to trace detailed events during the session. When the trace is disabled, only some basic events are traced.

hexa2 consists of 8 bits, each bit traces different events:

1 general tds detailed trace (rcv,snd,...).

2 trace commit manager.

3 A.M internal trace.

4 trace PPC conversation verbs.

5 trace PPC-PI verbs called in TPR.

6 trace ppc_object_management verbs.

7 is used to supply a TDS WARNING message MV37 when a failure is detected during the connection of an XCP1 correspondent. The MV37 warning may be de-activated by specifying TRACE =00.

8 is used (from TS7356) to supply a TDS WARNING message MV37 when a connection failure of a TM correspondent is detected. The MV37 warning may be de-activated by specifying TRACE =00.

RESTART

Used only at the request of the Service Center:

=1 Automatic restart of the TDS application

=0 Manual restart of the TDS application. See Paragraph 11.3.3 *Recoverable Abort Conditions*.

MAXIDLETIME

Modifies the inactive time limit in seconds defined by IDLE-TIME Clause of TDS Section.

NOTE:

Correspondents already in IDLE state before the dynamic modification of the idle TIME remain with their own idletime until they launch again a transaction and pass again in IDLE state.

XCP2TRC

This keyword allows the system to issue TX95 and PP07 to PP11 messages and PP06 message issued by the procedure H_PPC_RMPROC, and by the procedure H_PPC_SPM when the reason value is 12 (backout restore/norestore conflict). PP06 message is not filtered in other procedures. These messages are printed on the TDS master terminal and in the TDS JOR.

Value 80 enables message issuing.
Value 00 disables message issuing.

Values other than 0 and 80 are reserved for a future use.
Default value is 0 after TP7GEN or a TDS cold restart. After a TDS warm restart, the value remains unchanged from the previous session.

The XCP2TRC keyword is exclusive of other keywords except MAXIDLETIME.

Constraints:

- RPC_SIMUL can be specified with SIMUL but both are non-concurrent with all the other parameters. The new *nrpcsimu* (RPC simultaneity level) must be less than the new *nsimu* (TDS simultaneity level).
- When specifying SMLIB, the transaction which processes MDTDS is non-concurrent with all transactions.
- The number of specified values in SMLIB must be the same as the number of declared SM libraries in the JOBLIB statement of the TDS JCL job. Otherwise the TX61 message is returned.
- If the value specified for MAXCPU is less than 64 milliseconds, the timelimit of 64 milliseconds applies.
- MAXIDLETIME is non-concurrent with all the other parameters.

Output:

When the command executes normally, the TX54 message appears.

If the command cannot be executed, either the TX55 or TX61 message is displayed.

The return codes for this command are listed in Appendix G.

NOTE:

When either or both simultaneity levels are changed, WAITING TPR MEAN and WAITING RPC TPR MEAN values are reset. These two averages that can be displayed by [M] DISPLAY_TDS command, allow the user to see the effect of a change on the different simultaneity levels.

Examples:

```
S: MDTDS SIMUL=+8
   TX55 MODIFY_TDS COMMAND NOT PERFORMED
         RC=0CB80025->TP7  56,COUNTLIM

S: MDTDS SIMUL=2
   TX54 MODIFY_TDS COMMAND COMPLETED

S: MDTDS ACCOUNT
   TX54 MDTDS COMMAND COMPLETED

S: MDTDS ACCOUNT=0
   TX54 MDTDS COMMAND COMPLETED

S: MDTDS MAXCPU=752014
   TX54 MDTDS COMMAND COMPLETED
```

The TPR timelimit is 752014 milliseconds.

```
S: MODIFY_TDS DUMP=0 TDS=TEXT;
   TX54 MDTDS COMMAND COMPLETED

S: MDTDS MAXIDLETIME=600
   TX54 MDTDS COMMAND COMPLETED
```

The maximum inactive time is now 600 seconds.

```
S: MDTDS XCP2TRC=80:
```

Messages TX95 and PP07 to PP11 are issued.

```
S: MDTDS XCP2TRC=00:
```

Messages TX95 and PP07 to PP11 are inhibited.

Example of TPR Replacement:

At the beginning of the session, the job description includes:

```
JOBLIB SM,TDS1.SMLIB,DEBUG1.SMLIB,DEBUG2.SMLIB;
```

This means that the initial state of the libraries is as follows:

TDS1.SMLIB contains TPR1 to TPR5
DEBUG1.SMLIB contains a new version of TPR1.
DEBUG2.SMLIB is empty.

To test the new version of TPR1:

```
MODIFY_TDS SMLIB = (DEBUG1.SMLIB,TDS1.SMLIB,DEBUG2.SMLIB)
```

The following activities occur:

- The new TPR1 from DEBUG1.SMLIB library is used instead of the *live* TPR1 from TDS1.SMLIB library.
- The search order is now:

  ```
  DEBUG1.SMLIB TDS1.SMLIB DEBUG2.SMLIB
  ```

- TPR2, TPR3, TPR4, and TPR5 are taken from TDS1.SMLIB library when they are required since no copy exists in DEBUG1.SMLIB library.
- TPR1 in TDS1.SMLIB library is now unused. An unused TPR may be replaced by manipulating the library (the SM can be reloaded by SYSMAINT LOAD command).

NOTE:
Here the TPR is considered an SM and not a TPR.

- If the version of TPR1 in DEBUG1.SMLIB library is correct:

  An update to the TDS1.SMLIB library is necessary to install the new version of TPR1. This can be done either by linking the new version of TPR1 and storing it, or by copying it from the DEBUG1.SMLIB library.

  The new TPR1 is then loaded into backing store and the MODIFY_TDS...SMLIB=... is used to restore the original searching order. Since the original search rules are restored, TPR1 in the DEBUG1.SMLIB library becomes unavailable.

- If the version of TPR1 in DEBUG1.SMLIB library is incorrect:

  The new version of TPR1 must be made unavailable by altering the search rules to TDS1.SMLIB DEBUG1.SMLIB DEBUG2.SMLIB. The corrected version of TPR1 is compiled, linked, and put into DEBUG1.SMLIB library instead of the incorrect version. The testing is performed as above.

  The updated search rule is stored to be retrieved at next TDS warm restart.

- To choose an SM to search for, issue the command MODIFY_TDS...SMLIB=...

  After a series of MODIFY_TDS...SMLIB=... commands, an incorrect TPR could be selected, if an incorrect SM has been searched. To avoid losing track of which SM is searched, do not have more than 2 SM libraries.

  Wherever possible, use one *live* SM library and a separate SM library for debugging.

9.5.23 MODIFY_TDS_MOT (MDTMOT)

Purpose:

Sends a message to all users logged on to a TDS application, including those who log on later in the current session.

Syntax:

```
[M] { MDTMOT         }
    {                }
    { MODIFY_TDS_MOT }

    [{ MSG | MESSAGE }= char250]

    [ IMMED={ 0 | bool }]

    [ TDS={ #WTDS | name4 }]
```

Parameters:

MSG — No value for MESSAGE cancels previous message. Until TS 8560, this field must not contain any single quote (') character.

IMMED — *Effective only with active TM correspondents (as opposed to passive terminals) and meaningful only if a value is specified for MESSAGE:* Delivers the message immediately to all users even if a transaction is currently running for any user.

NOTE:

Each subsequent command submitted overrides the previous message.

During the login of a user (LOGON and RESTART transactions), no message is received. The message will be displayed when the login is completed.

Examples:

```
MDTMOT MSG='System Going Down in 10 Minutes';

S: MDTMOT 'GOOD MORNING'
   TX54 MDTMOT COMMAND COMPLETED
```

9.5.24 MODIFY_TDS_RESTART_OPTION (MDTRSO)

Purpose:

Suppresses all options invoked for the previous session and specifies the restart options of the next TDS session when the current TDS session terminates abnormally.

If the current session terminates normally, this command is ineffective and all values are taken from the options specified in the TERMINATE_TDS command.

Syntax:

```
[M] { MDTRSO                     }
    {                            }
    { MODIFY_TDS_RESTART_OPTIONS }

    [ MODE={ WARM | COLD }]

    [{ CN | CONNECT }={ PASSIVE | ALL | ACTIVE | NONE }]

    [{ IDLE | IDLE_TOO }={ 0 | bool }]

    [{ EXDISTX | EXEC_DISCONNECT_TX }={ 0 | bool }]

    [ TDS={ #WTDS | name4 }]
```

Parameters:

MODE	Restart mode of next TDS session.
COLD	TDS restarts with initial conditions as defined in TDSGEN. Dynamic modifications in the previous session are nullified.
WARM	*Default:* Reconnects PASSIVE correspondents that are not idle. See Paragraph 9.3.2 *Warm Restart*.
CONNECT	*Applicable to terminals and XCP1 correspondents on warm restart:* How TDS recovery routines handle correspondents:
=ALL	Reconnect all *active* and *passive* correspondents.
=ACTIVE	Reconnect only *active* correspondents.

=NONE	Prevent correspondents from connecting or being reconnected until [M] ALLOW_NEW_TDS_COR is issued.
=PASSIVE	*Default:* Reconnect only *passive* correspondents.
IDLE_TOO	How *idle* correspondents are handled:
=1	Also connects the correspondents not executing a transaction.
=0	*Default:* Reconnects only the correspondents that were not idle at abort to speed up restarting the TDS session.

EXEC_DISCONNECT_TX

=1	Executes the DISCNCT transaction at next restart for each correspondent disconnected when TDS was forced to terminate.
=0	*Default*

Constraints:

- MODE=COLD, CONNECT and EXEC_DISCONNECT_TX are mutually exclusive.
- IDLE_TOO is meaningless without ACTIVE or PASSIVE or ALL.

Examples:

```
MDTRSO MODE=WARM CN=ACTIVE;
```

Changes the restart of next TDS session to WARM and allows only active correspondents to be reconnected if the current TDS session terminates abnormally.

```
S: mdtrso cn=ALL idle
```

Allows all active and passive correspondents to reconnect including those correspondents not executing a transaction when the TDS session abnormally terminated.

```
TX54 MDTRSO COMMAND COMPLETED
```

9.5.25 MODIFY_TDS_XA (MDTDSXA)

Purpose:

Modifies the XA parameters of a TDS session.

Syntax:

```
[M] { MDTDSXA       }
    {               }
    { MODIFY_TDS_XA }

    [ XA_RESYNC_DELAY ]

    [ TDS={ #WTDS | name4 } ]
```

Parameters:

XA_RESYNC_DELAY Modifies the delay for XA resynchronization, expressed in seconds.

Constraints:

None.

Output:

When specified, the XA_RESYNC_DELAY is updated with the corresponding input value.

If the command cannot be completed, the following message is output:

`TX55 TDS : PL, MDTDSXA COMMAND NOT PERFORMED rc`

Example:

```
S: MDTDSXA XA_RESYNC_DELAY = 300 TDS = PL
   TX54 TDS : PL, MDTDSXA COMMAND COMPLETED
```

9.5.26 MODIFY_TX

Purpose:

Modifies the characteristics of a specified transaction or class of transactions. If no parameters are entered, the following can be performed:

- modify all parameters (except NON-CONCURRENT) of an existing transaction
- create a new transaction available for use until the next cold restart session.

Syntax:

```
[ M ] { MDTX      }
      {           }
      { MODIFY_TX }

        [ TX=name8 ]

        [{ TXC | TX_CLASS }={ 2 | char1 }]

        [{ AUTHCODE | AUTHORITY_CODE }=( dec2 [,dec2 ]...) ]

        [ LOCK=bool ]

        [{ VAL | VALIDATE }=bool ]

        [{ ACCNT | ACCOUNT }=bool ]

        [{ TPRACCNT | TPR_ACCOUNT }=bool ]

        [ XA = bool ]

        [ CMA = bool ]

        [ TDS={ #WTDS | name4 }]
```

Parameters:

TX	Name of the transaction to be modified or created.This name is up to 8 alphanumeric characters long.
TX_CLASS	Single alphabetic character identifying a particular class as defined in the CLASS clause in TRANSACTION SECTION. *Default:* Z for all transactions.

AUTHORITY_CODE Decimal ranging from 0 through 31 specifying new access rights for running a transaction.

The value specified cancels and replaces the existing authority code(s). To add or delete one or more codes without affecting the others, use MDTX without parameters (except for TX) and reply to the prompts as explained under *Usage*.

LOCK Prevents (LOCK=1) or allows the transaction or a class of transactions from starting. A transaction in progress when the command is issued, is allowed to complete.

VALIDATE *Applicable only if DEBUG has not been specified in TDSGEN:* Validate TDS-controlled files associated with the specified transaction or class of transactions on the current TDS session:

=0 Files cannot be modified until either VALIDATE=1 is issued or the end of the session.

=1 Files can be modified when the commitment unit ends normally. Currently running transactions are not affected.

ACCOUNT Accounting at transaction level. TPR accounting depends on TPR_ACCOUNT or the initial value set in WITH [TPR] ACCOUNTING Clause.

TPR_ACCOUNT Accounting at TPR level:

=0 Only transaction accounting records will be written to the system accounting file.

=1 *Meaningful only if ACCOUNT=1:* Both transaction and TPR accounting records are written in the system accounting file.

If ACCOUNT=0, the value specified in TPR_ACCOUNT is ignored and no accounting is performed.

XA If set to 1, specifies that the transaction activates the XA protocol.

CMA If set to 1, specifies that the transaction wishes to access a CMA database.

Constraints:

- This command is effective for transactions meeting the criteria at the time the command is issued. If creating or modifying a transaction, reissue a request with the correct value set in the LOCK parameter.
- TX and TX_CLASS are mutually exclusive.
- TX_CLASS=S is not allowed since it is the reserved class for system transactions.
- TX_CLASS=* denotes a requirement for all transactions of all classes, with the exception of "S" (system transaction class).
- Note that the maximum number of dynamically created transactions must be defined in TDSGEN for creating more than 10 transactions.
- It is possible to modify the value of the XA parameter only when XA is defined at TDS level. In this case, it is forbidden to change a non-XA transaction to an XA transaction if the XA session was not successfully initialized, or if this transaction has been declared "for inquiry".
- It is possible to modify the value of the CMA parameter only when at least one CMA transaction is defined during the generation phase (in the STDS file). If no CMA transaction exists at generation level and the master wishes to create one with this command, the TX55 message is displayed on the master terminal.
- XA and CMA are mutually exclusive

Output:

Either message TX52 or TX55 appears when the command is unsuccessful. To view the output of this command, use DISPLAY_TX.

Usage:

On issuing [M] MDTX **without parameters**, prompts appear for each parameter. All parameters in TRANSACTION SECTION can be issued except NON-CONCURRENT.

Use the following characters to respond to the parameter prompts:

=	Keep current value, also applicable if no response is entered
>	Set defaults for parameters following, also for end of MDTX.
/	Exit from MDTX without creating or validating.
<	Return the prompt to the preceding question.
0	Reset control for MAXIMUM NUMBER OF TPR.

For modifying SUPPRESS CONCURRENT ACCESS CONTROL and SHARED READ:

`[*] [ { + | - } ifn ]...,` where

`*`	Resets the file list
`+`	Adds the file to the list `(+ ifn)`
`-`	Removes the file from the list `(- ifn)`
`.`	Terminates list (or an empty line for no modification).

Example of Modifying File List:

`*+file1 +file2.`	List consist of *file1 file2*.
`+file1 -file2.`	List is as displayed (previous value) plus *file1* minus *file2*.

If an error occurs, only the line containing the error is ignored.

For the AUTHORITY CODE, the syntax to modify is:

`[*] [ { + | - } authority-code ]...,` where

`*`	Resets all authority code(s) for respecification.
`+`	Adds given authority code to the list
`-`	Removes given authority code from the list.
`.`	Terminates list or an empty line if no modification is required.

Example of Authority Code Prompts:

A list of the current authority codes is displayed:

`*+5 +12 +8` Authority codes list contains 5, 8 and 12.

Modify the list by entering:

`-13 +17 +31` Authority codes 17 and 31 are added to the previous list and 13 is removed.

A message will appear:

```
NEXT AUTHORITY CODE ?
```

Then enter:

`-8` Authority code 8 is removed from the list.

If an error occurs, only the line containing the error is ignored.

Example of Modifying Number of TPRs:

This prompt is displayed:

```
MAXIMUM NUMBER OF TPR ? (yyyyy)
```

where *yyyy* is current number of TPRs from 0 through 32767. 0 indicates no control.

Examples of Accounting:

`M MDTX X*Y* ACCOUNT=0;`
Accounting is not done.

`M MDTX T*R ACCOUNT=1 TPR_ACCOUNT=0;`
Transaction accounting is done.

`If WITH TPR ACCOUNTING`
is specified for a transaction:

Transaction and TPR accounting are done.

If issuing:
`M MDTX ACCOUNT=0;` No accounting is done.

If issuing:
`M MDTX...ACCOUNT=1;`
Only transaction accounting is done.
If issuing:
`M MDTX...ACCOUNT=1 TPR_ACCNT=1;`
Transaction + TPR accounting is done. Sets initial accounts state at TDSGEN.

Other Examples:

`S: MDTX * LOCK` Prevents all user transactions from starting.
`TX54 MDTX COMMAND COMPLETED`

`S: MDTX J* LOCK=0` Cancels effect of MDTX J* LOCK=1 for transactions beginning with character j.
`TX54 MDTX COMMAND COMPLETED`

`S: mdtx hhh validate`
Indicates that transaction *hhh* whose DEBUG option is to be removed, is unknown. Correct transaction name.
`TX52 MDTX COMMAND NOT PERFORMED UNKNOWN TX : HHH`

`S: mdtx hhh AUTHCODE=30`
The new authority code for transaction *hhh* is 30.
`TX54 MDTX COMMAND COMPLETED`

`S: MDTX` *hhh* `AUTHCODE=(1,5,10,31)`
The new authority codes are 1, 5, 10 and 31. Authority code 30 has been overwritten.

`S: MDTX TX = MENU XA = 1 TDS = PL`
The XA protocol is activated for the transaction menu.

`S: MDTX TX = MENU CMA = 1 TDS = TD07`
transaction menu can access a CMA database.

9.5.27 OPEN_COR_POOL (OCPOOL)

Purpose:

Opens one or several session pools between a local TDS application and a partner application.

Syntax:

```
[M] { OCPOOL         }
    {                }
    { OPEN_COR_POOL  }

    { COR | CORRESPONDENT }=name12

    [ POOL={ * | name8 }]

    [{ ATTR | ATTRIBUTE }=name2 ]

    [{ ACTSESS | ACTIVE_SESSION }=dec4 ]

    [ TDS={ #WTDS | name4 }]
```

Parameters:

CORRESPONDENT	XCP2 correspondent or a XCP1 primary correspondent.
POOL	*Applicable only to XCP2 correspondent:* * opens all pools of the specified correspondent.
ATTRIBUTE	*Applicable only to XCP1 correspondent:* Local extended address. *Default:* First two characters of specified correspondent.
ACTIVE_SESSION	*Applicable only to XCP1 correspondent:* Number of sessions to be immediately activated.

Constraints:

Total number of sessions must not exceed the maximum declared at the network generation for the XCP1 and XCP2 correspondents.

Output:

Depending on the parameters specified, the messages returned can be TX70, TX71, TX72, TX73, TX74, TX75, TX80 and TX56. See Appendix I.

See Appendix J for return codes sent by OPEN_COR_POOL.

For all error cases except TX80, please refer to the TDS COBOL Programmer's Guide.

If an error occurs, the CALL "OPEN-POOL" procedure (output parameters) give the significance of the TDS REASON.

For an XCP1 pool, the error message TX80 NOT CONNECTED displays a REASON value of 24 or 25. For more details, refer to the Master command MDTDS TRACE = 02.

Examples:

```
OCPOOL COR=STOCK POOL=POOL2;
```

Opens pool named POOL2 used by correspondent STOCK.

```
M OCPOOL POOL=TP3TOTP4 COR=TP4
OPEN_COR_POOL : 1 POOL(S) OPENED TOWARDS TP4
```

Appears at terminal of master controlling local application.

```
POOL TP3TOTP4 OPENED BY TP3
```

Appears at terminal of master controlling partner application.

9.5.28 OPEN_TDS_FILE (OTF)

Purpose:

Assigns and opens a file that has been previously closed and/or deassigned through the CLOSE_TDS_FILE command, or not opened at TDS startup.

Syntax:

```
[ M ] { OTF           }
      {               }
      { OPEN_TDS_FILE }

      IFN=name8

      [     { AP | APPEND }]        [{ FILE | EFN }=file78 ]
      [ PMD={ OU | OUTPUT }]
      [     { IN | INPUT  }]        [ TDS={ #WTDS | name4 }]
      [     { UP | UPDATE }]

      [       { MONITOR  }]         [         { WRITE   }]
      [ SHARE={ NORMAL   }]         [ ACCESS={ READ    }]
      [       { DIR      }]         [         { SPWRITE }]
      [       { ONEWRITE }]         [         { SPREAD  }]
```

Parameters:

IFN	File declared at TDSGEN to be assigned and opened.
FILE	External name of the file to be assigned and opened. See *GCL Programmer's Manual* for convention for file names.
PMD	Processing mode in which the file is to be opened:
ACCESS	Access mode of specified file. *Default:* WRITE
SHARE	Sharing mode of specified file: – If the file is uncataloged, or cataloged with SHARE=UNSPEC: . *default* is MONITOR for controlled files . *default* is NORMAL for non-controlled files. – If the file is cataloged with SHARE other than UNSPEC, the *default* is SHARE defined in catalog. Access mode becomes *exclusive* to the TDS application, namely, SPWRITE or SPREAD.

Constraints:

- OPEN_TDS_FILE is executed serially whereby all other processes are suspended until the command is executed.
- PMD={OU|AP} must not be used for TDS controlled files. Defaults for PMD are those specified at TDSGEN.
- Multi-volume files are supported only if they are cataloged.
- SUPPRESS CONCURRENT ACCESS CONTROL clause is no longer required for transactions, including special-purpose transactions, using files assigned in SHARE=MONITOR and ACCESS=READ.

Output:

The message *ifn* OPENED means that the file has been opened. If the file cannot be opened, the message *ifn* NOT OPENED rc=*xxxxxxxx* appears on the master terminal. See Appendix A for return codes.

Examples:

```
M OTF  IFN=ifn1  EFN=PKIT.PROD1;
```

Assigns and opens the file named PKIT.PROD1

```
S: otf t1
   TX64 OTF : T1 NOT OPENED   RC=0C860909->TP7    6,IFNNASG

S: otf T1 efn=pdl.uffind1
   TX63 OTF : T1 OPENED
```

9.5.29 OPEN_TDS_SOCKET (OTS)

Purpose:

Opens the TDS socket. See *TDS-TCP/IP User's Guide*.

9.5.30 PREVENT_NEW_TDS_COR

Purpose:

Prevents new correspondents, TCP ones included (See *TDS-TCP/IP User's Guide*), from logging onto a TDS application. Prevents allocating new XCP1 and XCP2 sessions by [M] MODIFY_COR_POOL, [M] OPEN_COR_POOL or corresponding CALLs described in *TDS COBOL Programmer's Guide*. For XCP2 correspondents, new conversations can be allocated using pools already opened.

When all current correspondents log off normally, TDS remains idle until [M] ALLOW_NEW_TDS_COR or [M] TERMINATE_TDS is issued.

Syntax:

```
[M] { PVNTC                }
    {                      }
    { PREVENT_NEW_TDS_COR }

    [{ TMC | TM_COR }=bool ]

    [{ X1C | XCP1_COR }=bool ]

    [{ X2C | XCP2_COR }=bool ]

    [ TDS={ #WTDS | name4 } ]

    [{ TCPIPC | TCPIP_COR }=bool ]
```

Parameters:

TM_COR	Terminal correspondents.
XCP1_COR	XCP1 Correspondents.
XCP2_COR	XCP2 Correspondents.
TCPIP_COR	TCP correspondents, see *TDS-TCP/IP User's Guide*.

Usage:

No parameters is the same as specifying all correspondents.

PVNTC X2C=1 does not prevent the XCP2 service pool from opening (SNASVCMG). This service pool is managed directly by XCP2.

Output:

The TX54 message indicates that the command has been taken into account.

Example:

```
M PVNTC TMC=1 TDS=PKIT;
```

Prevents the terminal correspondents from logging onto the TDS application named PKIT.

```
S: pvntc xcp2_cor
   TX54 PREVENT_NEW_TDS_COR COMMAND
   COMPLETED
```

9.5.31 SEND_TDS (SNDT)

Purpose:

Allows a TDS master connected through IOF to start a user transaction, or dialog with a system or user transaction. This is not strictly a master command.

Syntax:

```
{ SNDT     }
{          }
{ SEND_TDS }

{ MESSAGE | MSG }=char255

[ TDS={ #WTDS | name4 }]
```

Parameters:

MESSAGE String of up to 255 alphanumeric characters.

Constraints:

SEND_TDS is only available when the master operator is logged on under IOF and wishes to dialog with a TDS application.

Usage:

- When issuing SEND_TDS with a message of zero length, TDS simulates a break.
- Use double quotes to protect characters to be received *as-is* by TDS.
- Responses to SEND_TDS are always displayed asynchronously. See MAIL described in Paragraph 9.4.1.1 *Restrictions Under IOF*.
- SEND_TDS is useful for replying to TX57 message where scroll forward makes different parts of the message text visible. To exit, key in SEND_TDS 'N'.

Examples:

```
S: SNDT 'M SIMU +3' TDS=PKIT;
```

Increases the current number of simultaneity levels of the TDS application named PKIT by 3.

```
S: SNDT 'USCREATE SMITH 35' TDS=PKIT;
```

Activates the user transaction USCREATE with parameters SMITH AND 35 for the application named PKIT.

9.5.32 SEND_TDS_USER (SNDTU)

Purpose:

Sends a message to all users matching the asterisk convention.

Syntax:

```
[M] { SNDTU         }
    {               }
    { SEND_TDS_USER }

    { MESSAGE | MSG }= char250

    { USER | COR }=char12

    [ IMMED={ 0 | bool }]

    [ TDS={ #WTDS | name4 }]
```

Parameters:

MESSAGE alias MSG — Until TS 8560, this field must not contain any single quote (') character.

USER alias COR — User or Correspondent.

IMMED

=1 — The message is sent immediately to the terminal whether the IMMEDIATE DELIVERY is specified in the SERVICE MESSAGE generation clause (see TDS SECTION of STDS) or not.

=0 — Default value.

If IMMEDIATE DELIVERY is specified in the SERVICE MESSAGE generation clause (see TDS SECTION of STDS) , the message is sent at the end of the TPR in progress,

else, the message is delivered only when the last message of the transaction is sent.

Constraints:

This command is not effective with DUMMY, XCP1, or XCP2 correspondents.

Output:

If the specified user is not known to the TDS application, or if no user matching the star convention is found, the TX56 message appears.

NOTE:

During the login of a user (LOGON and RESTART transactions), no message is received. The message will be displayed when the login is completed.

Example:

```
M SNDTU MESSAGE='CONSULT ORACLE'
        USER=SAVINGS* IMMED=1;
```

Send the message CONSULT ORACLE immediately to all users whose names begin with SAVINGS.

9.5.33 SUPERVISE_TDS

Purpose:

Dynamically activates the supervision of the TDS session. This command should be used only at the request of the Service Center.

Syntax:

```
[ M ] { SPRVT         }
      {               }
      { SUPERVISE_TDS }

      [ SESSION=bool ]          [ PROCESS=bool ]

      [ LEVEL=dec1 ]            [ ABORT-LEVEL=dec1 ]

      [ DELAY=dec5 ]            [ TDS={ #WTDS | name4 }]
```

Parameters:

For a complete explanation of the SESSION, PROCESS, LEVEL, ABORT-LEVEL, and DELAY parameters, see Paragraph 3.3.12 *DYNAMIC-SUPERVISION Clause*.

The defaults when these parameters are omitted, are:
- the preceding SUPERVISE_TDS command (if any)
- the DYNAMIC-SUPERVISION Clause specified at TDSGEN (if any).

If these values do not exist, TDS automatically provides a complete set of defaults:

SESSION=0
PROCESS=0
LEVEL=4
ABORT-LEVEL=5
DELAY=20.

Usage:

Issue this command before a TDS application is deadlocked; otherwise the command SUPERVISE_TDS is not taken into account. To ensure that the supervision service is always activated, specify DYNAMIC-SUPERVISION Clause.

Issue this command whether DYNAMIC-SUPERVISION Clause was specified or omitted at TDSGEN.

Output:

Messages returned for this command are TX54 and TX55. The messages TX76, TX77, and TX78 may also appear when the supervision mechanism is active.

Constraints:

If RESNAV is returned, increase the current simultaneity level. Supervision requires at least one process.

Examples:

```
M SPRVT PROCESS=1 LEVEL=3 DELAY=10;
```

Activates dynamic process-supervision.

```
M SPRVT PROCESS=0;
```

Deactivates supervision.

9.5.34 TERMINATE_TDS (TTDS)

Purpose:

Closes the TDS session and:

- prevents new transactions from being scheduled,
- suppresses all the options invoked for the previous session and indicates the restart conditions for the next TDS session,
- allows transactions and commitment units in progress to complete normally or forcibly depending on STRONG,
- freezes correspondents progressively and logs them off,
- allows the master operator to enter commands until shutdown.

Syntax:

```
[M] { TTDS           }
    {                }
    { TERMINATE_TDS }

    [ STRONG={ 0 | bool }]

    [ MODE={ WARM | COLD }]

    [{ CN | CONNECT }={ PASSIVE | ALL | ACTIVE | NONE }]

    [{ IDLE | IDLE_TOO }={ 0 | bool }]

    [{ EXDISTX | EXEC_DISCONNECT_TX }={ 0 | bool }]

    [ TDS={ #WTDS | name4 }]
```

Parameters:

STRONG Determines how TDS is terminated:

=1 Forces TDS to shut down immediately.

CAUTION:
If a correspondent is executing a TPR which is looping, the TERMINATE_TDS command will not take effect until the TPR-TIME-LIMIT (defined at TDS generation time) is reached.

=0 *Default:* Allows all active transactions to complete before TDS terminates and user is logged off.

MODE Restart mode of next TDS session:

COLD TDS restarts with initial conditions as defined in TDSGEN. Dynamic modifications in the previous session are nullified.

WARM *Default:* Reconnects PASSIVE correspondents that are not idle. See Paragraph 9.3.2 *Warm Restart*.

CONNECT *Applicable to terminals and XCP1 correspondents on warm restart:* How TDS recovery routines handle correspondents:

=ALL Reconnect all *active* and *passive* correspondents.

=ACTIVE Reconnect only *active* correspondents.

=NONE Prevent correspondents from connecting or being reconnected until [M] ALLOW_NEW_TDS_COR is issued.

=PASSIVE *Default:* Reconnect only *passive* correspondents.

IDLE_TOO How *idle* correspondents are handled:

=1 Also connects correspondents who were not executing a transaction.

=0 *Default:* Reconnects only the correspondents that were not idle at abort to speed up restarting the TDS session.

EXEC_DISCONNECT_TX

=1	Executes the DISCNCT transaction at next restart for each correspondent disconnected when TDS was forced to terminate.
=0	*Default*

Constraints:

- MODE=COLD, CONNECT, EXEC_DISCONNECT_TX are mutually exclusive.
- IDLE_TOO is meaningless without ACTIVE or PASSIVE or ALL.
- EXEC_DISCONNECT_TX is meaningful only if STRONG=1.

Output:

When all correspondents are logged off, the SHUTDOWN transaction is started. On completion of the TDS job, the TX53 message appears on the master operator's terminal:

```
TX53 "TDS tdsname SHUTDOWN"
```

Examples:

- `S: LET #WTDS PKIT;`

  `S: TTDS;` — Shuts down the #WTDS application under the name PKIT.

  ```
  TX54 TTDS COMMAND COMPLETED
  TX53 TDS: PKIT SHUTDOWN
  ```

- `TTDS CN=NONE TDS=PK2;`

  Shuts down the application named PK2 and prevents correspondent from logging on until [M] ALLOW_NEW_TDS_COR is issued.

- `S: TTDS MODE=WARM CN=PASSIVE IDLE;`

  `TX54 TTDS COMMAND COMPLETED` — Stops the TDS application and ensures that only passive correspondents who were not executing a transaction are reconnected at warm restart of next TDS session.

  ```
  S:
  --> TX53 TDS : TEXT SHUTDOWN
  S:
  ```

9.5.35 UNLOAD_TDS_IQS (UNLDTIQS)

Purpose:

Unloads IQS objects previously loaded into type 2 segments.

Syntax:

```
[M] { UNLDTIQS       }
    {                }
    { UNLOAD_TDS_IQS }

    OBJECT=name31

    [ TDS={ #WTDS | name4 }]
```

See *IQS-V4/TDS User's Guide*.

Example:

```
M UNLDTIQS OBJECT='TRA=Q-AFT';
```

The after transaction trigger query Q-AFT is unloaded for the default TDS #WTDS. This is issued from the master mailbox.

9.5.36 UNLOAD_TDS_MEMORY (UNLDTMEM)

Purpose:

Cancels the effects of the LOAD_TDS_MEMORY command.

Syntax:

```
[M] { UNLDTMEM          }
    {                   }
    { UNLOAD_TDS_MEMORY }

    { TPR=name12                          }
    {                                     }
    { USE_PROC=name12                     }
    {                                     }
    {{ SHRDSTO | SHARED_STORAGE }=name12  }
    {                                     }
    { IFN=name8                           }
    {                                     }
    {{ TDAT | TDS_DATA }=bool             }

    [ TDS={ #WTDS | name4 }]
```

Parameters:

TPR	Transaction program.
USE_PROC	User procedure.
SHARED_STORAGE	SHARED STORAGE area.
TDS_DATA	Data structures used by TDS.
IFN	Control structures associated with the given file.

Constraints:

Parameters are mutually exclusive.

Output:

The results of the command are reported as follows:

- `UNLOAD_TDS_MEMORY unsuccessful; rc XXXXXXXX`
- `TDS_DATA unloaded from main memory`
- `xxxxxxxxxxxx unloaded from main memory`

Example:

```
M UNLDTMEM USE_PROC=CUST-PROC TDS=PKIT;
```

Remove the user procedure named CUST-PROC from non-swappable memory.

10. Using the Programmed Operator Facility

DOF 7-PO (Distributed Operator Facility) allows the TDS application to be automatically controlled by a Programmed Operator. This facility manages exchanges, responses, or unsolicited messages using services such as data enqueueing and addressee notification.

See *DOF 7-PO User's Guide*.

10.1 Connecting and Accessing a TDS Application

To use the Programmed Operator facility, the user must be the submitter of the TDS application and no master mailbox must be declared.

Only one Programmed Operator can access a TDS application at one time. When several Programmed Operators try to access the same TDS application, only the first access request known to TDS is accepted. Otherwise the TX59 error message appears.

When starting TDS using IOF, the TDS application must be available before commands can be sent using DOF 7-PO. Otherwise an error message appears.

When starting TDS through a Programmed Operator, TDS can send unsolicited messages to the Programmed Operator without waiting for a command to be entered. These unsolicited messages are also sent to the IOF master. Once the startup sequence ends, TDS can receive and process commands coming from the Programmed Operator.

At TDS warm restart, the state of the TDS application is restored to its previous state only if the submitter of the TDS application is the same as that in the last session. If the submitter of the TDS application is the same, TDS can send unsolicited messages to the Programmed Operator without waiting for the first command.

10.2 Dialoging with a TDS Application

When using master commands, the TDS parameter is mandatory because a Programmed Operator can control several TDS applications simultaneously.

All the master commands can be used except [M] EXEC_TDS and [M] MODIFY_TX which involve send/receive conversation(s) in the transaction. The message TX66 appears.

A DOF 7-PO application can issue commands to a TDS application over only one DOF 7-PO connection. The same connection must be used until the end of the current TDS session.

10.3 Programming Restrictions

The restrictions to using this facility are:

- user transactions cannot be started
- issuing the SEND_TDS command results in the message TX66 (1)
- special-purpose transactions such as BREAK, LOGON and LOGOUT cannot be used
- master transactions started by a Programmed Operator are not automatically restartable
- a transaction cannot be spawned to a Programmed Operator session.

To view the Programmed Operator, issue [M] LIST_TDS_COR under H_PMS. No other command has an effect on this correspondent except [M] TERMINATE_TDS.

As soon as the [M] TERMINATE_TDS is issued by either the Programmed Operator or by the Master Operator under IOF, no further commands requested by the Programmed Operator can be processed by TDS. If commands are sent, the message TX66 appears.

When a Programmed Operator accesses a TDS application, the Programmed Operator must have the turn, otherwise the command is rejected. EVERY clause of $H_PMSSENDCMD must also observe this rule.

Be careful when the Programmed Operator issues [M] LIST_COR_POOL, [M] LIST_TDS_COR, [M] LIST_TDS_FILE and [M] LIST_TDS_SPAWN. The DOF 7-PO message queue may overflow, causing the transaction to abort.

11. Dump Handling and TDS Errors

When errors and/or aborts occur, debugging information is stored in system dump file SYS.SPDUMP or in SYSOUT. This section explains the types of TDS application errors, where they are stored and how they are treated.

11.1 Handling Dumps of a TDS Session

TDS dumps are stored in a system library, SYS.SPDUMP. The system library SYS.SPDUMP records TDS Session aborts in subfiles named *ron_dsn_dpindex:*

- *ron* (Run Occurrence Number) of TDS
- *dsn* (Dynamic Step Number) starting at 1 and incremented when TDS step is repeated
- *dpindex* is the Dump Index starting from 1 and incremented for the next dump.

Previous files with ifn H_DPBIN or H_DUMP are now obsolete.

All TPR aborts are written in SYSOUT.

11.1.1 Dump Operations

When a dump is taken, the following message is printed on the TDS console:

DP05 *ron_dsn_dpindex* **PGID**=*xxxx lm_name* **DUMP CREATED**

Use these commands for the following actions:

LSDP RON=*ron*|*|ALL	list list information about a dump.
DDP DUMP_ID=*dump_id*	SYS.SPDUMP subfiles.
CDP DUMP_ID=*ron*	cancel dumps created for specified RON.
CDP DUMP_ID=*ron*:*dsn*:*dpindex*	cancel a dump.

See *System Operator's Guide*.

11.1.2 Saving Binary Dump

To save a binary dump, use the JCL command DP_SAVE with the DUMP_ID parameter. After this dump is saved, the subfile is deleted in SYS.SPDUMP.

11.2 Handling TPR DUMPs

Use `[M] MODIFY_TDS DUMP` if TDS is to produce a dump for TPRs.

When a TPR terminates abnormally with return codes TIMELIM, SNDVIOL, RCVVIOL and/or RESVIOL, and `[M] MODIFY_TDS DUMP=1` has been issued, TDS produces a dump of the TPR.

There is an exception condition PROCEXP, which always produces a dump even if `[M] MODIFY_TDS DUMP=0` is issued.

The most common PROCEXP reasons are:

13-01	Caused by COBOL when an I/O request leads to abnormal condition not handled by INVALID KEY or AT END clause and no USE procedure is defined.
06-00	Out of segment bounds.
09-01	Illegal decimal data; field defined as PIC 9 contains non-decimal data.
0E-01	Fault in data descriptor; the TPR calls a subprogram not linked with it and tries to access a file not described in STDS.
11-02	Out of array range; the subscript of an array is incorrect.
0C-01	Write violation in the TPR. Ensure that the parameter order in the Procedure Division is correct.

Sometimes an exception does not occur inside a TPR but in a service called by the TPR. Such cases usually lead to abnormal TDS termination and should be reported to the Service Center.

When the same exception occurs several times in a single TPR in the same TDS session, the corresponding dump is only done once.

In this context, "same exception" means that it is the same type, has the same address, and is in the same TPR (the same SM in the same library of SMs).

In this case, the following message is sent to the JOR of the TDS:

```
WARNING DUMP ALREADY CREATED FOR THE SAME EXCEPTION AT THE SAME
ADDRESS OF THE SAME TPR
```

NOTE:

If 50 different exceptions have already resulted in 50 different dumps, each new exception different than the 50 already recorded will lead to a dump being taken each time it occurs (as is the case up to TS7356).

11.3 Handling TDS Errors

There are two type of errors, fatal and non-fatal. Some of these errors are recoverable.

A summary of both types is presented in Appendix B.

11.3.1 Fatal Abort Errors

Fatal errors cause an abort after a dump of the TDS application is taken. The abort interrupts database activity until recovery routines are run. Transaction processing is stopped until the TDS application is restarted and the network is disconnected.

Fatal errors are flagged with the Action:

None - internal error

11.3.2 Non-fatal Abort Errors

Non-fatal errors are those that can be corrected by a specified action and where a dump may be provided without TDS aborting. These errors do not have an internal error flag.

11.3.3 Recoverable Abort Conditions

A fatal error can be dynamically recovered if all the following five conditions apply:

- UFAS-EXTENDED is installed.
- The abort is a single abort. If other error conditions are simultaneously detected in a concurrent simultaneity level, no recovery is possible.
- The abort does not occur during the TDS startup.
- The abort does not occur during a dynamic recovery phase.
- The TDS application is not *killed* through CANCEL_JOB (CJ) Directive.
- A minimum number (depending on the declared number of terminals) of commitment units have been executed since the last TDS session has been started or automatically restarted.

If the fatal abort can be recovered, no dump is produced, but the TDS debugging information is logged in the SYS.SWLOG. Else, a dump is produced, the TDS session aborts, and the users are disconnected from the TDS application.

A new TDS session can be started:

- either automatically when:
  - REPEAT is specified in the JCL
  - and RESTART is set by default to 1 in [M] MODIFY_TDS
- or manually by the system console operator.

When the new session is started, users can log on again. Users may be automatically reconnected, depending on the parameters specified in [M] MODIFY_TDS_RESTART_OPTION.

11.3.4 Dynamic Recovery

When the abort is recoverable, GCOS 7 does the following:

- switches to damaged mode, preventing the scheduling of new transactions, commitment units, or TPRs
- terminates abnormally the current TPRs active in the concurrent simultaneity levels
- runs dynamic dump routines once TDS is idle, to store the debugging information in the SYS.SWLOG file
- runs standard file recovery routines where files and database areas are protected by the HIGH or MEDIUM levels of security
- restarts the processing of transactions at the commitment point where they were interrupted.

The file *tdsname*.RECOV is used to speed up recovery.

11.3.5 Program Exceptions

A program exception may occur either outside or inside a TPR enclosure.

If the exception is outside a TPR enclosure, the error is considered as a system error and the recovery mechanism is activated if the conditions are satisfied.

If a program exception occurs inside, the error is considered as local and a local dump is provided on SYSOUT even if the error is encountered within a system module invoked by the TPR.

After the dump terminats, the process is restarted and either TDS aborts the current transaction, or the user-defined *on-abort-tpr* is executed.

11.3.6 Error Logging

The information for restartable aborts is stored in the SYS.SWLOG file with an error flag. The system logs information useful in finding the cause of the problem.

A. TDS Return Codes at File Opening

The return codes at file opening are given when:

- starting a TDS application
- and issuing [M] OPEN_TDS_FILE.

The following tables list return codes (*Ret-Code*) by their values and by their mnemonics.

List of Return Codes in Order of Value

Value *Ret-Code*	Explanation/Action
0200 BUSY	File is currently assigned to another job.
0460 CONFLICT	Internal error in the file's characteristics.
0900 NASSIGND	No ASSIGN for internal file name specified in [M] OPEN_TDS_FILE.
OA02 DVNAV	Specified device is not available - ensure syntax is correct.
0A03 MDNAV	Media is not available - issue CR MS*nn*.
0A06 FLNAV	File must be recovered before it can be opened. For IDS/II areas, the return code corresponds to one of the following: – the assigned area is not an IDS/II area – the assigned area name is not the name declared for this *ifn* – the user label of the file is invalid – the schema date is different from the sarea date. See the JOR for further explanations.

0A10 IFNNASG	This file was not assigned.
0A17 DATANAV	File data is lost due to a crash at creation.
1007 EFNUNKN	External file name is unknown - ensure syntax is correct.
1208 DVOV	Device overflow - too many devices indicated.
1219 TABOV	Either a TDS-controlled file is not UFAS-EXTENDED, or there is an internal error.
1223 ENTRYOV	Entry overflow - number of entries exceeds limit, retry after issuing [M] CLOSE_TDS_FILE DEASSIGN=1.
1462 SHLVVIOL	SHARE=MONITOR is not specified for a TDS-controlled file.
1501 ARVIOL	The requestor has not sufficient access rights on file specified. For more details on access rights, refer to the Data Security Facilities User's Guide.
1800 DVCERR	Device-class error - ensure syntax is correct.
1806 OPTERR	Missing or inadequate DEFINE statement. Required file integrity level cannot be obtained. No dynamic corrective action may be performed; include the DEFINE statement in next TDS session.
1820 EXTERR	Volume(s) of a multivolume file is missing.
1821 IFNERR	The syntax of the internal file name may be incorrect, or there may be an internal error.
1898 KEYERR	Source program record key and PREALLOC record key do not match.
1C02 WRONGORG	Error in file organization - usually non-UFAS-EXTENDED file was specified as a TDS-controlled file.
1C07 OPEN	File is already opened.

List of Return Codes in Alphabetical Order

Ret-Code	*Value*	Explanation/Action
ARVIOL	1501	The requestor has no sufficient access rights on file specified.
BUSY	0200	File is currently assigned to another job.
CONFLICT	0460	Internal error in the file's characteristics.
DATANAV	0A17	File data is lost due to a crash at creation.
DVCERR	1800	Device-class error - ensure syntax is correct.
DVNAV	OA02	Specified device is not available - ensure syntax is correct.
DVOV	1208	Device overflow - too many devices indicated.
EFNUNKN	1007	External file name is unknown - ensure syntax is correct.
ENTRYOV	1223	Entry overflow - number of entries exceeds limit, retry after issuing [M] CLOSE_TDS_FILE DEASSIGN=1.
EXTERR	1820	Volume(s) of a multivolume file is missing.
FLNAV	0A06	File must be recovered before it can be opened. For IDS/II areas, the return code corresponds to one of the following: – the assigned area is not an IDS/II area – the assigned area name is not the name declared for this *ifn* – the user label of the file is invalid – the schema date is different from the sarea date. See the JOR for further explanations.
IFNERR	1821	The syntax of the internal file name may be incorrect, or there may be an internal error.
IFNNASG	0A10	This file was not assigned.
KEYERR	1898	Source program record key and PREALLOC record key do not match.

MDNAV	0A03	Media is not available - issue CR MS*nn*.
NASSIGND	0900	No ASSIGN for internal file name specified in [M] OPEN_TDS_FILE.
OPEN	1C07	File is already opened.
OPTERR	1806	Missing or inadequate DEFINE statement. Required file integrity level cannot be obtained. No dynamic corrective action may be performed; include the DEFINE statement in next TDS session.
SHLVVIOL	1462	SHARE=MONITOR is not specified for a TDS-controlled file.
TABOV	1219	Either a TDS-controlled file is not UFAS-EXTENDED, or there is an internal error.
WRONGORG	1C02	Error in file organization - usually non-UFAS-EXTENDED file was specified as a TDS-controlled file.

B. TDS Error Messages

This Appendix lists TDS error messages in numerical order. All error messages are reported in the JOR and some appear on the screen.

A) TDS error messages displayed on the screen can be in one of three forms:

```
tdsname EMERGENCY SHUTDOWN REASON numeric-code
message-text
tdsname RESTARTABLE ABORT REASON prefix-numeric-code
msg-text
tdsname WARNING REASON numeric-code message-text
```

where

- *tdsname* identifies the TDS application in the PROGRAM-ID clause in TP7GEN.
- EMERGENCY SHUTDOWN is fatal: the TDS application has terminated abnormally.
- *prefix* can either be MU or MV.
- *numeric-code* is the message number.
- *message-tex*t is the contents of the TDS message.
- RESTARTABLE ABORT is fatal but the TDS application restarts automatically if UFAS-EXTENDED is used.
- WARNING does not prevent a TDS application from executing.

B) Error messages printed in the JOR can be in one of two forms:

```
WARNING hhmmss MUxx. message-text
FATAL hhmmss MUxx. message-text
```

where

- *WARNING* and **FATAL** are the same as described for error messages displayed on the screen.
- *hhmmss* is the time in hours, minutes, and seconds when the error occurred.
- the prefix MU or MV always appears both in the JOR and on the screen.

The table overleaf is a resume of error messages showing:

- the type of error message (*Warning* or **Fatal**)
- whether a dump or a logging mechanism is activated or not (dump=yes/no)

dump=yes means that:
- in case of a non recoverable abort, a dump is produced.
- in case of a recoverable abort, no dump is produced and the error is logged in the SYS.SWLOG file. Refer to the chapter 11.3 *Recoverable Abort Conditions*.

- which operators receive the error message.

B.1 MU and MV Error Messages

Code	Type	DUMP	Operator	Code	Type	DUMP	Operator
MU00	**Fatal**	no	Main	MU50	**Fatal**	yes	Master
MU01	**Fatal**	no	Main	MU51	**Fatal**	yes	Master
MU02	**Fatal**	yes	Main	MU52	*Warn*	no	Main
MU03	**Fatal**	no	Master	MU53	*Warn*	no	Main + Master
MU04	**Fatal**	yes	Master	MU54	**Fatal**	yes	Master
MU05	**Fatal**	yes	Master	MU55	*Warn*	no	Master
MU06	**Fatal**	yes	Master	MU56	*Warn*	yes	Master
MU07	**Fatal**	yes	Master	MU57	*Warn*	no	Main
MU08	*Warn*	no	Main	MU58	**Fatal**	yes	Master
MU09	*Warn*	no	Master	MU59	**Fatal**	yes	Master
MU10	**Fatal**	yes	Master	MU60	*Warn*	no	Main
MU11	*Warn*	yes	Master	MU61	**Fatal**	yes	Master
MU12	**Fatal**	yes	Main	MU62	*Warn*	no	Main + Master
MU13	**Fatal**	no	Main	MU63	**Fatal**	yes	Main + Master
MU14	**Fatal**	no	Main	MU64	**Fatal**	yes	Main + Master
MU15	**Fatal**	no	Main	MU65	**Fatal**	yes	Main + Master
MU16	**Fatal**	no	Main	MU66	**Fatal**	yes	Main + Master
MU17	**Fatal**	no	Master	MU67	*Warn*	no	Main + Master
MU18	**Fatal**	yes	Master	MU68	**Fatal**	yes	Main + Master
MU19	**Fatal**	yes	Master	MU69	**Fatal**	yes	Master
MU20	**Fatal**	no	Main + Master	MU70	**Fatal**	yes	Master
MU21	*Warn*	no	Master	MU71	**Fatal**	no	Main
MU22	*Warn*	no	Master	MU72	**Fatal**	yes	Master
MU23	*Warn*	no	Main	MU73	**Fatal**	yes	Main + Master
MU24	*Warn*	no	Main	MU74	*Warn*	yes	Main + Master
MU25	*Warn*	no	Main	MU75	*Warn*	no	Main + Master
MU26	*Warn*	no	Main	MU76	*Warn*	yes	Main + Master
MU27	*Warn*	no	Main + Master	MU77	*Warn*	no	Main + Master
MU28	*Warn*	no	Master	MU78	*Warn*	no	Main + Master
MU29	*Warn*	no	Main	MU79	*Warn*	no	Main + Master
MU30	**Fatal**	yes	Master	MU80	**Fatal**	yes	Master
MU31	*Warn*	no	Master	MU81	**Fatal**	yes	Master
MU32	*Warn*	no	Master	MU82	**Fatal**	yes	Master
MU33	*Warn*	no	Master	MU83	**Fatal**	yes	Master
MU34	*Warn*	no	Master	MU84	**Fatal**	no	Master
MU35	*Warn*	no	Master	MU85	*Warn*	no	Main
MU36	*Warn*	no	Main	MU86	**Fatal**	yes	Master
MU37	**Fatal**	no	Main	MU87	**Fatal**	yes	Master
MU38	**Fatal**	yes	Main + Master	MU88	*Warn*	no	Master
MU39	*Warn*	yes	Master	MU89	*Warn*	no	Main + Master
MU40	*Warn*	no	Master	MU90	**Fatal**	no	Master
MU41	*Warn*	yes	Master	MU91	*Warn*	no	Main
MU42	**Fatal**	yes	Main + Master	MU92	**Fatal**	yes	Main
MU43	*Warn*	no	Main	MU93	**Fatal**	no	Master

Code	Type	DUMP	Operator	Code	Type	DUMP	Operator
MU44	*Warn*	yes	Main + Master	MU94	*Warn*	no	Main + Master
MU45	**Fatal**	yes	Master	MU95	*Warn*	no	Main + Master
MU46	*Warn*	no	Main + Master	MU96	*Warn*	yes	Master
MU47	**Fatal**	yes	Main + Master	MU97	*Warn*	yes	Master
MU48	**Fatal**	yes	Master	MU98	*Warn*	yes	Master
MU49	**Fatal**	yes	Master	MU99	**Fatal**	yes	Main
MV00	**Fatal**	no	Main	MV25	**Fatal**	yes	Main
MV01	*Warn*	yes	Master	MV26	*Warn*	no	Main
MV02	*Warn*	no	Main + Master	MV27	**Fatal**	yes	Main + Master
MV03	*Warn*	yes	Master	MV28	*Warn*	no	Main + Master
MV04	*Warn*	no	Main + Master	MV29	**Fatal**	no	Main
MV05	**Fatal**	no	Main + Master	MV30	**Fatal**	no	Main
MV06	*Warn*	no	Master	MV31	*Warn*	no	Main + Master
MV07	*Warn*	no	Master	MV32	Info	no	Main + Master
MV08	*Warn*	no	Master	MV33	*Warn*	no	Main + Master
MV09	**Fatal**	yes	Master	MV34	*Warn*	no	Main + Master
MV10	**Fatal**	no	Main	MV35	*Warn*	no	Main + Master
MV11	*Warn*	yes	Main + Master	MV37	*Warn*	no	Main + Master
MV12	*Warn*	no	Main + Master	MV39	Warn	no	Main
MV13	*Warn*	no	Main	MV40	**Fatal**	yes	Main + Master
MV14	*Warn*	no	Main	MV41	**Fatal**	yes	Main
MV15	*Warn*	no	Main	MV60	**Fatal**	yes	Master
MV16	*Warn*	no	Main + Master	MV61	Warn	no	Main + Master
MV17	*Warn*	no	Main + Master	MV62	**Fatal**	no	Master
MV18	*Warn*	no	Main	MV63	Warn	no	Main + Master
MV19	*Warn*	yes	Main + Master	MV70	Warn	no	Master
MV20	**Fatal**	yes	Main + Master	MV79	Warn	no	Main + Master
MV21	**Fatal**	no	Main	MV80	Warn	no	Main + Master
MV22	**Fatal**	no	Main				
MV23	*Warn*	no	Main				
MV24	**Fatal**	no	Main				
MV38	*Warn*	no	Main + Master				
MV42	*Warn*	no	Main + Master				

MU00. CONNECTION TO COMMUNICATIONS NOT ESTABLISHED.

Meaning: When the return code is TP7 11,ALREADY, the meaning of the message is that one attempts to launch an already running TDS.

Action: When the return code is TP7 11,ALREADY there is nothing to do. Else run the NETGEN utility before starting the TDS session.

MU01. OMH COMMAND FOR MASTER OF TDS CANNOT BE CREATED.

Action: Ensure that *tdsname* is not the same as a command name. Delete dynamic commands created by other jobs. Consult the Service Center.

MU02. UNSUCCESSFUL TDS TABLE ENTRY CREATION.

Meaning: If the return code is USERUNKN, ensure that the TDS was not started by a user under the OPERATOR project with the Station attribute whose value is a "site name" (not MAIN). For more information, see the paragraph 5.3 "Starting a TDS Session". In other cases, this is an internal error.

MU03. UNABLE TO SEND MESSAGE TO MASTER.

Action: None - internal error.

MU04. A TDS TASK HAS NOT BEEN INITIATED.

Action: None - internal error.

MU05. CANNOT ENTER IN MONOSIMULTANEITY PROCESSING.

Action: None - internal error.

MU06. ERROR WHILE RETRIEVING RECORD FROM TDS SYSTEM FILE.

Action: None - internal error.

MU07. ERROR WHILE WRITING RECORD TO TDS SYSTEM FILE.

Action: None - internal error.

MU08. ERROR WHILE ACCESSING RECORD IN TDS SYSTEM FILE.

Action: None - internal error.

MU09. UNEXPECTED AFTER IMAGE FILE FROM AFTER JOURNAL. PRINT SYS.JADIR

Meaning: Error during dynamic rollforward.

Action: Print the SYS.JADIR directory.

MU10. NO LINK TO USER JOURNAL UCA.

Meaning: No journal was defined at TDSGEN, but the job description includes ASSIGN USERJRNL, DUMMY.

Action: Remove this JCL statement and restart the TDS application.

MU11. ILLEGAL TYPE OF RECORD.

Action: None - internal error.

MU12. UNABLE TO OPEN SYSOUT.

Action: None - internal error.

MU13. UNSUCCESSFUL WRITE ON AFTER JOURNAL.

Action: None - internal error.

MU14. UNABLE TO OPEN BEFORE JOURNAL.

Action: None - internal error.

MU15. UNABLE TO INITIALIZE AFTER JOURNAL FOR USER JOURNAL USE.

Action: Ensure that there is at least one file protected by After Journal in the TDS. If not, add to the JCL:

DEFINE USERJRNL,JOURNAL=AFTER;

MU16. MASTER MAILBOX NOT OPENED.

Action: Ensure that the master mailbox name is cataloged in the APPLIST parameter of the SITE.CATALOG.

MU17. TDS MAILBOX NOT OPENED.

Action: Ensure that the tdsname is cataloged in the APPLIST parameter of the SITE.CATALOG.

MU18. UNSUCCESSFUL CLOSE OF AFTER JOURNAL.

Action: None - internal error.

MU19. UNSUCCESSFUL CLOSE OF BEFORE JOURNAL.

Action: None - internal error.

MU20. PLEASE CHECK M.I.

Meaning: The product (Marketing Identifier) has not been purchased for the site.

Action: Verify if the product was purchased or not.

MU21. ERROR WHILE READING DEBUGFILE.

Action: None - internal error.

MU22. STATIC/DYNAMIC ASSIGN CONFLICT ON IFN:*xxxxxxxx* EFN:*xxxxxxxxxxxxxxxx*.

Meaning: There is a conflict at TDS warm restart between the ASSIGN statement of the job description and the dynamic file assignment of the previous TDS session (external file names or catalogs are not the same as those assigned dynamically). Access to the file is denied.

MU23. CANNOT RESTORE POOL OF SESSIONS.

Action: Reissue [M] OPEN_COR_POOL or the [M] MODIFY_COR_POOL to reestablish the pool of sessions.

MU24. CANNOT RESTORE COMMON STORAGE AT RESTART.

Action: None. The common-storage is reset to zero.

MU25. CANNOT ACTIVATE SPAWNING TRANSACTION AT RESTART.

Action: None - internal error. The spawned transactions are lost.

MU26. CANNOT SWITCH SHARABLE MODULE LIBRARIES AT RESTART.

Meaning: This warning message may appear when a switching of sharable module libraries (MDTDS SMLIB=(.....,....) command) has been issued during any TDS session following the last TDS cold restart.

It means that an abnormal condition has been detected at TDS warm restart when trying to restore the order of sharable module libraries (such as those stored at the end of the previous TDS session).

Action: Check that at least one module of each sharable module library is present, using a DC BS=SM command.

Load (if necessary) unloaded modules (at least one) using SYSMAINT or MNSYS processor.

Enter a new MDTDS SMLIB=(.....,....) command.

MU27. SWAP DIRECTORY NOT AVAILABLE.

Meaning: An error has occurred at TDS warm restart; TDS has overridden the swap directory *tdsname*.RECOV; the TDS application proceeds normally.

MU28. CANNOT ENTER CORRESPONDENT xxxxxxxxxxxx IN TDS TABLE.

Action: None - internal error.

MU29. COLD RESTART FORCED DUE TO UNSUCCESSFUL FILE RECOVERY.

Action: Check the GCOS 7 restart conditions. Some files may be inconsistent as TDS recovery procedures do not occur at GCOS 7 warm restart.

MU30. PROCESS EXCEPTION INSIDE TDS CODING.

Action: Reload TDS, internal error within TDS or UFAS-EXTENDED.

MU31. DEBUG FILE CANNOT BE OPENED.

Action: Ensure that the ASSIGN DBUGFILE statement in the TDS step enclosure is correct. A detailed reason will be given in the TDS JOR.

MU32. ERROR WHILE OPENING DEBUG SUBFILE.

Action: None - internal error. Return code is displayed at the user terminal.

MU33. ERROR WHEN CLOSING DEBUG SUBFILE.

Action: None-internal error reported but does not prevent TDS from executing.

MU34. ERROR WHILE WRITING TO DEBUG FILE.

Action: None-internal error reported but does not prevent the TDS application from executing. The return code is displayed on the user terminal. The DEBUG file is probably full.

MU35. ERROR WHILE SENDING FOR TRACES.

Action: None-internal error reported but does not prevent the TDS application from executing.

MU36. CANNOT CLEAR SYSTEM FILE AT COLD RESTART.

Action: None-internal error which must be reported.

MU37. TDS APPLICATION NOT GENERATED.

Meaning: The TDS application is not generated.

Action: Run the TP7GEN utility.

MU38. ERROR WHILE ADDRESSING FILE STRUCTURES.

Action: If a hyphen is used in an internal file name specified at TP7GEN, ensure that a reserved word follows the hyphen.

MU39. ERROR DETECTED BY IDS2 ACCESS METHOD.

Action: None - internal error.

MU40. CANNOT CREATE MEMORY AREAS.

Action: None - internal error.

MU41. TDS-IDS INTERFACE ERROR.

Action: None - internal error.

MU42. SEGMENT SIZE CANNOT BE INCREASED OR CREATED.

Action: None - internal error.

MU43. INIT PROCEDURE NOT FOUND.

Meaning: The procedure for transaction initialization was not found as requested in TP7GEN.

Action: Check the TP7GEN report file and re-run the TP7GEN utility to include this procedure in the TDS load module. This error does not prevent the TDS application from executing.

MU44. UNABLE TO GET ADDRESSABILITY TO TDS ERROR PROCEDURE.

Action: None - internal error.

MU45. ERROR WHILE READING FILE CONTROL STRUCTURE.

Action: None - internal error.

MU46. *xxxxxxxx* SCHEMA STRUCTURES NOT AVAILABLE.

Action: Ensure that the named schema is correctly declared. If necessary, resubmit TP7GEN to correct the error. When the USE IDS-SUB-SCHEMA Clause is used in the STDS, ensure that the schema corresponds to a Full-IDS/II schema.

MU47. ABORT DURING WARM RESTART. NEXT RESTART IS COLD.

Action: None. The system encountered a fatal error during the recovery processing and restarts cold without shutdown.

MU48. INTERNAL ERROR IN TDS - COMMIT-MANAGER.

Action: None - internal error.

MU49. CANNOT CREATE IDS FD.

Action: None - internal error.

MU50. CANNOT BUILD ROLLFORWARD CONTROL SEGMENT.

Action: None - internal error.

MU51. UNSUCCESSFUL REBUILD OF TERMINAL TABLE AT RESTART.

Action: Run the TP7GEN utility and perform a cold restart.

MU52. ***xxxxxxxx*** **ASSIGNMENT IMPOSSIBLE AT RESTART.**

Action: Reload TDS and perform a warm restart with all available used files. If unsuccessful, perform a cold restart.

MU53. **TERMINAL WRITER STRUCTURE INITIALIZATION FAILED.**

Action: None - internal error. The GTWriter is not available for the TDS session.

MU54. **INTERNAL ERROR DURING CLOSING PROCESS.**

Action: None - internal error.

MU55. **USER JOURNAL MISSING - USED IN PREVIOUS SESSION.**

Action: Insert the ASSIGN USERJRNL, DUMMY; statement in the job description; perform a warm restart. Otherwise, perform a TDS cold start with the same JCL.

MU56. **UNEXPECTED CALL TO FILE ERROR HANDLER.**

Action: None - internal error. A dump is provided. The TDS session resumes.

MU57. **CANNOT COMPLETE FILE RECOVERY.IFN =** ***xxxxxxxx*****.**

Action: None-internal error which does not prevent the TDS application from executing. Ensure that the specified ifn is available and perform the recovery procedures if necessary.

MU58. **INTERNAL ERROR IN M TRANSACTION.**

Action: None - internal error.

MU59. **INTERNAL ERROR IN VCAM INTERFACE [VCAM STATUS (HEXA)=*xx*].**

Action: None - internal error.

MU60. **DEFINE EXISTS BUT JOURNAL NOT DECLARED.**

Action: Remove the ASSIGN USERJRNL, DUMMY; statement or regenerate the TDS application with user journalization.

MU61. **ERROR IN FORM ACTIVATION.**

Action: None - internal error.

MU62. H_SWAP*xx* FILE NOT INITIALIZED.

Meaning: Depends on the return code displayed at the console. Does not prevent a TDS application from executing if there is more than one swap file. The return code TP7 13, FLNAV means that the <tdsname>. RECOV or the <tdsname>.SWAP file location have changed since the last TDS session.

The location of a <tdsname>.SWAP (or of the <tdsname>.RECOV) file within the media may have changed due to the DELETE_FILE/BUILD_FILE commands or the SAVE_DISK/RESTORE_DISK (with the REORG option) commands.

Action: Run the TP7GEN utility.

MU63. ERROR DETECTED WHILE ACCESSING TDS SWAP FILE.

Meaning: This message can concern the tdsname.RECOV file, or the *tdsname*.SWAP file.

Action: None - internal error.

MU64. IO FAILURE ON SWAP FILE (C.CODE=*xxxxxxx*). SEARCH ADR+CCCC/TT/RR.

Meaning: Error detected on I/O. H_SWAPxx is the ifn of the swap file. C.CODE is the I/O completion code. CCC/TT/RR is the (decimal) address of the last searched cylinder/track/record.

Action: See PRLOG for reasons.

MU65. NO SWAP AVAILABLE FOR TDS.

Meaning: This message concerns the <tdsname>.RECOV file or the tdsname.SWAP file.

Action: Run the TP7GEN utility.

MU66. SWAP CAPACITY NOT COMPATIBLE WITH TDS TABLE.

Action: None - internal error.

MU67. SWAP FILE TOO SMALL-*xxxx* SESSIONS MAX-*xxx* CYL MISSING.

Meaning: This message appears only for VBO swap files which are not large enough to support the generated configuration.

NOTE:

The SESSION MAX number gives the maximum number of contexts that can be allocated.

One swap context is needed for each correspondent (TM, XCP2 and VIRTUAL terminals) and for each session of a POOL (XCP1 and DUMMY)

Furthermore, for each correspondent, several swap contexts may be needed. For each occurrence of a special purpose transaction such as *break* and *logon*, a swap context is allocated and kept until the end of the transaction.

A correspondent abnormally disconnected will keep all its previous allocated swap contexts.

Action:
- Either reallocate a new file with TP7PREP
- or update the Network Generation with NETGEN.

MU68. TIMER REQUEST FAILED.

Action: None - internal error.

MU69. TDS PROTOCOL ERROR.

Action: None-internal error in implementing the cooperative application protocol.

MU70. TDS TABLE INCONSISTENCY.

Action: None - internal error.

MU71. PMOS DYNAMIC COMMAND FOR MASTER CANNOT BE CREATED.

Action: Ensure that the tdsname is not the same as a command name. Delete dynamic commands created by other jobs. Consult the Service Center.

MU72. ERROR IN STATIC ASSIGN:
NO FILE MUST BE ASSIGNED IN THE JCL TP7JCLBAC.

Action: None - internal error.

MU73. INTERNAL ERROR IN UNLOCKING USER RESOURCES.

Action: None - internal error.

MU74. TDS WARNING ***DDM00001 *xxxxxxxxxx//xxxxxxxxxx* RC=*xxxxxxxx*.

Meaning: An error detected by IDS is reported to TDS, a dump is taken. Refer to *Messages and Return Codes Directory* (class DDM0001).

NOTE:
The whole text may be edited by several consecutive MU74 messages.

Action: None - internal error.

MU75. NO MORE TABLE SPACE FOR TRANSACTION STACKING.

Meaning: A transaction is prevented from executing due to lack of TDS table space.

Action: Increase the value specified in the NUMBER OF TERMINALS clause or the TMSESS clause in NETGEN.

MU76. NO MORE SWAP SPACE FOR TRANSACTION STACKING.

Meaning: The TDS swap file is full. This is detected when processing special purpose transactions such as *break* and *logon* for which the allocation of a new swap context is requested.

Action: Increase the TDS swap size by executing the TP7PREP utility and re-run TP7GEN.

MU77. CANNOT START IQS SESSION FOR VIEW=*xxxxxxxx*.

Action: Refer to *IQS/TDS User's Guide*.

MU78. CANNOT INITIALIZE IQS SESSION FOR FILE=*xxxxxxxx*.

Action: Refer to *IQS/TDS User's Guide*.

MU79. ERROR ON PMOS COMMAND RECEIPT.

Meaning: Interface error concerning a DOF 7-PO command.

Action: Consult the Service Center. Use [M] SEND_TDS command to enter the command in line mode.

MU80. ERROR DETECTED WHILE DYNAMICALLY ASSIGNING TDS SYSTEM FILE.

Action: None - internal error.

MU81. ERROR DETECTED WHILE OPENING TDS SYSTEM FILE.

Action: None - internal error.

MU82. ERROR DETECTED WHILE CLOSING TDS SYSTEM FILE.

Action: None - internal error.

MU83. UFAS/GAC OR COMMIT-MANAGER CONTROL STRUCTURES DAMAGED.

Action: None - internal error. A recovery procedure will dynamically restart the system with no shutdown of the network.

MU84. ABORT DUE TO SWAP REORGANIZATION, NEXT RESTART WILL BE COLD.

Meaning: This message appears the first time a TDS application is run after UFAS-EXTENDED has been installed at the site and the TP7GEN utility was not run.

Action: Run the TP7GEN utility.

MU86. ORACLE INITIALIZATION FAILED.

Meaning: The SM containing ORACLE is not loaded; or the correct library is not specified in the JOBLIB statement.

Action: Ensure that the SM containing ORACLE is loaded; if not, ensure the library in the JOBLIB statement is correct.

MU87. ERROR DETECTED IN ORACLE/TDS PROTOCOL.

Action: None - internal error

MU88. UNSUCCESSFUL RELEASE OF DYNAMIC PMOS COMMAND.

Meaning: Internal error at job termination.

Action: None.

MU89. SWAP FILE TOO SMALL-*xxxx* SESSIONS MAX-*xxxxxx* BLOCKS MISSING.

Meaning: This message appears only for FBO swap files which are not large enough to support the generated configuration.

NOTE:

The SESSION MAX number gives the maximum number of contexts that can be allocated.

One swap context is needed for each correspondent (TM, XCP2 and VIRTUAL terminals) and for each session of a POOL (XCP1 and DUMMY)

Furthermore, for each correspondent, several swap contexts may be needed. For each occurrence of a special purpose transaction such as *break* and *logon*, a swap context is allocated and kept until the end of the transaction.

A correspondent abnormally disconnected will keep all its previous allocated swap contexts.

Action: *The number of missing blocks to support the configuration and the number of possible users for the file* enable the administrator to re-allocate a new file by either using TP7PREP (updating STDS if necessary) and then resubmitting TP7GEN or just modifying NETGEN.

MU90. TP7GEN NECESSARY DUE TO UNSUCCESSFUL SYSTEM FILE RECOVERY.

Meaning: After a GCOS 7 clean restart, some TDS system files may be inconsistent because TDS recovery procedures were not performed.

Action: Run the TP7GEN utility.

MU91. XCP2 SERVICE UNAVAILABLE - INTERNAL STATUS=VVV-WWW[-XXX-YYY-ZZZ].

Where:

vvv is the sicid of the TP7 procedure that detects the error.

www is an internal error number in the procedure.

xxx and yyy are the two statuses PPC_CODE and PPC_SUBCODE described in the STATUS SYMBOLIC VALUE paragraph in the documentation *CPI-C/XCP2 User's Guide (47 A2 14UT).*

zzz is an internal complementary code for debugging purpose.

Action:

Ensure that:
- an XCP2WKS was declared for the TDS application in the NETGEN service
- the JPPC job is running
- the XCP2 option was specified at preparation time.

MU92. SYSTEM ERROR IN XCP2 SERVICE - INTERNAL STATUS = *xxx-xxx*.

Action: None - internal error.

MU93. ERROR ON SYSTEM FILE PROTECTION.

Action: Ensure that the TDS system file (CTLM) is protected by the Before Journal.

MU94. SWAP FILE TOO BIG: SWAP xx NOT USED.

Action: Reduce the size of the swap files. Try to balance the size of each swap file. This message may appear several times with a different number of swap files.

MU95. INCOMPLETE ROLLFORWARD.
RECOVER FILES PROTECTED WITH AFTER JOURNAL.

Action: Perform the recovery procedures for the files which are used with the After Journal.

MU96. UNSUCCESSFUL RELEASE OF MEMORY.

Action: None-internal error which does not prevent TDS from executing.

MU97. UNSUCCESSFUL REQUEST OF MEMORY.

Action: None-internal error which does not prevent TDS from executing.

MU98. TDS SIMULTANEITY SYNCHRONIZATION ERROR.

Action: None-internal error which does not prevent TDS from executing.

MU99. ERROR IN ADMINISTRATION OF TDS-HA

Meaning: Internal protocol error in the HA context. For example, the TDS was declared WATCHED BY CMSC but was not started by CMSC.

Action: If the cause of the error cannot be found, contact the Service Center.

MV00. UNABLE TO ACCESS TO SECURITY PROCEDURES.

Meaning: On a site declared with the SECUR'ACCESS option, the SM containing the SECUR'ACCESS TPR is not present; or the library containing the SM is not specified properly in the JOBLIB statement.

Action: Ensure that the SM containing the SECUR'ACCESS TPR is present; otherwise ensure that the library containing this SM is actually specified in the JOBLIB statement.

MV01. DPF SYSTEM ERROR.

Meaning: Internal error in DPF.

Action: Consult the Service Center.

MV02. UNABLE TO SEND A CATALOGED MESSAGE OR PMOS RESPONSE.

Meaning: TDS fails sending a message to the master using DOF 7-PO.

Action: Consult the Service Center.

MV03. SUPERVISION INITIALIZATION FAILED AT RESTART.

Meaning/Action: A system error prevents the supervision facility from starting, but does not prevent TDS from executing.

MV04. UNABLE TO TRANSLATE A MNMC/PMOS COMMAND INTO TDS COMMAND.

Meaning: This incident does not prevent the command being executed.

Action: Consult the Service Center.

MV05. SYSTEM ERROR IN PROCESS.

Action: None - internal error.

MV06. NO MORE SUBFILES AVAILABLE IN DEBUG FILE FOR PRINT OPTION.

Meaning: There are too many subfiles being used with the trace mechanism. The limit is 100 subfiles.

Action: Reduce the number of subfiles.

MV07. NO MORE SUBFILES AVAILABLE IN DEBUG FILE FOR PCF OPTION.

Meaning: There are too many subfiles being used with the trace mechanism. The limit is 100 subfiles.

Action: Reduce the number of subfiles.

MV08. TDS { WARM|COLD } RESTART. TS = *xxxx*

Meaning: This message appears only in the JOR and indicates at what time the TDS application started and the current Technical Status (XXXX).

MV09. GAC INITIALIZATION FAILED.

Meaning: The TDS application cannot start.

Action: Consult the Service Center.

MV10. JOURNAL INITIALIZATION FAILED

Meaning: An internal error prevents the TDS application from starting.

Action: Consult the Service Center.

MV11. ERROR WHILE WRITING TO SECURITY LOG FILE

Meaning: An internal error occurs while TDS is recording information in the Log C2 file.

Action: Consult the Service Center.

MV12. JPPC SERVICE UNAVAILABLE OR OVERLOADED; IQS RUNS IN DEGRADED MODE

Meaning: This is a warning message. IQS (only version 41) cannot use the JPPC service, it uses the GMEM file.

Action: Verify that the JPPC is not accessible.

MV13. TPR NOT INCLUDED

Action: None-internal error.

MV14. UNABLE TO CREATE A CACHE FOR CONTEXTS

Action: None - internal error.

MV15. UNABLE TO EXPAND THE CACHE OF CONTEXTS

Action: None-internal error.

MV16. CANNOT CLOSE ALL XCP2 POOLS, RETRY LATER OR SUBMIT TTDS STRONG

Meaning: This is a warning message. A protocol problem prevents the normal ending of a TDS application using XCP2 after a TERMINATE_TDS command.

Action: Try the TTDS command later or issue with the STRONG option.

MV17. INCONSISTENCY IN XCP2 SERVICE AT WARM RESTART – COLD RESTART MANDATORY

Meaning: This is a warning message. A protocol or configuration problem prevents the normal start of the XCP2 service.

Action: Restart TDS in cold mode.

MV18. MLDS FILE NO MORE SUPPORTED IFN=*xxxxxxxx*

Meaning: MLDS files are not supported in this release of GCOS 7.

Action: None possible.

MV19. H_SWAPxx ERROR DETECTED BY TDS:
EXPECTED ADR=*cccc/tt/rr* READ=*cccc/tt/rr*

Meaning: H_SWAPxx is the ifn of the swap file. No error has been detected by the I/O interface, but the address read form the controller is different from the computed address. CCCC/TT/RR is Cylinder, Track, and Record information.

Action: None.

MV20. JPPC SERVICE UNAVAILABLE OR OVERLOADED FOR XCP2

Meaning: Fatal, dump. The capacity of the JPPC is not sufficient to start the TDS using XCP2.

Action: Verify the configuration of the application requested segments from the JPPC job or increase its configuration.

MV21. TP7GEN MAY BE NECESSARY AFTER A CHANGE OF TECHNICAL STATUS

Meaning: The application has been generated on a previous release not compatible with the current release.

Action: Regenerate the application with the TP7GEN utility.

MV22. WRONG DEFINE STATEMENT FOR H_SWAPxx. RC=DFASG 39, OVRVIOL

Meaning: A wrong DEFINE statement concerning the swap file xx is present in the JCL for the TDS job.

Action: Modify the JCL for the TDS job. When working with the USER JOURNAL check that the TDS-V5 statement:
DEFINE H_SWAP[nn],JOURNAL=AFTER;
has been replaced by the new TDS-V6/V7 statement:
ASSIGN USERJRNL,DUMMY;

MV23. START UP NOT FINISHED (CHECK MASTER CONNECTION)

Meaning: The CMSC command (used to manage the TDS service, for example TSRV=TERMINATE_SERVICE) is postponed since the startup has not completed.

Action: Verify that the master operator is well connected and wait until the startup terminates.

MV24. UNDEFINED NUMBER OF TERMINALS – CHECK THE NETWORK GENERATION (OR TDSGEN)

Meaning: The number of terminals is not declared (neither in the network generation nor in the NUMBER OF TERMINALS clause). TDS cannot be run.

Action: Declare the number of terminals in the NUMBER OF TERMINALS clause. See Section 3 *Generating a TDS Application*.

MV25. TDS INITIALIZATION FAILED. RETRY LATER. RC=*xxxxxxxx*

Meaning: The TDS cannot be started because of missing resources. If this message appears just after the message MU83, this means that the TDS cannot be started because of a GAC reinit in progress.

Action: Retry later. Refer to *GAC-EXTENDED User's Guide* and *Messages and Return Codes Directory* according to the return code indicated in the message.
If necessary, reconfigure the TDS environment.

MV26. UNABLE TO CONNECT MASTER TERMINAL, STARTUP ON DUMMY SESSION.

Meaning: In an HA environment, at restart after takeover, the startup transaction has been executed on a dummy session because the reconnection to the Master Terminal was unsuccessful.

Action: Start a dummy session.

MV27. SWAP INCONSISTENCY DETECTED, NEXT RESTART WILL BE COLD.

Meaning: Fatal error detected on context processing at TDS warm restart.

Action: None. Next restart will be forced at COLD.

MV28. ***xxxxxxxx*** **SCHEMA STRUCTURES FOUND UNAVAILABLE SINCE TDS GENERATION**

Meaning: Initialization of IDS schema structure has failed; this schema cannot be accessed in this TDS and other sessions:
- on first startup of TDS after TP7GEN; the return code gives the reason
- this warning message is sent at each startup of the NEXT TDS session; return code is TP7 11, NOTDONE.

Action: Perform the following:
- correct the original incident to make all databases accessible
- run TP7GEN, startup the TDS session and then verify that no further MV28 messages appear.

MV29. **TOO MANY DUMMY/VIRTUAL CORRESPONDENTS FOR NUMBER OF TERMINALS.**

Meaning: The total number of virtual correspondents in all families and the maximum number of DUMMYs exceeds the number specified in the NUMBER OF TERMINALS Clause (if this clause is not overridden by a NETGEN directive).

Action: Increase the number of terminals, or decrease the number of virtual correspondents, or use the NETGEN directive.

MV30. **ERROR WHEN ALLOCATING OR CONNECTING DUMMY/VIRTUAL CORRESPONDENTS.**

Meaning: TDS cannot connect all the virtual correspondents or DUMMYs; because the TDS tables are full, or because the swap file is too small.

Action: Decrease the number of virtual correspondents, or increase the size of the swap file.

MV31. **TDS MAILBOX TABLE IS FULL, USE CTC COMMAND ON FROZEN CORRESPONDENT**

Meaning: The maximum size of the TDS MAILBOX table has been reached during a connection.

Action: If there are many frozen correspondents, use the CTC command to clean the table entries.

MV32. RESTARTABLE ABORT RECOVERY PHASE.

Meaning: A restartable abort is beginning its recovery phase.

Action: None.

MV32. RESTARTABLE ABORT RESTART PHASE.

Meaning: The recovery phase is finished, and the Commitment Units are ready to restart.

Action: None.

MV33. DUE TO GENERATION VALUES, NO CONNECTION LEFT FOR TM CORRESPONDENTS.

Meaning: This message appears when the computed number of TM Correspondents is 0:
- either at TDS startup
- or after [M] ALLOW_NEW_TDS_COR has been issued on TM Correspondents.

Any attempt to connect terminals to the TDS will be rejected with the code 0006.

[M] TERMINATE_TDS CONNECT=NONE has been performed on the previous session or TDS has been started up with CN=NONE in OPTIONS for the JCL on starting the TDS job.

The computed number of TM Correspondents is 0 when:
- either the TMSESS value of the TDSWKS Directive in the NETGEN for this TDS is 0
- or, if no TDSWKS Directive exists, the *number of terminals* supplied in the STDS results in no TM correspondents left (after decrementing the DUMMY maximum and subtracting 1, if TDS MASTER MAILBOX is used).

Action: None if this situation is expected. Otherwise increase the TMSESS value of the TDSWKS Directive according to the number of terminals expected and resubmit NETGEN.

MV34. H_REINIT TRANSACTION CANNOT BE LAUNCHED - WARM RESTART MANDATORY

Meaning: The H_REINIT transaction defined in the TDS generation to be used as a STARTUP transaction (in case of RESTARTABLE ABORT) cannot be executed due to lack of space in the TDS Tables or in the SWAP file, or due to the fact that the TDS has been generated with a single level of simultaneity, or if there is no simultaneity left to execute the transaction.

Action: Restart TDS in WARM mode.

MV35. TX: *txname* MUST BE DECLARED RPC TO USE RPC SERVICES (TPR: *tprname*)

Meaning: The transaction does not specify RPC SERVICE USED but one of its TPRs has attempted to run an RPC service while running on a non-RPC process.

The transaction then aborts with the following message that appears on the screen and also in the JOR:

ABORT(*x.xx.xxxx*) **RC=*xxxx*1463->TP7 X, PMDVIOL(***tprnm***).**

Action: Specify the RPC SERVICE USED Clause in the MESSAGE Statement of the Transaction Section and rerun TP7GEN.

MV37. CNCT DBG FROM:XXXXXXXX C=000 ST=00X R=0000X

Meaning: This version of the MV37 message applies to Technical Statuses before TS 7356 (MDTDS TRACE = 02).
The "XXXXXXXX" value is CONNECT or CPCNCT to identify the TDS procedure,
"C" is the decimal value of an internal error code (if not zero),
"ST" is the VCAM status (hexadecimal), and
"R" is the VCAM reason (hexadecimal).

Action: This information is used by your Customer Service Center. To de-activate this message, enter the command MDTDS TRACE=00.

MV37. **CNCT DBG:ZZZZZZZZZZZZ-->>V=YYYYYY, I=CCC, S=XXXX(HEXA), REASON=RRRR(HEXA)**

or

MV37. **CNCT DBG:ZZZZZZZZZZZZ-->>V=YYYYYY, I=CCC, S=XXXX(HEXA), ADDR=FFFFFFFF**

Meaning: This version of the MV37 message applies to Technical Statuses from TS 7356 (MDTDS, TRACE = 02 for XCP1 correspondents, TRACE = 01 for TM correspondents, and TRACE = 03 for TM and XCP1 correspondents).
The "ZZZZZZZZZZZZ" value is ********** (no correspondent name available), **CPCNCT** (error detection in H_TP7_CPCNCT component), or CORRESPONDENT NAME,
"V=YYYYYY" is the verb name (CNxxxx or CPxxxx from H_TP7_CONNECT or H_TP7_CPCNCT, RJCP or CLCP from H_TP7_CONNECT rejection,
"I=CCC" is an internal error code (decimal),
"REASON=RRRR" is the VCAM reason,
"S=XXXX" is the rejection code from TDS if V = RJCP or CLCP, or the VCAM status (hexadecimal).
"ADDR=FFFFFFFF" is the address of the connection rejection (if the value of ADDR is not FFFFFFFF).

Action: This information is used by your Customer Service Center. To de-activate this message, enter the command MDTDS TRACE=00.

MV38. **TIMEOUT UPON SYNCHRONOUS SEND. USER=user_name, TPR=tpr_name**

Meaning: Warning message.
This message is sent when the TDS generation clause "MAXIMUM WAIT-TIME UPON SYNCHRONOUS SEND IS *nmin"* is present and when the *nmin* value is exhausted during a SEND verb (please refer to the clause description in Chapter 3).
This means that network acknowledgement was not received during the WAIT-TIME.

MV39. COLD RESTART FORCED DUE TO ABORT DURING PREVIOUS COLD RESTART.

Meaning: An abort occurred during the previous TDS cold restart phase. The current TDS restart mode is forced to cold in order to ensure the TDS next session consistency.

Action: None - Files are recovered to their last commitment point; TDS is cold restarted.

MV40. IDS/GAC ERROR TPR:XXXXXXXXXXX, XXX SCHEMA/COMMIT, XXX USABLE SCHEMAS.

Meaning: Fatal abort.

The swap area used to save IDS/ll currencies, after a TPR execution is not sufficient.

The size of this area is computed at the beginning of the first TDS session following TP7GEN. This size depends on schema descriptions and depends on the STDS clause MAXIMUM NUMBER OF SCHEMA IS *maxschema* PER COMMITMENT.

TPR:XXXXXXXXXXX is the name of the faulty TPR.

XXX SCHEMA/COMMIT is the maximum number of schemas per commitment unit:

If the clause MAXIMUM NUMBER OF SCHEMA IS n PER COMMITMENT is absent from the STDS generation member, the value XXX is 001. If the clause is present XXX is the value n.

XXX USABLE SCHEMAS is the number of schemas currently usable for the TDS session.

Action: Check the value XXX SCHEMA/COMMIT is correctly defined.

If not, increase this value and run TP7GEN.

If XXX is correct, contact the Service Center.

MV41. XXXXXXXXXXX CANNOT BE MASTER OF TDS (NOT MASTER IN A PREVIOUS SESSION)

Meaning: Since TS 7458, this message is issued at TDS warm restart when the master of the current TDS session (whose name is XXXXXXXXXXX) was an ordinary user in a previous session. As he is still known by the TDS as an ordinary correspondent, he cannot be the master of the starting session (in this case, the return code DUPNAME is displayed).

MV42. TIMEOUT UPON CONNECTION TO NODE=xxxx MBX=mailbox_name, (TPR=tpr_name)

Meaning: Warning message.
This message is sent when the TDS generation clause "MAXIMUM WAIT-TIME UPON SYNCHRONOUS SEND IS *nmin"* is present and when the *nmin* value is exhausted during an asynchronous outward CONNECTION (please refer to the clause description in Chapter 3).
NODE=xxxx is the name of the destination node.
This means that the connection destination did not answer during the WAIT-TIME.

MV60. XA SERVICE DAMAGED

Meaning: A protocol error has been made by TDS-XA or a swap failure occurred.

Action: None. Internal error.

MV61. XA SESSION CANNOT BE INITIALIZED FOR ORACLE

Meaning: The SM containing ORACLE7/TDS is not loaded, or the correct library is not specified in the JOBLIB statement of the TDS job.

Action: Ensure that the SM containing ORACLE7/TDS-XA is loaded. If it is, check JOBLIB statement of TDS job.

MV62. ERROR DURING XA RECOVERY PHASE

Meaning: ORACLE7/TDS-XA has been asked to perform XA recovery in a bad context or with incorrect parameters which involved a protocol error.

Action: None. Internal error.

MV63. XA FUNCTION IS NOT AVAILABLE

Meaning: The product TDS-XA (Marketing Identifier) has not been purchased for the site.

Action: Verify if the product was purchased or not.

MV70. CMA INITIALISATION FAILED RC=XXXXXXXX

Meaning: CMA cannot be initialized.

Action: Ensure that the SM H_SM_CMA is loaded in BKST. If it is, check JOBLIB statement of TDS job.

MV79. TCP/IP COMMUNICATION IS NOT INITIALIZED ERRNO=XXXXX ON XXXXXXXXXXXX

Meaning: An error occured during TDS-TCP/IP initialization:
ERRNO=XXXXX is the error code returned by the invoked Sockg7 function, see *TDS-TCP-IP User's Guide*.
XXXXXXXXXXXX is the name of the invoked Sockg7 function which encountered a problem.

Action: Check <<hosts>> and <<services>> configuration files used for the chosen TCP/IP way of communication.
Check Open7 and Sockg7 products are active and ready on the site.

MV80. TCP-IP FUNCTION IS NOT AVAILABLE

Meaning: The product TDS/TCP-IP (Marketing Identifier) has not been purchased for the site, see *TDS-TCP-IP User's Guide*.
The H_SM_DCM SM of SYS.DCM.SYSTEM is not loaded.

Action: Verify if the product was purchased or not.
Ensure that the SM containing H_SM_DCM is loaded.

B.2 PPC Error Messages

In the following messages:

xx	is the decimal value of the SCID of the procedure whose name is NNNNNNNN (PPC_COM_S for example),
corname	is a character field which identifies the correspondent name edited on a maximum of 12 characters. If the correspondent is not identified, "CORR=UNKNOWN" is displayed.
edited_G4	Example: RC=350504C2->PPC 5,ABOPROC
poolname	is a character field which identifies the name of the pool. It is edited on a maximum of 8 characters. If the pool is not identified, "POOL=UNKNOWN" is displayed.
reason	is a decimal value which identifies the error in this procedure and is edited on 3 characters with space left. Example: REASON=7
compl	is a complementary code. It is edited on 4 characters with zero left. Example: COMPL (or CPL) = 0128
text	(if present) will be an explanation about this error. It is edited on a variable field with a maximum length of 72 characters.
tpname	is a character field which identifies the name of the tp. It is edited on a maximum of 64 characters.
username	is a character field which identifies the name of the user that initiated the tp. It is edited on a maximum of 12 characters.
xx(procname)	identifies the procedure that detected the error. **xx** is a decimal number edited on 2 characters with space left. **procname** is the name of the procedure (header H_is suppressed) edited on variable field. Example: SICID=5(PPC_COM_S) or SICID=7(PPC_DIALOG)
function	is a character field which identifies the requested function such as "attach local" or "attach remote".

Messages PP07, PP08, PP09, PP10, PP12 and TX95 are sent to the TDS master and printed in the TDS jor according to the value set in the keyword XCP2TRC in the master command MDTDS. If the first bit of the field associated with XCP2TRC is set to 1, messages are sent. If it is set to 0 (default), messages are not sent.

PP05 INTERNAL ERROR SCID = xx (procname) REASON = reason [COMPL = compl] CORR=corname POOL = poolname RC=edited_G4

If reason is equal to 8 or 9, the complementary code is a VCAM status.

For other values of reason, compl value has an internal meaning.

Action: This is a system error, contact your supplier.

PP06 PPC PROTOCOL ERROR SCID=xx(procname) REASON = reason [COMPL = compl] CORR=corname POOL=poolname RC=edited_G4

Meaning: This message is sent when a xcp2 session is disconnected upon a protocol violation or a time out when waiting for a confirm response.

Action: This is a system error, contact your supplier.

NOTE:

some PP06 messages are sent only if the "MDTDS XCP2TRC=80" command has been entered.

- in PPC_RMPROC procedure
- in PPC_SPM procedure when reason = 12 in case of backout restore/no restore conflict.

PP07 SESSION DISCONNECTED BY LOCAL CORR = corname POOL = Poolname SCID = xx(procname) REASON = reason COMPL = compl RC=edited_G4 [TEXT=text]

Meaning: This is an information message for customer service.

For the procname PPC_CNCT:

- only the reason = 03 is issued from this procedure.
- COMPL = 166 means that a time out occurs during a connection.

For the procname PPC_DIALOG

REASON = 1
text = "BIS SENT". Normal disconnection processing.

REASON = 2
text = "V_MGCLOSED RECEIVED". The session has been closed abnormally.

REASON = 3
There is no text. COMPL is an internal error code.
- If COMPL is <128, a PP05 message is sent.
- If COMPL is >128, a PP06 message is sent.

Action: None.

PP08 SESSION DISCONNECTED BY REMOTE CORR = corname POOL = poolname SCID = xx(procname) REASON = reason COMPL = compl

or

PP08 SESSION DISCONNECTED BY REMOTE CORR = corname POOL = poolname SCID = xx(procname) DATA LEVEL 5 RECEIVED

or

PP08 SESSION DEACTIVED CORR = corname POOL = poolname SCID = xx(procname) REASON = reason COMPL = compl

Meaning: This is an information message for customer service.

Some reasons are VCAM codes. For example:

For the procname PPC_DIALOG
the reason 4 is set when the result of $H_RECVIT is V_ABNTERM.
In this case COMPL is the interrupt value.

For the procname PPC_CNCT
the reason 6 is set when the result of $H_RECVIT is V_ABNTERM
In this case COMPL is the interrupt value.
Note: for interrupt value see documentation *Network System Message and Return Codes OSI/DSA 39* A2 26DM).

For the procname PPC_RMPROC
the reason 70 is set upon a session deactivation, the COMPL value is an internal deactivation reason.

Action: None.

PP09 SESSION CONNECTION REJECTED BY LOCAL CORR = corname POOL = poolname SCID = xx(procname) REASON = reason COMPL = compl RC=edited_G4 [TEXT=text]

Meaning:	Issued in PPC_CNCT procedure.
reason = 2:	some PPC tables cannot be allocated.
compl =	internal information on the control block that cannot be created.
text =	"TABLE OVERFLOW.RM REQUEST REJECTED".
reason = 4	
compl =	"82"X negociation winner/loser on pool (internal error).
text =	none. Other values are internal error.
reason = 5:	session opening is rejected
compl =	H_OPMG VCAM primitive status (MGI_REASON). NOTE: see PP10 message for common values of MGI_REASON.
text =	"OPENMG REJECTED".
reason = 7:	connection rejected by local acceptor.
compl =	"89"X remote own_unique_name equal local own_unique_name.
text =	LOCAL ACCEPTOR.none.
Action:	Contact your supplier.

PP10 SESSION CONNECTION REJECTED BY REMOTE CORR = corname POOL = poolname SCID = xx(procname) REASON = reason COMPL = compl [TEXT=text]

Meaning: A negative acknowledgment is received in response to the H_OPMG VCAM primitive.

Sent by the procname PPC_CNCT with the reason 01.

compl gives the VCAM reason of the reject (MGI_REASON).

"0001" x =	abnormal rejection reason.
"0002" x =	destination node not operable.
"0003" x =	destination node saturated.
"0004" x =	destination mailbox unknown
"0005" x =	destination mailbox not operable.
"0006" x =	destination mailbox saturated.
"0007" x =	destination application saturated.
"0009" x =	dialog rejection (negotiation result).
"000A" x =	presentation rejection (negotiation result).
"0015" x =	time out when waiting for acknowledgment.
"0017" x =	access right violation.
"0018" x =	security violation. Verify that TDSname_ADM (where TDSname is the name of the local TDS) is correctly cataloged in the remote site. See *CPI-C/XCP2 User's Guide*, paragraph "Site Catalog Configuration".
"0040" x =	destination node unknown.
"0041" x =	path to destination node not available.
"0042" x =	duplicate user identifier.
text =	NEGATIVE OPENACK RECEIVED

Action: According to the complementary code, modify application, catalog, NETGEN directives or call supplier.

PP11 function REFUSED BY LOCAL PPC CORR = corname POOL = poolname TP = tpname US = username SCID = xx(procname) RS = reason CPL = compl RC =edited_G4 TXT=text

For the procname PPC_PTM

reason = 1: function = ATTACH_LOCAL

text = MAXIMUM TP REACHED

Action: Check the MAXTX keyword of the XCP2WKS directive in NETGEN.

reason = 2: function = ATTACH_REMOTE. Internal error.

text = PROBLEM WITH TIPS

For the procname PPC_RMPROC

reason = 1: function = ATTACH_REMOTE

text = MAXIMUM TP REACHED

reason = 2: function = ATTACH_LOCAL. Case syncpoint and defer resynchronisation.

text = MAXIMUM TP REACHED

reason = 3: function = ATTACH_ALLOC

text = POOL BEING CLOSED

reason = 4: function = ATTACH_ALLOC

text = NO FREE WINNER SESS

reason = 5: function = ATTACH_ALLOC. Internal error.

text = ALLOC CONV FAILED

reason = 6: function = ATTACH_ALLOC. The Terminate_Tds strong command is in progress..

text = CLOSE STRONG PNDG

reason = 7: function = ATTACH_ALLOC. Internal error: troubles when getting partner address.

text = LOOK PARTNER FAILED

reason = 8: function = ATTACH_ALLOC. Internal error: return code "SYSOVLD" when creating pool table.

text = CREATE PARTN FAILED

reason = 9:	function = ATTACH_ALLOC. Internal error: return code "NOTDONE" when creating pool table.
text =	CREATE PARTN FAILED
reason = 10:	function = ATTACH_ALLOC. Internal error: correspondent not found.
text =	POOL UNKNOWN
reason = 11:	function = ATTACH_ALLOC. Internal error.
text =	NO SESSION ON POOL
reason = 12:	function = ATTACH_ALLOC. Internal error: return code "SYSOVLD" when creating pool table.
text =	CREATE POOL FAILED
reason = 13:	function = ATTACH_ALLOC. Internal error: return code "NOTDONE" when creating pool table.
text =	CREATE POOL FAILED
reason = 14:	function = ATTACH_ALLOC.
text =	STOP PENDING
reason = 15:	function = ATTACH_ALLOC.
text =	PARALL POOL CLOSED

For the procname PPC_TPI

Meaning:	Some control are not satisfied before starting the invoked transaction.
function =	ATTACH_REMOTE
text =	SYSTEM TP WITH TIPS or CHECKING NOT OK
reason = 5:	though the checking of the userid has been declared mandatory in NETGEN (CONV_USERID=MANDATORY in XCP2WKS or XCP2POOL directive), the partner invokes (CMINIT for CPI-C and H_PPC_INVOKE for PPC-PI) a transaction with the security option at "NONE" value.

reason = 20:	the partner invokes a transaction with the security option set at "PRESENT" value ("PROGRAM" value for CPI-C) but the password is not passed in the security parameters.
reason = 23:	the partner invokes a transaction with the security option at "SAME" value though the clauses CONV_USERID=MANDATORY and CONV_VERIFIED_ACCEPTED= 0 are used in the XCP2WKS or XCP2POOL directive of NETGEN.
reason = 24:	return code not "DONE" from the catalog when controlling security parameters: – USER, PROJECT, BILLING provided through the invoke verb are not correctly cataloged – USERUNKN: user is not cataloged. – PROJUNKN: project is not cataloged. – APPLUNKN: application not found in catalog. – RLPUNKN: XCP2 mailbox (mailbox declared in NETGEN in the XCP2WKS directive of the TDS on which the transaction is to be running) is not in the application list of the project of the user for whom the invoked transaction is to be run.
reason = 26:	provides password though INVOKE (CMINIT) verb is not the cataloged one.
reason = 27:	catalog not validated.
reason = 28:	synchronization level received from partner not supported (other than: none, confirm or syncpoint).
	Other values are internal status.
Action:	Contact your supplier

PP12 PPC WARNING SCID = xx(procname) REASON = reason RC = edited_G4 TEXT=text

Meaning: A PPC function cannot be executed because some problem when allocating PPC table.

For the procname PPC_RMPROC

text = TABLE OVERFLOW or FUNCTION NOT DONE

Action: Contact your supplier

PP13 PPC EXCEPTION cc.tt AT ADDRESS xxxxxxxx

cc.tt is the exception class/type,
xxxxxxxx is the address of the exception.

Meaning: An exception has occurred in PPC code. A log is taken (stacks and PPC tables). PP13 is used for debugging purpose.

Action: Contact your supplier.

B.3 TP7 Error Message Sent by TDS

TP7 TDS-RECOVERY RC=*return code*.

This is a warning message. This message appears in the JOR during recovery phase at end of job, restartable abort, or after a crash. An abnormal return code indicates that some problem has occurred during the recovery phase. Refer to messages coming from Before and After Journals.

TP7 DTP HEURISTIC ROLLBACK: …

TP7 XA_COMMIT REQUESTED AFTER HEURISTIC ROLLBACK: …

TP7 XA_ROLLBACK REQUESTED AFTER HEURISTIC ROLLBACK: …

These messages are related to DTP / XA TDS function. See *XA TDS User's Guide*.

C. Structure of Records in the User Journal

The structure of the records in the User Journal is given in order of their J-TYPE:

```
01......INPUT MESSAGE.
02......REPLY MESSAGE.
04......OUTPUT MESSAGE.
08......USER JOURNALIZATION.
09......STATISTICAL OUTPUT.
10......REPLY MESSAGE WITH DEVICE HEADER LOGGING.*
11......TILS SIMULATION RECORD.
12-15...CONNECTION RECORD.
16, 17..DISCONNECTION RECORD.
```

NOTE:

***** Applies only if USE DEVICE-HEADER-LOGGING specified and replaces TYPE 02.

J-TYPE 01 Input Message

```
01 J-REC.
  02 J-HEADER.
    03 J-TYPE   PIC 99.      "01" *
    03 J-TXNU   COMP-2.      Transaction serial number
    03 J-TPNU   COMP-1.      TPR serial number
    03 J-USID   PIC X(8)     User identification
    03 J-FLAG   PIC X.       Flag (C if complete
                             record, else T)
    03 J-TIME   COMP-2.      Time in millisecond units
    03 J-TDSN   PIC X(4)     TDSname
    03 J-DATE.               Date in European format
      04 J-DAY    PIC 99.      Day
      04 J_MONTH  PIC 99.      Month
      04 J_YEAR   PIC 99.      Year
    03 J-LGTH   COMP-1.      Data length.
  02 J-DATA
    03 J-MSGID  PIC X(8).    Transaction name
    03 J-MSGLG  COMP-1.      Message length
    03 J-MSGTX  PIC X (**).  Message text
```

NOTE:

***** Record logged if USER-JOURNAL WITH [INPUT] MESSAGE LOGGING specified.

****** Denotes value obtained in J-LGTH.

J-TYPE 02 Reply Message

```
01 J-REC.
  02 J-HEADER.
    03 J-TYPE   PIC 99.      "02" *
    03 J-TXNU   COMP-2.      Transaction serial number
    03 J-TPNU   COMP-1.      TPR serial number
    03 J-USID   PIC X(8)     User identification
    03 J-FLAG   PIC X.       Flag (C if complete
                             record, else T)
    03 J-TIME   COMP-2.      Time in millisecond units
    03 J-TDSN   PIC X(4)     TDSname
    03 J-DATE.               Date in European format
      04 J-DAY    PIC 99.      Day
      04 J-MONTH  PIC 99.      Month
      04 J-YEAR   PIC 99.      Year
    03 J-LGTH   COMP-1.      Data length.
  02 J-DATA
    03 J-MSGTX  PIC X (**).  Reply message text of
                             variable length
```

NOTE:

***** Record logged if USER-JOURNAL WITH [INPUT] MESSAGE LOGGING specified and if USE DEVICE-HEADER-LOGGING omitted.

****** Denotes value obtained in J-LGTH.

J-TYPE 04 Output Message

```
01 J-REC.
  02 J-HEADER.
    03 J-TYPE   PIC 99.      "04" *
    03 J-TXNU   COMP-2.      Transaction serial number
    03 J-TPNU   COMP-1.      TPR serial number
    03 J-USID   PIC X(8)     User identification
    03 J-FLAG   PIC X.       Flag (C if complete
                             record, else T)
    03 J-TIME   COMP-2.      Time in millisecond units
    03 J-TDSN   PIC X(4)     TDSname
    03 J-DATE.               Date in European format
      04 J-DAY    PIC 99.      Day
      04 J-MONTH  PIC 99.      Month
      04 J-YEAR   PIC 99.      Year
    03 J-LGTH   COMP-1.      Data length.
  02 J-DATA.
    03 J-MSGTX  PIC X (**).  Output message text
```

NOTE:

***** Record logged if USER-JOURNAL WITH [INPUT] MESSAGE LOGGING specified.

****** Denotes value obtained in J-LGTH.

J-TYPE 08 User Journalization

```
01 J-REC.
  02 J-HEADER.
    03 J-TYPE   PIC 99.      "08" *
    03 J-TXNU   COMP-2.      Transaction serial number
    03 J-TPNU   COMP-1.      TPR serial number
    03 J-USID   PIC X(8)     User identification
    03 J-FLAG   PIC X.       Flag (C if complete
                             record, else T)
    03 J-TIME   COMP-2.      Time in millisecond units
    03 J-TDSN   PIC X(4)     TDSname
    03 J-DATE.               Date in European format
      04 J-DAY    PIC 99.      Day
      04 J-MONTH  PIC 99.      Month
      04 J-YEAR   PIC 99.      Year
    03 J-LGTH   COMP-1.      Data length.
  02 J-DATA.
    03 J-USRTX  PIC X (**).  User record
```

NOTE:

***** Record logged if USER-JOURNAL specified regardless of MESSAGE LOGGING.

****** Denotes value obtained in J-LGTH.

J-TYPE 09 Statistical Output

```
01 J-REC.
  02 J-HEADER.
    03 J-TYPE   PIC 99.      "09" *
    03 J-TXNU   COMP-2.      Transaction serial number
    03 J-TPNU   COMP-1.      TPR serial number
    03 J-USID   PIC X(8)     User identification
    03 J-FLAG   PIC X.       Flag (C if complete
                             record, else T)
    03 J-TIME   COMP-2.      Time in millisecond units
    03 J-TDSN   PIC X(4)     TDSname
    03 J-DATE.               Date in European format
      04 J-DAY    PIC 99.      Day
      04 J-MONTH  PIC 99.      Month
      04 J-YEAR   PIC 99.      Year
    03 J-LGTH   COMP-1.      Data length
  02 J-DATA.
    03 J-MSGLG  COMP-1       Output message length
```

NOTE:

***** Record logged if USER-JOURNAL WITH [INPUT] MESSAGE LOGGING specified.

J-TYPE 10 Reply Message with Device-Header-Logging

```
01 J-REC.
  02 J-HEADER.
    03 J-TYPE       PIC 99.       "10" *
    03 J-TXNU       COMP-2.       Transaction serial number
    03 J-TPNU       COMP-1.       TPR serial number
    03 J-USID       PIC X(8)      User identification
    03 J-FLAG       PIC X.        Flag (C if complete
                                  record, else T)
    03 J-TIME       COMP-2.       Time in millisecond units
    03 J-TDSN       PIC X(4)      TDSname
    03 J-DATE.                    Date in European format
      04 J-DAY        PIC 99.       Day
      04 J-MONTH      PIC 99.       Month
      04 J-YEAR       PIC 99.       Year
    03 J-LGTH       COMP-1.       Data length
  02 J-DATA.
    03 J-DVCHDRLG   COMP-1.       Device Header length
    03 J-DVCHDR     PIC X(16).    Device Header text
    03 J-MSGLG      COMP-1.       Message length
    03 J-MSGTX      PIC X (**).   Reply message text
```

NOTE:

***** Record logged if both USER-JOURNAL WITH [INPUT] MESSAGE LOGGING and USE DEVICE-HEADER-LOGGING specified.

****** Denotes value obtained in J-MSGLG.

J-TYPE 11 TILS Simulation Record

```
01 J-REC.
  02 J-HEADER.
    03 J-TYPE     PIC 99.        "11" *
    03 J-TXNU     COMP-2.        Transaction serial number
    03 J-TPNU     COMP-1.        TPR serial number
    03 J-USID     PIC X(8)       User identification
    03 J-FLAG     PIC X.         Flag (C if complete
                                 record, else T)
    03 J-TIME     COMP-2.        Time in millisecond units
    03 J-TDSN     PIC X(4)       TDSname
    03 J-DATE.                   Date in European format
      04 J-DAY      PIC 99.        Day
      04 J-MONTH    PIC 99.        Month
      04 J-YEAR     PIC 99.        Year
    03 J-LGTH     COMP-1.        Data length
  02 J-DATA.
    03 J_VERSION    PIC X.
    03 J_C12USERID  PIC X(12).   User ID in 12 characters
    03 J_TYPE       PIC X(2).    Type (01, 02, 04, 09, 10, BK**)
    03 J_MSGID      PIC X(8).    TX name if the J_TYPE field of
                                 J_DATA=1
    03 J_DVCHDRLG   COMP-1.      Length of device header
    03 J_DVCHDR     PIC X(16).   Device header
    03 J_MSGLEN     COMP-1.      Message length
    03 J-MSGTX      PIC X (***). Input or reply message text
```

NOTE:

***** Record logged if both USER-JOURNAL WITH [INPUT] MESSAGE LOGGING and USE TILS-SIMULATION specified.

****** Used for BREAK simulation.

******* Appears only if J_TYPE (of J_DATA) equals 1, 2, 4, or 10.

J-TYPE 12, 13, 14, and 15 Connection Record

```
12 : FIRST LOGON WITH TDS AS ACCEPTOR
14 : FIRST LOGON WITH TDS AS INITIATOR
13 : RELOGON     WITH TDS AS ACCEPTOR
15 : RELOGON     WITH TDS AS INITIATOR

01 J_REC.
  02 J_HEADER.
    03 J_TYPE           PIC 99.     "12", "13", "14", or "15"
    03 J_TXNU           COMP-2.     Transaction serial number
    03 J_TPNU           COMP-1.     TPR serial number
    03 J_USID           PIC X(8).   User identification
    03 J_FLAG           PIC X.      Flag (C if complete
                                    record, else T)
    03 J_TIME           COMP-2.     Time in millisecond units
    03 J_TDSN           PIC X(4).   TDSname
    03 J_DATE.                      Date in European format
      O4 J_DAY            PIC 99.     Day
      04 J_MONTH          PIC 99.     Month
      04 J_YEAR           PIC 99.     Year
    03 J_LGTH           COMP-1.     Data length
  02 J_DATA .
    03 J_VERSION        PIC X(1).   Version number is 1, 2, or 3
    03 J_C12USERID      PIC X(12).  Userid in 12 characters
    03 J_TERMTYPE       PIC X(8).   Type of terminal
    03 J_MODEL          COMP-1.     DSA model
    03 J_USERINFO       PIC X (*).  Userinfo string at connection.
                                    Length depends on J_VERSION:
                                    if J_VERSION=1, PIC X(10)
                                    if J_VERSION=2, PIC X(22)
                                    if J_VERSION=3, PIC X(22)
    03 J_CORTYPE        PIC X(1).   Correspondent type:
                                    O = OMH
                                    M = Master mailbox
                                    D = Dummy
                                    Y = Temporary
                                    T = Terminal manager
                                    X = XCP
                                    A = Application
    03 J_SESTYPE        PIC X.      A = Active, P = Passive
    03 J_INOUTCAP       PIC X.      O = Output capability
                                    B = Both capability
    03 J_EDITMODE       PIC X       E = Edited mode
                                    U = Unedited mode
    03 J_VIP            PIC X.      V = VIP
    03 J_DVTYPE.                    Device type
      04 J_DISPLAY        PIC X.      D = Display
      04 J_KEYBRD         PIC X.      K = Keyboard
      04 J_PRINTER        PIC X.      P = Printer
      04 J_DISKET         PIC X.      T = Diskette
      04 J_CASSET         PIC X.      C = Cassette
```

```
   03 J_DVNB           COMP-1.     Device number (slave/master)
   03 J_MASTMBX        PIC X(8).   Master mailbox name
   03 J_PROJECT        PIC X(12).  Project
   03 J_BILLING        PIC X(12).  Billing
   03 J_TERMINALID     PIC X(8).   Terminal name field
                                   if J_VERSION=3
   03 J_AUTHORITYCODES COMP-2.     Authorities field
                                   if J_VERSION=3
```

NOTE:

***** Denotes value obtained in J_VERSION.

J-TYPE 16 and 17 Disconnection Record

```
16 : LOGOUT RECORD (BYE)
17 : DISCONNECT RECORD

01 J_REC.
  02 J_HEADER.
    03 J_TYPE       PIC 99.     "16" or "17"
    03 J_TXNU       COMP-2.     Transaction serial number
    03 J_TPNU       COMP-1.     TPR serial number
    03 J_USID       PIC X(8).   User identification
    03 J_FLAG       PIC X.      Flag (C if complete
                               record, else T)
    03 J_TIME       COMP-2.     In millisecond units
    03 J_TDSN       PIC X(4).   TDSname
    03 J_DATE.                  Date in European format
      04 J_DAY        PIC 99.     Day
      04 J_MONTH      PIC 99.     Month
      04 J_YEAR       PIC 99.     Year
    03 J_LGTH       COMP-1.     Data length
  02 J_DATA.
    03 J_VERSION    PIC X.
    03 J_C12USERID  PIC X(12).  Userid in 12 characters
    03 J_SEQUENCE   PIC X(2).   Type of sequence:
                                 DT = Disconnect terminal
                                 MC = M CANCEL
                                 TT = Term terminal
                                 LT = Logout terminal
                                 LX = Logout XCP1
                                 NM = Disconnection - no message
```

D. TDS Statistics Reports

At the end of the TDS session, a statistics report is produced. The statistics report consists of two parts:

- the FILE_OPENING_REPORT,
- and the EXECUTION_REPORT.

The EXECUTION report shows the number of transactions and TPRs executed by each user. It includes RPC statistics when the RPC SIMULTANEITY Clause exists in the STDS.

It includes DTP statistics as well when the MAXIMUM DTP-WAITTIME clause is specified. See *XA TDS User's Guide*.

Example of Syntax of File_Opening Statistics:

```
******  FILE_OPENING_REPORT  ******

H_FORM NOT OPENED     RC=OC8E0909->TP7     14,IFNNASG
T1 NOT OPENED         RC=OC8E0909->TP7     14,IFNNASG
T3 NOT OPENED         RC=OC8E0909->TP7     14,IFNNASG
T4 NOT OPENED         RC=OC8E0909->TP7     14,IFNNASG
```

Example of Syntax of Execution_Report Statistics:

```
*****************************************************************
*****************************************************************
****GCOS7                                                    ****
****                          T P 0 6                        ****
****                    VERSION: HALU   DATED: SEP   24 93   ****
*****EXECUTION_REPORT*********************************************
*****************************************************************

USER_NAME    LAST_TERM       TX_COUNT       TPR_COUNT
   STDS1     BP03D21D            4              9

******   GENERAL STATISTICS  ******

TDSNAME ........................... : TP06
# OF TRANSACTIONS ................. : 8
# OF USED TPR ..................... : 14
# OF EXCHANGES .................... : 7
# OF COMMITMENT UNITS ............. : 13
TOTAL TPR ELAPSE ................. : 27044
TOTAL TPR CPU .................... : 998
# OF NON-CONCURRENCE CONFLICTS ..... : 0
# OF SERIALIZATIONS ............... : 0

******   RPC   STATISTICS  ******

# OF TRANSACTIONS ................. : 0
# OF USED TPR ..................... : 0
TOTAL TPR ELAPSE ................. : 0
TOTAL TPR CPU .................... : 0

*******  ABORT STATISTICS  *******

# OF ABORTED TRANSACTIONS .......... : 1
# OF ABORTED TPR ................... : 1
# OF BUFFER OVERFLOWS .............. : 0
# OF DEADLOCKS ..................... : 0
# OF LOCKED PAGE TABLE OVERFLOWS ... : 0
DIRTY READ ABORTS ................. : 0
# OF LONG WAIT ..................... : 0
# OF PSEUDO-BUFFERS ................ : 0
```

Description:

USER_NAME	User identification.
LAST_TERM	Name of terminal associated with the user name at log-off.
TX_COUNT	Number of transactions executed by user since the last connection.
TPR_COUNT	Number of TPRs executed by user since the last connection.

General Statistics

TDSNAME	Name of TDS application.
# OF TRANSACTIONS	Total number of transactions, including the TDS-supplied transactions, run during the session.
# OF USED TPRs	Total number of TPRs whether terminated normally or not.
# OF EXCHANGES	Total number of successful exchanges.
# OF COMMITMENT UNITS	Total number of commitment units which completed successfully.
TOTAL TPR ELAPSE	Accumulated elapsed time, in milliseconds, used in running TPRs.
TOTAL TPR CPU	Accumulated CPU time, in milliseconds, used in running TPRs.
# OF NON-CONCURRENT CONFLICTS	Total number of conflicts due to non-concurrency.
# OF SERIALIZATIONS	Total number of TPRs requested to be restarted after an abort in serialized mode.

RPC Statistics

# OF TRANSACTIONS	Total number of RPC transactions run during the session.
# OF USED TPR	Total number of RPC TPRs whether terminated normally or not.
TOTAL TPR ELAPSE	Accumulated elapsed time, in milliseconds, used in running RPC TPRs.
TOTAL TPR CPU	Accumulated CPU time, in milliseconds, used in running RPC TPRs.

Abort Statistics

# OF ABORTED TRANSACTIONS	Total number of transactions aborted.
# OF ABORTED TPR	Total number of TPRs aborted. When a TPR aborts, its transaction does not necessarily abort.
# OF BUFFER OVERFLOWS	Total number of TPRs that aborted due to lack of buffers in the pool. TDS either restarted the transaction using the Before Journal or aborted the transaction.
# OF DEADLOCKS	Total number of commitment units that aborted because of the return code DEADLOCK from GAC. The commitment units aborted because DIE-WAIT mechanisms are not taken into account.
# OF LOCKED PAGE TABLE OVERFLOWS	Total number of aborts due to an insufficient number of locked pages (RC=COUNTOV or TABOV).
DIRTY READ ABORTS	Total number of TPRs using IDS/II database that abort in dirty read mode and restarted automatically. The abort/restart prevents database inconsistencies.
# OF LONG WAIT	Total number of TPRs that abort due to concurrent access conflicts.
# OF PSEUDO-BUFFERS	Set to 0.

Example of Syntax of Debugging Statistics:

When TDS abort occurs, the DEBUG INFO section is printed. A line is printed for each user who is not frozen at the time of shutdown.

```
***** DEBUG INFO *****

USER_NAME  TX_NAME  STATE  ST_TPHLK  ST_ACT  TPH_NCAREA  TPH_FLAGS
  NG2      USEDU    02    118000E0  18400000 0000000FFFFFFFFFFFFF
```

Description:

USER_NAME User identification.

TX_NAME Name of transaction activated by the user.

STATE Stage of execution at which the transaction was being processed:

01 No transaction is active for the user.

02 Transaction is active for the user and at commitment point.

03 Transaction active for the user but not at commitment point.

04 TPR is executing for the user.

05 TPR code is processed but its associated TDS code has not completed execution.

06 TDS is about to activate an interrupt transaction at a commitment point for the user.

07 TDS is about to activate an interrupt transaction for the user who currently has no active transaction.

Other data may appear such as ST_TPHLK, ST_ACT, TPH_NCAREA and TPH_FLAGS for Service Center personnel. See the Section 7 *Optimizing a TDS Application*.

E. TDS Accounting Records

Accounting records written in the system accounting file include:

- TDS session record when the TDS session terminates.
- User session record when a user disconnects normally (BYE) or forces disconnection (CANCEL).
- Transaction record at the end of a transaction. This record must be specifically requested at TP7GEN or dynamically at the master terminal.
- TPR record at the end of a TPR. This record must be specifically requested at TP7GEN or dynamically at the master terminal.

Time in these records is expressed in milliminutes or milliseconds according to the CONFIG statement SITEOPT. The default time-unit is thousandths of minutes.

TDS Session Accounting Record:

```
01 TDS-ACCOUNTING-RECORD.
  02 RECORD-TYPE         PIC X(2)    VALUE "20".
  02 STANDARD-HEADER.
    03  TDS-USER         PIC X(12).  * User name of TDS job
    03  TDS-PROJECT      PIC X(12).  * Project under which job runs
    03  TDS-BILLING      PIC X(12).  * Billing of job
    03  TDS-JOBID        PIC X(8).   * Identification of job
    03  TDS-RON          PIC 9(4).   * Run Occurrence Number
    03  TDS-REPEAT       PIC X.      * "R" if job repeated
                                     * "space" if first occurrence
    03  TDS-DSN          PIC 9(3).   * Dynamic job step number
    03  TDS-DATE         PIC 9(6).   * Date in yy.mm.dd format
    03  TDS-TIME         PIC 9(6).   * Time in hh.mm.ss format
  02 USER-ID             PIC X(12).  * Submitter userid
  02 PROJECT-ID          PIC X(12).  * Submitter project
  02 BILLING-ID          PIC X(12).  * Submitter billing
  02 TERMINAL-ID         PIC X(24).  * Spaces
  02 TERMINATION-STATUS  PIC X(8).   * MU91.NS if normal
                                       termination
                                     * MU91.FS if abnormal
                                       termination

  02 NB-OF-TRANSACTIONS                 COMP-2.
  02 NB-OF-TPR                          COMP-2.
  02 NB-OF-COMMITMENTS                  COMP-2.
  02 NB-OF-DIALOG                       COMP-2.
  02 TPR-ELAPSE-TIME                    COMP-2.
  02 TPR-CPU-TIME                       COMP-2.
  02 ABORT-COUNTS.
    03 NB-OF-ABORT-TPR                  COMP-2.
    03 NB-OF-ABORT-TRANSACTIONS         COMP-2.
    03 NB-OF-BUFFER-OVERFLOWS           COMP-2.
    03 NB-OF-DEADLOCKS                  COMP-1.
    03 NB-OF-DEFERRED-UPDATES-ABORT     COMP-1.
    03 NB-OF-CONCURRENCY-TABLE-OVERFLOW COMP-1.
    03 NB-OF-LONG-WAIT                  COMP-1.
    03 NB-OF-DIRTY-READ-ABORT           COMP-1.
    03 NB-OF-SERIALIZED-COMMITMENTS     COMP-1.
    03 NB-OF-NON-CONCURRENCY-ABORTS     COMP-1.
  02 TDS-NAME                           CHAR(4).
```

User Session Accounting Record:

```
01 USER-ACCOUNTING-RECORD.
  02 RECORD-TYPE         PIC X(2)    VALUE "21".
  02 STANDARD-HEADER.
    03  TDS-USER         PIC X(12).  * User name of TDS job
    03  TDS-PROJECT      PIC X(12).  * Project under which job runs
    03  TDS-BILLING      PIC X(12).  * Billing of job
    03  TDS-JOBID        PIC X(8).   * Identification of job
    03  TDS-RON          PIC 9(4).   * Run Occurrence Number
    03  TDS-REPEAT       PIC X.      * "R" if job repeated
                                       "space" if first occurrence
    03  TDS-DSN          PIC 9(3).   * Dynamic job step number
    03  TDS-DATE         PIC 9(6).   * Date in yy.mm.dd format
    03  TDS-TIME         PIC 9(6).   * Time in hh.mm.ss format
  02 USER-ID             PIC X(12).
  02 PROJECT-ID          PIC X(12).
  02 BILLING-ID          PIC X(12).
  02 TERMINAL-ID         PIC X(24).
  02 TERMINATION-STATUS  PIC X(8).   * MU90.BYE if normal
                                       disconnection (BYE)
                                       MU90.CAN if forced
                                       disconnection (CANCEL)
  02 NB-OF-TRANSACTIONS  COMP-2.
  02 NB-OF-TPR           COMP-2.
  02 NB-OF-COMMITMENTS   COMP-2.
  02 NB-OF-DIALOG        COMP-2.
  02 TPR-LAPSE-TIME      COMP-2.
  02 TPR-CPU-TIME        COMP-2.
  02 TDS-NAME            CHAR(4).
```

Transaction Accounting Record:

```
01 TRANSACTION-RECORD.
  02 RECORD-TYPE         PIC X(2)    VALUE "23".
  02 STANDARD-HEADER.
    03  TDS-USER         PIC X(12).  * User name of TDS job
    03  TDS-PROJECT      PIC X(12).  * Project under which job runs
    03  TDS-BILLING      PIC X(12).  * Billing of job
    03  TDS-JOBID        PIC X(8).   * Identification of job
    03  TDS-RON          PIC 9(4).   * Run Occurrence Number
    03  TDS-REPEAT       PIC X.      * "R" if job is repeated
                                       "space" if first occurrence
    03  TDS-DSN          PIC 9(3).   * Dynamic job step number
    03  TDS-DATE         PIC 9(6).   * Date in yy.mm.dd format
    03  TDS-TIME         PIC 9(6).   * Time in hh.mm.ss format
  02 USER-ID             PIC X(12).
  02 PROJECT-ID          PIC X(12).
  02 BILLING-ID          PIC X(12).
  02 TERMINAL-ID         PIC X(24).
  02 TRANSACTION-NAME    PIC X(8).
  02 TERMINATION-STATUS  PIC X(4).   * "spaces" if normal
                                       "ABT" for abort
  02 NB-OF-TPR           COMP-2.
  02 NB-OF-COMMITMENTS   COMP-2.
  02 NB-OF-DIALOGS       COMP-2.
  02 TPR-LAPSE-TIME      COMP-2.
  02 TPR-CPU-TIME        COMP-2.
  02 ACCOUNT-ID.
    03 ACCOUNT-NAME      PIC X(12).  } See explanation at
    03 ACCOUNT-PROJECT   PIC X(12).  } the end of this
    03 ACCOUNT-BILLING   PIC X(12).  } Appendix.
```

TPR Accounting Record:

```
01 TPR-RECORD.
  02 RECORD-TYPE          PIC X(2)    VALUE "24".
  02 STANDARD-HEADER.
    03 TDS-USER           PIC X(12).  * user name of TDS job
    03 TDS-PROJECT        PIC X(12).  * Project under which job runs
    03 TDS-BILLING        PIC X(12).  * Billing of job
    03 TDS-JOBID          PIC X(8).   * Identification of job
    03 TDS-RON            PIC 9(4).   * Run Occurrence Number
    03 TDS-REPEAT         PIC X.      * "R" if job repeated
                                       " " if first occurrence
    03 TDS-DSN            PIC 9(3).   * Dynamic job step number
    03 TDS-DATE           PIC 9(6).   * Date in yy.mm.dd format
    03 TDS-TIME           PIC 9(6).   * Time in hh.mm.ss format
  02 USER-ID              PIC X(12).
  02 PROJECT-ID           PIC X(12).
  02 BILLING-ID           PIC X(12).
  02 TERMINAL-ID          PIC X(24).
  02 TRANSACTION-NAME     PIC X(8).
  02 TERMINATION-STATUS PIC X(4).    * "ETPR": normal end of TPR
                                       "ECMT": normal end of
                                        commitment unit
                                       "RTPR": rollback TPR
                                       "ABT" : fatal TPR abort
                                       "RCMT": rollback commitment
  02 TPR-NAME             PIC X(12).
  02 TPR-LAPSE-TIME       COMP-2.     Time in thousandths of
                                      minutes.
  02 TPR-CPU-TIME         COMP-2.     Time in thousandths of
                                      minutes.
  02 ACCOUNT-ID.
    03 ACCOUNT-NAME       PIC X(12).
    03 ACCOUNT-PROJECT    PIC X(12).
    03 ACCOUNT-BILLING    PIC X(12).
```

How to Use ACCOUNT-ID Structure:

The following two cases describe how to use the `ACCOUNT-ID` structure.

CASE 1: The reported transaction has been spawned

The transaction account record contains the identity (name, project, and billing) of the user sending the spawned transaction.

For example, a user U1, runs transaction TX1 which spawns a transaction TX2 towards user U2. The transaction accounting record associated with TX2 contains the identity of both:

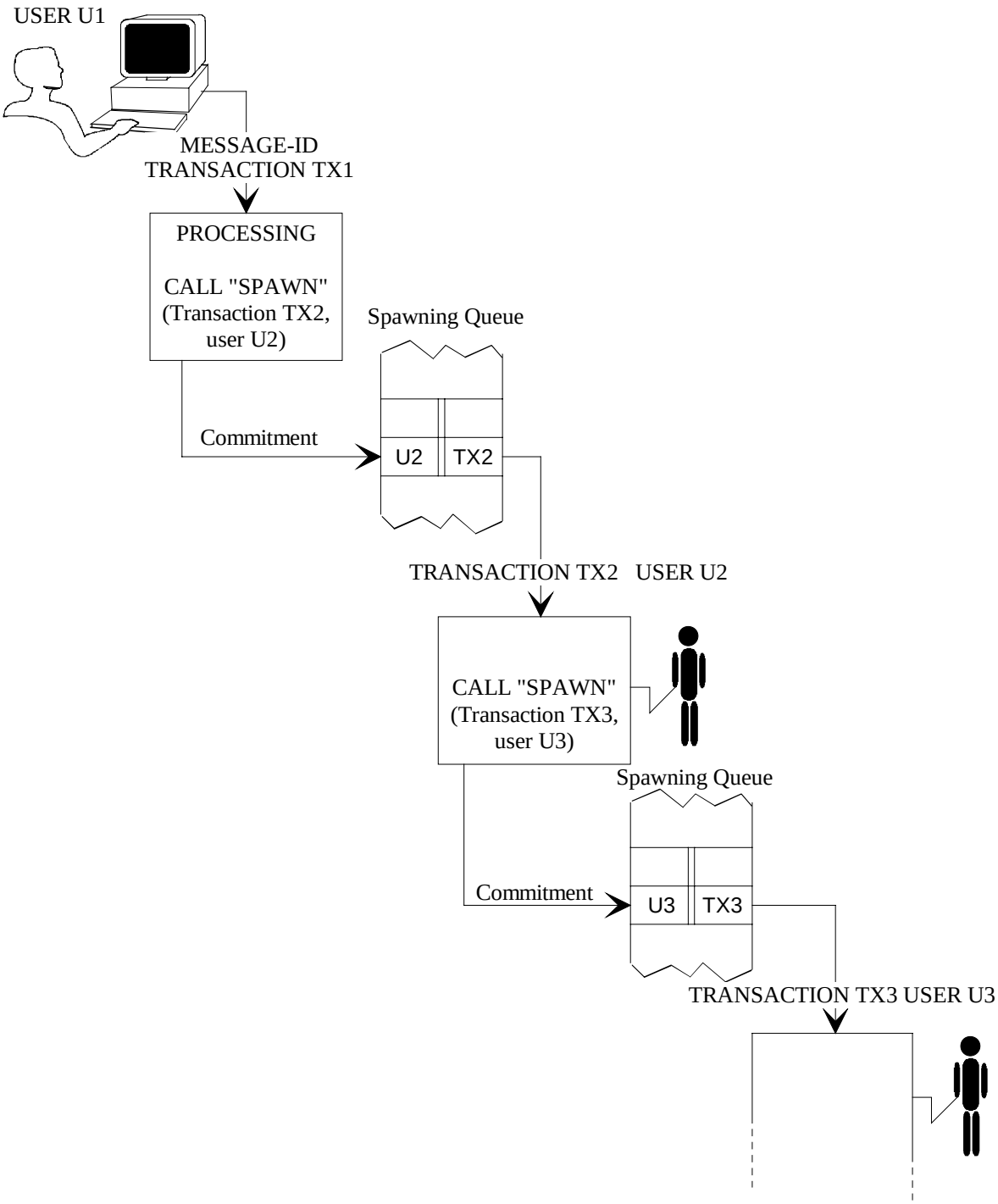
- U2 (user-id, project-id, and billing-id)
- U1 (account-name, account-project, and account-billing)

When successive spawnings are executed, the account-structure always contains the identity of the *original initiator* of the spawn, being the identity of the first user starting the first transaction.

U1 runs a transaction TX1 which spawns TX2 towards U2. Then U2 runs transaction TX2 which spawns transaction TX3 towards U3.

The transaction accounting record associated with TX3 contains the identity of both:

- U3 (user-id, project-id, and billing-id)
- U1 (account-name, account-project, and account-billing).


USER U1
MESSAGE-ID
TRANSACTION TX1
PROCESSING
CALL "SPAWN"
(Transaction TX2,
user U2)
Spawning Queue
Commitment
U2
TX2
TRANSACTION TX2 USER U2
CALL "SPAWN"
(Transaction TX3,
user U3)
Spawning Queue
Commitment
U3
TX3
TRANSACTION TX3 USER U3

CASE 2: The reported transaction has been started through the XCP2 verb INVOKE

The account structure contains the user's identity specified in the INVOKE parameters. The user-id field contains the name of the partner correspondent, being the application invoking the transaction.

For example, TDS1 and TDS2 are two TDS applications. In TDS1, a transaction TX1 is run by user U1 (name=U1, project=P1, billing=B1).

At a given time, TX1 needs some processing by transaction TX2 which belongs to TDS2. TX1 invokes TX2 by giving the following security parameters:

- User's name = Ux
- Project = Px
- Billing = Bx

TDS2 starts TX2. TX2 runs on behalf of user Ux and holds a conversation with correspondent TDS1. When TX2 ends, the transaction accounting record for TX2 contains:

- Userid = TDS1
- Project-id = Px
- Billing = Bx
- Account-name = Ux
- Account-project = Px
- Billing = Bx

In all other cases, the ACCOUNT structure is set to blanks. See *CPI-C/XCP2 User's Guide*.

F. Error Messages Produced by the Master Commands

Master commands can cause one or more error messages to appear. Messages delivered exclusively to the master operator cannot be redefined, see Paragraph 3.3.48 *SERVICE MESSAGE Clause*. The following is a list of standard error messages in alphabetical order:

AT LEAST ONE PARAMETER MUST BE PRESENT

ERRONEOUS VALUE FOR: *keyword*

MANDATORY VALUE MISSING FOR KEYWORD: *keyword*

MISSING RIGHT PARENTHESIS IN A LIST

NESTED LISTS ARE NOT ALLOWED FOR KEYWORD: *keyword*

NOT ENOUGH VALUES GIVEN FOR: *keyword*

TOO LONG PARAMETER: *parameter*

TOO MANY PARAMETERS

TOO MANY VALUES GIVEN FOR KEYWORD: *keyword*

UNKNOWN PARAMETER: *parameter*

VALUE LENGTH EXCEEDS DECLARED MAXIMUM LENGTH

parameter-x AND *parameter-y* ARE MUTUALLY EXCLUSIVE

parameter MUST BE type

parameter SPECIFIED MORE THAN ONCE

keyword SPECIFIED MORE THAN ONCE

G. Return Codes at Modify_TDS SM Library

This Appendix lists the return codes which may occur when issuing [M] MODIFY_TDS.

The following tables list return codes (*Ret-Code*) by their values and by their mnemonics.

NOTE:
On TDS warm restart, all libraries affected by [M] MODIFY_TDS during the previous session must be loaded into BKST. If this is not done, unpredictable results may be obtained.

List of Return Codes in Order of Value

Value *Ret-Code*	Explanation/Action
0012 ALREADY	Libraries requested are already in correct order - issue DTDS SMLIB to display and verify order.
0020 BUSY	One of the specified libraries is already used by another process group.
0802 INDOUT	Internal error - index out of bounds.
0110 NOTDONE	Function not performed - one of the SMs cannot be connected or disconnected.
0A16 SMNAV	Sharable module is not available - the SM of one of the two libraries is in debugging mode.
1853 ARGERR	Argument error - conflict in SMLIB list, one of the libraries in the list is not correct.
1878 NOMATCH	One of the specified libraries is not loaded in BKST or specified in the JOBLIB statement.

List of Return Codes in Alphabetical Order

Ret-Code *Value*		Explanation/Action
ALREADY	0012	Libraries requested are already in correct order - issue DTDS SMLIB to display and verify order.
ARGERR	1853	Argument error - conflict in SMLIB list, one of the libraries in the list is not correct.
BUSY	0020	One of the specified libraries is already used by another process group.
INDOUT	0802	Internal error - index out of bounds.
NOMATCH	1878	One of the specified libraries is not loaded in BKST or specified in the JOBLIB statement.
NOTDONE	0110	Function not performed - one of the SMs cannot be connected or disconnected.
SMNAV	0A16	Sharable module is not available - the SM of one of the two libraries is in debugging mode.

H. TDS Generation Source

```
 TDS SECTION.
 PROGRAM-ID. MAIT.
 NUMBER TERMINALS 200.
 SIMULTANEITY 25.
 RESERVE 300 AREAS.
 MESSAGE-LENGTH 6500.
 TPR-TIME-LIMIT 250000.
 MASTER MAILBOX IS MAI.
 USE FORMS.
 SERVICE-MESSAGE IMMEDIATE DELIVERY.
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
 SELECT EXTERNAL UFAS-RELATIVE
    ASSIGN TO DIRECT2
    ORGANIZATION IS UFF RELATIVE
    ACCESS MODE IS DYNAMIC RELATIVE KEY IS ZKEY
    FILE STATUS IS FSTATUS.
*END
 SELECT EXTERNAL SEQUENTIAL-INDEX
     ASSIGN TO ISEQ2
     ORGANIZATION IS UFF INDEXED
     ACCESS MODE IS DYNAMIC
     RECORD KEY IS KEY-INDEX
     ALTERNATE RECORD SECONDARY-1 DUPLICATES
     FILE STATUS IS FSTATUS.
*END
 TDS-FILE-DEFINITION.
*
*
*        FD FOR UFAS FILES
*
 FD UFAS-RELATIVE
     BLOCK CONTAINS 2 RECORDS
     RECORD CONTAINS 45 TO 2000 DEPENDING ON RECORD-LENGTH
     LABEL RECORDS STANDARD.
 01  UFAS-RELATIVEA.
     02 UFAS-RELATIVE-NUMBER          PIC X(10).
     02 ORGAD                         PIC A(10).
     02 DATD                          PIC X(10).
     02 TIME-DIRECT                   PIC X(15).
     02 FILLER                        PIC X(52).
     02 FILLER                        PIC X(1903).
 *END
```

```
 FD SEQUENTIAL-INDEX
     BLOCK 1 RECORDS
     RECORD CONTAINS 45 TO 2000 CHARACTERS DEPENDING ON
RECORD-LENGTH
     LABEL RECORDS STANDARD.
 01  SEQUENTIAL-INDEXA.
     02 KEY-INDEX                  PIC X(10).
     02 ORGAI                      PIC A(10).
     02 DATI                       PIC X(10).
     02 TIME-INDEX.
      03 FILLER         PIC X(3).
      03 SECONDARY-1    PIC X(2).
      03 FILLER         PIC X.
      03 SECONDARY-2    PIC X(2).
      03 FILLER         PIC X.
      03 SECONDARY-3    PIC X(2).
      03 FILLER         PIC X(4).
     02 FILLER          PIC X(52).
     02 FILLER          PIC X(1903).
     *END
*
 PROCESSING-CONTROL.
 FILE-INTEGRITY FOR UFAS-RELATIVE NONE.
 FILE-INTEGRITY FOR SEQUENTIAL-INDEX NONE.
 WORKING-STORAGE.
 77 RECORD-LENGTH    PIC 9(5).
 77 ZKEY             PIC 9(10).
 77 FSTATUS          PIC XX.
 77   NEXTKEY        PIC 9(10).
 01  COMMN-STR.
     05   NUMEROX.
     10   NUMERON    PIC 9(5).
     05   FILLER     PIC X(95).
     *END
 TRANSACTION SECTION.
 MESSAGE "MOTI" ASSIGN TO TPMONI-UIUD
 IMPLICIT COMMITMENT
 AUTHORITY-CODES 1
 TRANSACTION-STORAGE TS-MOTI SIZE 4200.
 01 TRANSACTION-STORAGE.
  02  TERMN.
   03 FILLER           PIC X(8).
   03 SDEST            PIC X(4).
  02 LOOP-TABLE.
   03 LOOP-INSTRUCTION PIC X(80) OCCURS 50 TIMES.
  02 AIGIMP            PIC A.
  02 AIG-INCREMENT-LENGTH PIC A.
  02 AIG-LOOP             PIC X.
  02 NUMBER-OF-LOOP-INSTR PIC 99.
  02 NUMBER-OF-LOOP PIC 9(10).
  02 INDEX-LOOP  PIC 99.
  02 MEM-LOOP    PIC X.
  02 CPT-GAC     PIC 99.
  02 COMINFO     PIC 99.
  *END
```

```
MESSAGE "TDSBD-W1" ASSIGN TO TPMONI-UIUD FOR DEBUG
IMPLICIT COMMITMENT
CLASS "A"
AUTHORITY-CODES 1
TRANSACTION-STORAGE TS-TDSBD-W1 SIZE 4200.
MESSAGE "TDSBD-W2" ASSIGN TO TPMONI-UIUD FOR DEBUG
IMPLICIT COMMITMENT
CLASS "A"
AUTHORITY-CODES 1
TRANSACTION-STORAGE TS-TDSBD-W2 SIZE 4200.
MESSAGE "TDSBD-W3" ASSIGN TO TPMONI-UIUD
IMPLICIT COMMITMENT
CLASS "A"
AUTHORITY-CODES 1
TRANSACTION-STORAGE TS-TDSBD-W3 SIZE 4200.
MESSAGE "TDSBD-W4" ASSIGN TO TPMONI-UIUD FOR DEBUG
IMPLICIT COMMITMENT
CLASS "B"
AUTHORITY-CODES 1
TRANSACTION-STORAGE TS-TDSBD-W4 SIZE 4200.
MESSAGE "TDSBD-W5" ASSIGN TO TPMONI-UIUD FOR DEBUG
IMPLICIT COMMITMENT
CLASS "B"
AUTHORITY-CODES 1
TRANSACTION-STORAGE TS-TDSBD-W5 SIZE 4200.
MESSAGE "TDSBD-W6" ASSIGN TO TPMONI-UIUD
IMPLICIT COMMITMENT
CLASS "B"
AUTHORITY-CODES 1
TRANSACTION-STORAGE TS-TDSBD-W6 SIZE 4200.
```

I. Messages Produced by Master Commands

TX*nn* messages appear in response to Master commands.

For each command issued, one of the following messages appears:

- TX54 to indicate that the command is successfully executed.
- TX55 to indicate that the command cannot be executed.

For messages in the range TX50-TX99, the TDS application name appears under IOF.

When a TDS application is started with a master mailbox, TX messages appear on the submitter's console without their TXnn prefixes.

Example of Use of TX Messages:

Consider [M] DISPLAY_TDS.

Messages do not appear with their TXnn prefixes. The combinations of these messages depend on the parameter(s) entered with the command and the TDS Generation options declared in the STDS file, for example:

- When the SIMUL parameter is specified with the RPC SIMULTANEITY Clause, messages TX30 *type 3*, TX09, TX42 and TX54 appear.
- When the SIMUL parameter is specified without the RPC SIMULTANEITY Clause, only the messages TX30 *type 3*, TX09 and TX54 appear.
- When the STATUS parameter is specified, messages TX30, TX05, TX06, TX07, TX04, TX43 and TX54 appear.
- When the RPC_STATUS parameter with the RPC SIMULTANEITY Clause, messages TX40, TX41 and TX54 appear.

DOF 7-PO (Programmed Operator) related messages

- TX59 appears when more than one programmed operator tries to run a TDS application.
- TX60 appears when a command cannot be taken into account because the preceding command is not yet completed.
- TX66 appears when the command is not supported by a programmed operator or when shutdown is in progress.

Miscellaneous Messages

- TX02 is an unsolicited message containing miscellaneous information.
- TX50 appears only during the startup of a TDS application.
- TX57 replies to commands for listing and displaying such as [M] DISPLAY_TX, [M] LIST_TDS_FILE, [M] LIST_TDS_COR and [M] LIST_TDS_SPAWN where scrolling forward enables displaying preceding parts of the message text. This message is not sent to a programmed operator.
- TX62 is an unsolicited messages and does not appear as a reply to a command.
- TX63 and TX64 appear during the startup of a TDS application and when [M] OPEN_TDS_FILE or [M] CLOSE_TDS_FILE are executed.
- TX68 and TX69 are unsolicited messages and do not appear as a reply to a command.

TX00

Convention used for null response to PMOS.

TX01 **REQUESTED TDS *tds* NOT AVAILABLE**

Response to all commands when the TDS is not started or not valid.

TX02 *message*

See text of message.

TX04 **POOL USED (KB) =** *poolused* **POOL SIZE (KB) =** *poolsize*
PSEUDO BUFFERS = *pseudobuf*

Response to DISPLAY_TDS STAT.

TX05

```
INIT.SIMU.COUNT = init_simu      CUR.SIMU.COUNT  = cur_simu
ACC.SESS.ALLOC  = sess_alloc     ACC.SESS.REJEC  = sess_rjct
USED TX COUNT   = txnb           TX ABORT.COUNT  = txabtnb
USED TPR COUNT  = tprnb          TPR ABORT COUNT = tprabtnb
COMMIT COUNT    = cmtnb          DIALOG COUNT    = dialnb
```

Response to DISPLAY_TDS STAT.

TX06

```
TPR ELAPSED TIME = tprelps       TPR CPU TIME     = tprcpu
DEADLOCK COUNT   = ddlcknb       NON CONCUR WAIT  = nconcnb
TABOV ABT COUNT  = tabovnb       WDNAV ABT COUNT  = wdnavnb
LGWAITABT COUNT  = lgwaitnb      DIRTY READ ABORT = dirtabtnb
BUFOVABT COUNT   = bufovnb       SERIALIZATION    = serialnb
```

Response to DISPLAY_TDS STAT.

TX07

```
MAX TM SES     = mxtmnb          CUR TM SES    = curtmnb
MAX XCP1 SES   = mxxcp1nb        CUR XCP1 SES  = curxcp1nb
MAX XCP2 SES   = mxxcp2nb        CUR XCP2 SES  = curxcp2nb
MAX VIRT SES   = dummax          CUR VIRT SES  = virtsesnb
PMOS COR COUNT = pmossesnb       MAX IDLE TIME = idletime
```

Response to DISPLAY_TDS STAT.

TX08 *authcode1 authcode2 ... authcode32*

Response to DISPLAY_TX *spec-tx* AUTHCODE.

TX09

```
MAX.SIMU.COUNT    = init_simu    CUR.SIMU.COUNT  = cur_simu
FROZEN SIMU COUNT = frz_simu
```

Response to DISPLAY_TDS SIMUL.

NOTE:
The last 2 counters are always set to zero because they are not used.

TX10

```
SM LIBRARY 1:  smlib1
SM LIBRARY 2:  smlib2
SM LIBRARY 3:  smlib
```

Response to DISPLAY_TDS SMLIB.

TX11 **XCP2 SERVICE USED** : *xcp2used*
PRIORITY : *priority*
TRANSACTION STORAGE SIZE : *txstosz*
ACCOUNTING : *account*
FORM : *form*

Response to DISPLAY_TX *spec-tx*.

TX12 **MESSAGE** : *name of the transaction*
FIRST ASSIGNED TPR : *name of the tpr*
CLASS : *n (alphabetic)*
IMPLICIT COMMITMENT : { **Y** | **N** }
AUTOMATIC UNMAPPING : { **Y** | **N** }

Response to DISPLAY_TX *spec-tx*.

TX13 **DISPLAY_TX:** *txnm message-text*

where message-text is:

(1) **FILE SECURITY OPTION:**
SUPPRESS { **BEFORE JOURNAL** | **DEFERRED UPDATES** }
(2) **LIST OF AUTHORITY CODES**
(3) **NO CLAUSE SPECIFIED**
(4) **SHARED READ FOR:**
(5) **SUPPRESS CONCURRENT ACCESS CONTROL FOR:**
(6) **TX [MANUALLY] NON CONCURRENT WITH:**
(7) **TX MANUALLY NON CONCURRENT WITH ALL** [**TX**]
(8) **USE DEFERRED UPDATES** [**EXCEPT FOR:**]

Response to

(1) DISPLAY_TX *spec-tx* FSEC
(2) DISPLAY_TX *spec-tx* AUTHCODE
(3) DISPLAY_TX *spec-tx* {FSEC|FSHRD|TXNCONC|FSCAC}
(4) DISPLAY_TX *spec-tx* FSHRD
(5) DISPLAY_TX *spec-tx* FSCAC
(6) DISPLAY_TX *spec-tx* TXNCONC
(7) DISPLAY_TX *spec-tx* TXNCONC
(8) DISPLAY_TX *spec-tx* FSEC.

TX14 *txnm1 txnm2 txnm3 txnm4 txnm5 txnm6*

Response to DTX *.

TX15 *ifn1 ifn2 ifn3 ifn4 ifn5 ifn6*

Response to DTX *file-options* or LSTF * DTLD=0.

```
TX16      LOCKED TRANSACTION  : lock value     { Y | N }
          FOR DEBUG TX        : validate value { Y | N }
          FOR INQUIRY TX      : inquiry value  { Y | N }
          HIDDEN (IN MENU) TX : hidden value   { Y | N }
```

Response to DISPLAY_TX *spec-tx*.

```
TX17      entry-tabulation
```

where entry-tabulation is:

```
(1) IFN    OPEN     MONITORED    PMD
(2) ifn    open     monitored    pmd
```

Response to

(1) LSTF *startx* DTLD.

(2) LSTF * DTLD.

```
TX18      ---------------------------------------------
          ----- TDS: tds    date_time             -----
          -----               heading             -----
          ---------------------------------------------
```

where *heading* is:

(1) **LIST OF CLOSED FILES**

(2) **LIST OF FILES**

(3) **LIST OF OPENED FILES**

Response to

(1) LSTF * OPENED=0.

(2) LSTF *.

(3) LSTF * OPENED.

```
TX19      XCP2 HEURISTIC                    : {COMMIT | BACKOUT | NONE}
          NO DEFERRED RESYNCHRONIZATION     : { Y | N }
          NOT RESTARTABLE COMMITMENT (XCP2): { Y | N }
```

Response to DISPLAY-TX (DTX) *spec-tx* for XCP2 transaction.

```
TX20      SWAP_FILE  ACT     LOGGED_CNTXT  OCCUPANCY   ALLOC_FACTOR
TX21      swapifn    active  ctxnb         occup%      alfactor%
```

Response to DISPLAY_TDS SWAP.

NOTE:

The number of user contexts displayed may not correspond to the number of users logged on because TDS dynamically reports all contexts including breaks and logons.

If only swap file is declared, the allocation factor is set to 100%.

TX22 *cornm1 cornm2 cornm3 cornm4 cornm5*

Response to LIST_TDS_COR * DTLD=0.

TX23 **CORRESPONDENT ADDRESS TX_COUNT TPR_COUNT STATUS TX_NM**
TX24 *cornm coraddr txnb tprnb status txnm*

Response to LIST_TDS_COR * { TM | * } DTLD.

TX25
```
-----------------------------------------
----- TDS: tds    date_time           -----
-----              heading            -----
-----------------------------------------
```

where *heading* is:
(1) **CHARACTERISTICS OF THE TRANSACTION**
(2) **LIST OF CORRESPONDENTS**
(3) **LIST OF POOLS**
(4) **LIST OF SESSIONS WITH SPAWNS**
(5) **LIST OF SPAWNEES WITH QUEUE LENGTHS**
(6) **LIST OF TRANSACTIONS.**

Response to
(1) DTX *spec-tx*.
(2) LIST_TDS_COR.
(3) LIST_COR_POOL.
(4) LIST_TDS_SPAWN DTLD=0.
(5) LIST_TDS_SPAWN DTLD.
(6) DTX *.

TX26 **CORRESPONDENT ADDRESS COR_BACKUP**
cornm coraddr corbck

Response to LIST_TDS_COR * TM DTLD NG.

TX27 **CORRESPONDENT ADDRESS NB_OF_POOLS**
cornm coraddr poolnb

Response to LIST_TDS_COR * XCP2 DTLD NG=0.

TX28 **STATE:** *state* / **TYPE:** *cor-typ* / **LIST:** *list* / **OPTION:** *option*

Response to LIST_TDS_COR * TYPE=*.

TX29 **CORRESPONDENT ADDRESS MXALCSES CUMALCSES REJALCSES**

cornm coraddr mxalses cumalcses rejalcses

Response to LIST_TDS_COR * XCP1 DTLD NG=0.

TX30
```
-------------------------------------------
----- TDS: tds    date_time            -----
-----               heading            -----
-------------------------------------------
```

where *heading* is:

(1) **CHARACTERISTICS OF THE POOL**
(2) **CURRENT SEARCH RULES OF SM LIB**
(3) **CURRENT SIMULTANEITY LEVELS**
(4) **GENERAL TDS STATISTICS**
(5) **LIST OF SWAP FILES**

Response to
(1) LIST_COR_POOL
(2) DISPLAY_TDS SMLIB
(3) DISPLAY_TDS SIMUL
(4) DISPLAY_TDS
(5) DISPLAY_TDS SWAP.

TX31 *poolnm1 poolnm2 poolnm3 poolnm4 poolnm5 poolnm6*

Response to LIST_COR_POOL cor POOL=*.

TX32 **POOL MAXSESNB MINWINSC MINWINTG AUTOACT DRAINSC DRAINTG MAXSYNC**
poolnm maxsesnb minwinsc minwintg autoact drainsc draintg maxsync

Response to LIST_COR_POOL *cor pool* DTLD NG.

See CALL "DISP-POOL" Procedure in *TDS COBOL Programmer's Guide*.

TX33 **CORRESPONDENT ADDRESS COR_BACKUP PRIM ACTIVE INITWK**
cornm coraddr corbck prim activ initwk

Response to LIST_TDS_COR * XCP1 DTLD NG.

TX34 **CORRESPONDENT ADDRESS COR_BACKUP PRIM PRL WIN SYNLVL**
cornm coraddr corbck prim prl win synlv

Response to LIST_TDS_COR * XCP2 DTLD NG.

See *CPI-C/XCP2 User's Guide*.

TX35 **POOL CURSESNB CURWINSC CURWINTG TRSTSESNB**
poolnm cursesnb curwinsc curwintg trstssnb

Response to LIST_COR_POOL cor pool DTLD NG=0.

See CALL "DISP-POOL" Procedure in *TDS COBOL Programmer's Guide*.

TX36 **ATTRIB ACTSESNB FRZSESNB TRSTSESNB**
attrib actsesnb frzsesnb trstssnb

Response to LIST_COR_POOL *xcp1-cor* DTLD NG=0.

See CALL "DISP-POOL" Procedure in *TDS COBOL Programmer's Guide*.

TX38 **COR HIGH_SPW_NB MED_SPW_NB LOW_SPW_NB TIMER_SPW_NB**
cornm highspnb medspnb lowspnb timspnb

Response to LIST_TDS_SPAWN * DTLD.

TX39 **RPC SERVICE USED:** *rpcused*

Response to DISPLAY_TX *spec-tx*.

TX40
```
-----------------------------------------
-- TDS=tdsname  hh:mm:ss Month Day, Year --
-----      TDS RPC STATISTICS       -------
-----------------------------------------
```

Response to DISPLAY_TDS RPC_STATUS.

TX41
```
USED RPC TX COUNT   =rpctxnb   USED RPC TPR COUNT =rpctprnb
MAX RPC TX COUNT    =mxrpctxnb CUR RPC TX COUNT   =currpctxnb
WAITING TPR MEAN    =wtprmean  RPC TPR ELAPSE TIME=rpctprelps
WAITING RPC TPR MEAN=wrpctprmean
```

Response to DISPLAY_TDS RPC_STATUS.

TX42 **MAX RPC SIMU COUNT=***max_rpcsimu* **CUR RPC SIMU COUNT=***cur_rpcsimu*

Response to DISPLAY_TDS SIMUL.

TX43 **WAITING TPR MEAN =** *wtprmean*

Response to DISPLAY_TDS STATUS.

TX44 **XA SERVICE USED :** *usexa* {Y|N}

Response to DISPLAY_TX *spec-tx*.

TX45

```
-----------------------------------------
-- TDS=tds     hh:mm:ss Month Day, Year  --
-----             heading           -------
-----------------------------------------
```

where *heading* is:

(1) **CURRENT XA PARAMETERS**
(2) **CURRENT XA PARAMETERS AND RM**

Response to

(1) DTDSXA RM_STAT=0.
(2) DTDSXA RM_STAT=1.

TX46 **XA_RESYNC_DELAY =** *xarsyncdlay*

Response to DISPLAY_TDS_XA.

TX47 **RM_IDENT CNCT SES RESYNC START**

TX48 *rmident* ***cnct ses resync start***

Response to DISPLAY_TDS_XA RM_STAT=1.

TX49 **NO XA RM YET CONNECTED**

Response to DISPLAY_TDS_XA RM_STAT=1 instead of TX47 and TX48.

TX50 **TDS:***tds* { **COLD RESTART IS PERFORMED** }
{ **STARTED WITH MASTER MAILBOX:** *mbx* }
{ **STARTED; YOU ARE MASTER TERMINAL OPERATOR** }
{ **WARM RESTART IS PERFORMED** }

Startup message.

TX51 *cornm* **DISCONNECTED FROM** { *coraddr* | **TDS** *tds* }

Unsolicited message. Perhaps a response to the CANCEL_TDS (CTC).

TX52 *cmdnm* **COMMAND NOT PERFORMED,** { **INVALID** | **UNKNOWN** } **TX:** *txnm*

Response to DTX or MDTX.

TX53 **TDS:***tdsname* **SHUTDOWN**

Shutdown message, after TTDS.

TX54 **TDS:***tdsname*, *cmdnm* **COMMAND COMPLETED**

Response to normal execution.

TX55 **TDS:***tdsname*, *cmdnm* **COMMAND NOT PERFORMED** *rc*

Command has failed to execute. Check the return code and the sicid.

TX56 *cmdnm* { **UNKNOWN OR NO MATCHING OBJECT:** *objnm* }
{ **WRONG TYPE OF CORRESPONDENT** }

Response to commands which deal with correspondents.

TX57 *cmdnm* **SCROLLING UP MORE? (ENTER Y/N)**

Startup message or response to some list or display commands.

TX58 **OVERFLOW ON PMOS COMMAND/RESPONSES QUEUE**

TX59 **ONLY ONE PROGRAMMED OPERATOR ALLOWED TO CONTROL TDS**

Response occurs when several programmed operators try accessing a TDS application.

TX60 **COMMAND** *cmdnm* **IGNORED; RETRY LATER**

Returned when a command is submitted by an IOF or PMOS correspondent who has not the turn, that is, when a transaction is already running for its current session.

TX61 **SM LIBRARY NOT FOUND; ISSUE DISPLAY_TDS TO CHECK**

Response to MDTDS SMLIB.

TX62 **SWAP CONTEXT MAXIMUM REACHED**

Unsolicited message.

TX63 cmdnm: ifn action

where *action* is:
(1) **CLOSED**
(2) **OPENED**

Response to
(1) CLTF or startup message: file assigned and closed
(2) OTF or startup message.

TX64 *cmdnm*: *ifn* **NOT** *action* *gr4*

where *action* is:
(1) **CLOSED**
(2) **OPENED**

Response to
(1) CLTF
(2) OTF or startup message.

TX65 **UNKNOWN MASTER COMMAND:** *cmdnm*

TX66 *cmdnm* **COMMAND NOT** *result*

where *result* is:
(1) **PERFORMED; SHUTDOWN IN PROGRESS**
(2) **SUPPORTED FOR PROGRAMMED OPERATOR**

Response to
(1) TTDS command has been issued.
Programmed operator cannot submit another command.
(2) EXEC_TDS or MDTX for programmed operator.

TX67 *cmdnm* *tobjnm* *action*

where *action* is:
(1) **LOADED INTO MAIN MEMORY**
(2) **UNLOADED FROM MAIN MEMORY**

Response to
(1) LDTMEM.
(2) UNLDTMEM.

TX68 *cornm*: **UNRECIO ON** { **BEFORE JOURNAL** }
{ **CI NUMBER:** *cinb* **FOR** *efn* }

Unsolicited message - The input/output operation failed.

TX69 *reason*: *txnm*, **SPAWN CANNOT BE STARTED FOR** *cornm*

where *reason* is:

- **ACCESS DENIED TO TX**
- **UNKNOWN TX**

TX70 *poolnb* **POOL(S) OPENED TOWARD** *cornm*

Response to OCPOOL.

TX71 **POOL** *poolnm* { **CLOSED** | **MODIFIED** | **OPENED** } **BY** *cornm*

Unsolicited message which appears on the remote correspondent following a command which acts on pools.

TX72 *cmdnm* **COMMAND NOT PERFORMED FOR** *poolnm reason*

where *reason* is:
(1) **NETGEN REASON:** *netgen*
(2) **PPC REASON:** *ppc_rc/ppcreason CC=cplcode*
(3) **TDS REASON:** *tdsreason*

Response to

(1) OCPOOL
(2) MDCPOOL
(3) CLCPOOL.

cplcode is an internal value giving an error number inside the H_PPC_COM_S procedure.

Action: contact your technical support.

Refer to Appendix K.

TX73 *cmdnm* **COMMAND NOT PERFORMED** *reason*

where *reason* is:
(1) **NETGEN REASON:** *netgen*
(2) **PPC REASON:** *ppc_rc/ppcreason CC=cplcode*
(3) **TDS REASON:** *tdsreason*

Response to
(1) OCPOOL
(2) MDCPOOL
(3) CLCPOOL.

cplcode is an internal value giving an error number inside the H_PPC_COM_S procedure.

Action: contact your technical support.

Refer to Appendix K.

TX74 *cmdnm* **COMMAND NOT PERFORMED FOR** *cornm* **TDS REASON:** *tdscode*

Response to MDCPOOL, or CLCPOOL for XCP1 or DUMMY correspondents.

See Appendix K.

TX75 *cmdnm* **LIMIT IS REACHED, ONLY** *sessnb* { **CREATED** | **DELETED** }

Response to MDCPOOL, or CLCPOOL for XCP1 or DUMMY correspondents.

TX76 **TDS:** *tds* **SUPERVISION** { **ACTIVATED** | **DEACTIVATED** }

Response to SPRVT.

TX77 **TDS:** *tds* **SUPERVISION** { **ABNORMALLY DEACTIVATED** } *gr4*
{ **CANNOT BE ACTIVATED** }

Response to SPRVT.

TX78 **TDS:** *tds* **ERROR LEVEL** *erlvl*
DETECTED BY SUPERVISION OF { **SESSION** | **PROCESS** }

Unsolicited message sent by the supervision manager.

TX79 *cmdnm*: **NO SPAWNED TRANSACTION ON THE SPECIFIED USER**

Response to OCPOOL, MDCPOOL for xcp1, dummy cor.

TX80 *cmdnm*: *cornm* **{ CONNECTED** | **NOT CONNECTED; REASON:** *reason* }

Response to OCPOOL, MDCPOOL for xcp1, dummy cor.

TX91 CUR TCP SES = *cur_tcpses*

Response to DISPLAY_TDS_STATUS.

TX92 **TDS SOCKET** *action*

where *action* is:

(1) **BEING CLOSED**
(2) **CLOSED**
(3) **OPENED**.

Unsollicited message which follows CLTS or OTS commands when they are successful.

TX93
```
-----------------------------------------
-- TDS=tdsname hh:mm:ss Month Day, Year  --
-----  CURRENT TCP/IP PARAMETERS    -------
-----------------------------------------
```

Response to DISPLAY_TDS_SOCKET STATUS.

TX94 **SOCKET INTERFACE IS =** *s_interface*

Response to DISPLAY_TDS_SOCKET STATUS.

TX95 *cmdnm*: **ATTACH RM REFUSED COR**=cornm **POOL**=poolnm **TP**=tpnm **US**=usernm **RS**=reason **CC**=cplcode **RC**=rc **TXT**=text.

This message is sent to the master when a user is unable to start a transaction on a remote application .

This message is filtered: the TDS Master command "MDTDS XCP2TRC=80" must be issued to allow it to be displayed.

reason is an internal error number in the procedure H_TP7_EVTMGT.

cplcode

text:

- TXINIT ERROR
  The significant CC values are:
  CC=9 attempting to invoke the "CLOSE" transaction
  CC=8 the transaction is declared FOR INQUIRY and the asked security level is different from NONE
  CC=16 the transaction name is illegal (ex: LOGON, RESTART, SYNCPEVT,...) or the transaction is locked, or mismatch between authority codes of the invoked transaction and those defined in the catalog. XCP2 mailbox with TDSCODE must be declared in application list in the project of the user for whom the TX is to be running. See *CPI-C/XCP2 User's Guide* (paragraph Authority Codes Checking).
- CANNOT ALLOCATE SESSION ENTRY
  CC=17 internal error. TDS table entry creation is not possible.
- CANNOT ALLOCATE TRANSACTION PHASE ENTRY
  CC=17 internal error. TDS table entry creation is not possible.
- CANNOT ALLOCATE COMMIT ENTRY
  CC=17 internal error. TDS table entry creation is not possible.
- TERMINATE TDS IN PROGRESS
  CC=16 TTDS weak or strong has been issued.
- TRUNC FROM H_PPC_GET_TX
  CC=6 internal error: length or number of TIPs (Transaction Input Parameters) is not correct.

- TPNAME>8
  CC=9 transaction name exceeds 8 characters.
- TPNAME NOT FOUND IN TX TABLE
  CC=9
- TX NOT USABLE FOR XCP2
  CC=4 generation clause "XCP2 SERVICE USED" missing for the invoked transaction.
- INVALID PARTNER FROM H_PPC_DISPL_COR
  CC=2 internal error

Action: For internal error, contact your supplier, otherwise modify the application.

TX99 *cmdnm*: **POOLREQ REFUSED POOL**=poolnm **RC**=rc **RS**=reason **CC**=cpicode **TXT**=text

This message is sent to the master when a problem occurs during the processing of a remote pool request.

reason is an internal error number.

cplcode is the requested function:

CC=0 for OPENING the POOL (OCPOOL command)
CC=1 for MODIFYING the POOL (MDCPOOL command)

CC=2 for CLOSING the POOL (CLCPOOL command)

text:

- ERROR: INIT_XCP2_VALUES
  Internal error. Communication with partner is not established or is not available.
- ERROR: GET_POOL_REQUEST_PPC
  Internal error. Abnormal return code returned from PPC when trying to get information on pool.
- ERROR: OPEN_POOL_XCP2_ACCEPTOR
  Internal error. If the return code is not "ABOPROC" this message is preceded by the message TX02, TX56, TX72 or TX73.
- ERROR: MOD_POOL_XCP2_ACCEPTOR
  Internal error. This message is preceded by the message TX72 or TX73.

- ERROR: CLOSE_POOL_XCP2_ACCEPTOR
  Internal error.

- OPEN POOL REJECTED DUE TO PVNTC
  Pool request is rejected due to the previous master command PVNTC.

- ERROR: H_PPC_REPLY_POOL
  Internal error.

Action: If internal error, contact your supplier.

List of Metalanguage Terms

action	action taken on command execution
activ	active
active	pertains to active swap file
actsesnb	current number of connected sessions
alfactor%	activity distribution over all swap files
attrib	pool attribute
authcodei	authority code of the transaction in a list
autoact	number of winner auto-active sessions
bufovnb	number of transactions aborted because of buffer overflow
cinb	number of the CI
cmdnm	name of command
cmtnb	cumulated number of commitments declared
cnct	number of physical ORACLE connections
coraddr	address of the correspondent
corbck	correspondent backup
cornm	name of the correspondent
cornmi	correspondent in a list
cpicode	complementary code
ctxnb	number of user contexts, usually 2 per user.
cumalcses	number of sessions allocated and deallocated so far
cur_rpcsimu	current RPC simultaneity count
cur_simu	current simultaneity level
cur_tcpses	current number of TCP sessions
currpctxnb	current RPC transaction count
cursesnb	current number of sessions
curtmnb	current number of TM (Terminal Manager) sessions
curwinsc	current number of winner sessions for source

curwintg	current number of winner sessions for target
curxcp1nb	current number of XCP1 sessions
curxcp2nb	current number of XCP2 sessions
dialnb	cumulated number of conversations held.
dirtabtnb	number of times access is denied due to unstable data
ddlcknb	number of times a TPR has been aborted due to the return code DEADLOCK received from GAC
drainsc	drain source: how local allocation requests are handled
draintg	drain target: how remote allocation requests are handled
dummax	the maximum number of virtual (dummy) sessions
efn	external file name
erlvl	error level
frz_simu	frozen simumltaneity level
frzsesnb	current number of frozen sessions
gr4	contents of return code
heading	title in the TDS header banner
highspnb	number of high spawns
ifn	internal file name
ifni	ifn in a list
init_simu	initial simultaneity level
initwk	initwork applicable to the active correspondent
lgwaitnb	number of transactions aborted due to a longwait
lowspnb	number of low spawns
max_rpcsimu	maximum RPC simultaneity count
maxsesnb	maximum number of sessions
maxsync	maximum sync_level: – C for confirm – S for syncpoint
mbx	name of the master mailbox
medspnb	number of medium spawns
message-text	text of the message variant
minwinsc	minimum number of winner sessions for source
minwintg	minimum number of winner sessions for target
mxalcses	maximum allocated sessions reached by correspondent
mxrpctxnb	maximum RPC transaction count
mxtmnb	maximum number of TM (Terminal Manager) sessions

mxxcp1nb	maximum number of XCP1 sessions
mxxcp2nb	maximum number of XCP2 sessions
nconcnb	number of transactions wait in non-concurrent access
netgen	reason given in the Network Generation
objnm	name of object
occup%	rate of occupation regarding the size of the swap file
pmossesnb	maximum number of PMOS correspondents
poolnb	number of pools
poolnm	name of the pool
poolnmi	name of the pool in a list
poolsize	total size in kbytes of the pool
ppc_rc	return code given by PPC
ppcreason	return code and reason given by PPC
poolused	size in kbytes of the pool occupied
prim	primary
prl	parallel
pseudobuf	number of pseudo-buffers
reason	gives cause why command is rejected
rc	return code
rejalcses	number of allocation requests rejected.
resync	XA resynchronization
result	result of command execution
rmident	identification of the XA Resource Manager (ORACLE database)
rpctprelps	RPC tpr elapse time
rpctprnb	used RPC tpr count
rpctxnb	used RPC transaction count
rpcused	RPC service used
s_interface	socket interface chosen for TCP-IP way of communication: OPEN7 or GTXI
serialnb	number of processes executed consecutively
ses	number of active XA sessions
sess_alloc	total number of sessions allocated and deallocated
sess_rjct	number of allocation requests rejected
sessnb	number of sessions
smlibi	name of the SM library.
start	status of last XA session start attempt

status	status of the user: "F" = in frozen state "G" = waiting for a gac event "I" = idle (logged but no transaction running) "P" = processing "T" = waiting for a timer event "V" = waiting for a vcam event "W" = waiting for a TDS or other event
swapifn	ifn of swap file
synlv	syncpoint level
tabovnb	number of transactions aborted on table overflow
tds	name of the TDS application
tdscode	reason given by TDS
tdsreason	reason given by TDS
text	additional information
timspnb	number of timer spawns
tobjnm	name of the TDS object
tpnm	name of the transaction program
tprabtnb	cumulated number of tpr's aborted
tprcpu	time taken by CPU to execute tpr's
tprnb	number of tpr's or the cumulated number of tpr's run
tprelps	time taken by tpr
trstssnb	current number of transient sessions
txabtnb	cumulated number of transactions aborted
txnb	number of cumulated transactions run
txnm	name of the transaction
txnmi	transaction in a list
txstosz	transaction storage size
usernm	name of the user
virtsesnb	current number of virtual (dummy) sessions.
wdnavnb	cumulated number of times transaction is rolled back due to lack of space to run Deferred Updates
win	winner
wrpctprmean	waiting RPC tpr mean
wtprmean	waiting tpr mean
xatsyncdlay	XA resynchronisation delay value

J. Return Codes on XCP2 Master Commands

This Appendix describes the return codes that may appear when the [M] OPEN_COR_POOL, [M] CLOSE_COR_POOL and [M] MODIFY_COR_POOL fail to execute. These message appear in one of two formats:

cmd COMMAND NOT PERFORMED FOR *poolnm* PPC REASON: *ppc_rc/ppcreason*

cmd COMMAND NOT PERFORMED PPC REASON: *ppc_rc/ppcreason*

A message is sent in the first type of format when the command is attached to a named XCP2 pool. Otherwise messages appear in the second format. The part of the message in lowercase indicates the variable part of the message. Uppercase letters indicate the fixed part of the message.

When ppc_rc = PPC XX, NOTDONE, *ppcreason* equals *code, subcode* where code and subcode are numbers (subcode is set to 0 when the code is self-explanatory). For each command, the meanings of code and subcode are given.

The following tables give symbolic names of *codes* and *subcodes* and their *values*. Use these lists to find the correspondence between *values* and *codes/subcodes* for [M] OPEN_COR_POOL, [M] CLOSE_COR_POOL and [M] MODIFY_COR_POOL.

Codes/Subcodes and Values of XCP2 Master Command

Code	Value		Subcode	Value
alloc-fail	2		ok	0
command-race-reject	5		inv-mode	6
dealloc-abend	7		nortry	15
param-error	9		retry	23
pool-closed	10		sync-nosup	27
resfail	16		max-number-conv	36
tp-node-ses-limit	18		limit-conflict	46
unrec-mode-name	20			
tp-node-stopping	25			
resync-fail	26			

Effect of Codes/Subcodes on COR_POOL Master Commands

Code/Subcode	OPEN	CLOSE	MODIFY
2/15	*yes*	*yes*	*yes*
2/23	*yes*	*yes*	*yes*
2/27	*yes*	*yes*	*yes*
2/36	*yes*	*yes*	*yes*
5/ 0	*yes*	*yes*	*yes*
7/ 0	*yes*	*yes*	*yes*
9/46	***no***	***no***	*yes*
10/ 0	*yes*	***no***	***no***
16/15	*yes*	*yes*	*yes*
18/ 0	*yes*	***no***	*yes*
20/ 0	*yes*	*yes*	*yes*
25/ 0	*yes*	*yes*	*yes*
26/ 0	*yes*	***no***	***no***

Explanation of Code and Subcode

2/15 `alloc-fail / nortry`
2/23 `alloc-fail / retry`
2/27 `alloc-fail / sync-nosup`
2/36 `alloc-fail / max-number-conv`

These combinations report that PPC cannot establish a dialog with its partner.

5/ 0 `command-race-reject / ok`

Another master command is requested at same time for the same pool.

7/ 0	`dealloc-abend / ok` The system transaction was interrupted.
9/46	`param-error / limit-conflict` A modification cannot be requested.
10/ 0	`pool-closed / ok` Remote application rejects the request because: – either it issued [M] PREVENT_NEW_TDS_COR or the corresponding COBOL call – or it has not yet started such as waiting for the master operator to log on.
16/15	`resfail / nortry` As a result of a protocol error, the session fails.
18/ 0	`tp-node-ses-limit / ok` Maximum session number requested for pool exceeds maximum session number authorized for the TP-node.
20/ 0	`unrec-mode-name / ok` Remote application cannot identify pool specified by input parameters.
25/ 0	`tp-node-stopping / ok` Request cannot be performed since TP-node is stopped.
26/ 0	`resync-fail / ok` An incident ocurred while pool was being opened during the resynchronization step.

K. TDS Main Disconnection Codes

The Main disconnection codes at the close of the TDS session are:

`35:`	Protocol violation detected on Data Receive
`36-38:`	Protocol violation detected on Interrupt Processing
`39:`	Disconnection on submission of TTDS, CTC commands
`3C:`	A TM correspondent, which is not master, is not allowed to connect to a TDS-Web using the USE TDS-SPECIALIZED-WEB7 clause. For a TS > 9662, the code 3C is sent. For TS = 9662, the code 3D (instead of 3C) is sent.
`3D:`	Connection denied (Authority Code 0 for User). See also 3C.
`3E:`	Device Header not found by VCAM but has been negotiated
`3F:`	Level 3 data (EGI) *received* after *send* Level 5 (EOS) by TDS
`43:`	VCAM Check sent by VCAM

Complementary codes are available in the System Trace if [M] MDTDS TRACE=FF is issued.

L. TDS Generation Keywords

Keywords used in the TDS Generation should not be used to name files or ifn's.

```
ABORT
ABORT-LEVEL
ACCESS
ACCOUNTING
ACTIVE
ADDED
AND
ARE
AREA
AREAS
ASSIGN
ATTACH
AUTHORITY-CODES
AUTOMATIC

AUXILIARY

BACKOUT
BATCH
BEFORE
BTNS
BY

CANCELCTX
CD-IN
CLASS
CMA
CMA-DEF
CMA-ENDDEF
CMSC
COMMIT
COMMITMENT
COMMON-STORAGE
CONCURRENT
CONTROL
DYNAMIC-SUPERVISION
DYNAMICALLY

ENTRIES
EXCEPT
EXTEND
EXTENSION
EXTERNAL

FD
FILE-CONTROL
FILE-DEFINITION
FILE-INTEGRITY
FILE-OPENING
FILE-OPENING OPTION
FILES
FOR
FORCE
FORM
FORMAT
FORMATTED

GATEWAY
GXTI

HEADER
HEURISTIC
HIDDEN
HIGH
HOST

IDLE-TIME
IDS-DEFINITION
IMMEDIATE
IMPLICIT
```

```
CONTROLLED
CONVERSATIONS
CORRESPONDENT

DATE
DB
DDMMYY
DEBUG
DEFAULT
DEFERRED
DELAY
DELIVERY
DUMMY
JOURNAL
JPPC

LAST
LEVEL
LOCKED
LOGGING

MAILBOX
MANUAL
MANUALLY
MASTER
MAXIMUM
MEDIUM
MEMORY
MESSAGE
MESSAGE-LENGTH
MMDDYY
MODULES
MONO-PHASE
MSEC

NO
NO-DEFER-RESYNC
NO-MESSAGE-COMMIT
NO-RESTART
NON-CONCURRENT
NONE
NOT-WARM-RESTARTABLE
NUMBER

OF
OPEN7
OPS

INCLUDED
INITIALIZATION
INIT-WORK
INPUT
INPUT-OUTPUT
INQUIRY
IPS
IQS
IQS-DEFINITION
IS
ITX


SCHEMA
SECONDARY
SECTION
SEGMENTS
SELECT
SERVICE
SERVICE-MESSAGE
SESSION-CONTROL
SESSION
SESSIONS
SHARABLE
SHARED
SHARED-STORAGE
SIMULTANEITY
SIZE
SPECIAL-CHAR
SUPPRESS

TABLE
TCP-IP
TDS
TDSTX-MESSAGE
TDS-FILE-DEFINITION
TDS-RPG-FILE
TERMINAL
TERMINALS
THINK-TIME
TO
TPR
TPR-NAME
TPR-TIME-LIMIT
TRACE
TRAILER
TRANSACTION
```

OPTION
ORACLE-DEF
ORACLE-ENDDEF
ORGANIZATION
OTX
OUTPUT

PAGES
PASSIVE
PER
PRIMARY
PRIORITY
PRIVATE-STORAGE
PROCESS
PROCESSING-CONTROL
PROCESSING-MODE
PROGRAM-ID
PROMPT
PROTOCOL

READ
REALMS
RELEASE
RESERVE
RFU
ROLE
ROLL-BACK
RPC
RPG-FILE
RPG-SPECIAL-FILE

TRANSACTIONS
TRANSACTION-MENU
TRANSACTION-STORAGE
TYPE

UNITS
UNMAPPING
UPDATES
USE
USED
USER-JOURNAL

VIEWS
VIRTUAL

WAIT
WATCHED
WITH
WORKING-STORAGE
WATCHTIME

XA
XA-RESYNC-DELAY
XCP1
XCP2
XCP2-SYNCPEVT
XCP2-WAITTIME

YYMMDD

M. TP7PREP Errors and Responses

The destination of the message depends on the type of message and on the setting of the MSG parameter:

- the submitter's terminal and/or the report file
- or only the report file.

The *Destination* is given for each message.

TP7PREP messages prefixed with **TV***nn* have the format of the following example:

TV02 **UNABLE TO READ STEP OPTIONS. CHECK STEP JCL STATEMENT**
RC=*cccccccc-> sicid, return code*

where:

- **TV02** is the numerical order in which these messages are listed
- **RC** represents the Return Code introducing:
  - *cccccccc* as the hexadecimal return code
  - *sicid* as the System Integration Component issuing the message
  - *return code* as the mnemonic.

The *Type* of message determines whether the TP7PREP utility can continue running. All ***Fatal*** messages terminate TP7PREP. *Warning* messages do not stop TP7PREP.

TV01 **UNABLE TO OPEN REPORT FILE. CHECK ASSIGN JCL STATEMENT**
RC=*cccccccc-> sicid, return code*

Destination: Depending on MSG parameter, to the submitter's terminal and/or the Report File.

Type: ***Fatal***

Meaning: An internal error was encountered when opening the report file.

Action: Check the ASSIGN JCL statement of the report file.

TV02 **UNABLE TO READ STEP OPTIONS. CHECK STEP JCL STATEMENT**
RC=*cccccccc-> sicid, return code*

Destination: Depending on MSG parameter, to the submitter's terminal and/or the Report File.

Type: ***Fatal***

Meaning: An internal error was encountered when using the step OPTIONS.

Action: Check the JCL statement of the step.

TV03 **INCORRECT ARGUMENT. CONSULT THE REPORT FILE**

Destination: Depending on MSG parameter, to the submitter's terminal and/or the Report File.

Type: ***Fatal***

Meaning: An internal error was encountered during parameters analysis.

Action: Consult the report file.

TV04 **UNABLE TO ACCESS efn**
RC=*cccccccc-> sicid, return code*

Destination: Depending on MSG parameter, to the submitter's terminal and/or the Report File.

Type: ***Fatal***

Meaning: This message is sent when trying to retrieve the private catalog directory.

Action: Refer to error and return code documentation.

TV05 **NO** *tdsname*.{ **CTLM** | **CTLN** | **RECOV** | **SWAP** } **ALLOCATED**

Destination: Depending on MSG parameter, to the submitter's terminal and/or the Report File.

Type: ***Fatal***

Meaning: The file specified has not been allocated.

Action: Launch TP7PREP with appropriate parameters.

TV20 **UNSUCCESSFUL ALLOCATION FOR** ***efn***
RC=*cccccccc-> sicid, return code*

Destination: Depending on MSG parameter, to the submitter's terminal and/or the Report File.

Type: ***Fatal*** or Warning

Meaning: An error was encountered during allocation phase.

Action: See *Messages and Return Codes Directory*.
See Paragraph 2.4.2 Syntax of TP7PREP Parameters.
Note that the PPCLOG member relates to the XCP2 parameter.

TV21 **WAITS FOR VOLUME** *volnm*

Destination: Depending on MSG parameter, to the submitter's terminal and/or the Report File.

Type: *Warning*

Meaning: The volume is not currently accessible.

Action: As appropriate.

TV22 **WAIT TIME LIMIT EXCEEDED**

Destination: Depending on MSG parameter, to the submitter's terminal and/or the Report File.

Type: ***Fatal***

Meaning: WAIT time limit is reached.

Action: As appropriate.

TV23 **UNABLE TO** { **ASSIGN** | **CLOSE** | **DEASSIGN** | **INIT** | **OPEN** } *efn*
RC=*cccccccc-> sicid, return code*

Destination: Depending on MSG parameter, to submitter's terminal and/or the Report File.

Type: ***Fatal***

Meaning: Depends on failure.

Action: Refer to *Messages and Return Codes Directory*.

TV24 ***sfn*** **CREATED INTO** *efn*

Destination: To the Report File.

Type: Information

Meaning: *sfn* means TP7GEN *tdsname* created into SLLIB file.

Action: None.

TV25 **SUCCESSFUL INITALIZATION OF** *sfn*

Destination: To the Report File.

Type: Information

Meaning: *sfn* means *tdsname*.SMLIB.

Action: None.

TV26 **UNSUCCESSFUL UPDATE IN CATALOG: INCRSIZE, FOR** *efn.* **BYPASSED**
RC=*cccccccc->sicid, return code*

Destination: To the submitter's terminal and the Report File.

Type: Warning

Meaning: Information could not be modified in the catalog for the file specified. INCRSIZE information relates to a file allocated on an FBO disk, for which the conversion in the matching unit (blocks) could not be recorded in the catalog. The real information, located on the disk label, may be displayed using the LSF command.

Action: Refer to error and return code documentation to correct the problem.

TV40 **TP7PREP *text* FOR TDS:** *tdsname*

Destination: To the submitter's terminal and the Report File.

Type: depends on *text:*
ABORTED : ***Fatal***
COMPLETED : Information
COMPLETED WITH WARNINGS: *Warning*

Meaning: For *Warning*, previous WARNINGs have already appeared.

Action: As appropriate

TV50 **MISSING MANDATORY KEYWORD** *keyword*

Destination: To the Report File.

Type: Information

Meaning: An internal error was encountered when analyzing the step option.

Action: Modify TP7PREP step JCL statement.

TV51 **SYNTAX ERROR FOR** *keyword*

Destination: To the Report File.

Type: Information

Meaning: An internal error was encountered when analyzing the step option.

Action: Modify TP7PREP values.

TV52 **ERRONEOUS VALUE FOR** *keyword*

Destination: To the Report File.

Type: Information

Meaning: An internal error was encountered when analyzing the step option.

Action: Modify TP7PREP values.

TV53 **UNKNOWN KEYWORD** *keyword*

Destination: To the Report File.

Type: Information

Meaning: An internal error was encountered when analyzing the step option.

Action: Modify TP7PREP step JCL statement.

TV54 **ILLEGAL DEVICE VOLUME** *volattr*

Destination: To the Report File.

Type: Information

Meaning: The device is not available.

Action: Change TP7PREP values.

TV55 *volnm* **IS NOT A RESIDENT VOLUME DISK**

Destination: To the Report File.

Type: Information

Meaning: The volume does not belong to the resident volumes list.

Action: Change the volume for a resident one.

TV56 **{ SUCCESSFUL ALLOCATION |**
{ SUCCESSFUL | UNSUCCESSFUL } DEALLOCATION } FOR *efn*

Destination: To the Report File.

Type: Information

Meaning: As in message.

Action: None.

TV57 KEYWORD *keyword* **NO LONGER USED**

Destination: To the Report File.

Type: *Warning*

Meaning: *Keyword* is no longer used.

Action: Modify step options parameters.

TV58 THE NUMBER OF SWAP FILES IS LIMITED TO *number*

Destination: To the Report File.

Type: ***Fatal***

Meaning: Number depends on Technical Status.

Action: Modify NBSW parameter.

TV59 WRONG SYSFILE, FILESTAT VALUE FOR HA USE

Destination: To the Report File.

Type: ***Fatal***

Meaning: All files must be cataloged when HA is used.

Action: Correct as appropriate.

TV60 WRONG FILEP VALUE WITH NBSW VALUE

Destination: To the Report File.

Type: ***Fatal***

Meaning: Number of SWAP files is inconsistent with the name of the SWAP file. For example, SWAP5 file is inconsistent with NBSW=1.

Action: Correct as appropriate.

TV61 **WRONG FILEP VALUE WHEN** *keyword=value*

Destination: To the Report File.

Type: ***Fatal***

Meaning: This message signals an inconsistency between the user-defined *value* for the *keyword* identifying the TP7PREP parameter. For example, it is inconsistent to specify FILEP=GMEM and GMEM=N, or to process FILEP=SLLIB when DEAL=Y.

Action: Correct as appropriate.

TV87 **TP7PREP ABORTED**

Destination: To the submitter's terminal and the Report File.

Type: ***Fatal***

Meaning: Final abort message, preceded by at least one fatal message which report the error cause.

Action:
1. Correct the error cause reported by the preceding message
2. If necessary, reset the <tdsname> directory in a coherent state
3. Start TP7PREP again.

TV88 **EXCEPTION** ***cd02*** **AT ADDRESS** *address*

Destination: To the Report File.

Type: ***Fatal***

Meaning: Error cannot be corrected by user.

Action: Call the Service Center.

TV89 **ABNORMAL** *gr4* **AT ADDRESS** *address*

Destination: To the Report File.

Type: ***Fatal***

Meaning: Error cannot be corrected by user.

Action: Consult the Service Center.

N. TP7GEN Errors and Responses

N.1 TP7GEN LM Messages

All error messages are reported in the JOR and some of them which are messages issued by the TP7GEN JCL appear at the submitter's terminal.

TP7GEN messages prefixed with **TG***nn* have the format of the following example:

TG05 *sev* **WRONG OR NOT REACHED** *term*.

where:

- **TG05** is the numerical order in which these messages are listed
- *sev* stands for *severity* ranging from 0 (information) through 4 (fatal) indicated by as many asterisks
- *term* (metalanguage) can also be according to the context of the message:
  - *clause*
  - *file*
  - *option*
  - *tdsobj*
  - *text*
  - *value.*

The *sev* of the message determines whether the TDS application can continue running. Severities 0 through 3 serve as information and warnings, and can be displayed accompanying fatal messages of severity 4.

Other TP7GEN messages without the prefix **TG***nn* can appear during the execution of the utility. See Subsection N.2.

TG01 *sev* **UNEXPECTED CLAUSE OR STATEMENT.**

Type: ***Fatal****: sev=4*

Meaning: This message appears after a statement or a clause not recognized by the syntactic analyzer. Only a limited set of clauses or statements can be specified, according to the convention for chaining clauses and statements.

Action: Correct the chaining conventions for clauses and statements for this part of the STDS file.

TG02 *sev* **FATAL ERROR. PROCESSING STOPPED.**

Type: Information: *sev*=0

Meaning: Message complements a previous fatal error which deprives TP7GEN of essential information. TP7GEN aborts.

Action: Correct the error reported before this message, and restart TP7GEN. Further errors in STDS file may be discovered during this new run, since the previous one could not complete the analysis.

TG03 *sev* **RESULT JOB ABORTED.**

Type: Information: *sev*=0

Meaning: Message complements a previous message flagging a major error. TP7GEN aborts.

Action: Correct the error reported before this message, and restart TP7GEN. Further errors in STDS file may be discovered during this new run, since the previous one could not complete the analysis.

TG04 *sev* **UNEXPECTED TDS GENERATION DATA EXHAUSTION.**

Type: ***Fatal****: sev=4*

Meaning: TP7GEN has reached an unexpected STDS End-Of-File during the analysis:
– either because the STDS file is incomplete
– or due to a break in the chaining of clauses.

Action: Check the STDS file. If it is complete, correct the errors reported before TG04.

TG05 *sev* **WRONG OR NOT REACHED** *term*.

where *term* is one of the following:

- { . | , }
- **DOUBLE QUOTE**
- **IDENTIFIER**
- **INTEGER**
- **KEY-WORD**
- **LEXICAL UNIT**
- **LITERAL BETWEEN DOUBLE QUOTES**

Type: Information: *sev*=2

Meaning: Either a unit is missing or the syntax analyzer cannot recognize it. For example, an expected . is missing, or an expected integer is not a numerical string, or a reserved keyword instead of an identifier.

Action: Check clause or statement syntax, and restart TP7GEN.

TG06 *sev* **ERRONEOUS SYNTAX IN CLAUSE OR STATEMENT.**

Type: Information: *sev*=3

Meaning: The current clause or statement does not conform to the accepted syntax.

Action: Check the structure of the clause or statement in the documentation and restart TP7GEN.

TG07 *sev* **FILE DEFINITION WITHOUT SELECT DECLARATION.**

Type: Information: *sev*=3

Meaning: In a PROCESSING-MODE or a FILE-INTEGRITY statement, the *filename* specified does not match any declared in a SELECT entry of a FILE-CONTROL statement.

Action: Check that the filename of the current clause or statement is the same than that defined in a previous SELECT EXTERNAL *filename* entry of the FILE-CONTROL statement.

TG08 *sev* **INVALID BUFFER RESERVATION.**

Type:	Information: *sev*=3
Meaning:	Number of buffers reserved in RESERVE AREA clause exceeds 20,000 units.
Action:	Check the number of buffers in RESERVE AREA clause.

TG09 *sev* **INVALID TPR TIME LIMIT OPTION.**

Type:	Information: *sev*=3
Meaning:	Number of seconds given in the MAXIMUM IDLE-TIME clause is equal or greater than 1,000,000.
Action:	Reduce the idle time in the MAXIMUM IDLE-TIME clause.

TG10 *sev* **INVALID SERVICE-MESSAGE NUMBER.**

Type:	Information: *sev*=2
Meaning:	The number in the SERVICE-MESSAGE clause is not in the range of 1 through 33. It identifies the service-message to modify.
Action:	Correct the SERVICE-MESSAGE clauses (format 2) concerned.

TG11 *sev* **INVALID SPECIAL CHARACTER IDENTIFIER.**

Type:	Information: *sev*=2
Meaning:	The *mnemonic* parameter of the SPECIAL-CHAR clause is too long (*length* > 31 characters).
Action:	Correct the *mnemonic* length in the SPECIAL-CHAR clause.

TG12 *sev* **INVALID SPECIAL CHARACTER VALUE.**

Type: Information: *sev*=2

Meaning: the *code* parameter of the SPECIAL-CHAR clause exceeds 32 characters.

Action: Correct the *mnemonic* length in the SPECIAL-CHAR clause.

TG14 *sev* **SCHEMA NAME TOO LONG OR TOO MANY DECLARED SCHEMAS.**

Type: Information: *sev*=3

Meaning: Name in DB statement of the IDS-DEFINITION clause specifying the name of an IDS schema exceeds 30 characters, or there are more than 32 declared schemas.

Action: Correct the length of the *schemaname* parameters, and the number of DB *schemaname* statements.

TG16 *sev* **DEFAULT VALUE ASSUMED.**

Type: Information: *sev*=2

Meaning: Message complements another signaling that thresholds have been exceeded for a given parameter or no value has been provided. TP7GEN continues with the default value.

Action: Adjust the value to conform to parameter thresholds in the STDS file.

TG17 *sev* **OBSOLETE CLAUSE OR SECTION, SUPPOSED TO BE REPLACED BY NETGEN DIRECTIVE(S)**

Type: Information: *sev*=1 or 2 depending on the clause or section

Meaning: The previous clause, statement, or section could be replaced by an equivalent declaration in the NETGEN domain. If this declaration is done in NETGEN, the STDS clause or section is ignored. If the severity is 2, remove the clause or section from the STDS file, and place the equivalent declaration in NETGEN instead.

Action: If the severity is 2, remove the clause, statement, or section from the STDS file and check that the equivalent declaration has been done in NETGEN.

TG18 *sev* **NO MORE RELEVANT CLAUSE OR SECTION, TO BE REMOVED.**

Type: Information: *sev*=2

Meaning: TP7GEN no longer supports a clause, statement or section.

Action: Remove the clause, statement or section of the STDS file.

TG19 *sev* **FETCHING NEXT** *term*.

where *term* is one of the following:

- **CLAUSE OR STATEMENT**
- **MANDATORY CLAUSE**
- **PARAGRAPH**
- **PARAMETER**
- **PERIOD**

Type: Information: *sev*=0

Meaning: Message complements another signaling a syntactic error. TP7GEN skips the current part of the STDS file to continue the analysis.

Action: Correct the syntactic error and restart TP7GEN.

TG20 *sev* **ASSUMED VALUE IS:** *value*.

where *value* is one of the following:

- { **0** | **1** | **10** | **64** | **512** }
- { **U** | **I-O (DEFAULT)** }
- **PRIVATE-STORAGE SIZE**
- **STO**

Type: Information: *sev*=2

Meaning: Message complements another signaling that thresholds have been exceeded for a given parameter, or that its value was not provided. TP7GEN decides use the value specified.

Action: Adjust the value to conform to parameter thresholds in the STDS file.

TG21 *sev* **ILLEGAL** *text*.

where *text* is one of the following:

- **MAXIMUM XCP2 WAITTIME: MUST BE < 1000**
- **NUMBER OF DYNAMICALLY CREATED TRANSACTIONS: MUST BE <= 50. 10 ASSUMED**
- **NUMBER OF VIRTUAL CORRESPONDENTS. MUST BE <= 100**
- **NUMBER OF XCP1 SESSIONS: MUST BE <= 10. DEFAULT ASSUMED**
- **NUMBER OF XCP2 CONVERSATIONS: MUST BE <=10. DEFAULT ASSUMED**
- **SYNCPOINT EVENT IDENTIFIER: MUST BE <= 10. IGNORED**
- **VALUE FOR INTEGER PARAMETER. IGNORED**
- **ILLEGAL VALUE FOR INTEGER PARAMETER. IGNORED.**

Type: Information: *sev*=2

Meaning: The value given in the STDS file for the specified parameter does not fit the documented thresholds.

Action: Adjust the value to conform to parameter thresholds in the STDS file, and restart TP7GEN.

TG22 *sev* **ERRONEOUS OR NOT DECLARED:** *tdsobj*.

where *sev* depends on *tdsobj*:

- *sev*=**2** *tdsobj*=**SELECT**
- *sev*=**3** *tdsobj*=**TRANSACTIONS**
- *sev*=**2** *tdsobj*=**VIEW, FILE, OR AREA. FETCHING NEXT STATEMENT**
- sev=**3** *tdsobj*=**SIMULTANEITY LEVEL**

Type: Information: *sev*=2 or 3

Meaning: An essential element has not been declared where expected in the STDS file. The element can be either mandatory in any STDS file, or conditionally needed inside a paragraph of the STDS file. For example: a transaction referred to in the NON-CONCURRENT clause must be declared in a MESSAGE clause of the TRANSACTION SECTION.

Action: If mandatory, specify the element. If optional, either declare it, or remove its reference if not necessary.

TG23 *sev* **TOO LONG PARAMETER:** *tdsobj*.

where *sev* depends on *tdsobj*:

- *sev*=**3** *tdsobj*=**FILE-NAME**
- *sev*=**2** *tdsobj*=**HOST GATEWAY NAME. MAXIMUM LENGTH IS 8 CHARACTERS**
- *sev*=**3** *tdsobj*=**INTERNAL FILE-NAME**
- *sev*=**2** *tdsobj*=**MESSAGE LIBEL**
- *sev*=**3** *tdsobj*=**MESSAGE-ID OR TPRNAME. IGNORED**
- *sev*=**2** *tdsobj*=**PROCEDURE NAME. IGNORED**
- *sev*=**3** *tdsobj*=**REALM NAME**
- *sev*=**2** *tdsobj*=**SCHEMA NAME**
- *sev*=**2** *tdsobj*=**VIRTUAL FAMILY NAME**

Type: Information: *sev*=2 or 3

Meaning: *tdsobj* character string exceeds its maximum length in the current statement or clause of STDS file.

Action: Adjust the length for the erroneous parameter, and restart TP7GEN.

TG24 *sev* **DUPLICATED PARAMETER:** *tdsobj*.

where *sev* depends on *tdsobj*:

- *sev*=**2** *tdsobj*=**AUTHORITY CODE**
- *sev*=**3** *tdsobj*=**FILE NAME**
- *sev*=**2** *tdsobj*=**HOST GATEWAY NAME**
- *sev*=**3** *tdsobj*=**INTERNAL FILE NAME**
- *sev*=**3** *tdsobj*=**MESSAGE-ID**
- *sev*=**2** *tdsobj*=**PROCEDURE SYMBOL**
- *sev*=**3** *tdsobj*=**REALM NAME**
- *sev*=**2** *tdsobj*=**USE CHARACTER(S). IGNORED**
- *sev*=**2** *tdsobj*=**VIRTUAL FAMILY NAME**

Type: Information: *sev*=2 or 3

Meaning: Parameter recurs in the STDS file.

Action: Check and correct the declaration of the parameter in the STDS file, and restart TP7GEN.

TG25 *sev* **INVALID PARAMETER:** *tdsobj*.

where *sev* depends on *tdsobj*:

- *sev*=**2** *tdsobj*=**AUTHORITY CODE. IGNORED**
- *sev*=**2** *tdsobj*=**MESSAGE-NAME. IGNORED**
- *sev*=**2** *tdsobj*=**NUMBER OF TERMINALS. IGNORED**
- *sev*=**4** *tdsobj*=**PROGRAM-ID. IGNORED**
- *sev*=**2** *tdsobj*=**PROMPT. IGNORED**
- *sev*=**2** *tdsobj*=**THINK-TIME. IGNORED**
- *sev*=**2** *tdsobj*=**TRANSACTION CLASS. IGNORED**
- *sev*=**2** *tdsobj*=**TRANSACTION-STORAGE SIZE. IGNORED**
- *sev*=**2** *tdsobj*=**USE CHARACTER(S). IGNORED**
- *sev*=**2** *tdsobj*=**USE PROCEDURE NAME. IGNORED**

Type: Information: *sev*=2
Fatal: *sev*=4

Meaning: Parameter identified either is not of the expected type such as numeric or alphanumeric, or is out of limits, or is incompatible with the TDS or the GCOS environment.

Action: Correct the parameter for type, limit, consistency with other TDS elements, or with the GCOS environment.

TG26 *sev* **ERRONEOUS PARAMETER:** *tdsobj*.

where *sev* depends on *tdsobj*:

- *sev*=**2** *tdsobj*=**COMMON-STORAGE SIZE. IGNORED**
- *sev*=**2** *tdsobj*=**CONCURRENT MESSAGE LENGTH**
- *sev*=**2** *tdsobj*=**LOCKED PAGES NUMBER. IGNORED**
- *sev*=**2** *tdsobj*=**MAILBOX NAME. DEFAULT ASSUMED**
- *sev*=**2** *tdsobj*=**MAXIMUM NUMBER OF TPRS. CLAUSE IGNORED**
- *sev*=**2** *tdsobj*=**MAXIMUM SIMULTANEOUS GTWRITER REPORTS SET TO TDS SIMULTANEITY NUMBER**
- *sev*=**2** *tdsobj*=**MESSAGE LENGTH. DEFAULT ASSUMED**
- *sev*=**2** *tdsobj*=**NUMBER OF BEFORE JOURNAL COMMITMENT UNITS. NONE ASSUMED**
- *sev*=**2** *tdsobj*=**NUMBER OF DUMMY CORRESPONDENTS**
- *sev*=**2** *tdsobj*=**NUMBER OF JPPC IQS SEGMENTS. NONE ASSUMED**
- *sev*=**2** *tdsobj*=**NUMBER OF LOCKED PAGES. 512 ASSUMED**
- *sev*=**2** *tdsobj*=**NUMBER OF SCHEMAS ACCESSED PER COMMITMENT**
- *sev*=**3** *tdsobj*=**NUMBER OF TPR MODULES**
- *sev*=**2** *tdsobj*=**NUMBER OF TRACE ENTRIES**
- *sev*=**2** *tdsobj*=**PRIORITY NUMBER. IGNORED**
- *sev*=**2** *tdsobj*=**SESSION NUMBER. DEFAULT ASSUMED**
- *sev*=**2** *tdsobj*=**SHARED-STORAGE NAME. IGNORED**
- *sev*=**3** *tdsobj*=**SIMULTANEITY NUMBER**

Type: Information: *sev*=2 or 3

Meaning: Parameter identified is:
- not the expected type such as numeric or alphanumeric
- or out of limits
- or inconsistent with elements of TDS already declared.

Action: Correct the parameter for type, limit, and consistency with other TDS elements.

TG26 *sev* **ERRONEOUS PARAMETER: RPC SIMULTANEITY NUMBER.**

Type: Information: *sev*=2

Meaning: The *nrpcsimu* of the RPC SIMULTANEITY Clause is greater than or equal to the *nsimu* of the SIMULTANEITY Clause. See Paragraph 3.3 *Syntax of TDS Section*.

Action: Ensure that *nrpcsimu* is less than *nsimu*. Rerun TP7GEN.

TG27 *sev* **MANDATORY CLAUSE NOT REACHED OR WRONG:** *clause*.

where *sev* depends on *clause*:

- *sev*=**3** *clause*=**AUTHORITY-CODES**
- *sev*=**3** *clause*=**FILE-CONTROL OR IDS-DEFINITION**
- *sev*=**3** *clause*=**INPUT-OUTPUT SECTION**
- *sev*=**3** *clause*=**MESSAGE**
- *sev*=**3** *clause*=**MESSAGE-LENGTH**
- *sev*=**4** *clause*=**PROGRAM-ID**
- *sev*=**3** *clause*=**RESERVE AREAS**
- *sev*=**3** *clause*=**SIMULTANEITY**
- *sev*=**3** *clause*=**TDS SECTION**
- *sev*=**3** *clause*=**TPR-TIME-LIMIT**
- *sev*=**3** *clause*=**TRANSACTION SECTION**

Type: Information: *sev*=3
Fatal: *sev*=4

Meaning: A mandatory clause is:
– not recognized because of a syntactic error (missing or wrong keyword) in the statement immediately preceding this message
– or not at the right place in the STDS file
– or recognized but contains a syntactic error reported in a message immediately following this message.

Action: Proceed as follows:
– Insert the mandatory clause in the right place and with the right syntax.
– If already present, check the syntax of the clause preceding the message.

TG28 *sev* **WRONG SYNTAX IN CLAUSE OR STATEMENT:** *clause*.

where *sev* depends on clause:

- *sev*=**3** *clause*=**AREAS ARE**
- *sev*=**3** *clause*=**ATTACH**
- *sev*=**3** *clause*=**AUTOMATIC UNMAPPING**
- *sev*=**3** *clause*=**BEFORE JOURNAL COMMITMENT UNITS**
- *sev*=**3** *clause*=**CANCELCTX AT RECONNECTION**
- *sev*=**3** *clause*=**CD-IN DATE**
- *sev*=**3** *clause*=**CLASS**
- *sev*=**3** *clause*=**COMMITMENT**
- *sev*=**3** *clause*=**COMMON STORAGE**
- *sev*=**3** *clause*=**CONTROLLED COMMON-STORAGE**
- sev=**3** clause=**DB**
- *sev*=**3** *clause*=**DEFAULT TRANSACTION-STORAGE**
- *sev*=**3** *clause*=**DYNAMIC SUPERVISION**
- sev=**3** clause=**FD**
- *sev*=**3** *clause*=**{ FILE-DEFINITION | FILE-INTEGRITY }**
- *sev*=**3** *clause*=**FILES ARE**
- *sev*=**3** *clause*=**FORM. IGNORED**
- *sev*=**3** *clause*=**FORMATTED MESSAGE-LENGTH**
- *sev*=**3** *clause*=**HOST GATEWAY**
- *sev*=**3** *clause*=**IDS-DEFINITION**
- *sev*=**3** *clause*=**IMPLICIT COMMITMENT**
- *sev*=**3** *clause*=**IQS-DEFINITION**
- *sev*=**3** *clause*=**MASTER MAILBOX**
- *sev*=**3** *clause*=**MAX XCP2 WAITTIME**
- *sev*=**3** *clause*=**MAXIMUM**
- *sev*=**3** *clause*=**MAXIMUM**
  { IDLE-TIME | DYNAMICALLY ADDED TRANSACTIONS |
  LOCKED PAGES. IGNORED | TPR |
  NUMBER OF TPR WITH NO UNMAPPING |
  PRIVATE-STORAGE SIZE | TRANSACTION-STORAGE SIZE }
- *sev*=**3** *clause*=**NON-CONCURRENT**
- *sev*=**3** *clause*=**NUMBER**
- *sev*=**3** *clause*=**NUMBER { IQS JPPC SEGMENTS | TPR SHARABLE MODULES }**
- *sev*=**3** *clause*=**NUMBER OF { TERMINALS | DUMMY | LOCKED PAGES | VIRTUAL }**
- *sev*=**3** *clause*=**ORACLE-DEF**
- *sev*=**3** *clause*=**PRIORITY. IGNORED**
- *sev*=**3** *clause*=**PRIVATE-STORAGE**
- *sev*=**3** *clause*=**{ PROCESSING-CONTROL | PROCESSING-MODE }**
- *sev*=**3** *clause*=**PROMPT**
- *sev*=**3** *clause*=**REALMS**

- *sev*=**3** *clause*=**RPC SERVICE**
- *sev*=**3** *clause*=**RPC SIMULTANEITY**
- *sev*=**3** *clause*=**SELECT { EXTERNAL | XCP2-SYNCPEVT }**
- *sev*=**3** *clause*=**SERVICE-MESSAGE**
- *sev*=**3** *clause*=**{ SHARED READ | SHARED-STORAGE }**
- *sev*=**3** *clause*=**SPECIAL-CHAR**
- *sev*=**3** *clause*=**SUPPRESS { BEFORE JOURNAL | CONCURRENT | DEFERRED UPDATES }**
- *sev*=**3** *clause*=**{ TDS-FILE-DEFINITION | TDSTX-MESSAGE }**
- *sev*=**3** *clause*=**THINK-TIME**
- *sev*=**3** *clause*=**{ TRANSACTION STORAGE SIZE | *lvl* TRANSACTION-STORAGE }**
- *sev*=**3** *clause*=**{ USE | USE DEFERRED UPDATES }**
- *sev*=**3** *clause*=**USER JOURNAL**
- *sev*=**3** *clause*=**VIEWS ARE. FETCHING NEXT STATEMENT**
- *sev*=**3** *clause*=**WITH ACCOUNTING**
- *sev*=**3** *clause*=**WORKING-STORAGE**
- *sev*=**3** *clause*=**XCP1 MAILBOX**
- *sev*=**3** *clause*=**XCP2 SERVICE**
- *sev*=**2** *clause*=**FILE-OPENING OPTION**
- *sev*=**2** *clause*=**XA-RESYNC-DELAY**
- *sev*=**2** *clause*=**XA SERVICE**
- *sev*=**2** *clause*=**CMA SERVICE**
- *sev*=**2** *clause*=**TCP-IP PROTOCOL**

Type: Information: *sev*=3 or 2

Meaning: Indicates the clause or statement in which a syntax error has been detected. The nature of the error may be detailed in an another message.

Action: Correct the syntax of the clause or statement, and rerun TP7GEN.

Check that no TDS Generation Keyword is misused in the faulty clause or statement (refer to Appendix L).

TG29 *sev* **ILLEGAL CLAUSE IN THIS CONTEXT:** *lvl* **TRANSACTION-STORAGE.**

Type: Information: *sev*=2

Meaning: The clause in the context of TRANSACTION-STORAGE is either wrong or missing, out of range or skipped.

Action: Ensure that the preliminary declarations(s) are set in their right place in the STDS file. Correct syntactic errors, and restart TP7GEN.

TG29 *sev* **ILLEGAL CLAUSE IN THIS CONTEXT: INQUIRY TX CANNOT USE RPC**

Type: Information: *sev*=3

Meaning: The Clauses RPC SERVICE USED and FOR INQUIRY are mutually exclusive when RPC simultaneity regulation is required (RPC SIMULTANEITY IS *nrpcsimu*).

Action: Suppress at least one of these three Clauses according to what is needed.

TG29 *sev* **ILLEGAL CLAUSE IN THIS CONTEXT: INQUIRY TX CANNOT USE XA.**

Type: Information: *sev*=2

Meaning: The clauses XA SERVICE USED and FOR INQUIRY are mutually exclusive.

Action: Suppress at least one of these two clauses according to what is needed.

TG30 *sev* **IGNORED.**

Type: Information: *sev*=0

Meaning: Complements message detecting an erroneous clause, statement or parameter. TP7GEN decides not to use the information specified. This message does not have a severity assigned to it.

Action: Adjust the value in the STDS file to conform to parameter thresholds.

TG31 *sev* **RESUMING ANALYSIS AT CLAUSE:***clause*.

Where clause is one of the following:

- **RESERVE AREAS.**
- **MESSAGE-LENGTH**
- **TPR-TIME-LIMIT**
- **INPUT-OUTPUT SECTION.**
- **TRANSACTION SECTION.**
- **MESSAGE.**
- **AUTHORITY-CODES.**

Type: Information: *sev*=0

Meaning: Message displayed in complement of a message signaling a syntactic error. This message informs about the decision to skip the current part of the STDS file, in order to carry on the analysis. This message has no severity of its own, but rather the severity of the complement message.

Action: Correct the syntactic error and restart TP7GEN.

TG51 *sev* **CLAUSE OR STATEMENT APPLYING ONLY TO TDS FILES: { SHARED READ | SUPPRESS CONCURRENT ACCESS CONTROL }.**

Type: Information: *sev*=0

Meaning: The current clause or statement is not used inside a TDS file declaration block, as it should be.

Action: Remove the clause or statement, or declare the file a TDS file, then restart TP7GEN.

TG52 *sev* **SEGMENT { CREATION | EXPANSION } FAILED FOR TRANSACTION TABLE WITH THE RETURN CODE:** *code*

Type: ***Fatal****: sev*=4

Meaning: A problem of memory availability has occurred during TP7GEN.

Action: With the help of the *reason code* and *sicid*, determine if it is a temporary or a permanent problem. Try later if it is temporary problem. If the problem persists, contact the Service Center.

TG53 *sev* **SEGMENT { CREATION | EXPANSION } FAILED WITH THE RETURN CODE:** *code*

Type: ***Fatal****: sev=4*

Meaning: A problem of memory availability has occurred during TP7GEN.

Action: With the help of the *reason code* and *sicid*, determine if it is a temporary or a permanent problem. Try later if it is temporary problem. If the problem persists, contact the Service Center.

TG54 *sev* **{ CLOSE | OPEN | PUT ON } ACTION ON { COBOL | SYSTEM } LIBRARY FAILED WITH THE RETURN CODE:** *code*

Type: ***Fatal****: sev=4*

Meaning: A problem on a file action has occurred, with:
– OPEN for library with SYSTEM
– CLOSE for library with SYSTEM and COBOL
– OPENS and PUT ON for a subfile
accompanied by a reason reported in the *return code*.

Action: With the *return code* and *sicid*, ensure that the TDS library or the subfile is available.

TG55 *sev* **{ CLOSE | GET | OPEN | PUT | PUT ON | PUTX } ACTION ON { EDITION | SYSIN | TDS SYSTEM } FILE FAILED WITH THE RETURN CODE:** *code*

Type: ***Fatal****: sev=4*

Meaning: A file system problem has occurred on a TDS internal file operation, with:
– OPEN + CLOSE for EDITION, SYSIN + TDS SYSTEM
– PUT for EDITION; PUT ON and PUTX for TDS SYSTEM
– GET for SYSIN,
accompanied by a reason reported in the *return code*.

Action: With the help of the *return code* and *sicid*, ensure that the TDS system file is available.

TG56 *sev* **MISSING PARAMETER: STORAGE NAME. IGNORED.**

Type: Information: *sev*=2

Meaning: The current clause or statement needs a parameter not previously declared. For example, the *lvl* TRANSACTION-STORAGE paragraph requires a TRANSACTION-STORAGE name previously declared.

Action: Ensure that the parameter required is in the right place and has previously been correctly declared.

TG57 *sev* **MISSING CLAUSE OR STATEMENT:** *text*

where *text* is one of the following:

- **USE FORMS, CURRENT CLAUSE IGNORED**
- **USE ORACLE, NO XA SERVICE AVAILABLE**

Type: Information: *sev*=2

Meaning: The current clause or statement depends on a previous declaration of the clause or statement specified in the message, and is necessary to prepare the appropriate context. The previous declaration is wrong or missing, out of range or skipped.

Action: Ensure that the preliminary declarations(s) are set in their right place in the STDS file. Correct syntactic errors, and restart TP7GEN.

TG58 *sev* **OUT OF RANGE:** tdsobj

with *tdsobj* one of the following:

- **MAXIMUM PRIVATE-STORAGE SIZE.**
- **LEVEL OF SUPERVISION.DEFAULT ASSUMED.**
- **CONTROLLED COMMON-STORAGE SIZE.**
- **MAXIMUM TRANSACTION-STORAGE SIZE.**
- **XA-RESYNC-DELAY. DEFAULT ASSUMED.**

Type: Information: *sev*=2

Meaning: The parameter specified in the message is outside the minimum and maximum thresholds allowed.

Action: The size is more than 65527; reduce its value, or check value of level of supervision, or xa-resync-delay depending on message text, and restart TP7GEN.

TG59 *sev* **MISMATCH BETWEEN** *option*.

where *sev* depends on *option*:

- *sev*=**2** *option*=**CONTROL ACCESS AND SHARED READ**
- *sev*=**2** *option*=**I-SPEC AND F-SPEC FILE-NAME**
- *sev*=**2** *option*=**O-SPEC AND F-SPEC FILE-NAME**
- *sev*=**4** *option*=**PROGRAM-IO AND TDS-NAME**
- *sev*=**3** *option*=**VBO DISK FOR RELOV FILE AND FBO DISK(S) FOR SWAP FILES**
- *sev*=**3** *option*="**SUPPRESS" AND "USE" DEFERRED UPDATE OPTIONS**
- *sev*=**3** *option*="**SUPPRESS BEFORE JL" AND "SUPPRESS DEFERRED UPDATES" OPTIONS**
- *sev*=**2** *option*="**XA" AND "CMA"**
- *sev*=**2** *option*="**HEURISTIC COMMIT" AND "CMA"**

Type: Information: *sev*2 or 3
***Fatal**: sev*=4

Meaning: Inconsistency between two linked elements of TDS.

Action: Locate the linked elements, ensure that they are compatible and, restart TP7GEN.

- *sev*=**3** *option*=**MISMATCH BETWEEN MAXIMUM TRANSACTION-STORAGE SIZE AND DEFAULT TRANSACTION-STORAGE SIZE**

Action: Default transaction-storage size is greater than maximum transaction-storage size, reduce its value or increase maximum.

- *sev*=**2** *option*=**MISMATCH BETWEEN PRIVATE-STORAGE SIZE AND MAXIMUM TRANSACTION-STORAGE SIZE**

Action: Private-storage size is greater than maximum transaction-storage size. It is ignored; reduce its value or increase maximum.

- *sev*=**2** *option*=**MISMATCH BETWEEN PRIVATE-STORAGE SIZE AND DEFAULT TRANSACTION-STORAGE SIZE**

Action: Private-storage size is greater than default transaction-storage size. It is ignored; reduce its value or increase default.

- *sev*=**3** *option*=**MISMATCH BETWEEN MAXIMUM PRIVATE-STORAGE SIZE AND MAXIMUM TRANSACTION-STORAGE SIZE**

Action: Maximum private-storage size is greater than maximum transaction-storage size. Reduce its value or increase maximum transaction-storage size.

- *sev*=**3** *option*=**MISMATCH BETWEEN MAXIMUM PRIVATE-STORAGE SIZE AND PRIVATE-STORAGE SIZE**

Action: Private-storage size is greater than maximum private-storage size. Reduce its value or increase maximum.

- *sev*=**3** *option*=**MISMATCH BETWEEN TRANSACTION-STORAGE SIZE AND MAXIMUM TRANSACTION-STORAGE SIZE**

Action: Transaction-storage size of this transaction is greater than maximum transaction-storage size specified in the TDS section. Increase the maximum.

- *sev*=**3** *option*=**MISMATCH BETWEEN TRANSACTION-STORAGE SIZE AND PRIVATE-STORAGE SIZE**

Action: Transaction-storage size of this transaction is less than private-storage size specified in the TDS section. Increase its value.

- *sev*=**3** *option*=**MISMATCH BETWEEN TRANSACTION-STORAGE SIZE AND MAXIMUM PRIVATE-STORAGE SIZE**

Action: Maximum private-storage size of this transaction is too big. Therefore, a system transaction has a transaction storage greater than 65527. Reduce its value.

- *sev*=**3** *option*=**MISMATCH BETWEEN MAXIMUM OF TRANSACTION-STORAGE SIZES AND MAXIMUM TRANSACTION-STORAGE SIZE**

Action: The following relation is not true:
Maximum transaction-storage size greater than MAX (transaction-storage sizes) - private-storage size + maximum private-storage size. Adjust the various sizes so that it becomes true.

Type: Information: *sev*=2 or 3

Meaning: Inconsistency between two linked elements of TDS.

TG61 *sev* ***END DELIMITER MISSING.**

Type: ***Fatal****: sev*=4

Meaning: The current paragraph is not ended by the "*END" convention, as it should be.

Action: Check the presence of the ending convention mark at the end of the paragraph.

TG62 *sev* **TOO LONG UNIT: TRUNCATED TO MAXIMUM SIZE.**

Type: Information: *sev*=2

Meaning: The character string specified in STDS file exceeds its maximum and is truncated.

Action: Correct the character string for length and consistency, then restart TP7GEN.

TG64 *sev* **UNABLE TO INITIALIZE *file* WITH THE RETURN CODE:** *code*

where *sev* depends on *file*:

- *sev*=**4** *file*=**RECOV FILE**
- *sev*=**3** *file*=**SWAP FILE**

Type: Information: *sev*=3
Fatal: sev=4

Meaning: A file system problem has occurred on a TDS internal file operation. The type of the TDS file is reported in the *file* parameter.

Action: With the help of the *reason code*, ensure that the TDS system file is available.

TG65 *sev* **TOO MANY DECLARED** *tdsobj*.

where *sev* depends on *tdsobj*:

- *sev*=**3** *tdsobj*=**FILES. MAXIMUM IS 512.**
- *sev*=**2** *tdsobj*=**HOST GATEWAY NAMES. MAXIMUM IS 100.**
- *sev*=**3** *tdsobj*=**USE PROCEDURES. MAXIMUM IS 256**
- *sev*=**2** *tdsobj*=**VIRTUAL CORRESPONDENTS. AVAILABLE NUMBER OF TERMINALS EXCEEDED.**
- *sev*=2 *tdsobj*=**VIRTUAL FAMILY NAMES. MAXIMUM IS 5.**
- *sev=3 tdsobj=TRANSACTIONS.* **MAXIMUM IS 3000.**

Type: Information: *sev*=2 or 3

Meaning: The maximum number of declarations for the specified TDS object is exceeded.

Action: Reduce the number of declarations.

TG66 *sev* **NOT ASSIGNED: COBOL LIBRARY. NO ACTION.**

Type: Information: *sev*=2

Meaning: The TDS COBOL library is not correctly defined or cataloged

Action: With the help of the system reason code, check the COBOL library availability.

TG67 *sev* **INVALID PROCESSING MODE. { I-O | U } DEFAULT VALUE ASSUMED.**

Type: Information: *sev*=2

Meaning: Anomaly in the processing mode:
- either the processing mode value given in the PROCESSING-MODE clause is not INPUT-OUTPUT, INPUT, OUTPUT or EXTEND
- or the type of file is incompatible with the processing mode value.

Action: Check that the processing mode specified:
- is in the list of authorized values
- and is compatible with the file type, for example, TDS-controlled files and IDS/II realms can be only INPUT or INPUT-OUTPUT modes.

TG68 *sev* **EXPECTED: I-SPEC OR O-SPEC.**

Type: Information: *sev*=3

Meaning: A parameter (integer or option) is not correctly declared

Action: Declare the specified parameter and restart TP7GEN.

TG69 *sev* **FILES NOT DECLARED, ALTHOUGH EXPECTED.**

Type: Information: *sev*=3

Meaning: A TDS object, referred to in the STDS file has not been declared in the proper section.

Action: Declare the required object in its proper section, for example, files in the Input-Output Section, or remove their references.

TG70 *sev* **UNEXPECTED STATEMENT. FETCHING NEXT CLAUSE OR STATEMENT.**

Type: Information: *sev*=3

Meaning: The current declaration is not recognized as being a clause or a statement. To continue the analysis of STDS file, strings are skipped until the next period, assuming that a clause or statement is starting there.

Action: Correct or remove the unrecognized clause or statement, then restart TP7GEN.

TG71 *sev* **TOO LONG TRANSACTION-STORAGE NAME. TRUNCATED TO MAXIMUM SIZE.**

Type: Information: *sev*=2

Meaning: The maximum threshold for a transaction storage name is exceeded.

Action: Specify maximum size and restart TP7GEN.

TG72 *sev* **TRANSACTION-STORAGE SIZE TOO SMALL. PRIVATE-STORAGE SIZE ASSUMED FOR IT.**

Type: *Warning: sev*=1

Meaning: *transaction-storage size* is smaller than *private-storage size*. *private-storage size* is used for determining *transaction-storage size*.

Action: Set *transaction-storage size* larger than *private-storage size*.

TG73 *sev* **PRIVATE-STORAGE WITHOUT TRANSACTION-STORAGE. SIZE OF PRIVATE-STORAGE ASSUMED FOR TRANSACTION-STORAGE.**

Type: Information: *sev*=2

Meaning: *transaction-storage size* is either missing, incorrect, out of range or skipped due to syntactic errors. Its value taken as 0 is therfore smaller than the *private-storage size* declared. *private-storage size* is used for determining *transaction-storage size*.

Action: Ensure that *transaction-storage size* is declared with a value valid and syntactically correct, and restart TP7GEN.

TG74 *sev* **MAILBOX NAME TOO LONG. TRUNCATED TO 8 DIGITS.**

Type: Information: *sev*=2

Meaning: STDS file contains a mailbox name longer than 8 characters. Only the first 8 characters are taken into account.

Action: Ensure that all the names of the mailboxes dedicated to TDS are limited to 8 characters.

TG75 *sev* **ERRONEOUS PARAGRAPH:** *text*.

where *text* is one of the following:

- **ORACLE-DEF...ORACLE-ENDDEF**
- **ORACLE-DEF...ORACLE-INDDEF, TOO LONG.**
- **ERRONEOUS PARAGRAPH: ORACLE-DEF...ORACLE-ENDDEF (EXTERNAL ENTRY POINT NOT FOUND)**
- **MESSAGE.**

Type: Information: *sev*=3

Meaning: An error in the structure of the current paragraph is due to:
– a size threshold overflow of this paragraph
– an unexpected delimiter
– TP7GEN cannot successfully call the ORACLE/TDS component
– or the absence of an expected delimiter character inside this paragraph.

Action: Correct the following:
– the general structure of the current paragraph
– the validity of the delimiters inside and at the end of the paragraph
then rerun TP7GEN.

TG76 *sev* **NOTIFICATION:** *notification*.

Type: Information: *sev*=2

Meaning: *notification* reports a non-standard feature discovered during the semantic analysis phase of TP7GEN. TP7GEN processing is not interrupted, but its result is not supported.

Action: Correct the anomaly and restart the generation.

TG99 *sev* **INTERNAL ERROR: CONTACT YOUR BULL SUPPORT REPRESENTATIVE.**

Type:	Information: *sev*=3
Meaning:	An unexpected internal error was detected during execution of TP7GEN.
Action:	Note in the report file between which clauses this message occurs and submit it to the Service Center.

N.2 TP7GEN JCL Messages

The following messages appears both in the JOR and on submitter's terminal to provide global information about TP7GEN execution.

The metalanguage term *tds* is the name of the TDS application specified by the user when submitting TP7GEN.

ERROR IN SECURITY PROCESSING FOR TDS *tds*

Meaning: Optional SA7 subfile (TDS security) was invoked in the TP7GEN JCL arguments but the SA7 job has reported an error greater than severity 2.

Action: Consult the JOR and report files for messages concerning the SA7 job execution. Correct the parameters for the SA7 environment and resubmit TP7GEN.

ERROR(S) SEVERITY 2 IN TDS *tds* **GENERATION**

Meaning: Errors of severity 2 and less have been detected during the analysis by TP7GEN Load Module. This message is followed by the message `GENERATION ENDED WITH SEVERITY 2 ERROR(S)` which means that the generation is not considered `SUCCESSFUL` even though the TDS LM is linked and its environment built.

Action: Consult the appropriate report file for messages concerning STDS analysis. Correct errors of severity 2, then resubmit TP7GEN JCL. Repeat the procedure until the message: *tds* `GENERATION SUCCESSFUL` appears.

ERROR(S) SEVERITY 3 IN TDS *tds* **GENERATION**

Meaning: Errors of severity 3 and less have been detected during the analysis by TP7GEN Load Module. This message is followed by `JOB ABORTED WITH SEVERITY 3` which means that the generation stopped before building a TDS Load Module and a TDS environment.

Action: Consult the appropriate report file for messages concerning STDS analysis. Correct other errors of severity 3 and less if any, then resubmit TP7GEN JCL. Repeat the procedure until the message: *tds* `GENERATION SUCCESSFUL` appears.

TDS *tds* **FATAL ABORT**

Meaning: This message reports that one fatal error was detected during the STDS analysis, preventing TP7GEN to carry on the analysis.

Action: Consult in the appropriate report file, messages about TP7GEN LM execution. Correct the cause of the fatal error in STDS file. Also correct other errors of severity 2 and more if any, then resubmit TP7GEN JCL. Repeat the procedure until the message: *tds* GENERATION SUCCESSFUL appears.

TDS *tds* **GENERATION SUCCESSFUL**

Meaning: This message reports that no more severity 2 and upper error were found in STDS file, and that a reliable TDS Load Module and environment is built.

Action: TP7EXEC can be started.

O. TDS Rejected Connection Codes

TDS sends the following hexadecimal codes which give the main reasons for rejected connections:

`0B:`	Lack of system resources (TDS mailbox full) *or* connection to a master mailbox rejected because a dummy session already exists. (High Availability context).
`0C:`	Connection already exists or is in progress.
`10:`	TM connection is denied for user *`tdsname`*`_ADM`.
`11:`	XCP1 protocol error.
`12:`	Reconnection error.
`13:`	Other.

P. Messages Produced by Default H_XAEVT Transaction

This transaction is only useful for a TDS using XA commitment protocol with ORACLE7.

A basic transaction called H_XAEVT is provided by ORACLE7 (refer to the ORACLE7/TDS User's Guide). It may be customized so that events that could lead to data inconsistency can be handled. Although using this transaction is not mandatory, the default action is limited to sending a message (see below) to the master and writing this message in the JOR (see the TDS Cobol Programmer's Guide).

TX90. DESYNC XAEVT. TPR: *tprnm* USER: *usernm* TDS_XA_STATUS: *tdsxast* XA_GLOBAL_STATUS: *xagblstat*

Action: Desynchronization has been detected in the XA protocol and asynchronous resynchronization has been launched. TDS_XA_STATUS indicates whether TDS took the decision to roll back or to commit the desynchronized CU to assist the database administrator in taking a heuristic decision.

TX90. RESYNC XAEVT. TPR: *tprnm* USER: *usernm* TDS_XA_STATUS: *tdsxast* XA_GLOBAL_STATUS: *xagblstat*

Action: Resynchronization is completed. Check XA_GLOBAL_STATUS to find out the result of resynchronization.

Glossary

C

CI
Control Interval

D

DWS
Declared Working Set

H

HA
High Availibility

I

I/O
Input-output operation

IQS
Interactive Query System

J

JAS
Journalization Advanced Service

M

MMS
Multimedia Support

MNJAS
MAINTAIN_JAS

P

PPC
Program-to-Program Communication

R

RPC
Remote Procedure Call

RPS
Rotational Position Sensing

S

SBR
System Behavior Reporter

SM
Sharable Module

SMLIB
Sharable Module Library

STE
Segment Table Entry

T

TILS
Transactional and Interactive Load Simulator

Index

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W

X

Vos remarques sur ce document / Technical publications remarks form

Titre / Title :	**TDS Administrator's Guide**

N° Référence / Reference No. :	**47 A2 32UT Rev07**

Date / Dated :	**June 2001**

ERREURS DETECTEES / ERRORS IN PUBLICATION

AMELIORATIONS SUGGEREES / SUGGESTIONS FOR IMPROVEMENT TO PUBLICATION

Vos remarques et suggestions seront attentivement examinées. Si vous désirez une réponse écrite, veuillez indiquer ci-après votre adresse postale complète.
Your comments will be promptly investigated by qualified personnel and action will be taken as required. If you require a written reply, furnish your complete mailing address below.

NOM / NAME : _____ DATE : _____

SOCIETE / COMPANY : _____

ADRESSE / ADDRESS : _____

_____

Remettez cet imprimé à un responsable Bull S.A. ou envoyez-le directement à :
Please give this technical publications remarks form to your Bull S.A. representative or mail to:

Bull S.A.
CEDOC
Atelier de reprographie
357, Avenue Patton BP 20845
49008 ANGERS Cedex 01
FRANCE

Bull HN Information Systems Inc.
Publication Order Entry
FAX: (800) 611-6030
MA30/415
300 Concord Rd.
Billerica, MA 01821
U.S.A.