



GT.M V5.5-000 Release Notes

Contact Information

GT.M Group
Fidelity Information Services, Inc.
2 West Liberty Boulevard, Suite 300
Malvern, PA 19355
United States of America

GT.M Support for customers: +1 (610) 578-4226
gtmsupport@fisglobal.com
Switchboard: +1 (610) 296-8877
Website: <http://fis-gtm.com>

Legal Notice

Copyright © 2012 Fidelity Information Services, Inc. All Rights Reserved

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts and no Back-Cover Texts.

GT.M™ is a trademark of Fidelity Information Services, Inc. Other trademarks are the property of their respective owners.

This document contains a description of GT.M and the operating instructions pertaining to the various functions that comprise the system. This document does not contain any commitment of FIS. FIS believes the information in this publication is accurate as of its publication date; such information is subject to change without notice. FIS is not responsible for any errors or defects.

Revision History		
Revision 1.4	27 July 2012	<ul style="list-style-type: none">In GTM-3873, corrected the optional percentage range of the TRUNCATE qualifier.In Platforms, updated the section on IBM System p AIX.
Revision 1.3	March 22, 2012	<ul style="list-style-type: none">In Conventions, added a new table that summarizes the new and revised replication terminology introduced in V5.5-000.Marked GTM-4337 and GTM-6664 as .
Revision 1.2	February 27, 2012	<ul style="list-style-type: none">In Platforms, changed the minimum supported Sun SPARC Solaris to version 10 (Update 6 and above).Marked GTM-5797, GTM-6026, and the Error Messages section as .
Revision 1.1	February 23, 2012	In Change History, corrected the summary note of GTM-7082.
Revision 1.0	February 22, 2012	First published version.

Table of Contents

Conventions	1
Release Overview	3
Platforms	3
Platform support lifecycle	6
Migrating to 64-bit platforms	6
.....	7
Internationalization (Collation)	7
Environment Translation	7
Recompile	8
Rebuild Shared Libraries or Images	8
Additional Installation Instructions	8
UNIX	8
OpenVMS	9
Upgrading to GT.M V5.5-000	9
Stage 1: Upgrading your Global Directory	9
Stage 2: Upgrading your Database Files	10
Stage 3: Upgrading your Replication Instance File	12
Stage 4: Upgrading your Journal Files	12
Stage 5: Upgrading your Trigger Definitions	12
Managing M mode and UTF-8 mode	13
Compiling ICU	14
Setting the environment variable TERM	14
Installing Compression Libraries	15
Change History	17
V5.5-000	17
M-Database Access	23
M-Other Than Database Access	27
Utilities-MUPIP	33
Utilities-Other Than MUPIP	39
Error Messages 	43
ACTLSTEXP	43
BADREGION	43
COREINPROGRESS	43
DBCDBNOCERTIFY	43
DBFLCORRP	44
DBROLLEDBACK	44
DSEWCREINIT	44
FMLLSTPRESENT	44
GTMASSERT2	45
IGNBMPMRKFREE	45
INSNOTJOINED	45
INSROLECHANGE	45
INSUNKNOWN	46
INVTRCGRP	46

JIUNHNDINT	46
JOBEXAMFAIL	46
JNLALLOCGROW	46
JNLNOREPL	47
JRTNULLFAIL	47
JNLRECINCMPL	47
LOCKSPACEFULL	47
LOCKSPACEINFO	48
LOCKSPACEUSE	48
MAXSEMGETRETRY	48
MUINFOINT6	48
MUTRUNC1ATIME	49
MUTRUNCBACKINPROG	49
MUTRUNCERROR	49
MUTRUNCFAIL	49
MUTRUNCNOSPACE	49
MUTRUNCNOTBG	50
MUTRUNCNOV4	50
MUTRUNCPERCENT	50
MUTRUNCSSINPROG	50
MUTRUNCSUCCESS	50
NOLOCKMATCH	50
NONASCII	51
NOREPLCTDREG	51
NORESYNCSUPPLONLY	51
NORESYNCPDATERONLY	51
NOSUPPLSUPPL	51
ORLBKCMPLT	52
ORLBKFRZOVER	52
ORLBKFRZPROG	52
ORLBKNOSTP	52
ORLBKNOV4BLK	53
ORLBKSTART	53
ORLBKTERMNTD	53
PERMGENDIAG	53
PERMGENFAIL	53
PRIMARYISROOT	54
RCVR2MANY	54
RCVRMANYSTRMS	54
REPL2OLD	54
REPLINSTDBSTRM	55
REPLINSTMISMATCH	55
REPLUPGRADEPRI	56
REPLUPGRADESEC	56
RESOLVESEQNO	56
RESOLVESEQSTRM	56
RESUMESTRMNUM	56

REUSEINSTNAME	56
RLBKCONFIGBNDRY	57
RLBKSTRMSEQ	57
RNDWNSKIPCNT	57
RSYNCSTRMSUPPLONLY	57
RSYNCSTRMVAL	58
SECNOTSUPPLEMENTARY	58
SEMWT2LONG	58
SHMREMOVED	58
STRMNUMMISMATCH1	58
STRMNUMMISMATCH2	59
STRMSEQMISMATCH	59
SUPRCVRNEEDSSUPSRC	59
SRVLCKWT2LNG	60
STRMNUMIS	60
TRIGMODREGNOTRW	60
UPDSYNC2MTINS	60
UPDSYNCINSTFILE	60
ZDATEBADDATE	61
ZDATEBADTIME	61
ZTRIGNOTRW	61
Documentation Addendum	63
Journal Extract Format	63
Shared Resource Authorization Permissions	66
Areas affected by ONLINE ROLLBACK	68

Conventions

Command Syntax: UNIX syntax (that is, lowercase text and "-" for flags/qualifiers) is used throughout this document. OpenVMS accepts both lowercase and uppercase text; flags/qualifiers on OpenVMS should be preceded with "/" rather than a "-".

Program Names: When referring to a GT.M program or function, the reference is in upper case, for example, MUIP BACKUP. When a specific example is provided, the lower case UNIX command names are used, for example, mupip backup -database ACN,HIST /backup

Reference Number: Reference numbers used to track software enhancements and customer support requests appear in parentheses ().

Platform Identifier: If a new feature or software enhancement does not apply to all platforms, the relevant platform or platforms appear in brackets [].

GT.M runs on a wide variety of UNIX/Linux implementations as well as OpenVMS. Consult FIS for currently supported versions. Each GT.M release is extensively tested by FIS on a set of specific versions of operating systems on specific hardware architectures (the combination of operating system and hardware architecture is referred to as a platform). This set of specific versions is considered Supported. There will be other versions of the same operating systems on which a GT.M release may not have been tested, but on which the FIS GT.M support team knows of no reason why GT.M would not work. This larger set of versions is considered Supportable. There is an even larger set of platforms on which GT.M may well run satisfactorily, but where the FIS GT.M team lacks the knowledge to determine whether GT.M is Supportable. These are considered Unsupported. Contact FIS GT.M Support with inquiries about your preferred platform.

Sections and items marked as 🟢 are updated in the manuals.

The following table summarizes the new and revised replication terminology introduced in GT.M V5.5-000.

Pre V5.5-000	Pre V5.5-000 qualifier	V5.5-000	V5.5-000 qualifier
originating instance or primary instance	-rootprimary	originating instance or originating primary instance. Within the context of a replication connection between two instances, an originating instance is referred to as source instance or source side. For example, in an B<-A->C replication configuration, A is the source instance for B and C.	-updok (recommended) -rootprimary (still accepted)

Conventions

Pre V5.5-000	Pre V5.5-000 qualifier	V5.5-000	V5.5-000 qualifier
replicating instance (or secondary instance) and propagating instance	N/A for replicating instance or secondary instance. -propagateprimary for propagating instance	replicating instance. Within the context of a replication connection between two instances, a replicating instance that receives updates from a source instance is referred to as receiving instance or receiver side. For example, in an B<-A->C replication configuration, both B and C can be referred to as a receiving instance.	-updnok
N/A	N/A	supplementary instance. For example, in an A->P->Q replication configuration, P is the supplementary instance. Both A and P are originating instances.	-updok

Release Overview

V5.5-000 provides a new type of replication, described in more detail in the Supplementary Instance Technical Bulletin. With Supplementary Instance (SI) replication, GT.M provides a mechanism to replicate to a supplementary instance which can execute its own business logic and commit database updates, for example, to provide reporting, decision support, data warehousing, auditing, and so on. In the course of that work, we have improved other aspects of replication.

MUPIP REORG now has a -TRUNCATE option, to return unused space at the end of a database file to the host file system.

An enhancement to the DO command allows for label invocations using DO to no longer require parentheses for calls with no actualist. This change also improves the performance of routine invocation.

MUPIP ROLLBACK now has a field test grade functionality implementation of an -ONLINE option. This anticipates future GT.M enhancements and is not intended to be used in production as currently implemented.

The encryption reference plug-in supports a method you can employ if you want to pass an obfuscated password between unrelated processes, or even from one system to another.

As usual, release includes many smaller improvements and corrections. For a comprehensive list, refer to Change History.

Platforms

Over time, computing platforms evolve. Vendors obsolete hardware architectures. New versions of operating systems replace old ones. We at FIS continually evaluate platforms and versions of platforms that should be Supported for GT.M. In the table below, we document not only the ones that are currently Supported for this release, but also alert you to our future plans given the evolution of computing platforms. If you are an FIS customer, and these plans would cause you hardship, please contact your FIS account executive promptly to discuss your needs.

As of the publication date, FIS supports this release on the following hardware and operating system versions. Contact FIS for a current list of supported platforms.

Platform	Supported Versions	Notes
Hewlett-Packard Integrity IA64 HP-UX	11V3 (11.31)	-
IA64 GNU/Linux - Red Hat Enterprise Linux	Red Hat Enterprise Linux 5.6	GT.M should also run on recent releases of other major Linux distributions with a contemporary 2.6 Linux kernel, glibc (version 2.5-24 or later) and ncurses (version 5.5-24 or later). FIS has verified that GT.M passes comprehensive testing on RHEL 5.x on machines that have single cells (no

Platform	Supported Versions	Notes
		<p>more than 8 CPUs). Multi-cell machines are not considered suitable for production use until they are certified.</p> <p>Although this platform remains at present fully supported, the announcement that it will not be supported by Red Hat Enterprise Linux 6 makes it likely that future GT.M releases will not be supported on this platform.</p> <p>Please contact your FIS account manager if you need GT.M on this platform.</p>
Hewlett-Packard PA-RISC HP-UX	11.11	<p>GT.M supports UTF-8 mode and M mode on this platform subject to the following:</p> <ul style="list-style-type: none"> • Problems with HP-UX 11.11 prevent FIS from testing 4 byte UTF-8 characters. FIS understands that the issue is resolved in HP-UX 11.31. At this time, GT.M is still untested and not formally supported on HP-UX 11.31; however, we are not aware of anything that would prevent this GT.M release from working correctly on that newer version. <p>Running GT.M on HP-UX 11i requires that patch PHKL_28475 be applied to the system. This patch fixes a problem with the lseek64() C library call that GT.M uses. A system without this patch gives fairly consistent database errors of varying types, structural damage, and in general does not work correctly for any but the most simplistic usage. The swlist -p command (executed as root) can be used to determine if this patch has been applied. Since recent "BATCH" and "GOLDEN" patches may contain this patch, your system may already have this patch applied but may not list it separately. Contact your HP support channel for more information.</p> <p>GT.M does not support database encryption on this platform.</p> <p>GT.M does not support the Memory Mapped (MM) Access Method on this platform.</p> <p>Although this platform remains at present fully supported with respect to bug fixes, owing to its looming sunset by HP, new functionality is supported on this platform only for FIS' convenience. It is likely that future GT.M releases will not be supported on this platform.</p> <p>Please contact your FIS account manager if you need GT.M on this platform.</p>
Hewlett-Packard Alpha/AXP Tru64 UNIX	5.1B	<p>GT.M supports M mode but not UTF-8 mode on this platform. GT.M does not support database encryption on this platform.</p> <p>Although this platform remains at present fully supported with respect to bug fixes, owing to its looming sunset by HP, new functionality is supported on this platform only for FIS' convenience. It is likely that future GT.M releases will not be supported on this platform.</p>

Platform	Supported Versions	Notes
		Please contact your FIS account manager if you need GT.M on this platform.
Hewlett-Packard Alpha/AXP OpenVMS	7.3-1 / 7.3-2 / 8.2 / 8.3	<p>GT.M supports M mode but not UTF-8 mode on this platform. GT.M does not support several recent enhancements on this platform, including but not limited to database encryption, on-line backup, multi-site replication, PIPE devices and triggers.</p> <p>If you need to work with external calls written in C with Version 6.x of the C compiler on Alpha OpenVMS, then you must carefully review all the provided kits for that product and apply them appropriately.</p> <p>Although this platform remains at present fully supported with respect to bug fixes, owing to its looming sunset by HP, new functionality is supported on this platform only for FIS' convenience. Future GT.M releases may not be supported on this platform. Regardless of ongoing plans for support of the OpenVMS platform itself, the next GT.M release will likely no longer support OpenVMS 7.x. Please contact your FIS account manager if you need ongoing support for GT.M on this platform.</p>
IBM System p AIX	6.1, 7.1	<p>Since GT.M processes are 64-bit, FIS expects 64-bit AIX configurations to be preferable.</p> <p>While GT.M supports both UTF-8 mode and M mode on this platform, there are problems with the AIX ICU utilities that prevent FIS from testing 4-byte UTF-8 characters as comprehensively on this platform as we do on others.</p> <p>Running GT.M on AIX 7.1 requires APAR IZ87564, a fix for the POW() function. This fix is present in AIX 7.1 TL01 SP04 and above. It can also be applied to prior AIX 7.1 releases. To verify that this fix has been installed, execute instfix -ik IZ87564.</p>
GNU/Linux on IBM System z	SuSE Linux Enterprise Server 10.3 and 11 or later	<p>GT.M support starts at SuSE Linux Enterprise Server 10 Service Pack 3. On SuSE Linux Enterprise Server 11 or later, we require the installation of the SuSE provided libelf0-0.8.10-37.10, or later, package. Future GT.M releases are likely to no longer support SLES 10 and to support only SLES 11.</p> <p>Please contact your FIS account manager if you need GT.M on this platform.</p>
IBM System z z/OS	V1R11	Please contact your FIS account manager if you need GT.M on this platform.
Sun SPARC Solaris	10 (Update 6 and above)	The deprecated DAL calls operate in M mode but not in UTF-8 mode. Please refer to the Integrating External Routines chapter in the Programmer's Guide for appropriate alternatives.
x86_64 GNU/Linux	Red Hat Enterprise Linux 5.6; Ubuntu 8.04 LTS through	To run 64-bit GT.M processes requires both a 64-bit kernel as well as 64-bit hardware.

Platform	Supported Versions	Notes
	10.10; SuSE Linux Enterprise Server 11	<p>GT.M should also run on recent releases of other major Linux distributions with a contemporary 2.6 Linux kernel, glibc (version 2.5-24 or later) and ncurses (version 5.5 or later).</p> <p>To install GT.M with Unicode (UTF-8) support on RHEL 5.4, in response to the installation question Should an ICU version other than the default be used? (y or n) please respond y and then specify the ICU version (for example, respond 3.6) to the subsequent prompt Enter ICU version (ICU version 3.6 or later required. Enter as <major-ver>.<minor-ver>):</p> <p>Future GT.M releases may no longer support Red Hat Enterprise Linux 5.x. Please contact your FIS account manager if you need ongoing support for GT.M on RHEL 5.x. We anticipate support for RHEL 6 in the future.</p>
x86 GNU/Linux	Red Hat Enterprise Linux 5.5	<p>This 32-bit version of GT.M runs on either 32- or 64-bit x86 platforms; we expect the X86_64 GNU/Linux version of GT.M to be preferable on 64-bit hardware.</p> <p>GT.M should also run on recent releases of other major Linux distributions with a contemporary 2.6 Linux kernel, glibc (version 2.5-49 or later) and ncurses (version 5.5-24 or later). The minimum CPU must have the instruction set of a 686 (Pentium Pro) or equivalent.</p> <p>Future GT.M releases may no longer support Red Hat Enterprise Linux 5.x. Please contact your FIS account manager if you need ongoing support for GT.M on RHEL 5.x. We anticipate support for RHEL 6 in the future.</p>

Platform support lifecycle

FIS usually supports new operating system versions six months after stable releases are available and we usually support each version for a two year window. GT.M releases are also normally supported for two years after release. While FIS will attempt to provide support to customers in good standing for any GT.M release and operating system version, our ability to provide support is diminished after the two year window.

GT.M cannot be patched, and bugs are only fixed in new releases of software.

Migrating to 64-bit platforms

The same application code runs on both 32-bit and 64-bit platforms. Please note that:

- You must compile the application code separately for each platform. Even though the M source code is exactly the same, the generated object modules are different even on the same hardware architecture - the object code differs between x86 and x86_64.
- Parameter-types that interface GT.M with non-M code using C calling conventions must match the data-types on their target platforms. Mostly, these parameters are for call-ins, external calls,

internationalization (collation), and environment translation and are listed in the tables below. Note that most addresses on 64-bit platforms are 8 bytes long and require 8 byte alignment in structures whereas all addresses on 32-bit platforms are 4 bytes long and require 4-byte alignment in structures.

Call-ins and External Calls

Parameter type	32-Bit	64-bit	Remarks
gtm_long_t	4-byte (32-bit)	8-byte (64-bit)	gtm_long_t is much the same as the C language long type, except on Tru64 UNIX, where GT.M remains a 32-bit application.
gtm_ulong_t	4-byte	8-byte	gtm_ulong_t is much the same as the C language unsigned long type.
gtm_int_t	4-byte	4-byte	gtm_int_t has 32-bit length on all platforms.
gtm_uint_t	4-byte	4-byte	gtm_uint_t has 32-bit length on all platforms

 If your interface uses gtm_long_t or gtm_ulong_t types but your interface code uses int or signed int types, failure to revise the types so they match on a 64-bit platform will cause the code to fail in unpleasant, potentially dangerous and hard to diagnose ways.

Internationalization (Collation)

Parameter type	32-Bit	64-bit	Remarks
gtm_descriptor in gtm_descript.h	4-byte	8-byte	Although it is only the address within these types that changes, the structures may grow by up to 8 bytes as a result of compiler padding to meet platform alignment requirements.

 Assuming other aspects of their code are 64-bit capable, collation routines should require only recompilation.

Environment Translation

Parameter type	32-Bit	64-bit	Remarks
gtm_string_t type in gtmxc_types.h	4-byte	8-byte	Although it is only the address within these types that changes, the structures may grow by up to 8 bytes as a result of compiler padding to meet platform alignment requirements.

 Assuming other aspects of their code are 64-bit capable, environment translation routines should require only recompilation.

Recompile

- Recompile all M and C source files.

Rebuild Shared Libraries or Images

- Rebuild all Shared Libraries (UNIX) or Shareable Executable Images (OpenVMS) after recompiling all M and C source files.

Additional Installation Instructions

To install GT.M, see the "Installing GT.M" section in the GT.M Administration and Operations Guide. For minimal down time upgrade a current replicating instance, restart replication, once that replicating instance is current, switch it over to originating instance and upgrade the prior originating instance to become a replicating instance, at least until it's current.

UNIX

- FIS strongly recommends installing each version of GT.M in a separate (new) directory, rather than overwriting a previously installed version. If you have a legitimate need to overwrite an existing GT.M installation with a new version, you must first shut down all processes using the old version. FIS suggests installing GT.M V5.5-000 in a Filesystem Hierarchy Standard compliant location such as **/usr/lib/fis-gtm/V5.5-000_arch** (for example, **/usr/lib/fis-gtm/V5.5-000_x86** on 32-bit Linux systems). A location such as **/opt/fis-gtm/V5.5-000_arch** would also be appropriate. Note that the **arch** suffix is especially important if you plan to install 32- and 64-bit versions of the same release of GT.M on the same system.
- Use the MUPIP RUNDOWN command of the old GT.M version to ensure all database files are cleanly closed.
- In UNIX editions, make sure gtmsecshr is not running. If gtmsecshr is running, first stop all GT.M processes including the DSE, LKE and MUPIP utilities and then perform a **MUPIP STOP *pid_of_gtmsecshr***.

 Never replace the binary image on disk of any executable file while it is in use by an active process. It may lead to unpredictable results. Depending on the operating system, these results include but are not limited to denial of service (that is, system lockup) and damage to files that these processes have open (that is, database structural damage).

Additional Information for JFS1 on AIX

If you expect a database file or journal file to exceed 2GB with older versions of the JFS file system, then you must configure its file system to permit files larger than 2GB. Furthermore, should you choose to place journal files on file systems with a 2GB limit, since GT.M journal files can grow to a maximum size of 4GB, you must then set the journal auto switch limit to less than 2 GB.

OpenVMS

To upgrade from a GT.M version prior to V4.3-001, you must update any customized copy of **GTM \$DEFAULTS** to include a definition for **GTM\$ZDATE_FORM**.

You can ignore the following section if you choose the standard GT.M configuration or answer yes to the following question:

```
Do you want to define GT.M commands to the system
```

If you define GT.M commands locally with **SET COMMAND GTM\$DIST:GTMCOMMANDS.CLD** in **GTMLOGIN.COM** or other command file for each process which uses GT.M, you must execute the same command after installing the new version of GT.M before using it. If you define the GT.M commands to the system other than during the installation of GT.M, you must update the system DCLTABLES with the new **GTMCOMMANDS.CLD** provided with this version of GT.M. See the OpenVMS "Command Definition, Librarian, and Message Utilities Manual" section on "Adding a system command." In both cases, all GT.M processes must match the proper **GTMCOMMANDS.CLD** with the version of GT.M they run.

Upgrading to GT.M V5.5-000

The GT.M database consists of four types of components- database files, journal files, global directories, and replication instance files. The format of some database components differs for this GT.M version between 32-bit and 64-bit GT.M releases for the x86 GNU/Linux platform.

GT.M upgrade procedure for V5.5-000 consists of 5 stages:

- Stage 1: Upgrading your Global Directory
- Stage 2: Upgrading your Database Files
- Stage 3: Upgrading your Replication Instance File
- Stage 4: Upgrading your Journal Files
- Stage 5: Upgrading your Trigger Definitions

Read the upgrade instructions of each stage carefully. Your upgrade procedure for GT.M V5.5-000 depends on your GT.M upgrade history and your current version.

Stage 1: Upgrading your Global Directory

FIS strongly recommends you make a copy of your Global Directory before upgrading it. There is no single-step way to downgrade a Global Directory to an earlier format.

To upgrade from any prior GT.M version:

- Open your Global Directory with the GDE utility program from GT.M V5.5-000.

- Run the EXIT command. This command automatically, even with no other intervening commands, upgrades the Global Directory.

To switch between 32- and 64-bit global directories on the x86 GNU/Linux platform:

- Open your Global Directory with the GDE utility program on the 32-bit platform.
- On GT.M versions that support it, execute the SHOW -COMMAND -FILE=<file-name>. The command file file-name now holds the current Global Directory settings.
- On GT.M versions that don't support GDE SHOW -COMMAND, use SHOW -ALL and either edit the result into an appropriate command file or use them to guide you in manually reentering the information into GDE.
- Open GDE on the 64-bit platform. If you have a command file from 2. or 3., execute @<file-name> and then run the EXIT command. These commands automatically recreate the Global Directory. Otherwise use the GDE output from the old global directory to replicate the configuration in the new environment.

The same procedure applies in the reverse direction.

If you inadvertently open a global directory in an earlier format, with no intention of upgrading it, execute the QUIT command rather than the EXIT command.

If you inadvertently upgrade a global directory, perform the following steps:

- Open the global directory with GDE from V5.5-000.
- Execute the SHOW -COMMAND -FILE=<file-name> command to create a command file to use to recreate the global directory on the older version. If the old version is significantly out-of-date, you may need to edit the command file to remove text that only applies to more recent versions. Alternatively you can use the output from SHOW -ALL or SHOW -COMMAND as a guide to manually reestablish the Global Directory for the older version.

Stage 2: Upgrading your Database Files

You need to upgrade your database files only when there is a block format upgrade (such as V4->V5). However, some versions, for example, database files which have been initially been created with V4 (and subsequently upgraded to a V5 format) may additionally need a MUPIP REORG -UPGRADE operation to upgrade previously used but free blocks that may have been missed by earlier upgrade tools.

To upgrade from a GT.M version prior to V5.000:

- Upgrade your database files using in-place or traditional database upgrade procedure depending on your situation. For more information on in-place/traditional database upgrade, see Database Migration Technical Bulletin.
- Run the MUPIP REORG -UPGRADE command. This command upgrades all V4 blocks to V5 format.

 Databases created with GT.M releases prior to V5.0-000 and upgraded to a V5 format retain a maximum size limit of 64M (67,108,864) blocks.

To upgrade from GT.M V5.0*/V5.1*/V5.2*/V5.3*/V5.4*:

No database file upgrade procedure is necessary if you upgrade from GT.M V5.0-000 or later to V5.3-000 or later. However, you may need to run the MUPIP REORG -UPGRADE command to upgrade any previously used but free blocks that may have been missed during earlier upgrade cycles. You do not need to run MUPIP REORG -UPGRADE in either of the following situations:

- A database was created by a V5 MUPIP CREATE
- A database has previously been completely processed by a MUPIP REORG -UPGRADE from V5.3-003 or later

If you have already run the MUPIP REORG -UPGRADE command in a version prior to V5.3-003, subsequent versions cannot determine whether or not it was done correctly and records warnings in the operator log for running MUPIP REORG -UPGRADE. Therefore, you must either:

- Run the MUPIP REORG -UPGRADE command again
- Run the DSE CHANGE -FULLY_UPGRADED=1 command to stop the warnings

 Do not run the DSE CHANGE -FILEHEADER -FULLY_UPGRADED=1 command unless you are absolutely certain you have previously successfully run to completion MUPIP REORG -UPGRADE from V5.3-003 or later. Using this command inappropriately may lead to database integrity issues.

For additional upgrade considerations, refer to Database Compatibility Notes.

 If your application code uses triggers, when upgrading to V5.5-000 from V5.4-001, triggers must be extracted from the database and deleted therein using the prior GT.M version, and the loaded with the new GT.M version.

Database Compatibility Notes

- Changes to the database file header may occur in any release. GT.M automatically upgrades database file headers as needed. Any changes to database file headers are upward and downward compatible within a major database release number, that is, although processes from only one GT.M release can access a database file at any given time, processes running different GT.M releases with the same major release number can access a database file at different times.
- Databases created with V5.3-004 and later can grow to a maximum size of 224M (234,881,024) blocks. This means, for example, that with an 8KB block size, the maximum database file size is 1,792GB; this is effectively the size of a single global variable that has a region to itself; a database consists of any number of global variables. A database created with GT.M versions V5.0-000 through V5.3-003 can be upgraded with MUPIP UPGRADE to increase the limit on database file size from 128M to 224M blocks.
- Databases created with V5.0-000 through V5.3-003 have a maximum size of 128M (134,217,728) blocks. GT.M versions V5.0-000 through V5.3-003 can access databases created with V5.3-004 and later as long as they remain within a 128M block limit.

Stage 3: Upgrading your Replication Instance File

V5.5-000 requires new replication instance files. Instructions for creating them are in the Supplementary Instance Replication Technical Bulletin. Shut down all Receiver Servers on other instances looking for updates from this instance, shut down this instance, recreate the instance file and then restart any Receiver Server for this instance with the `-UPDATERESYNC` qualifier.

-  Without the `UPDATERESYNC` qualifier, the replicating instance synchronizes with the originating instance using state information from both instances and potentially rolling back information on the replicating instance. The `UPDATERESYNC` qualifier declares the replicating instance to be in a wholesome state matching some prior (or current) state of the originating instance; it causes MUPIP to update the information in the replication instance file of the originating instance and not modify information currently in the database on the replicating instance. After this command, the replicating instance catches up to the originating instance starting from its own current state. Use `UPDATERESYNC` only when you are absolutely certain that the replicating instance database was shut down normally with no errors, or appropriately copied from another instance with no errors.
-  You must always follow the steps in the Multi-Site Replication technical bulletin when migrating from a logical dual site (LDS) configuration to an LMS configuration, even if you are not changing GT.M releases.

Stage 4: Upgrading your Journal Files

On every GT.M upgrade:

- Create a fresh backup of your database.
 - Generate new journal files (without back-links).
-  This is necessary because MUPIP JOURNAL cannot use journal files from a release other than its own for `RECOVER`, `ROLLBACK`, or `EXTRACT`.

Stage 5: Upgrading your Trigger Definitions

If you are upgrading from V5.4-002A/V5.4-002B to V5.5-000, you do not need to extract and reload triggers.

You need to extract and reload your trigger definitions only if you are upgrading from V5.4-000/V5.4-000A/V5.4-001 to V5.5-000. This is because multi-line xecutes for triggers necessitated a change in the internal storage for triggers, rendering triggers created in V5.4-000/V5.4-000A/V5.4-001 incompatible with V5.4-002/V5.4-002A/V5.4-002B/V5.5-000.

To extract and reapply the trigger definitions on V5.5-000 using MUPIP TRIGGER:

1. Execute a command like `mupip trigger -select="*" trigger_defs.trg`. The output file `trigger_defs.trg` now holds all trigger definitions.
2. Place `-*` at the beginning of the `trigger_defs.trg` file to remove the old trigger definitions before reapplying them.

3. Run **mupip trigger -triggerfile=trigger_defs.trg** to reapply your trigger definitions.

To extract and reapply the trigger definitions on a V5.5-000 replicating instance using `$ZTRIGGER()`:

1. Shut down the instance using the older version of GT.M.
2. Execute a command like **mumps -run %XCMD 'i \$ztrigger("select")' > trigger_defs.trg**. The output file `trigger_defs.trg` now holds all trigger definitions.
3. Turn off replication on all regions.
4. Run **mumps -run %XCMD 'i \$ztrigger("item","-*")** to remove the old trigger definitions.
5. Perform the upgrade procedure applicable for V5.5-000.
6. Run **mumps -run %XCMD 'if \$ztrigger("file","trigger_defs.trg")** to reapply your trigger definitions.
7. Turn replication back on.
8. Connect to the originating instance.

Managing M mode and UTF-8 mode

On selected platforms, with International Components for Unicode (ICU) version 3.6 or later installed, GT.M's UTF-8 mode provides support for Unicode (ISO/IEC-10646) character strings. On other platforms, or on a system that does not have ICU 3.6 or later installed, GT.M only supports M mode.

On a system that has ICU installed, GT.M optionally installs support for both M mode and UTF-8 mode, including a `utf8` subdirectory of the directory where GT.M is installed. From the same source file, depending upon the value of the environment variable `$gtm_chset`, the GT.M compiler generates an object file either for M mode or UTF-8 mode. GT.M generates a new object file when it finds both a source and an object file, and the object predates the source file and was generated with the same setting of `$gtm_chset/$ZCHset`. A GT.M process generates an error if it encounters an object file generated with a different setting of `$gtm_chset/$ZCHset` than that processes' current value.

Always generate an M object module with a value of `$gtm_chset/$ZCHset` matching the value processes executing that module will have. As the GT.M installation itself contains utility programs written in M, their object files also conform to this rule. In order to use utility programs in both M mode and UTF-8 mode, the GT.M installation ensures that both M and UTF-8 versions of object modules exist, the latter in the `utf8` subdirectory. This technique of segregating the object modules by their compilation mode prevents both frequent recompiles and errors in installations where both modes are in use. If your installation uses both modes, consider a similar pattern for structuring application object code repositories.

GT.M is installed in a parent directory and a `utf8` subdirectory as follows:

- Actual files for GT.M executable programs (`mumps`, `mupip`, `dse`, `lke`, and so on) are in the parent directory, that is, the location specified for installation.

- Object files for programs written in M (GDE, utilities) have two versions - one compiled with support for Unicode in the utf8 subdirectory, and one compiled without support for Unicode in the parent directory. Installing GT.M generates both the versions of object files, as long as ICU 3.6 or greater is installed and visible to GT.M when GT.M is installed, and you choose the option to install Unicode support.
- The utf8 subdirectory has files called mumps, mupip, dse, lke, and so on, which are relative symbolic links to the executables in the parent directory (for example, mumps is the symbolic link ../mumps).
- When a shell process sources the file gtmprofile, the behavior is as follows:
 - If \$gtm_chset is "m", "M" or undefined, there is no change from the previous GT.M versions to the value of the environment variable \$gtmroutines.
 - If \$gtm_chset is "UTF-8" (the check is case-insensitive),
 - \$gtm_dist is set to the utf8 subdirectory (that is, if GT.M is installed in /usr/lib/fis-gtm/gtm_V5.5-000_i686, then gtmprofile and gtmshrc set \$gtm_dist to /usr/lib/fis-gtm/gtm_V5.5-000_i686/utf8).
 - On platforms where the object files have not been placed in a libgtmutil.so shared library, the last element of \$gtmroutines is \$gtm_dist(\$gtm_dist/..) so that the source files in the parent directory for utility programs are matched with object files in the utf8 subdirectory. On platforms where the object files are in libgtmutil.so, that shared library is the one with the object files compiled in the mode for the process.

For more information on gtmprofile and gtmshrc, refer to the Basic Operations chapter of UNIX Administration and Operations Guide.

Compiling ICU

GT.M versions prior to V5.3-004 require exactly ICU 3.6, however, V5.3-004 (or later) accept ICU 3.6 or later. For sample instructions to download ICU, configure it not to use multi-threading, and compiling it for various platforms, refer to Appendix C: Compiling ICU on GT.M supported platforms of the UNIX Administration and Operations Guide.

Although GT.M can use ICU, ICU is not FIS software and FIS does not support ICU. The sample instructions for installing and configuring ICU are merely provided as a convenience to you.

Also, note that download sites, versions of compilers, and milli and micro releases of ICU may have changed ICU since the embedded dates when these instructions were tested making them out-of-date. Therefore, these instructions must be considered examples, not a cookbook.

Setting the environment variable TERM

The environment variable \$TERM must specify a terminfo entry that accurately matches the terminal (or terminal emulator) settings. Refer to the terminfo man pages for more information on the terminal settings of the platform where GT.M needs to run.

- Some terminfo entries may seem to work properly but fail to recognize function key sequences or position the cursor properly in response to escape sequences from GT.M. GT.M itself does not have any knowledge of specific terminal control characteristics. Therefore, it is important to specify the right terminfo entry to let GT.M communicate correctly with the terminal. You may need to add new terminfo entries depending on your specific platform and implementation. The terminal (emulator) vendor may also be able to help.
- GT.M uses the following terminfo capabilities. The full variable name is followed by the capname in parenthesis:

```
auto_right_margin(am), clr_eos(ed), clr_eol(e1), columns(cols), cursor_address(cup),
  cursor_down(cud1),
  cursor_left(cub1), cursor_right(cuf1), cursor_up(cuu1), eat_newline_glitch(xenl),
  key_backspace(kbs), key_dc(kdch1),key_down(kcud1), key_left(kcub1), key_right(kcuf1),
  key_up(kcuu1),
  key_insert(kich1), keypad_local(rmkx),keypad_xmit(smxx), lines(lines).
```

GT.M sends keypad_xmit before terminal reads for direct mode and READs (other than READ *) if EDITING is enabled. GT.M sends keypad_local after these terminal reads.

Installing Compression Libraries

If you plan to use the optional compression facility for replication, you must provide the compression library. The GT.M interface for compression libraries accepts the zlib compression libraries without any need for adaptation. These libraries are included in many UNIX distributions and are downloadable from the zlib home page. If you prefer to use other compression libraries, you need to configure or adapt them to provide the same API provided by zlib. Simple instructions for compiling zlib on a number of platforms follow. Although GT.M can use zlib, zlib is not FIS software and FIS does not support zlib. These instructions are merely provided as a convenience to you.

If a package for zlib is available with your operating system, FIS suggests that you use it rather than building your own.

Solaris/cc compiler from Sun Studio:

```
./configure --sharedmake CFLAGS="-KPIC -m64"
```

HP-UX(IA64)/HP C compiler:

```
./configure --sharedmake CFLAGS="+DD64"
```

AIX/XL compiler:

```
./configure --sharedAdd -q64 to the LDFLAGS line of the Makefilemake CFLAGS="-q64"
```

Linux/gcc:

```
./configure --sharedmake CFLAGS="-m64"
```

z/OS:

Refer to the steps FIS used to install zlib on z/OS in the GT.M for z/OS technical bulletin.

By default, GT.M searches for the libz.so shared library (libz.sl on HPUX PA-RISC) in the standard system library directories (for example, /usr/lib, /usr/local/lib, /usr/local/lib64). If the shared library is installed in a non-standard location, before starting replication, you must ensure that the environment variable \$LIBPATH (AIX and z/OS) or \$LD_LIBRARY_PATH (other UNIX platforms) includes the directory containing the library. The source and Receiver Server link the shared library at runtime. If this fails for any reason (such as file not found, or insufficient authorization), the replication logic logs a DLLNOOPEN error and continues with no compression.

Change History

V5.5-000

Fixes and enhancements specific to V5.5-000 are:

Id	Prior Id	Category	Summary
GTM-2296	C9905-001114	Utilities	Improved DSE FIND -SIBLING functionality.
GTM-3873	C9B09-001740	MUPIP	New -TRUNCATE qualifier for MUPIP REORG in UNIX.
GTM-4033	C9B12-001841	MUMPS	TROLLBACK to a \$TLEVEL other than zero (0) leaves \$REFERENCE empty which is the same behavior as a full TROLLBACK to 0=\$TEVEL.
GTM-4202	C9C03-001934	MUPIP	Correction to NOREPLCTDREG error message context in UNIX.
GTM-4337	C9C05-002005	DB	UNIX gtm_db_startup_max_wait environment variable to control waiting on resource conflicts.
GTM-5797 	S9F12-002581	MUPIP	New -REGION qualifier for MUPIP REORG operations.
GTM-5996	C9H04-002848	MUMPS	Interrupted GT.M HANG commands account for the time spent handling the interrupt when the hang resumes.
GTM-5997	C9H04-002849	MUMPS	Interrupted timed GT.M LOCK commands account for the time spent waiting before the interrupt when the LOCK resumes its wait.
GTM-6185	S9I05-002991	MUMPS	GT.M evaluates command postconditionals for invalid commands at runtime before generating an INVCMD error.
GTM-6224	S9I08-002697	Utilities	Improved the setting of Journaling attributes with GDE.
GTM-6228	C9I08-003014	MUMPS	GT.M properly manages stack frames when fielding CTRAP interrupts from the console.
GTM-6306	C9I12-003062	DB	UNIX MUPIP JOURNAL -ROLLBACK -ONLINE capability and associated primarily operational changes.
GTM-6340	C9J02-003091	MUPIP	Supplementary Instances and operational changes primarily to replication.
GTM-6736	S9L01-002801, C9J03-003101	MUPIP	New JNLNOREPL error for the Source Server and PREVJNLLINKSET message to the operator log.

Id	Prior Id	Category	Summary
GTM-6449	S9J08-002739	MUMPS	<CTRL-D> at the GT.M Direct Mode prompt on an empty line terminates GT.M and returns control to the shell in UNIX.
GTM-6569	C9K03-003250	MUMPS	In UNIX, GT.M defers MUPIP STOP for a time window during which a process should not be terminated.
GTM-6614	C9K05-003274	MUMPS	In UNIX, GT.M ignores SIGINT (control-C) if \$PRINCIPAL is not a terminal.
GTM-6622	S9K06-002778	MUMPS	GT.M better handles some error processing cases.
GTM-6650	C9K07-003295	MUPIP	A Receiver Server started with the -UPDATERESYNC qualifier starts every connection handshake (with a Source Server) during its own lifetime with that characteristic.
GTM-6664	C9K08-003307	Utilities	Enhancements for configure, gtmprofile, and gtminstall scripts in UNIX.
GTM-6711	C9K10-003334	MUMPS	When returning from code invoked by MUPIP INTRPT (clearing \$ZININTERRPT), GT.M implicitly clears any error(s) detected while 1=\$ZININTERRUPT and sends a JIUNHNDINT error notification to the operator log.
GTM-6719	C9K11-003340	MUMPS	Improved management of internal timers in UNIX.
GTM-6725	C9K12-003344	MUPIP	Improved handling of database migration from V4 format.
GTM-6738	C9L01-003352	MUMPS	New %RANDSTR string generator utility function.
GTM-6759	S9L03-002806	MUMPS	Improved handing of TPTIMEOUT interrupts.
GTM-6771	C9L03-003376	MUMPS	In UNIX, ZMESSAGE treats any non-GT.M error as a trappable or fatal event.
GTM-6811	C9L05-003414	DB	Improved ftok algorithm for shared resources on UNIX.
GTM-6813	C9L05-003416	MUMPS	Change to actualist-formallist handling mostly affecting DO
GTM-6819	S9L06-002815	MUMPS	New gtm_trace_gbl_name environment variable and M-profiling enhancements.
GTM-6826 🟢	S9L06-002819	MUPIP	GT.M no longer requires that an external filter treat its output as a transaction in UNIX.
GTM-6834	C9L06-003421	MUPIP	MUPIP RECOVER -BACKWARD restricts processing that identifies broken transactions to the most recent journal file(s).
GTM-6836	C9L06-003423	MUMPS	Improved SOCKET device handling of WRITE /LISTEN and WRITE /WAIT.

Id	Prior Id	Category	Summary
GTM-6841	C9L06-003428	MUPIP	The Source Server no longer hangs while waiting for a REPL_CMP_SOLVE message in UNIX.
GTM-6845	C9L06-003432	MUMPS	The GT.M parser now appropriately recognizes the end of the last source line in a routine.
GTM-6849	S9L07-002821	Utilities	Corrected typographical errors in three messages.
GTM-6851	S9L07-002822	MUMPS	Improved error handling for \$ZDATE().
GTM-6854	S9L07-002823	MUMPS	With VIEW "FULL_BOOLEAN", GT.M now evaluates functions with side effects - extrinsic functions (\$\$), external functions (\$&) and \$INCREMENT() in left-to-right order.
GTM-6856	S9L07-002824	MUMPS	GT.M correctly compiles expressions like if a<\$\$x(b-c) when specifying FULL_BOOLEAN.
GTM-6860	C9L07-003439	Utilities	DSE DUMP -FILE -ALL output prints journal record types in hexadecimal.
GTM-6896	--	MUPIP	Fixes to UNIX cross-endian replication.
GTM-6898	--	MUPIP	Improved replication of duplicate KILLS.
GTM-6935	--	MUMPS	MERGE no longer gives an error when the source and destination variables are identical.
GTM-6939	--	Utilities	%RO no longer strips trailing spaces or program text starting with a quoted semi-colon
GTM-6953	--	DB	Database encryption reference plug-in improvements in UNIX.
GTM-6957	--	MUMPS	\$ZTRNLNM() recognizes all the documented keywords including all the OpenVMS keywords.
GTM-6958	--	MUPIP	MUPIP INTEG and MUPIP JOURNAL -ROLLBACK (or -RECOVER) now operate successfully on database files larger than 1TB.
GTM-6994	--	MUMPS	64-bit GT.M (all except HP Alpha (OpenVMS and Tru64), HP PA-RISC (HP-UX), and 32-bit x86 (Linux)) now passes 64-bit long and ulong values between C and M though the call-out and call-in interfaces.
GTM-6996	--	MUMPS	ZSHOW "D" that includes one or more SOCKET devices no longer causes memory leak.
GTM-7002	--	MUMPS	Fixes for INDEPENDENT and RENAME device parameters for the PIPE device in UNIX.
GTM-7004	--	MUPIP	Closed tiny hole in MUPIP FREEZE.

Id	Prior Id	Category	Summary
GTM-7005	--	Utilities	More meaningful GDE error messages for encryption and MM access method set.
GTM-7007	--	MUMPS	Corrected handling of subscripted FOR control variables in XECUTE strings.
GTM-7008	--	MUMPS	Fix for XECUTE "DO @x" where x doesn't specify an actualist.
GTM-7016	--	MUPIP	New -STDIN qualifier for MUPIP LOAD in UNIX.
GTM-7018	--	MUMPS	Duplicate SET optimization now the default.
GTM-7020	--	DB	UNIX trigger updates don't get spurious TRIGDEFBAD errors.
GTM-7021	--	Utilities	Cleaner operation and organization of the UNIX installation kit.
GTM-7027	--	Utilities	LKE now issues NOLOCKMATCH when a non-existing lock is searched with SHOW or attempted to be released with CLEAR.
GTM-7029	--	Utilities	GT.M build successful on Ubuntu 11.10 with GCC 4.6.
GTM-7034	--	MUPIP	MUPIP JOURNAL RECOVER (or ROLLBACK) - BACKWARD now processes multi-region broken TP transactions appropriately.
GTM-7073	--	Utilities	Improved LKE message when no matching LOCK found.
GTM-7077	--	MUPIP	MUPIP EXTRACT -FORMAT=BIN now handles database blocks of maximum size.
GTM-7078	--	MUMPS	New GTMASSERT2 message to produce more information about a failing condition.
GTM-7079	--	MUPIP	MUPIP JOURNAL -SHOW=ALL output now shows counts of all Record Types correctly up to 1 billion.
GTM-7082	--	MUPIP	In UNIX, -extract="-stdout" writes to standard output (stdout) instead of writing to a file named -stdout.
GTM-7110	--	Utilities	In UNIX, GT.M utilities now give a CLIERR error when an equal-sign (=) follows a qualifier without an immediately following value.
GTM-7122	--	DB	GT.M handles journal file switches for some complex permissions patterns.
GTM-7123	--	MUMPS	DEVOPENFAIL for a deficient PIPE device specification

Id	Prior Id	Category	Summary
GTM-7158	--	DB	MUPIP TRIGGER, \$ZTRIGGER() and ZTRIGGER now operate correctly on a read-only database in UNIX.
GTM-7176	--	MUMPS	Improved error handling for ZBREAK and ZPRINT().
GTM-7179	--	MUMPS	Report global gets from MERGE in database counters displayed with ZSHOW "G".
GTM-7185	--	MUMPS	FOR evaluates a termination value specified with a subscripted local variable at loop initiation.
GTM-7188	--	MUMPS	Error messages report the address of an event in hexadecimal in UNIX.
GTM-7196	--	MUMPS	M-profiling corrections.
GTM-7206	--	MUMPS	Resolved an interaction between ZGOTO 0:entryref and triggers in UNIX.
GTM-7210	--	MUMPS	Better context for OpenVMS UNDEF error messages.
GTM-7214	--	MUMPS	LOCK management avoids a possible problem.
GTM-7215	--	MUMPS	Change to ZQUIT_ANYWAY behavior
GTM-7224	--	Utilities	The GT.M distribution kit now includes a README.txt file.
GTM-7228	--	MUMPS	The LOCK (and ZALLOCATE and ZDEALLOCATE) commands manage LOCK resource names more efficiently in LOCK_SPACE.
GTM-7230	--	MUPIP	Update Process and journal recover/rollback correctly handle NULL journal record type.
GTM-7234	--	MUMPS	Improved parameter passing with nested JOB'd processes in UNIX.

M-Database Access

- GT.M components use the `$gtm_db_startup_max_wait` environment variable to determine how long to wait for a resolution of any resource conflict when they first access a database file. GT.M uses semaphores maintained using UNIX Inter-Process Communication (IPC) services to ensure orderly initialization and shutdown of database files and associated shared memory. Normally the IPC resources are held in an exclusive state only for minuscule intervals. However, under unusual circumstances that might include extreme large numbers of simultaneous database initializations, a long-running MUPIP operation involving standalone access (like `INTEG -FILE` or `RESTORE`), an OS overload or an unpredicted process failure the resources might remain unavailable for an unanticipated length of time. `$gtm_db_startup_max_wait` specifies how long to wait for the resources to become available:
 - -1 - Indefinite wait until the resource becomes available; the waiting process invokes the `gtm_proctuckexec` mechanism, if configured, at 48 and 96 seconds.
 - 0 - No wait - if the resource is not immediately available, give a `DBFILERR` error with an associated `SEMWT2LONG`
 - > 0 - Seconds to wait - rounded to the nearest multiple of eight (8); if the specification is 96 or more seconds, the waiting process invokes the `gtm_proctuckexec` mechanism, if configured, at one half the wait and at the end of the wait; if the resource remains unavailable, the process issues `DBFILERR` error with an associated `SEMWT2LONG`

The default value for the wait if `$gtm_db_startup_max_wait` is not defined is 96 seconds. Previously, a process waited 60 seconds for one semaphore and immediately tried to acquire the other semaphore. If either of these attempts failed, it variously gave `CRITSEMFAIL` or `REQRUNDOWN` in addition to the above mentioned errors. [UNIX](GTM-4337) 

- `MUPIP JOURNAL -ROLLBACK` recognizes the `-ONLINE` qualifier which specifies that `ROLLBACK` run without requiring standalone access to the database and the replication instance file. GT.M provides the `$ZONLNRLBK` intrinsic special variable which it increments every time the process detects a concurrent `MUPIP JOURNAL -ONLINE -ROLLBACK`. If the logical state of the database after the completion of `MUPIP JOURNAL -ONLINE -ROLLBACK` matches the logical state of the database at the start of `MUPIP JOURNAL -ONLINE .ROLLBACK`, that is: only removes or commits an uncommitted TP transaction or non-TP mini-transaction, any transaction (TP or Non-TP) incurs a restart and nothing more. If the `MUPIP JOURNAL -ONLINE -ROLLBACK` changes the logical state of the database GT.M affects concurrent processing by incrementing `$ZONLNRLBK` and, in a TP transaction including trigger code within a transaction: restarts the transaction, or in a Non-TP mini-transaction, including within an implicit transaction caused by a trigger: issues a `DBROLLEDBACK` error, which, in turn, invokes the error trap if `$ETRAP` or `$ZTRAP` are in effect. Any utility/command attempted while `MUPIP JOURNAL -ONLINE -ROLLBACK` operates waits for `ROLLBACK` to complete; the `$gtm_db_startup_max_wait` environment variable (see `GTM-4337`) configures the wait. The default behavior of `MUPIP JOURNAL -ROLLBACK` is `-NOONLINE`, which requires standalone access on the database and replication instance file. Previously, `ROLLBACK` always required standalone access. At this time, FIS considers `MUPIP JOURNAL -ROLLBACK -ONLINE` to be of field test grade quality; please do not use this in production

without doing comprehensive evaluation and testing. We would especially appreciate any feedback on this feature. In its current form, we expect its main usefulness would be to automatically rollback a replicating instance in response to a rollback on the originating instance.

If ROLLBACK (either- NOONLINE or -ONLINE) terminates abnormally (say because of a kill -9), it leaves the database in a potentially inconsistent state indicated by the FILE corrupt field in the database file header. When a ROLLBACK terminates leaving this field set, all other processes receive DBFLCORRP errors any time they attempt to interact with the database. You can clear this condition as following in descending order of risk:

- Rerun ROLLBACK to completion
- MUPIP SET -FILE -PARTIAL_RECOV_BYPASS
- DSE CHANGE -FILEHEADER -CORRUPT=FALSE -NOCRIT

However, the MUPIP and DSE actions do not ensure that the database has consistent state; check for database integrity with MUPIP INTEG.

The project to introduce this feature also addressed the following changes:

GT.M manages database and replication locks more carefully. Previously, in some rare cases, a conflict between internally managed locks for a region and the replication [journal] pool could result in a deadlock.

MUPIP now correctly manages control structures if it doesn't succeed in allocating shared memory. Previously such an unusual failure leaked a small amount of memory.

MUPIP RESTORE now ensures that until the last of the incremental BACKUP files are restored, the process continues to operate in a standalone mode. Previously, it was possible for a concurrent process to sneak in while restore was switching input files and start accessing the database causing inconsistency.

MUPIP now appropriately interprets MUPIP INTEG -ONLINE as equivalent to MUPIP INTEG -ONLINE -REGION and treats the response at the "File or Region:" prompt as a region. Previously, MUPIP treated MUPIP INTEG -ONLINE as MUPIP INTEG -FILE and assumed any input provided for "File or Region:" to be a file rather than a region.

If the target database is frozen when MUPIP JOURNAL -RECOVER and -NOONLINE -ROLLBACK start, they issue a DBFRZRESETFL informational message and unfreeze the database. Previously, RECOVER did not issue any such informational message when it performed the unfreeze.

MUPIP REPLIC -START -LISTENPORT -RECEIV now supports a new optional qualifier -AUTOROLLBACK[=VERBOSE] which allows the Receiver Server to run ONLINE FETCHRESYNC ROLLBACK on receiving a REPL_ROLLBACK_FIRST message from the Source Server. The optional value keyword VERBOSE allows the ONLINE FETCHRESYNC ROLLBACK to provide more diagnostic information. Previously, on receiving REPL_ROLLBACK_FIRST message, the Receiver Server halted and required FETCHRESYNC ROLLBACK to be run before restarting the Receiver Server.

DSE ALL -WCREINIT and DSE WCREINIT now sends an operator log - DSEWCREINIT - to record this unusual event. Previously, no such message was sent to the operator log.

For more information on the areas affected by ONLINE ROLLBACK, refer to Documentation Addendum. [UNIX] (GTM-6306)

- GT.M now uses a revised filename-to-key (ftok) algorithm to locate shared resources on all UNIX platforms. The algorithm takes into account the identity of the file (inode number, generation number) and the identity of the file system containing it (major device, minor device). The previous algorithm had a higher probability of collisions between databases on different devices, with a consequent higher probability of synchronization problems.

 A prior GT.M release using the old algorithm may conflict with the current release if they concurrently attempt to initialize shared resources. Take care to avoid using prior and current GT.M versions at the same time on the same database. As with any hash algorithm, there is a very small probability of collisions with the new algorithm too; this can be detected using the "ftok" utility included with GT.M. In the event more than one database file produces the same ftok hash, copy the database file to a new location and use the copy, which will have a different key, instead of the original.

In addition, MUPIP RUNDOWN on Unix now outputs a VERMISMATCH error after two shared memory size messages when an older GT.M is using the database. Previously, MUPIP would have only output the shared memory size messages, or in some cases (AIX or Solaris with a 32-bit prior version) succeeded, leading to database corruption.

 It is still possible to corrupt a database by running an older version of MUPIP RUNDOWN while a current GT.M has the database open. Be certain to shut down all GT.M processes before switching GT.M versions. [UNIX] (GTM-6811)

- The encryption reference plug-in supports a method you can employ if you want to pass an obfuscated password between unrelated processes (e.g., a child process with a different userid invoked via a sudo mechanism), or even from one system to another (e.g., over an ssh connection). When the environment variable `gtm_obfuscation_key` names a file readable by the process, the plug-in uses an SHA-512 hash of the file's contents as the XOR mask for the obfuscated password in the environment variable `gtm_passwd`. When `gtm_obfuscation_key` does not point to a readable file, the plug-in creates an XOR mask based on the userid and inode of the mumps executable, as it did in previous versions, and then computes an SHA-512 hash of the XOR mask to use as a mask. In addition, the maskpass program handles passphrases containing blanks. Previous versions did not support blanks in the password, did not hash the mask before performing the XOR obfuscation and used a mask that was unique to the process and so could not be used to pass the password except to another process with the same user on the same GT.M installation. [UNIX] (GTM-6953)
- If the duplicate SET optimization is enabled, and if a SET causes no change in the value of a preexisting node, that update now invokes any associated triggers. Previously it did not invoke the triggers in this case. It is now up to the trigger code to check if the value of the node did or did not change due to this update and take an appropriate course of action. This also makes trigger invocation behavior consistent between an originating and its replicating instances. Previously each GT.M process on the originating

primary used to be able to control whether triggers were invoked or not for duplicate sets whereas the Update Process on the replicating instance (which performs updates on behalf of all the GT.M processes on the originating primary) inherited only one setting for trigger invocation (in case of duplicate sets) and hence did not always mirror the GT.M process in terms of trigger invocations. This could lead to potential out-of-sync situations between the originating instance and its replicating instance due to different trigger code being executed on the two instances and in addition caused a huge journal file on the replicating instance compared to the journal file on the replicating instance.

The environment variable `gtm_gvdupsetnoop` is no longer honored. By default, the duplicate set optimization is now enabled (i.e. equivalent to `gtm_gvdupsetnoop` previously being set to 1). The only way to control this feature is with a VIEW command (View "GVDUPSETNOOP":0 disables the optimization, View "GVDUPSETNOOP":1 enables it). (GTM-7018)

- Trigger updates from MUPIP TRIGGER and \$ZTRIGGER() now handle concurrency conflicts with appropriate transaction restarts. Previously concurrency conflicts with these update could cause misleading TRIGDEFBAD errors [UNIX] (GTM-7020)
- GT.M now correctly allows all processes with read-write access to a database file to switch journal files as long as they have read-write access to the journal file directory. Previously, in an edge case, GT.M did not permit a process with read-write access to a database file to perform a needed journal file switch. As part of this change, we have generated documentation to enumerate the various cases for ownership and permissions of shared resources. This documentation is reproduced in the Shared Resource Authorization Permissions section. (GTM-7122)
- MUPIP TRIGGER, \$ZTRIGGER(), and the ZTRIGGER command now operate correctly on a read-only database or in an environment where some databases are read-only and others are not. Previously, these functions ignored the read-only setting of the database updating shared memory anyway. If other processes had the database open read-write, the changes could be flushed to the database. The ZTRIGGER command assumed it had read-write access which could cause flushing issues. [UNIX] (GTM-7158)

M-Other Than Database Access

- TROLLBACK to a \$TLEVEL other than zero (0) now leaves \$REFERENCE empty which is the same behavior as a full TROLLBACK to \$TEVEL=0. Previously such a TROLLBACK left \$REFERENCE with a value that was a function of the last rolled back action. (GTM-4033)
- Interrupted GT.M HANG commands now account for the time spent handling the interrupt. Previously, the HANG command resumed using the full specified wait time thus potentially hanging indefinitely if hit with a stream of interrupts. (GTM-5996)
- Interrupted timed GT.M LOCK commands now account for the time spent waiting before the interrupt when the LOCK resumes its wait. The time spent handling the interrupt is ignored thus the locks specified in the command may be waited on during several intervals which add up to the wait time specified in the LOCK command. Note that while handling the interrupt, the lock requests (and any partial reservations) are released. Previously, the LOCK command restarted the full wait time thus never completing if additional interrupts arrived. (GTM-5997)
- GT.M now evaluates command postconditionals for invalid commands at runtime before generating a GTM-E-INVCMD error. If GT.M evaluates the postconditional to false, GT.M does not raise an error and continues execution. GT.M generates compile time GTM-E-INVCMD messages as before, though if the command includes a postconditional the compilation continues. Previously, GT.M considered the invalid command a run time error regardless of the command postconditional. (GTM-6185)
- GT.M properly manages stack frames when fielding CTRAP interrupts from the console. Previously, if a \$ETRAP did not handle a CTRAP error, GT.M mishandled the re-throw of the error in frames associated with an indirection, for example DO @X, subsequently causing a GTMASSERT or address exception (UNIX SIG-11 or OpenVMS ACCVIO). (GTM-6228)
- <CTRL-D> at the GT.M Direct Mode prompt on an empty line terminates GT.M and returns control to the shell. FIS thanks Sam Habel for this enhancement. Previously GT.M ignored <CTRL-D> in this circumstance. As in prior versions, if there is a character at the cursor when you enter <CTRL-D> in direct mode, GT.M erases it; if there are characters on the line but the character position is at the end of the line, GT.M ignores <CTRL-D>. Note: to delete the prior character on the line, use the key defined in your TERMINFO as key_backspace. [UNIX] (GTM-6449)
- There are a few very small and infrequent windows in GT.M code (expected to be under a microsecond on contemporary hardware) during which a process should not be terminated (hence the recommendation to use kill -9 on GT.M processes only as a last resort). When a MUPIP STOP is received within such a window, GT.M now defers process termination until it is outside the window. If, for some reason, the process is stuck within such a window of protected code, and receives three MUPIP STOP signals, it stops immediately. Previously, GT.M processes responded to MUPIP STOP signals immediately, and if one received a MUPIP STOP within such a window, it might generate a GTMASSERT or other failure while attempting to terminate. This change should require no operational changes in GT.M because of the tiny windows within which MUPIP STOP is deferred---the three STOP technique being required only to recover from unanticipated out-of-design conditions. [UNIX] (GTM-6569)

- GT.M ignores SIGINT (control-C) if \$PRINCIPAL is not a terminal. Previously receipt of this signal blocked subsequent out-of-band events such as MUPIP INTRPT or TPTIMEOUT. [UNIX] (GTM-6614)
- GT.M better handles some error processing cases. Previously, GT.M used to fail with addressing exceptions (UNIX SIG-11 or OpenVMS ACCVIO) while capturing error context information for \$STACK(). Other changes have addressed the known errors in that category but this improves the robustness of GT.M by giving it a better chance to continue operating even in adverse circumstances. (GTM-6622)
- When returning from code invoked by MUPIP INTRPT (clearing \$ZININTERRPT), GT.M implicitly clears any error(s) detected while 1=\$ZININTERRUPT and sends a JIUNHNDINT error notification to the operator log, for example:

```
Sep 26 11:41:09 flyingv GTM[15119]: %%GTM-E-JIUNHNDINT, An error during $ZINTERRUPT
processing was not handled: 150379874,jobintr2+4^jobintr,
%%GTM-E-ERRWZINTR, Error while processing $ZINTERRUPT
-GTM-E-LABELMISSING, Label referenced but not defined:
nonexistantrtn -- generated from 0x00007FEFD25B831A.
```

Prior versions followed the basic error handling pattern of re-throwing the most recent unhandled error at each level as long as \$ECODE is not equal to the empty string. [UNIX] (GTM-6711)

- GT.M now provides better management of timed events by deferring their processing during certain brief sections of code. Previously, GT.M could not keep timers alive while preventing various timer-driven events from interfering with the executing code; this limitation resulted in a number of very rare failures, caused by either a lost timer or a timed event delivered at an inappropriate moment, resulting in symptoms such as hung processes and GTMASSERTs. In addition, GT.M now raises the COREINPROGRESS error if an attempt to generate a core fails and triggers an attempt to generate another core. Previously, under certain very rare conditions, GT.M could get into a core generation loop potentially eating up disk space and process slots. [UNIX] (GTM-6719)
- The installation kit now includes a random string generator utility function called %RANDSTR (strlen,charranges,patcodes,charset). This generates a random string of length strlen from an alphabet defined by charset or by charranges and patcodes. If charset specifies a string of non-zero length, %RANDSTR generates the random string using the characters in charset, otherwise it takes its alphabet as specified by charranges and patcodes. charranges uses the same syntax used for FOR loop ranges, for example, 48:2:57 to select the even decimal digits or 48:1:57,65:1:70 to select hexadecimal digits. patcodes specifies pattern codes used to restrict the characters to those that match the selected codes. If charset is of zero length, and is passed by reference, %RANDSTR() initializes it to the alphabet of characters defined by charranges and patcodes. If not specified, strlen defaults to 8, charranges defaults to 1:1:127 and patcodes to "AN". (GTM-6738)
- GT.M now defers TPTIMEOUT interrupts as follows:
 - \$ETRAP has been SET since the last SET of \$ZTRAP, and \$ECODE="" and \$TLEVEL=0, GT.M defers TPTIMEOUT until one of those conditions changes; once \$TLEVEL=0 GT.M cancels rather than delivers any deferred TPTIMEOUT.

- If \$ETRAP is in control and \$ECODE="", GT.M defers TPTIMEOUT until one of those conditions changes.
- If \$ZININTERRUPT=1, GT.M defers TPTIMEOUT until that condition changes.

GT.M delivers interrupts including TPTIMEOUTs at the beginning of the next line, the beginning of the next FOR iteration or while waiting in a command with a timeout. Previously GT.M delivered TPTIMEOUT as described in the prior sentence without regard to the error, interrupt or transaction state. In addition, GT.M does not adjust state information related to transaction retry on any error, including on the error used to signal a TPTIMEOUT. This reduces the possibility of indefinite transaction restarts. This also means that the Principal device of a process in a transaction entering direct mode now displays a TPNOTACID error. Note that GT.M does not support transaction activity in Direct Mode. FIS recommends restricting Direct Mode interaction with transactions to investigating state for debugging purposes. Previously, GT.M released the database critical section on an error or interrupt. [UNIX] (GTM-6759)

- ZMESSAGE now treats any non-GT.M error as a trappable or fatal event. In prior releases, a ZMESSAGE argument with low order bits indicating success or information in the GT.M error designation scheme, for example ZMESSAGE 1, went to \$PRINCIPAL without activating an error trap. [UNIX] (GTM-6771)
- Label invocations using DO no longer require parentheses for calls with no actualist. If DO or a \$\$ that does not specify an actualist invokes a label with a formallist, the missing parameters are undefined in the called routine. As a result, a GT.M process no longer issues ACTLSTEXP and FLLSTPRESENT errors, regardless of the arguments passed and parameters expected. WARNING: If DO or \$\$ specifies a routine but no label using an actualist, then whether that routine's top label has a formallist or not, the actualist applies to it directly, whereas before the actualist would "fall through" to the first label with executable code. Previously, excluding parentheses from a DO invocation of an entryref with a formallist resulted in FLLSTPRESENT error for local labels, and ACTLSTEXP error for labels in external routines and indirect expressions. (GTM-6813)
- The gtm_trace_gbl_name environment variable enables GT.M tracing at process startup. Setting gtm_trace_gbl_name to a valid global variable name instructs GT.M to report the data in the specified global when a VIEW command disables the tracing, or implicitly at process termination. This setting behaves as if the process issued a VIEW "TRACE" command at process startup. However, gtm_trace_gbl_name has a capability not available with the VIEW command, such that if the environment variable is defined but evaluates to zero (0) or, only on UNIX, to the empty string, GT.M collects the M-profiling data in memory and discards it when the process terminates (this feature is mainly used for in-house testing). Note that having this feature activated for process that otherwise don't open a database file (such as GDE) can cause them to encounter an error. Previously, the only way to start tracing was with a VIEW command. As with the VIEW command, tracing initiated with gtm_trace_gbl_name can be redirected or turned off by a subsequent VIEW command.

In addition, if a process issues a malformed VIEW command that attempts to turn tracing off, GT.M issues an error but retains all accumulated profiling data and continues tracing. If the tracing is still enabled at the process shutdown and the trace start specified a reporting location, GT.M attempts to place the trace data there. Note that if there is a problem updating the specified trace-reporting

global variable, GT.M issues an error at process termination. Previously, any VIEW command that unsuccessfully attempted to transfer tracing data to a global variable turned tracing off and discarded all collected data. On UNIX, for each line executed, GT.M tracing now reports the elapsed time in addition to CPU times (user, system, and combined).

It also reports two aggregate entries for UNIX and one on OpenVMS, `"*RUN"` for the current process and, on UNIX, `"*CHILDREN"` for all of child processes spawned by the current process, each containing user, system, and combined CPU times. Previously, the trace facility did not report these details of the child processes for the ZSYSTEM command and PIPE devices that are closed before the trace data is reported. The `"CHILD"` category data excludes processes that result from the JOB command, PIPE devices OPENed with the INDEPENDENT device parameter and processes from PIPE devices that are still active. Note that M-profiling is currently disabled for DAL calls on OpenVMS. (GTM-6819)

- GT.M now correctly processes WRITE mnemonics for GT.M socket devices. Previously, a WRITE mnemonic usage such as WRITE /LISTEN or WRITE /WAIT without the depth value (for /LISTEN) or without a timeout (for /WAIT) could cause either a INVCTLMNE error or an addressing exception (signal-11 (UNIX) or ACCVIO (VMS)). (GTM-6836)
- The GT.M parser now appropriately recognizes the end of a source line. Previously it preprocessed the next character, which resulted in an inappropriate SPOREOL error if the two last lines in a routine had certain odd characteristics. The workaround was to add another line to the routine - either empty or comment lines work if a QUIT is not appropriate (GTM-6845)
- \$ZDATE() generates a ZDATEBADDATE for input date values greater than 31-Dec-999999 (364570088) or less than 01-JAN-1840 (-365), a ZDATEBADTIME for time values greater than a second before midnight (86399) or less than 0 (zero). Previously if the input was too large (either positive or negative), the process might produce meaningless output or terminate with a segmentation violation (UNIX SIG-11 or OpenVMS ACCVIO). Because invalid conditions that used to be ignored are now treated as errors, this change might require revisions to existing application code. (GTM-6851)
- GT.M now evaluates functions with side effects - extrinsic functions (\$\$), external functions (\$&) and \$INCREMENT() in left-to-right order when in full_boolean mode. In V5.4-002 it evaluated those functions early to ensure they were not skipped by the default GT.M short-circuit optimization, however this could cause their side-effects to occur out of order. (GTM-6854)
- GT.M correctly compiles expressions like `if a<$$x(b-c)` when specifying FULL_BOOLEAN. Previously this combination caused a GTMASSERT. (GTM-6856)
- MERGE now gives no error when the source variable and the destination variable are identical. Previously it would give a MERGEDESC error message. (GTM-6935)
- \$ZTRNLNM() now recognizes all the documented keywords including all the OpenVMS keywords. FULL and VALUE (the default) return the translation and LENGTH returns the length. Other keywords select default values. Many of the keywords return the empty string if the item to translate has no value. Previously it only recognized "FULL". Note that on UNIX \$ZTRNLNM() does not do iterative translation. [UNIX] (GTM-6957)
- 64-bit GT.M editions - currently those except HP Alpha (OpenVMS and Tru64), HP PA-RISC (HP-UX), and 32-bit x86 (Linux) - now pass 64-bit long and ulong values between C and M though the call-

out and call-in interfaces. GT.M represents values that fit in 18 digits as numeric values, and values that require more than 18 digits as strings. M code may use these strings in a numeric context, but the numeric result has less precision than the original value. Note that this change may expose latent problems if a call-in definition file contains incorrectly typed values. Prior versions truncated long and along values passed between C and M through the these interfaces to 32-bit int values even on 64-bit platforms. (GTM-6994)

- A ZSHOW "D" that includes one or more SOCKET devices manages memory in a stable fashion. In prior versions, each such ZSHOW leaked a small amount of memory. (GTM-6996)
- The INDEPENDENT device parameter now has the correct behavior for all shells. Previously, the subprocess started by the PIPE device would not continue to run after the PIPE device was closed for some shells, such as the Bourne shell. The RENAME device parameter now correctly renames a sequential file if it is followed by another device parameter. Previously, the new file name of the sequential file could contain information from the following parameter. [UNIX] (GTM-7002)
- GT.M now handles nesting of XECUTE strings that contain subscripted FOR control variables. In V5.4-002x such constructs could result in a memory segmentation violation (UNIX SIG-11 or OpenVMS ACCVIO). (GTM-7007)
- GT.M correctly handles an indirect DO without arguments (e.g. DO @x) inside an XECUTE argument. In prior versions, this construct caused a GTMASSERT or an addressing exception. (GTM-7008)
- GT.M now issues a DEVOPENFAIL error for a PIPE device if either the SHELL or STDERR device parameters are set to the null string. Previously, the open error was ignored and any subsequent ZSHOW "D" caused a SIG-11. In addition, the DEVOPENFAIL error message for a PIPE device now outputs the device name. Previously, it output the device type. [UNIX] (GTM-7123)
- GT.M uses an additional form of GTMASSERT error - GTMASSERT2, which adds information about the failing condition. This additional information may only be useful to FIS or others familiar with GT.M internals. For example:

```
%GTM-F-GTMASSERT2, GT.M V5.5-000 Linux x86 - Assert failed /usr/library/V994/src/
mdb_condition_handler.c
line 490 for expression (TPRESTART_STATE_NORMAL == tprestart_state)
%GTM-F-GTMASSERT2, GT.M V5.5-000 Linux x86 - Assert failed /usr/library/V994/src/
gds_rundown.c
line 543 for expression (NULL != csa->wbuf_dqd)
```

GTMASSERT and GTMASSERT2 indicate GT.M has detected a condition that seems to violate a design assumption and should be reported to your GT.M support channel. Prior versions did not provide the additional diagnostic information reported by GTMASSERT2. (GTM-7078)

- ZBREAK and ZPRINT() now issue errors when invoked with an indirect routine name that evaluates to the empty string other than a string literal, for example, the return from a \$GET(). Previously this construction caused a memory segmentation violation (UNIX SIG-11 or OpenVMS ACCVIO). (GTM-7176)

- The MERGE command now counts global gets as well as updates in the database counters displayed with ZSHOW "G" and DSE. Previously MERGE only counted the update portion of its work. (GTM-7179)
- FOR now evaluates a termination value specified with a subscripted local variable at loop initiation. Previously it incorrectly reevaluated the termination value at each iteration. (GTM-7185)
- On platforms where the information is unavailable error messages that report the address of an event now report a hexadecimal -1 (all F's). This information in a message is typically of use only to FIS or others who work with the GT.M implementation code. Previously GT.M reported a zero (all 0's) address in such cases, which called into question whether the error message was properly constructed. [UNIX] (GTM-7188)
- M-profiling no longer leaks memory. Previously, processes using M-profiling experienced a memory leak and could run out of memory when running code with repetitive function invocations. Also, M-profiling no longer receives a SIGSEGV signal during process termination. Previously, a tracing-enabled GT.M process interrupted at an inopportune time could die with a SIGSEGV condition. In addition, M-profiling correctly reports each label count entry. Previously, such it did not report entries for labels invoked with a DO command inside a FOR construct that used a variable rather than a literal for the termination value. (GTM-7196)
- GT.M completely removes all trigger artifacts during an unlink using the form ZGOTO 0:entryref. Previously, GT.M removed the code but left some trigger-related structures behind, which could cause memory segmentation violations (SIG-11) or other bad behavior. [UNIX] (GTM-7206)
- In OpenVMS, GT.M more reliably reports the name of an undefined unsubscripted local variable. Previously, depending on circumstances, the UNDEF error message sometimes failed to report the variable name. [OpenVMS] (GTM-7210)
- The GT.M LOCK manager now guards against unlikely, but possible, mismanagement of LOCK resources. This is not an issue that we are aware of anyone ever encountering, but is analogous to a relatively uncommon issue we saw with local variables on certain x86 Linux releases. Previously GT.M could, at least in theory, fail to properly maintain resource names. (GTM-7214)
- At runtime, gtm_zquit_anyway environment variable causes code of the form QUIT <expr> execute as if it were SET <tmp>=<expr> QUIT:\$QUIT tmp QUIT, where <tmp> is a temporary local variable in the GT.M runtime system that is not visible to application code. This mode no longer relies on the GT.M compiler. Previously, the gtm_zquit_anyway environment was used when compiling code, and caused code of the form QUIT <expr> to be compiled as QUIT:\$QUIT <expr> QUIT. While FIS normally avoids changing GT.M behavior in any way that is not upward compatible, in this instance FIS has worked with users of gtm_zquit_anyway to determine that this changed behavior is more appropriate to existing applications. (GTM-7215)
- The LOCK (and ZALLOCATE and ZDEALLOCATE) commands now manage LOCK resource names more efficiently in LOCK_SPACE. Previously they used more space than they really needed and could lose efficiency or generate an error when it was unnecessary. (GTM-7228)
- A JOB command with no parameters created by a JOB'd process with one or more parameters receives no parameters. Previously the JOB command with no parameters inappropriately received the first parameter of the JOB'd parent process. [UNIX] (GTM-7234)

Utilities-MUPIP

- MUPIP REORG -TRUNCATE[=*percentage*] specifies that REORG, after it has rearranged some or all of a region's contents, should attempt to reduce the size of the database file and return free space to the file system. The optional *percentage* (0-99) provides a minimum amount for the reclamation; in other words, REORG won't bother performing a file truncate unless it can give back at least this *percentage* of the file; the default (0) has it give back anything it can. TRUNCATE always returns space aligned with bit map boundaries, which fall at 512 database block intervals. TRUNCATE analyses the bit maps, and if appropriate, produces before image journal records as needed for recycled (formerly used) blocks; The journal extract of a truncated database file may contain INCTN records having the inctn opcode value 9 indicating that the specific block was marked from recycled to free by truncate. TRUNCATE only works on V5 databases - it does not work on database files that contain unconverted V4 format blocks. TRUNCATE does not complete if there is a concurrent online BACKUP or use of the snapshot mechanism, for example by INTEG. [UNIX] (GTM-3873)
- The NOREPLCTDREG error message now correctly identifies the reported context as the instance file. Previously since the introduction of multi-site replication on UNIX that error mislabeled the file as a global directory. The label has not changed on OpenVMS, where it is correct. [UNIX] (GTM-4202)
- MUPIP REORG now takes an optional -REGION qualifier. MUPIP REORG -REGION <region-list> reorganizes each global in the specified regions. Note that a global must be in the global directory in order for MUPIP REORG to process it. Previously, REORG would reorganize every global in the global directory unless specific globals were selected with -SELECT or EXCLUDE. Note that MUPIP REORG previously supported a -REGION qualifier, but only in conjunction with the -UPGRADE and -DOWNGRADE qualifiers. (GTM-5797) 🟢
- GT.M replication supports Supplementary Instances - instances which both receive updates from some other originating instance and local updates. Please refer to the Supplementary Instance Technical Bulletin for detailed information.

The project to introduce this feature also addressed the following changes most, but not all, of which apply only to UNIX editions:

- The -updateresync qualifier now requires a value, the name of an instance file. If you are following the previous procedure, you need to modify your scripts. As we bring you new Supplementary Instance replication, we regret our inability to maintain strict upward compatibility with prior GT.M releases with respect to this occasionally-used qualifier. [UNIX]
- The journal extract file format has changed- please see Documentation Addendum
- The argumentless MUPIP RUNDOWN command has been significantly reworked to work in various situations that it did not previously. [UNIX]
 - It now correctly cleans up orphaned journal pool shared memory and reports its actions. Previously it used to report a successful rundown when it actually did not do the rundown.

- It now gives more details in case it is not able to rundown: for example:
 - If rundown encounters a journal pool with a different format, it now indicates the format and the GT.M version that created the journal pool in addition to indicating the format that the current MUPIP can handle. Previous versions did not indicate any of this.
 - If rundown encounters a database shared memory created by a different version of GT.M, it now indicates the database shared memory ID. Previous versions did not indicate this.
- It now preserves shared memory IDs whose corresponding database or replication instance file cannot be opened (e.g. due to permissions). In GT.M V54002B, it silently deleted such shared memory segments.
- It now cleans up as many GT.M created orphaned ipc's (shared memory segments and semaphores) as possible. Previously, if it encountered an error while cleaning up one ipc it terminated with an error instead of proceeding to the next available ipc.
- The SHMREMOVED error message that RUNDOWN prints now indicates the database file or instance file name corresponding to the shared memory ID which it removed. Previously it did not provide this information.
- In a replicated environment, GT.M now maintains a non-decreasing timestamp sequence in the journal file in all cases. Previously in some rare cases of concurrent processes doing multi-region TP transactions, it was possible to write an out-of-order timestamp sequence, which in turn could cause a MUPIP JOURNAL ROLLBACK to treat transactions that could be replayed as lost.
- When a Source Server and Receiver Server connect with each other, they now correctly ensure their two instances are in sync before proceeding with replication. Previously the instance file histories did not contain sufficient detail to detect a number of mismatched conditions. [UNIX]
- MUPIP SET JOURNAL now issues a FILEEXISTS error in various situations when it is unsafe to create a new journal file by renaming the existing file because it is not a valid current generation journal file pointing to the specified database file. Examples of such unsafe situations are, (i) if one unintentionally specifies the new journal file name to be an existing actively used database file name, (ii) a backup database points to the same journal files as the source database and a journal switch is attempted on the backup database. Previous versions used to incorrectly allow such switches with potentially serious operational consequences. [UNIX]
- The PREVJNLLINKCUT message issued by a MUPIP SET JOURNAL command, now contains the correct new journal file name. Previously it displayed the journal file name stored in the database file header before the journal switch which might not always be the same as the new journal file name.
- MUPIP REPLIC -EDITINST command, which allows edits to a replication instance file, now supports the -NAME=<new-name> qualifier where the new-name value is required. This changes the instance name in the replication instance file header to the newly specified name. It is now possible for new instances to be created without requiring a new instance file, loss of instance history, and use of the -updateresync qualifier when starting the Receiver Server for the first time. The previous procedure

to create a new instance was to ship a backup of another instance's database and recreate the instance file on the target instance. Although you can still use the older procedure with a modification to add an instance file name argument to the `-updateresync` qualifier, FIS recommends the following procedure as more robust way to create a new instance:

- Use the MUPIP BACKUP command with the `-REPLINSTANCE` qualifier to backup the instance to be copied.
- Ship the backed up databases and instance file to the target system.
- Run the MUPIP REPLIC -EDITINST command on the backed up instance file to change the instance name to reflect the target instance. This makes the source replication instance file usable on the target instance while preserving the history records in the instance file.
- Create new journal files, start a passive Source Server and a Receiver Server (without an `-updateresync` qualifier).
- Allow a Source Server to connect.
- The MUPIP replication commands recognize the `-UPDOK` qualifier as a preferred synonym for the `-ROOTPRIMARY` qualifier; they also recognize the `-UPDNOTOK` qualifier as the negation of `-UPDOK` and the preferred synonym for `-PROPAGETPRIMARY`.
- The Receiver Server log now indicates the current journal sequence number of the replication instance at shutdown. Previously, it only printed the receive pool journal sequence number. [UNIX]
- The Update Process logs now indicate the current journal sequence of the replication instance at the time the Update Process is shutdown. Previously, it only logged how many sequence numbers it had processed.
- The Source Server and Receiver Server logs now indicate whether the other side is the same or opposite endianness relative to their side of the replication stream.
- Because it controlled a field in the file header that was relevant only for releases prior to V5.1-000, starting with V5.5-000, DSE ignores `CHANGE -FILE -DUALSITE_RESYNC_SEQNO`; V5.5-000 and later versions do not support replication with versions prior to V5.1-000.
- GT.M utilities like MUPIP, DSE, LKE etc. now produce time-ordered, and hence more easily understood, output in case the output and error streams (stdout and stderr) are redirected to the same logfile. Previously, in some cases, the logfile might have messages from the two streams mixed in the same file without maintaining the relative time order of when the process issued those messages. This meant a later message came ahead of an earlier message and resulted in confusing log files but only if both stdout and stderr were redirected to the same logfile. Since the replication server logfiles (Source Server, Receiver Server, Update Process) have this redirection inherently built in, they were also affected by this issue.
- Copying and pasting a sequence of shell commands, that include MUPIP commands, into a terminal session now works appropriately. In previous versions, a MUPIP command stopped execution of such a sequence of commands. [UNIX] (GTM-6340)

- The Source Server exits with error JNLNOREPL if it opens a journal file and finds that replication was not on for that journal. Also it retries any journal file read in the current journal file that returns an incomplete record; in UNIX, if it reads an incomplete record from a previous generation journal file, it exits with a JNLRECINCMPL error. Previously, these two conditions caused a GTMASSERT. Also, in UNIX, a Source Server does not switch to a new journal if it fails to open the currently active journal file. Additionally, MUPIP SET JOURNAL reports the PREVJNLLINKSET message in the operator log in addition to the operator terminal. Previously, those were only reported to the terminal. (GTM-6736)
- A Receiver Server started with the -UPDATERESYNC qualifier starts every Source Server connection handshake during its own lifetime with that characteristic. Previously, a Receiver Server based only the initial handshake for its first connection on the -UPDATERESYNC qualifier, which meant any errors that closed the connection caused it to reconnect in a way that likely caused a REPLINSTNOHIST error. (GTM-6650)
- A MUPIP action (RECOVER or ROLLBACK) that restores V4 format blocks to a database that had completed a transition from V4 to V5 format (using MUPIP REORG -UPGRADE) sets the indicator that the database still contains blocks in V4 format. Previously, such a sequence caused the database logic to incorrectly stop checking for V4 format blocks, which could, in rare cases, cause database corruption. The workaround was to take a BACKUP with fresh journal files immediately after completing the MUPIP REORG -UPGRADE. (GTM-6725)
- A replication filter no longer needs to package a simple update as a transaction. This change enables the use of filters such as the UNIX tee utility. GT.M also no longer requires the external filter pipe buffer be large enough to accommodate the entire largest transaction. If an external filter returns a bad record, GT.M issues a FILTERBADCONV error. Previously, GT.M required the external filter to return all transactions wrapped as TP transactions, even if they were originally non-TP single updates (mini-transactions), would hang if a transaction exceeded the buffer size for the pipe, and would loop indefinitely if the filter returned a defective record. Note that GT.M still requires that a filter maintain a one-to-one relationship between input and output transactions, as documented. [UNIX] (GTM-6826) 🟢
- MUPIP JOURNAL -ROLLBACK -FETCHRESYNC (or -RESYNC) now correctly recovers the database and creates a correct unreplicated (lost) transaction file in cases where a second command specifies a later point in time than the prior interrupted command. In prior versions, such a change sequence might create an incorrect recovery. In addition, MUPIP RECOVER -BACKWARD restricts processing that identifies broken transactions to the most recent journal file(s). Previously, backward recovery extended transaction validity checking all the way to the turnaround point at which it starts forward processing, which unnecessarily required more memory and potentially failed if that memory was not available. Also, now the -RESYNC ROLLBACK is more memory efficient and uses approximately same amount of memory as -FETCHRESYNC ROLLBACK uses. Previously -RESYNC required more memory than -FETCHRESYNC to accomplish the same thing. Note that -RESYNC is intended for use when advised by your GT.M support channel and is not recommended for routine normal use. (GTM-6834)
- The Source Server no longer hangs while waiting for a the message used in the initial handshake between a Source and Receiver Server to determine if compression can be turned on to send journal records across. In previous versions of GT.M, if a Receiver Server sent a flow control messages just before sending the REPL_CMP_SOLV message (which appears in the Receiver Server log), the Source Server hung. [UNIX] (GTM-6841)

- Cross-endian replication now works correctly in various scenarios. Previously, any abnormal events on the receiver side (e.g. draining the replication pipe in case of Update Process crash or a bad transaction etc.) could result in a hang on the Receiver Server. Note that cross-endian replication is handled robustly by GT.M V5.5-000 only if both the source and receiver sides run that version. If not, only a few combinations are supported. Below is the list.
 1. If GT.M V5.5-000 is on one side of a replication connection, the minimum version supported on the other side is GT.M V5.1-000 when both sides have the same endianness
 2. In case of a cross-endian connection involving GT.M V5.5-000 on the source side, the minimum version supported on the receiver side is GT.M V5.4-002. Even though versions prior to V5.4-002 work in most usual situations, they do not robustly handle abnormal events and might hang
 3. In case of a cross-endian connection involving GT.M V5.5-000 on the receiver side, the minimum version supported on the source side is V5.3-003 (the earliest version for which cross-endian replication is supported). [UNIX](GTM-6896)
- KILLS of non-existent global nodes (henceforth referred to as duplicate KILLS), which are possible for example if the global mapping is changed in the gld file on the replicating instance, are now handled robustly on the Update Process side by maintaining the journal sequence numbers of the originating instance and its replicating instances in sync. Previously, in this scenario, the Update Process printed a "Number of transactions from Primary which did NOT update JNLSEQNO on Secondary" message in its log indicating the sequence number at which the duplicate KILL occurred but continued processing even though the two instance sequence numbers were no longer in sync. This gave a false impression the two instances were in sync when they actually were not. Additionally, if an update that is played on the Update Process side does not increment the journal sequence number of the replicating instance, the Update Process prints the "Number of transactions from Primary which did NOT update JNLSEQNO on Secondary" message in its logfile and terminates. Previously, it used to continue processing thereby making it very difficult to get the two instances back in sync. (GTM-6898)
- MUPIP INTEG and MUPIP JOURNAL -ROLLBACK or -RECOVER handle database files larger than one terabyte. Previously GT.M gave a DBFILOPERR error with a secondary error of invalid argument (ENO22 on UNIX) when they attempted to access data higher than 1TB (GTM-6958).
- GT.M ensures not even a single update can occur during a MUPIP FREEZE. This issue was primarily addressed by C9J01-003076 in V5.3-004, however, we discovered one subtle case where under high update contention, a single update from MUPIP REORG might squeeze in right after a FREEZE. (GTM-7004)
- MUPIP LOAD can now read from the standard input by specifying the -STDIN qualifier instead of a file-name. [UNIX] (GTM-7016)
- MUPIP JOURNAL RECOVER or ROLLBACK -BACKWARD now processes multi-region TP transactions appropriately even if they are broken. Also, the Source Server avoids writes to the database and/or journal as much as possible. In versions starting with V5.4-000, RECOVER or ROLLBACK terminated with a GTMASSERT error if they encountered a complete transaction after a broken transaction. Previously the Source Server could write an INCTN journal record in cases where the database cache integrity was suspect. Both of these issues arose after processes were terminated abnormally, typically

with a kill -9 - FIS strongly recommends against using kill -9 on GT.M processes, as it can cause database damage when used to terminate a process that is actively updating a database. (GTM-7034)

- MUPIP EXTRACT -FORMAT=BIN now handles database blocks of maximum size. Previously it failed with a "%SYSTEM-E-ENO14, Bad address" error in UNIX or a "%RMS-F-RSZ, invalid record size" error in OpenVMS" (GTM-7077)
- MUPIP JOURNAL -SHOW=ALL output now shows counts of all Record Types correctly up to 1 billion. Previously, a Record Type count more than 1 million showed as '*****'. (GTM-7079)
- With a "-stdout" (case ignored) as the argument of its -EXTRACT qualifier, the MUPIP JOURNAL command now writes its output to standard output instead of creating a file with the name "-stdout". Although this alters existing GT.M behavior, FIS believes that using "-stdout" to direct output to stdout is more useful than creating a file with a name that makes it awkward to read and process. In addition, MUPIP JOURNAL -EXTRACT now ends with the last extracted journal record. Previously it placed a blank line at the end of the file. [UNIX] (GTM-7082)
- The Update Process and journal recover/rollback now correctly handle a NULL type journal record. Previously, they skipped a check on the maximum available space in the current journal file (based on the journal file autoswitchlimit) while writing the NULL journal record. This resulted in the journal file space getting filled up so eventually there is no room left in it for writing even one journal record. At that point, a journal file switch is unavoidable because there is no space but is also impossible because the switching routine needs to write one journal record to the current journal file. This caused them to go into an indefinite recursion eventually running out of C-stack space and terminating abnormally as well as leaving the journal file in an out-of-design state. Any other process that next does a global reference in this database ended up with the same situation leading to a cascade of process terminations and an unusable journal file. (GTM-7230)

Utilities-Other Than MUPIP

- DSE FIND -SIBLING now shows the current block in addition to any left and right siblings - all designated in hexadecimal. Previously the command only reported the two siblings when they were in the same branch of the tree, and while the representation was also in hexadecimal notation, it could be ambiguous. In addition, DSE FIND now correctly represents blocks in the global directory tree when they are selected. Previously FIND did not show block one and could represent other directory tree blocks as if they had been in a global tree. (GTM-2296)
- GDE and MUPIP now allow the user to set the journal extension size to a maximum of 1073741823 blocks on Unix; the VMS maximum remains at 65535 blocks. This feature required a change to the Global Directory (GLD) file format; the file must be upgraded using GDE. (Refer to the section on Upgrading Your Global Directory.) Note that older versions of GT.M cannot read the upgraded GLD files. Previously, GDE and MUPIP limited the journal extension size to 65535 blocks on all platforms.

GDE now supports AUTOSWITCHLIMIT as an option for the -JOURNAL region qualifier. GDE stores the value in the global directory, and MUPIP now uses the value from the global directory when initializing a region in place of the old default. However, the default AUTOSWITCHLIMIT value for GDE is the same as the old default for MUPIP CREATE, so the default result is unchanged. [UNIX]

When you specify values for ALLOCATION, EXTENSION, and AUTOSWITCHLIMIT for a region such that (ALLOCATION+EXTENSION>AUTOSWITCHLIMIT), either using GDE or MUPIP SET -JOURNAL, the utility sets ALLOCATION to match the AUTOSWITCHLIMIT, and outputs a GDE-I-JNLALLOCGROW or GTM-I-JNLALLOCGROW message, respectively. [UNIX]

At journal extension time, including journal autoswitch time, if (ALLOCATION +EXTENSION>AUTOSWITCHLIMIT) for a region, GT.M now uses the larger of EXTENSION and AUTOSWITCHLIMIT as the increment to warn of low available journal disk space. Otherwise, it uses the EXTENSION, as before. [UNIX]

The GDE CHANGE -REGION -JOURNAL command no longer forces the EXTENSION to be the same as the specified ALLOCATION if the EXTENSION is not also specified. The previous behavior was documented for OpenVMS, but not for UNIX. (GTM-6224)

- Corrected typos in the DBCDBNOCERTIFY and JOBEXAMFAIL error messages, a message from MUPIP RESTORE and, on OpenVMS, to the gtmkitinstal.com. (GTM-6849)
- On platforms where GT.M supports object code in shared libraries, the configure script which installs GT.M now automatically creates a shared library libgtmutil.so containing the object modules of the routines distributed with GT.M that are written in M (GDE and utility programs). The "configure" installation script asks a question as to whether to delete the .o files as part of the installation. They can safely be deleted as long as the \$gtmroutines environment variable (\$ZROUTINES intrinsic special variable) specifies the libgtmutil.so file rather than the installation directory (\$gtm_dist). The "gtminstall" installation script includes a --keep-obj command line switch and associated gtm_keep_obj environment variable (supporting "Y" and "N" values) that instructs the configure script not to delete

object files. Note that some platforms use a .sl or .dll extension for shared libraries, rather than .so. The gtmprofile script that you can source to set up a default GT.M environment automatically uses libgtmutil.so if it exists. In addition, if there are any shared libraries in the \$gtmdir/\$gtmver/o directory (\$gtmdir/\$gtmver/o/utf8 for processes in UTF-8 mode), the gtmprofile script automatically includes them in the gtmroutines environment variable. Previously gtminstall, gtmprofile and configure scripts did not deal with shared object libraries. Furthermore, the gtmprofile script now provides a framework to extend GT.M with plug-ins. Place the M code in \$gtm_dist/plugin/r, the M mode object code in the \$gtm_dist/plugin/o directory and UTF-8 mode object modules in the \$gtm_dist/plugin/o/utf8 directory. On platforms where GT.M supports object code in shared libraries, you can create shared libraries instead of .o files. The gtmprofile script now automatically includes these in the gtmroutines environment variable. [UNIX] (GTM-6664) 🟢

- DSE DUMP -FILE -ALL output prints journal record types in hexadecimal. Since V4.4-002 these numbers have been printed in decimal which, if the numbers exceeded seven characters, overflowed the format and filled the output field with asterisks (*). (GTM-6860)
- %RO now leaves trailing spaces intact and strips comments correctly. Previously it would strip trailing spaces from the end of each line in selected files, and it would mistake semicolons within quotes for comments. (GTM-6939)
- GDE now generates a FILENOTFND message when it fails to open a command file, and a CRYPTNOMM message when it attempts to save a global directory with both encryption and MM access method set. Previously it would generate the undocumented FNF and ENCNOMM messages, respectively. (GTM-7005)
- The size of the installation kit has been reduced by removing the file GTMDefinedTypesInit.o and creating a new GTMDefinedTypesInit for the distribution. This new kit contains the file GTMDefinedTypesInit.m from which GTMDefinedTypesInit.o can be created as needed. Previously, these were included in the installation kit. As the GTMDefinedTypesInit module is not needed for normal GT.M operation and is intended for use in diagnostic purposes under the guidance of FIS GT.M support, you do not need to download and install it except when so directed by FIS GT.M support. In addition, certain shell scripts have been removed from \$gtm_dist. These scripts are not documented in GT.M user documentation but were inadvertently included in GT.M distributions. Also, when lowercase utility programs are requested during a UTF-8 mode GT.M installation, the configure script now supplies the programs in \$gtm_dist/utf8 using symbolic links to the files in \$gtm_dist, as it does for the upper case names. Previously, the lowercase .m files were copied from \$gtm_dist. [UNIX] (GTM-7021)
- LKE now issues NOLOCKMATCH when a non-existing lock is searched with "show l[ock]=<resource>" or attempted to be released using "clear l[ock]=<resource>". Previously, it issued the following text: "No locks were found in <region>" (GTM-7027)
- GCC 4.6, which is part of the toolchain on Ubuntu 11.10, passes additional flags to the linker which prevent linking of libraries without direct dependencies. Previously, if a library was specified on the command line without direct dependencies, but that library's dependencies resolved a program dependency, the library would still be included. GT.M links libgtmshr against librt to get semaphore support, but semaphores are actually provided by libpthread, which is a dependency of librt. This caused builds to fail on Ubuntu 11.10 with a message that "sem_post" is not found. This change is of interest only for users who build GT.M from source. [UNIX] (GTM-7029)

- LOCK commands which find no available lock space send a LOCKSPACEFULL message to the operator log. To prevent flooding the operator log, GT.M suppresses further such messages until the lock space usage drops below 75% full. In addition, LKE SHOW displays lock space usage with a message in the form of: "%GTM-I-LOCKSPACEUSE, Estimated free lock space: xxx% of pppp pages." If the lock space is full, it also displays a LOCKSPACEFULL error. Previously, GT.M did not issue such messages. (GTM-7073)
- GT.M utilities now give a CLIERR error when an equal-sign (=) follows a qualifier without an immediately following value. Previously GT.M handled this case either with a different error or by ignoring the command without reporting any error. [UNIX] (GTM-7110)
- All GT.M distribution kits now include a README.txt file which clarifies license related text in the header of each source file and its relationship to the COPYING file. Previously, this information was not provided. Output from the configure script has been changed to omit the word "National" and to refer appropriately to UTF-8. [UNIX] (GTM-7224)

Error Messages

We are introducing the term "Group" to mean a set of one or more instances that replicate from or to one another and all represent the same logical state of an application. Operators can reconfigure the Group members to change which is the Originating Instance, how many Replicating Instances participate and how (the paths by which) the Replicating Instances receive their updates. Members cannot move from one Group to another although the resources they use can be moved by replacing all their state, using a backup that moves both database files and an instance file to create a new instance with a current or past state of the new Group. GT.M implicitly forms a Group by propagating the identification of the Originating Instance to each Replicating Instance. Supplementary Instances form groups that are different than non-Supplementary Instances; the originating member of a Supplementary group, started with -UPDOK, can replicate from other Groups and also perform local updates.

The new and revised error message(s) for V5.5-000 are as follows:

ACTLSTEXP

ACTLSTEXP, Actvallist expected

Obsolete Error: No longer issued by GT.M

BADREGION

BADREGION, Region is not BG, MM, or CM

LKE Error: The current global directory attempted to map a region with an access method other than those listed.

Action: Use LKE only with database mapped to one of the listed database access methods.

COREINPROGRESS

COREINPROGRESS, Previous core attempt failed; core generation bypassed.

Run Time Error: This indicates that the process, which failed, was unable to create a memory dump file and tried to create another one.

Action: Report the entire incident context to your GT.M support channel for further analysis.

DBCDBNOCERTIFY

DBCDBNOCERTIFY, Database xxxx HAS NOT been certified due to the preceding errors - rerun DBCERTIFY SCAN

Mupip Error: MUPIP UPGRADE for V5 triggers this error if it finds the DBCERTIFY CERTIFY command has not run to completion on database xxx.

Action: Complete the scan phase of DBCERTIFY by executing the DBCERTIFY SCAN command.

DBFLCORRP

DBFLCORRP, xxxx Header indicates database file is corrupt

Run Time/MUPIP Error: This indicates that a database operation tried to activate database file xxxx, which was previously marked as damaged.

Action: If ROLLBACK (either -NOONLINE or -ONLINE) terminates abnormally (say because of a kill -9), it leaves the database in a potentially inconsistent state indicated by the FILE corrupt field in the database file header. When an ROLLBACK terminates leaving this field set, all other processes receive DBFLCORRP errors any time they attempt to interact with the database. The best way to clear DBFLCORRP is by running another ROLLBACK. MUPIP SET -FILE -PARTIAL_RECOV_BYPASS and DSE CHANGE -FILE -CORRUPT=FALSE -NOCRIT can also clear this condition but these commands do not ensure that the database has consistent state so you should always run MUPIP INTEG after executing these commands.

DBROLLEDBACK

DBROLLEDBACK. Concurrent ONLINE ROLLBACK detected on one or more regions. The current operation is no longer valid>/error

Run Time Error: This indicates a non-TP mini-transaction attempted to interact with the database and found that a concurrent online rollback had taken the database to a state earlier than the one at the end of the last mini-transaction by this process unless there has been an intervening TP transaction.

Action: Application dependent - this error indicates a discontinuity in the database state that may cause inconsistent application data.

DSEWCREINIT

DSEWCREINIT, Database cache reinitialized by DSE for region rrrr

DSE Information: This indicates a DSE operator action to rebuild the database cache for region rrrr.

Action: None required.

FMLLSTPRESENT

FMLLSTPRESENT, The actual list is absent from a call to a label with a formal list: xxx

Obsolete Error: No longer issued by GT.M

GTMASSERT2

GTMASSERT2, GT.M eeee - Assert failed LLLL for expression (eeee)

Compile/Run Time Fatal Error: This indicates a design assumption failed at the location LLLL because the expression eeee was FALSE.

Action: Preserve the core (dump) files and report the entire incident context to your GT.M support channel for further analysis. If appropriate, verify database integrity by using the -FAST qualifier.

IGNBMPMRKFREE

IGNBMPMRKFREE, Ignoring bitmap free-up operation for region rrrr (dddd) due to concurrent ONLINE ROLLBACK

Run Time Information: A multi-node KILL bit map cleanup operation detected a concurrent online rollback in region rrrr mapped to database file dddd, and abandoned the cleanup, possibly leaving incorrectly marked busy errors.

Action: If there are incorrectly marked busy errors, match them with this cause and clean them up using DSE.

INSNOTJOINED

INSNOTJOINED, Replicating Instance RRRR is not a member of the same Group as Instance IIII

MUPIP Error: Issued by a Receiver Server or a MUPIP JOURNAL -ROLLBACK -FETCHRESYNC on instance RRRR attempted to connect with Instance IIII, and found they are members of different Groups. Only Supplementary Instances started with -UPDOK can accept updates from a different Group.

Action: Analyze the move and if appropriate, reinitialize the instance that is moving from one Group to another.

INSROLECHANGE

INSROLECHANGE, Supplementary Instance SSSS and non-Supplementary Instance IIII belong to the same Group

MUPIP Error: Issued by a Receiver Server or a MUPIP JOURNAL -ROLLBACK -FETCHRESYNC on Supplementary Instance SSSS attempted to connect to non-Supplementary Instance IIII, but found they have the same Group identification. Because supplementary and non-Supplementary Instances cannot belong to the same Group, one of these instances must have changed roles without appropriate re-initialization.

Action: Either reinitialize the instance that is changing roles or revert the inappropriate role change.

INSUNKNOWN

INSUNKNOWN, Supplementary Instance SSSS has no instance definition for non-Supplementary Instance III

MUPIP Error: Issued by a Receiver Server or a MUPIP JOURNAL -ROLLBACK -FETCHRESYNC on Supplementary Instance SSSS, started with -UPDOK, attempted to connect to non-Supplementary Instance III, but found it has no matching instance information.

Action: Take a backup of the database and replication instance file from a current instance on the non-Supplementary Group, load the backup data on the Supplementary Instance and start the Receiver Server on the supplementary instance using -UPDATERESYNC=<instbak.repl> where instbak.repl is the backup of the replication instance file taken along with the database backup.

INVTRCGRP

INVTRCGRP, Invalid trace group specified in \$gtm_trace_groups: gggg

Run Time Error: The process startup environment attempted to activate a diagnostic tracing facility but specified a group name of gggg and there is currently no such group.

Action: Check with your GT.M support channel for the currently available group names.

JIUNHNDINT

JIUNHNDINT, An error during \$ZINTERRUPT processing was not handled: eeee;

Run-time Error: When returning from code invoked by MUPIP INTRPT (clearing \$ZININTERRPT), GT.M implicitly clears any error(s) detected while 1=\$ZININTERRUPT, sends this error notification to the operator log and continues processing; eeee is the mnemonic for the unhandled error.

Action: Fix \$ZINTERRUPT handler to either not generate the error or to correctly handle it before returning to interrupted code

JOBEXAMFAIL

JOBEXAMFAIL, GT.M process aaaa executing \$ZJOBEXAM function failed with the preceding error message

Run Time Error: This is a secondary message that accompanies a \$ZJOBEXAM function error. This error message is sent to the operator log.

Action: Review the accompanying message(s) and take appropriate action.

JNLALLOCGROW

JNLALLOCGROW, Increased Journal ALLOCATION from [ssss blocks] to [aaaa blocks] to match AUTOSWITCHLIMIT for ffff nnnn

GDE or MUPIP Information: The utility increased the journal allocation value from ssss to aaaa for the journal files associated with ffff nnnn, which is either "database file" followed by a database file name or "region" followed by a region name. This indicates that the specified journal allocation and journal extension values combined exceed the specified journal autoswitchlimit and the utility has adjusted the journal allocation value accordingly.

Action: None required.

JNLNOREPL

JNLNOREPL, Replication not enabled for journal file jjjj (database file dddd)

MUPIP Error: Replication Source Server startup encountered a database dddd with journal file jjjj for which replication was turned off because of a journaling issue and has not since been re-enabled.

Action: Use MUPIP SET to re-enable replication. Take steps to ensure that there is sufficient management of journal file space to prevent a reoccurrence of this issue.

JRTNULLFAIL

JRTNULLFAIL, Applying NULL journal record failed. Failure code: xxxx.

MUPIP Error: Issued by an Update Process, MUPIP JOURNAL -ROLLBACK or MUPIP JOURNAL -RECOVER indicating it encountered a database problem when it attempted to play a NULL journal record into the database. xxxx contains the failure codes for the four attempts. It is very likely that the database may have integrity errors or that the process-private data structures are corrupted.

Action: Report the entire incident context to your GT.M support channel for further analysis.

JNLRECINCMPL

JNLRECINCMPL, Incomplete journal record at disk address aaaa for file jjjj while attempting to read seqno ssss

MUPIP Error: The replication Source Server had a problem with journal file jjjj at disk offset aaaa attempting to read the record with sequence number ssss.

Action: Report the entire incident context to your GT.M support channel for further analysis. Use MUPIP SET JOURNAL -EXTRACT to investigate the issue.

LOCKSPACEFULL

LOCKSPACEFULL, No more room for LOCK slots on database file ffff

Runtime Error: This indicates that the environment attempted more concurrent M LOCKs than the configured LOCK_SPACE for file ffff can support.

Action: Analyze the LOCK protocol for efficiency. Use mupip set -file -lock_space=size ffff to increase the lock space for region xxx. To avoid the same problem the next time you recreate the database, use GDE to make the analogous change to lock_space for the segment mapped to the ffff file in the global directory used to MUPIP CREATE this region

LOCKSPACEINFO

LOCKSPACEINFO, Region: rrrr: processes on queue: pppp/qqqq; LOCK slots in use: llll/kkkk; name space not full

Runtime Error: This indicates that the environment attempted more concurrent M LOCKs than the configured LOCK_SPACE for region rrrr can support. pppp processes are waiting on a lock. llll locks are in use. qqqq and kkkk indicate maximum number of process queue entries, and maximum number of locks respectively.

Action: Analyze the LOCK protocol for efficiency. Use mupip set -region -lock_space=size "rrrr" to increase the lock space for region rrrr. To avoid the same problem the next time you recreate the database, use GDE to make the analogous change to lock_space for the segment mapped to the ffff file in the global directory used to MUPIP CREATE this region.

LOCKSPACEUSE

LOCKSPACEUSE, Estimated free lock space: xxx% of pppp pages.

LKE Information: SHOW command displays the amount of free space along with the number of pages configured for lock space.

Action: If the free lock space report does not show a comfortable amount of free space, use MUPIP SET -LOCK_SPACE to increase the space; remember to also use GDE to revise the LOCK_SPACE in the global directory used to create the region in question so the change remains when the database is recreated.

MAXSEMGETRETRY

MAXSEMGETRETRY, Failed to get ftok semaphore after tttt tries because it is being continually deleted

Run Time Error: A process was unable to open a database file because on every one of tttt tries it found something kept deleting the IPC semaphore that gates access to the file.

Action: Check for one or more rogue processes disrupting IPC semaphore, or for damage to the Operating System semaphore services.

MUINFOINT6

MUINFOINT6 <tttt : vvvv [0x!hhhh] ; \$H=dddddd,ttttt

MUPIP Information: This is secondary information message that provides additional context for some other MUPIP message; tttt is explanatory text, vvvv is a numeric value, hhhh is the hexadecimal equivalent of vvvv, ddddddd and ttttt are a date and time in \$HOROLOG format.

Action: Refer to the preceding message.

MUTRUNC1ATIME

MUTRUNC1ATIME, Process with PID iiii already performing truncate in region rrrr

MUPIP Information: Issued when a REORG -TRUNCATE on a region rrrr detects some other active REORG process concurrently processing a truncation.

Action: No action required. The other process will complete the truncate.

MUTRUNCBACKINPROG

MUTRUNCBACKINPROG, Truncate detected concurrent backup in progress for region rrrr

MUPIP Information: REORG truncate process detected concurrent backup. Database file not truncated.

Action: Ensure the backup has completed and rerun MUPIP REORG -TRUNCATE command.

MUTRUNCERROR

MUTRUNCERROR, Truncate of region rrrr encountered service error eeee

MUPIP Error: This indicates that a system call failed during REORG truncate.

Action: Use the OS documentation to investigate the failure.

MUTRUNCFAIL

MUTRUNCFAIL, Truncate failed after reorg

MUPIP Error: This indicates that REORG encountered an unexpected error. Truncate may be partially complete.

Action: Review accompanying message(s) for more information.

MUTRUNCNOSPACE

MUTRUNCNOSPACE, Region rrrr has insufficient space to meet truncate target percentage of yyyy

MUPIP Information: Issued when REORG truncate determines that there is not enough free space at the end of the file; database file not truncated.

Action: If appropriate specify a larger threshold.

MUTRUNCNOTBG

MUTRUNCNOTBG, Region rrrr does not have access method BG

MUPIP Error: The truncate feature is only supported with the BG access method.

Action: Use the BG access method for files you wish to truncate.

MUTRUNCNOV4

MUTRUNCNOV4, Region rrrr is not fully upgraded from V4 format.

MUPIP Error: The truncate feature is only available for fully upgraded database files.

Action: In order to use truncate, first upgrade the database file to the current major version.

MUTRUNCPERCENT

MUTRUNCPERCENT, Truncate threshold percentage should be from 0 to 99

MUPIP Error: This indicates the the value entered for MUPIP REORG -TRUNCATE is invalid.

Action: Specify a valid threshold percentage.

MUTRUNCSSINPROG

MUTRUNCSSINPROG, Truncate detected concurrent snapshot in progress for region rrrr

MUPIP Information: REORG truncate process detected concurrent snapshot; database file not truncated.

Action: Ensure snapshot, for example a MUPIP INTEG, has completed and rerun the MUPIP REORG -TRUNCATE command.

MUTRUNCSUCCESS

MUTRUNCSUCCESS, Database file dddd truncated from oooo blocks to nnnn at transaction tttt

MUPIP Information: This operator log message indicates that the specified database file was truncated by MUPIP REORG as described by the message.

NOLOCKMATCH

NOLOCKMATCH, No matching locks were found in rrrr

LKE Information: SHOW or CLEAR, found that no LOCKs match the specified criteria in region rrrr; note that specifying no search criteria acts like a wildcard, checking all LOCKs in the region.

Action: If this is not the expected result, check the search criteria and / or research the LOCK protocol to validate its correct operation.

NONASCII

NONASCII, ssss is illegal for a oooo as it contains non-ASCII characters

GDE Error: The specification ssss contains non-ASCII characters which are required for an object of type oooo.

Action: Chose an object name or value containing only ASCII characters.

NOREPLCTDREG

NOREPLCTDREG, Replication subsystem found no region replicated for dddd ffff

MUPIP Warning: This indicates that the replication system is present, but no globals are configured for replication in ffff where dddd is "instance file" for UNIX and "global directory" for OpenVMS.

Action: Use MUPIP SET to specify which database regions to replicate.

NORESYNCSUPPLONLY

NORESYNCSUPPLONLY, NORESYNC only supported for Supplementary Instances

MUPIP Error: Issued by a Receiver Server on a non-Supplementary Instance when it is started with a -NORESYNC; -NORESYNC only applies to Supplementary Instances started with -UPDOK, not to non-Supplementary Instances.

Action: Use this qualifier only in a valid context.

NORESINCUPDATERONLY

NORESINCUPDATERONLY, NORESYNC qualifier only allowed on a Supplementary Instance which allows local updates

MUPIP Error: Issued by a Receiver Server when started with -NORESYNC on a Supplementary Instance but started without -UPDOK; -NORESYNC applies only to Supplementary Instances started with -UPDOK.

Action: Use this qualifier only in a valid context.

NOSUPPLSUPPL

NOSUPPLSUPPL, Instance ssss is configured to perform local updates, so it cannot receive from Supplementary Instance iiiii

MUPIP Error: Issued by a Receiver Server or a MUPIP JOURNAL -ROLLBACK -FETCHRESYNC on a Supplementary Instance ssss started with -UPDOK attempted to connect to instance iiiii, but found IIII is also a Supplementary Instance. A Supplementary Instance that permits local updates can only replicate updates that originate on a non-Supplementary Instance.

Action: Reconfigure the instances to a supported configuration.

ORLBKCMPLT

ORLBKCMPLT, ONLINE ROLLBACK completed successfully on instance iiiii corresponding to dddd

MUPIP Information: Issued by MUPIP ROLLBACK -ONLINE when it successfully completes work on database file dddd on instance iiiii.

Action: None required.

ORLBKFRZOVER

ORLBKFRZOVER, tttt : FREEZE on region rrrr (ddd) cleared

MUPIP Information: Issued by MUPIP ROLLBACK -ONLINE when it clears a FREEZE on region rrrr mapped to database file dddd; tttt is the time it cleared the FREEZE.

Action: None required.

ORLBKFRZPROG

ORLBKFRZPROG, tttt : waiting for FREEZE on region rrrr (ddd) to clear

MUPIP Information: Issued by MUPIP ROLLBACK -ONLINE when it encounters a region rrrr mapped to database file dddd which is frozen; tttt is the time it encountered the condition.

Action: ROLLBACK waits for a period determined by the gtm_db_startup_max_wait environment variable, after which it clears the FREEZE and proceeds. The ROLLBACK is inappropriate due to the conditions that lead to the FREEZE, cancel the ROLLBACK, otherwise cancel the FREEZE or wait for ROLLBACK to clear it automatically.

ORLBKNOSTP

ORLBKNOSTP, ONLINE ROLLBACK proceeding with database updates. MUPIP STOP will no longer be allowed

MUPIP Information: Issued by MUPIP ROLLBACK -ONLINE when it starts processing that cannot be interrupted without jeopardizing database integrity.

Action: Wait for the ROLLBACK to complete.

ORLBKNOV4BLK

ORLBKNOV4BLK, Region rrrr (dddd) has V4 format blocks. Database upgrade required. ONLINE ROLLBACK cannot continue

MUPIP Error: Issued by MUPIP ROLLBACK -ONLINE when it finds the region rrrr mapped to database file dddd contains V4 format blocks - online rollback does not support old format blocks.

Action: Upgrade the database to the current major version before attempting to use online rollback.

ORLBKSTART

ORLBKSTART, ONLINE ROLLBACK started on instance iiiii corresponding to dddd

MUPIP Information: Issued by MUPIP ROLLBACK -ONLINE when it starts work on database file dddd on instance iiiii.

Action: None required.

ORLBKTERMNTD

ORLBKTERMNTD, ONLINE ROLLBACK terminated on instance iiiii corresponding to dddd with the above errors

MUPIP Error: Issued by MUPIP ROLLBACK -ONLINE when it encounters issues that prevent it from operating on database file dddd on instance iiiii.

Action: Analyze and address the errors in the output preceding this message.

PERMGENDIAG

PERMGENDIAG, Permissions: Proc(uid:uuuu,gid:gggg), DB File(uid:vvvv,gid:hhhh,perm:pppp), Lib File(gid:iiii,perm:qqqq), Group Mem(opener:jjjj,owner:kkkk)

Run Time Information: This shows the permissions involved in a resource creation for the process, the associated database file, the libgtmsmr and the process group membership.

Action: Typically none, but if you have a permission issue use this key information for diagnosis.

PERMGENFAIL

PERMGENFAIL, Failed to determine access permissions to use for creation of xxxx for file yyyy

MUPIP/Run Time Error: This message indicates that GT.M was unable to determine the permissions to use when creating a file or resource associated with database file yyyy. xxxx may be "ipc resources", "journal file", "backup file", or "snapshot file".

Action: Note the user and group ownership of the database file and \$gtm_exe/libgtmshr.*, and the user and group permissions of the GT.M process, and report them to your GT.M support channel.

PRIMARYISROOT

PRIMARYISROOT, Attempted operation not valid on root primary instance xxxx

MUPIP Error: If a replication instance has local updates enabled, that is: the Source Server that created the journal pool was started with -UPDOK, issuing any Source Server command with the start, activate or deactivate qualifiers where the command explicitly specifies the propagateprimary qualifier or that qualifier is implicitly assumed by default, or, if this is a not a supplementary instance, attempting to start a Receiver Server causes MUPIP to issue this error.

Action: Do not start a Receiver Server on a root primary non-supplementary instance. Use rootprimary qualifier instead of propagateprimary in the source server command.

RCVR2MANY

RCVR2MANY, The instance already has the maximum supportable number of receiver servers nnnn active

MUPIP Error: Issued by a Receiver Server on a Supplementary Instance (started with -UPDOK) which found it exceeds nnnn, the currently permitted number of Receiver Servers.

Action: Reconfigure the instances to a supported configuration.

RCVRMANYSTRMS

RCVRMANYSTRMS, Receiver server now connecting to source stream NNNN but had previously connected to a different stream nnnn

MUPIP Error: Issued by a Receiver Server on a Supplementary Instance (started with -UPDOK) which had formerly connected to a source server corresponding to non-Supplementary stream nnnn, later disconnected and on reconnection found the Source Server corresponds to a different non-Supplementary stream NNNN.

Action: Mixing of non-Supplementary streams are not allowed in the same Receiver Server process. Restart the Receiver Server.

REPL2OLD

REPL2OLD, Instance IIII uses a GT.M version that does not support connection with the current version on iiiii.

MUPIP Error: Issued by a Source Server, Receiver Server or MUPIP JOURNAL -ROLLBACK - FETCHRESYNC on Instance iiiii attempted to connect to instance IIII, but found IIII is running an earlier version that does not support the current replication protocol. This can indicate either that the older version is just too old for any connection with the newer version (in case the older version is less than V5.1-000) or that the older version doesn't have the logic required to support a Supplementary Instance (in

case the older version is less than V5.5-000). Note that IIII may not be available if the older instance uses a version of GT.M less than V5.1-000.

Action: Upgrade the GT.M version on IIII to a version that can support communication with the current version or, if this is a Supplementary Instance, that can deal with a Supplementary Instance or choose another appropriate instance for the connection.

REPLINSTDBSTRM

REPLINSTDBSTRM, Replication instance file rrrr has seqno xxxx for Stream nnnn while database has a different seqno XXXX

MUPIP Error: Issued by the first source server started on a supplementary instance if the journal stream sequence numbers (for any non-supplementary stream from 0 through 15) stored in the instance file do not match those stored in the database file headers. This is possible if a database was recreated or refreshed from a backup on another instance without correspondingly recreating the instance file.

Action: If the database file is known to be accurate, recreate the instance file. If not, reinitialize this instance from a backup of some other instance in the same LMS Group (see Action section of REPLINSTDBMATCH error for more details on this).

REPLINSTMISMATCH

REPLINSTMISMATCH, Process has replication instance file ffff (jnlpool shmid = ssss) open but database dddd is bound to instance file gggg (jnlpool shmid =tttt)

Run Time Error: An update is being attempted on the replicated database dddd which is associated with the replication instance file ffff and journal pool shared memory id ssss but the currently updating process has a different replication instance file gggg or journal pool shmid tttt open.

Action: A replicated database can only be updated by processes that have the same replication instance file (defined by the environment variable gtm_repl_instance) open. Ensure the environment variable gtm_repl_instance is defined to be the same for all processes that update the same replicated database file. This error is also possible if the replication instance file was recreated (while processes were still accessing the replication instance). In this case, the name ffff and gggg would be the same but the corresponding journal pool shared memory ids would be different. To recover from this situation, shut down all processes accessing the instance from before and after the instance file recreate. Run an argumentless mupip rundown to clean up the older journal pool tttt and restart the instance. One more case of this error is when gggg is the empty string and tttt is 4294967295 (i.e. $2^{**32} - 1$). In this case, the Source Server was started with a global directory that (a) did not have this database file listed and hence this database was not associated with the journal pool that the Source Server created OR (b) had this database file listed but later the source server and this particular database file was shut down while other database files in this global directory were still active. To recover from (a), add this database file to the global directory used by the Source Server and restart the instance. The Source Server (which is the first process to start on a replicated instance) only binds replicated databases from its global directory to the journal pool that it creates. No other replicated database file can be bound with this journal pool. To recover from (b), restart the Source Server.

REPLUPGRADEPRI

REPLUPGRADEPRI, Attempted operation requires primary instance xxxx to support multi-site replication

Obsolete Error: No longer issued by GT.M

REPLUPGRADESEC

REPLUPGRADEPRI, Attempted operation requires primary instance xxxx to support multi-site replication

Obsolete Error: No longer issued by GT.M

RESOLVESEQNO

RESOLVESEQNO, Resolving until sequence number dddd [0xxxxx]

MUPIP Information: This indicates MUPIP JOURNAL ROLLBACK expects to do backward processing until it reaches sequence number with hexadecimal value xxxx (decimal value dddd). This is usually the common sequence number agreed upon between the primary and secondary by a -FETCHRESYNC rollback or the sequence number specified in a -RESYNC rollback.

Action: No action required.

RESOLVESEQSTRM

RESOLVESEQSTRM, Resolving until stream sequence number Stream nnnn : Seqno dddd xxxx

MUPIP Information: This indicates MUPIP JOURNAL ROLLBACK expects to do backward processing until it reaches the stream sequence number whose hexadecimal value is xxxx (decimal value dddd). This is usually the common stream sequence number agreed upon between the primary and secondary by a -FETCHRESYNC rollback or the stream sequence number specified in a -RESYNC rollback where -RSYNC_STRM is also specified.

Action: No action required.

RESUMESTRMNUM

RESUMESTRMNUM, Error with stream number specified in RESUME qualifier

MUPIP Error: Issued by a Receiver Server as an accompanying message to a UPDSYNCINSTFILE error. The stream number nnnn can be any integer value from -1 through 15.

Action: No action required for this message. Action is required for the preceding UPDSYNCINSTFILE error.

REUSEINSTNAME

REUSEINSTNAME, Error with instance name specified in REUSE qualifier

MUPIP Error: Issued by a Receiver Server when started with the -REUSE qualifier in case of either inappropriate use of this qualifier or an inappropriate instance name specified as a value to this qualifier.

Action: An accompanying GTM-I-TEXT message describes the particular error situation. Take appropriate corrective action based on that.

RLBKCONFIGBNDRY

RLBKCONFIGBNDRY, Rollback encountered journal records indicating current source iiiii replaced old source oooo; cannot rollback past sequence number ssss

MUPIP Error: Issued by MUPIP JOURNAL -ROLLBACK indicating it found a change in the source configuration from instance oooo to iiiii for stream ssss within the range of its attempt to process. Since it cannot dynamically reconfigure the source connections to deal with this, it stops.

Action: Investigate whether a shorter rollback that won't encounter this configuration change would be useful. If not, refresh this instance from a backup or use the -RSYNC_STRM qualifier if appropriate.

RLBKSTRMSEQ

RLBKSTRMSEQ, Stream journal seqno of the instance after rollback is Stream nnnn : Seqno dddd xxxxx

MUPIP Information: On a Supplementary Instance, MUPIP JOURNAL -ROLLBACK issues this message for each stream (from 0 through 15) that has at least one update. This message indicates how many updates in each stream this Supplementary Instance has processed.

Action: No action required.

RNDWNSKIPCNT

RNDWNSKIPCNT, A total of nnnn process(es) skipped database rundown due to a concurrent ONLINE ROLLBACK

Run Time Information: nnnn processes disconnected from a one or more databases while online rollback was running and therefore relied on online rollback or other remaining processes to complete any database shutdown.

Action: None required typically. If there's cleanup required, use MUPIP RUNDOWN.

RSYNCSTRMSUPPONLY

RSYNCSTRMSUPPONLY, RSYNC_STRM qualifier only supported for Supplementary Instances

MUPIP Error: Issued by MUPIP JOURNAL -ROLLBACK indicating the -RSYNC_STRM qualifier only applies to Supplementary Instances - Business Continuity instances require comprehensive synchronization.

Action: Reissue the command without the -RSYNC_STRM qualifier.

RSYNCSTRMVAL

RSYNCSTRMVAL, RSYNC_STRM qualifier can only take on a value from 0 to 15

MUPIP Error: Issued by a MUPIP JOURNAL -ROLLBACK -RESYNC command which also specifies -RSYNC_STRM with a stream number outside of the range of 0 through 15.

Action: Specify a stream number within the allowed range.

SECNOTSUPPLEMENTARY

SECNOTSUPPLEMENTARY, ssss is a Supplementary Instance and so cannot act as a source to non-Supplementary Instance iiiii

MUPIP Error: Issued by a Source Server on a Supplementary Instance ssss attempted to connect to a Replicating Instance iiiii, but found iiiii is not configured as a Supplementary Instance.

Action: Reconfigure the instances to a supported configuration.

SEMWT2LONG

SEMWT2LONG, Process wwwww waited ssss second(s) for the llll lock for region rrrr, lock held by pid pppp

Run Time Error: This indicates that the process pppp appears to be holding the llll control semaphore for region rrrr for longer than GT.M expects.

Action: Analyze the behavior of process pppp, and terminate it if appropriate. This error may indicate that the system is under-configured for the workload.

SHMREMOVED

SHMREMOVED, Removed Shared Memory id mmmm corresponding to file ffff

MUPIP Information: MUPIP RUNDOWN removed shared memory segment mmmm, corresponding to the file ffff, which could be a database file or a replication instance file with a GT.M signature because the resource was not actively in use.

Action: No action required.

STRMNUMMISMATCH1

STRMNUMMISMATCH1, Stream nnnn exists on the receiver instance file but is unknown on the source instance

MUPIP Error: Issued by a Source Server on a Supplementary Instance when it detects a non-Supplementary stream number nnnn (which can be any value from 1 through 15) exists on the receiving instance but not on the source instance. This indicates the two instances are not in sync at least with respect to stream nnnn, so replication cannot proceed.

Action: Reinitialize the receiving instance from a backup of the source instance and restart replication between the two instances.

STRMNUMMISMATCH2

STRMNUMMISMATCH2, Stream nnnn exists on the source instance file but is unknown on the receiver instance

MUPIP Error: Issued by a Source Server on a supplementary instance when it detects a non-Supplementary stream number nnnn (which can be any value from 1 through 15) exists on the source instance but not on the receiving instance. This indicates the two instances are not in sync at least with respect to stream nnnn and so replication cannot proceed.

Action: Reinitialize the receiving instance from a backup of the source instance and restart replication between the two instances.

STRMSEQMISMATCH

STRMSEQMISMATCH, Unable to play update on Stream nnnn with seqno xxxx as receiving instance has a different stream seqno XXXX

MUPIP Error: Issued by the Update Process on a supplementary instance started with -UPDNOTOK (that is, local updates are disabled) when it finds the source and receiving instances have different values of stream sequence number for the non-Supplementary stream nnnn (can be any value from 1 through 15). This indicates the two instances are not in sync at least with respect to stream nnnn and so replication cannot proceed.

Action: Reinitialize the receiving instance from a backup of the source instance and restart replication between the two instances.

SUPRCVRNEEDSSUPSRC

SUPRCVRNEEDSSUPSRC, Instance iiiii is not configured to perform local updates, so it cannot act as a receiver for non-Supplementary Instance ssss

MUPIP Error: Issued by a Receiver Server or a MUPIP JOURNAL -ROLLBACK -FETCHRESYNC on a Supplementary Instance ssss started with -UPDNOTOK attempted to connect to non-Supplementary Instance iiiii. A Supplementary Instance that does not permit local updates can only replicate from another Supplementary Instance.

Action: Reconfigure the instances to a supported configuration.

SRVLCKWT2LNG

SRVLCKWT2LNG, PID pppp is holding the source server lock. Waited for mmmm minute(s). Now exiting

MUPIP Error: Issued by MUPIP ROLLBACK -ONLINE when it finds process pppp has not released the journal pool resource for mmmm minutes.

Action: Investigate the state of process pppp and whether it should be stopped by operator action.

STRMNUMIS

STRMNUMIS, Stream # is ssss

MUPIP Information: Issued by a Receiver Server to designate a stream associated with the the immediately preceding message.

Action: Refer to the associated prior message.

TRIGMODREGNOTRW

TRIGMODREGNOTRW, Trigger(s) cannot be added/changed/deleted because region rrrr is read-only

Runtime Error: This error occurs when \$ZTRIGGER() or MUPIP TRIGGER attempts to write to read-only region rrrr.

Action: Check for appropriate global directory mapping and appropriate permissions on the database file mapped to region rrrr.

UPDSYNC2MTINS

UPDSYNC2MTINS, Can only UPDATERESYNC with an empty instance file

MUPIP Error: Issued by a Receiver Server started with the -UPDATERESYNC qualifier on a non-supplementary instance or on a Supplementary Instance started specifying the -UPDNOTOK qualifier, when the replication instance file on the Receiver side contains at least one history record. The purpose of -UPDATERESYNC is to unconditionally declare the instance state as a valid state in the set of current and prior states of the originating instance disregarding any history. Note that the receiver server on a Supplementary Instance started with -UPDOK does not issue this error.

Action: Verify that -UPDATERESYNC is appropriate and if so, recreate the instance file to discard the history then reissue the command. If the replication state is not a valid match to some available current or prior state of the originating instance, either do a normal resync or refresh the replicating instance to an appropriate state.

UPDSYNCINSTFILE

UPDSYNCINSTFILE, Error with instance file name specified in UPDATERESYNC qualifier

MUPIP Error: Issued by a Receiver Server when started with the -UPDATERESYNC qualifier in case of any error while processing the instance file specified as a value to this qualifier.

Action: An accompanying message, usually a GTM-I-TEXT message, describes the particular error situation. Take appropriate corrective action based on that.

ZDATEBADDATE

ZDATEBADDATE, \$ZDATE() date argument dddd is less than -365 (the \$HOROLOG value for 01-JAN-1840) or greater than 364570088 (the \$HOROLOG value for 31-DEC-999999)

Run Time Error: The value of the date portion of the argument dddd to \$ZDATE() is outside the range of values the function handles.

Action: Examine the code that created the value of dddd for errors or revise the design to create dates within the range supported by \$ZDATE().

ZDATEBADTIME

ZDATEBADTIME, \$ZDATE() time argument tttt is less than 0 or greater than 86399 (the \$HOROLOG value for a second before midnight)

Run Time Error: The value of the time portion of the argument tttt to \$ZDATE() is outside the range of values the function handles.

Action: Examine the code that created the value of tttt for errors or revise the design to create times within the range supported by \$ZDATE().

ZTRIGNOTRW

ZTRIGNOTRW, ZTRIGGER cannot operate on read-only region rrrr

Runtime Error: This error occurs when ZTRIGGER attempts to write to read-only region rrrr.

Action: Check for appropriate global directory mapping and appropriate permissions on the database file mapped to region rrrr.

Documentation Addendum

Journal Extract Format

To support supplementary instance replication, V5.5-000 added new fields and changed the record format of journal extracts. You may need to modify application code and shell scripts which read and process journal extracts. Here is the summary of the journal extract format changes in V5.5-000. New fields are in **bold** and changes are in *italics*.

Type	V5.4-002B	V5.5-000
Journal extract format (Plain)		
Header Label	GDSJEX05	GDSJEX06
NULL	00\time\tnum\pid\clntpid\jsnum	00\time\tnum\pid\clntpid\jsnum\strm_num\strm_seq
KILL	04\time\tnum\pid\clntpid\token_seq\updnum\nodeflags\node	04\time\tnum\pid\clntpid\token_seq\strm_num\strm_seq\updnum\nodeflags\node
SET	05\time\tnum\pid\clntpid\token_seq\updnum\nodeflags\node=sarg	05\time\tnum\pid\clntpid\token_seq\strm_num\strm_seq\updnum\nodeflags\node=sarg
TSTART	08\time\tnum\pid\clntpid\token_seq	08\time\tnum\pid\clntpid\token_seq\strm_num\strm_seq
TCOM	09\time\tnum\pid\clntpid\token_seq\partners\tid	09\time\tnum\pid\clntpid\token_seq\strm_num\strm_seq\partners\tid
ZKILL	10\time\tnum\pid\clntpid\token_seq\updnum\nodeflags\node	10\time\tnum\pid\clntpid\token_seq\strm_num\strm_seq\updnum\nodeflags\node
ZTWORM	11\time\tnum\pid\clntpid\token_seq\updnum\ztwormhole	11\time\tnum\pid\clntpid\token_seq\strm_num\strm_seq\updnum\ztwormhole
ZTRIG	12\time\tnum\pid\clntpid\token_seq\updnum\nodeflags\node	12\time\tnum\pid\clntpid\token_seq\strm_num\strm_seq\updnum\nodeflags\node
Journal extract format (Detail)		
Header Label	GDSJDX05	GDSJDX06
KILL	time\tnum\chksum\pid\clntpid\token_seq\updnum\nodeflags\node	time\tnum\chksum\pid\clntpid\token_seq\strm_num\strm_seq\updnum\nodeflags\node
SET	time\tnum\chksum\pid\clntpid\token_seq\updnum\nodeflags\node=sarg	time\tnum\chksum\pid\clntpid\token_seq\strm_num\strm_seq\updnum\nodeflags\node=sarg

Type	V5.4-002B	V5.5-000
TKILL	time\tnum\chksum\pid\clntpid\token_seq \updnum\nodeflags\node	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\nodeflags\node
UKILL	time\tnum\chksum\pid\clntpid\token_seq \updnum\nodeflags\node	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\nodeflags\node
TSET	time\tnum\chksum\pid\clntpid\token_seq \updnum\nodeflags\node=sarg	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\nodeflags\node=sarg
USET	time\tnum\chksum\pid\clntpid\token_seq \updnum\nodeflags\node=sarg	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\nodeflags\node=sarg
FSET	time\tnum\chksum\pid\clntpid\token_seq \updnum\nodeflags\node=sarg	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\nodeflags\node=sarg
GSET	time\tnum\chksum\pid\clntpid\token_seq \updnum\nodeflags\node=sarg	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\nodeflags\node=sarg
FKILL	time\tnum\chksum\pid\clntpid\token_seq \updnum\nodeflags\node	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\nodeflags\node
GKILL	time\tnum\chksum\pid\clntpid\token_seq \updnum\nodeflags\node	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\nodeflags\node
TSTART	time\tnum\chksum\pid\clntpid\token_seq	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq
TCOM	time\tnum\chksum\pid\clntpid\token_seq \partners\tid	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\partners\tid
ZKILL	time\tnum\chksum\pid\clntpid\token_seq \updnum\nodeflags\node	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\nodeflags\node
TZKILL	time\tnum\chksum\pid\clntpid\token_seq \updnum\nodeflags\node	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\nodeflags\node
UZKILL	time\tnum\chksum\pid\clntpid\token_seq \updnum\nodeflags\node	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\nodeflags\node
EPOC	time\tnum\chksum\pid\clntpid\jsnum \blks_to_upgrd\free_blocks\total_blks	time\tnum\chksum\pid\clntpid\jsnum\blks_to_upgrd \free_blocks\total_blks\fully_upgraded[\strm_num \strm_seq]...
ZTWORM	time\tnum\chksum\pid\clntpid\token_seq \updnum\ztwormhole	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\ztwormhole
NULL	time\tnum\pid\clntpid\jsnum	time\tnum\pid\clntpid\jsnum\strm_num\strm_seq
ZTRIG	time\tnum\chksum\pid\clntpid\token_seq \updnum\nodeflags\node	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\nodeflags\node
TZTWORM	time\tnum\chksum\pid\clntpid\token_seq \updnum\ztwormhole	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\ztwormhole

Type	V5.4-002B	V5.5-000
UZTWORM	time\tnum\chksum\pid\clntpid\token_seq \updnum\ztwormhole	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\ztwormhole
TZTRIG	time\tnum\chksum\pid\clntpid\token_seq \updnum\nodeflags\node	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\nodeflags\node
UZTRIG	time\tnum\chksum\pid\clntpid\token_seq \updnum\nodeflags\node	time\tnum\chksum\pid\clntpid\token_seq\strm_num \strm_seq\updnum\nodeflags\node

Legend (additions in **bold** and changes in *italics*):

strm_num	If replication is true and this update originated in a non-supplementary instance but was replicated to and updated a supplementary instance, this number is a non-zero value anywhere from 1 to 15 (both inclusive) indicating the non-supplementary stream number. In all other cases, this stream # value is 0. In case of an EPOCH record, anywhere from 0 to 16 such "strm_num" numbers might be displayed depending on how many sources of supplementary instance replication have replicated to the instance in its lifetime.
strm_seq	If replication is true and this update originated in a non-supplementary instance but was replicated to and updated a supplementary instance, this is the journal sequence number of the update on the originating non-supplementary instance. If replication is true and this update originated in a supplementary instance, this is the journal sequence number of the update on the originating supplementary instance. In all other cases, this stream sequence number is 0. Note that the journal seqno is actually 1 more than the most recent update originating on that stream number. In case of an EPOCH record, anywhere from 0 to 16 such "strm_seq" numbers might be displayed depending on how many sources of supplementary instance replication have replicated to the instance in its lifetime.
nodeflags	<p>Decimal number interpreted as a binary mask.. Currently only 5 bits are used.</p> <ul style="list-style-type: none"> • 00001 (1) => update journaled but NOT replicated (For example, update inside a trigger) • 00010 (2) => update to a global that had at least one trigger defined, even if no trigger matched this update • 00100 (4) => \$ZTWORMHOLE holds the empty string ("") at the time of this update or was not referenced during this update • 01000 (8) => update did not invoke any triggers even if they existed (For example, MUIPI LOAD) • 10000 (16) => whether the update (set or kill) is a duplicate. In case of a KILL, it is a kill of some non-existing node aka duplicate kill. Note that the dupkill occurs only in case of the Update Process. In case of GT.M, the KILL is entirely skipped. In both cases (duplicate set or kill), only a jnl record is written, the db is untouched. <p>Combinations of the above bits would mean each of the individual bit characteristics. For example, 00011 => update within a trigger context, and to a global with at least one trigger defined. For example, 00011 => update inside a trigger and to a global with at least one trigger</p>

	defined. Certain bit combinations are impossible. For example, 01001 since GT.M replicates any update that does not invoke triggers.
fully_upgraded	1 if the db was fully upgraded (indicated by a dse dump -file -all) at the time of writing the EPOCH

Shared Resource Authorization Permissions

GT.M uses several types of shared resources to implement concurrent access to databases. The first GT.M process to open a database file creates IPC resources (semaphores and shared memory) required for concurrent use by other GT.M processes, and in the course of operations GT.M processes create files (journal, backup, snapshot) which are required by other GT.M processes. In order to provide access to database files required by M language commands and administration operations consistent with file permissions based on the user, group and world classes, the shared resources created by GT.M may have different ownership, groups and permissions from their associated database files as described below. As an example of the complexity involved, consider a first process opening a database based on its group access permissions. In other words, the database file is owned by a different userid from the semaphores and shared memory created by that first process. Now, if the userid owning the database file is not a member of the database file's group, a process of the userid owning the database file can only have access to the shared resources if the shared resources have world access permissions or if they have a group that is guaranteed to be shared by all processes accessing the database file, even if that group is different from the database file's own group. Again, although FIS strongly recommends against running GT.M processes as root, a root first process opening the database file must still be able to open it although it may not be the owner of the database file or even in its group - but it must ensure access to other, non-root processes. Some things to keep in mind:

- Even a process with read-only access to the database file requires read-write access to the shared memory control structures and semaphores.
- Creating and renaming files (for example, journal files) requires write access to both the files and the directories in which they reside.
- If you use additional layered security (such as Access Control Lists or SELinux), you must ensure that you analyze these cases in the context of configuring that layered security.

GT.M takes a number of factors into account to determine the resulting permissions:

- The owner/group/other permissions of the database file
- The owner of the database file
- The group of the database file
- The group memberships of the database file's owner
- The owner/group/other permissions of the libgtmshr file
- The group of the libgtmshr file

- The effective user id of the creating process
- The effective group id of the creating process
- The group memberships of the creating process' user

The following table describes how these factors are combined to determine the permissions to use:

Database File Permissions	Opening process is owner of database file?	Owner is member of group of database file?	Opening process is a member of database file group?	Opening process is not owner but a member of database file group?	Execution of GT.M restricted to members of a group?
<i>Group of Resource</i>		<i>IPC Permissions</i>		<i>File Permissions</i>	
-r*-r*-r*-	-	-	Y	-	-
<i>Group of database file</i>		<i>-rw-rw-rw</i>		<i>-r*-r*-r*</i>	
-rw-rw-r*	-	-	N	-	-
<i>Current group of process</i>		<i>-rw-rw-rw</i>		<i>-rw-rw-rw</i>	
-rw-rw-rw	-	-	N	-	-
<i>Current group of process</i>		<i>-rw-rw-rw</i>		<i>-rw-rw-rw</i>	
-rw-rw-rw	Y	Y	-	-	-
<i>Group of database file</i>		<i>-rw-rw-rw</i>		<i>-r*-r*----</i>	
-r*-r*----	Y	N	-	-	N
<i>Current group of process</i>		<i>-rw-rw-rw-</i>		<i>-rw-rw-rw-</i>	
-r*-r*----	Y	N	-	-	Y
<i>Group to which GT.M is restricted</i>		<i>-rw-rw----</i>		<i>-rw-rw----</i>	
-r*-r*----	-	Y	-	Y	-
<i>Group of database file</i>		<i>-rw-rw----</i>		<i>-r*-r*----</i>	
-r*-r*----	-	N	-	Y	N
<i>Group of database file</i>		<i>-rw-rw-rw-</i>		<i>-rw-rw-rw-</i>	
-r*-r*----	-	N	-	Y	Y
<i>Group to which GT.M is restricted</i>		<i>-rw-rw----</i>		<i>-rw-rw----</i>	
---r*----	-	N	-	Y	-
<i>Group of database file</i>		<i>-rw-rw----</i>		<i>---r*----</i>	
-r*-----	Y	-	-	-	-

Database File Permissions	Opening process is owner of database file?	Owner is member of group of database file?	Opening process is a member of database file group?	Opening process is not owner but a member of database file group?	Execution of GT.M restricted to members of a group?
<i>Group of Resource</i>		<i>IPC Permissions</i>		<i>File Permissions</i>	
<i>Current group of process</i>		<i>-rw-----</i>		<i>-rw-----</i>	

- The resulting group ownership and permissions are found by matching the database file permissions, then determining which question columns produce the correct "Y" or "N" answer; "-" answers are "don't care".
- An asterisk ("*") in the Database File Permissions matches writable or not writable. An asterisk in the Resulting File Permissions means that GT.M uses the write permissions from the database file.
- GT.M determines group restrictions by examining the permissions of the libgtmsshr file. If it is not executable to others, GT.M treats it as restricted to members of the group of the libgtmsshr file.
- Group membership can either be established by the operating system's group configuration or by the effective group id of the process.
- A GT.M process requires read access in order to perform write access to database file - a file permission of write access without read access is an error.
- GT.M treats the "root" user the same as other users, except that when it is not the file owner and not a member of the group, it is treated as if it were a member of the group.

Areas affected by ONLINE ROLLBACK

Area Impacted by MUPIP JOURNAL -ONLINE - ROLLBACK	Effect of MUPIP JOURNAL -ONLINE -ROLLBACK
\$ZONLNRLBK	Incremented by any online rollback that takes the database back to a prior state
M run-time actions that do not interact with the target database	None
M run-time database interactions	Suspended during the online rollback. Restart if the online rollback does not take the database back to a prior state or if within an explicit TP transaction. Otherwise, DBROLLEDBACK error if the interaction is a non-TP mini-transaction. Note that triggers act like an extension of their originating event: they get the same treatment as the trigger event would get - if they are within an implicit transaction triggered outside a TP transaction, the process gets a DBROLLEDBACK error.

Area Impacted by MUPIP JOURNAL -ONLINE -ROLLBACK	Effect of MUPIP JOURNAL -ONLINE -ROLLBACK
M run-time larger KILLS	Because of the way GT.M manages multi-node KILLS, it possible for online rollback to remove bit-map cleanup while leaving the original KILL. This creates incorrectly marked busy errors, which represent unused, but protected space - they are benign, but should be cleaned up in a timely manner.
GT.CM GNP Server	Like the M run-time, except it sends any DBROLLEDBACK error to the client processes.
Starting most GT.M actions from the shell, including operator Utilities - see below for exceptions	Deferred until the ONLINE ROLLBACK is complete. The environment variable gtm_db_startup_max_wait determines the deferred time. For more information, refer to GTM-4337.
MUMPS	None unless, and until, the process tries to interact with the target database
Source Server	If the online rollback takes the database back to a prior state, disconnect from the receiving instance and restart the connection with a message that a rollback occurred.
Update Process	If the online rollback takes the database back to a prior state, suspend processing and notify the Receiver Server that there was a rollback on this receiving instance.
Receiver Server	When notified by the Update Process, disconnect from the source instance and reconnect with a message to restart transmission from the new current sequence number. When it receives a message the source instance has rolled back and it was started with -AUTOROLLBACK, initiate an online rollback on the receiving instance, otherwise shutdown in anticipation of an operator action.
LKE	None if already running before online rollback starts
<p data-bbox="198 1394 800 1482">Running GT.M utility commands that are mutually exclusive with MUPIP JOURNAL -ONLINE -ROLLBACK and require standalone access. For example:</p> <ul data-bbox="198 1514 800 1801" style="list-style-type: none"> <li data-bbox="198 1514 427 1539">• MUPIP UPGRADE <li data-bbox="198 1570 475 1596">• MUPIP DOWNGRADE <li data-bbox="198 1627 800 1688">• MUPIP JOURNAL -ROLLBACK or MUPIP JOURNAL -RECOVER <li data-bbox="198 1719 456 1745">• MUPIP ENDIANCVT <li data-bbox="198 1776 443 1801">• MUPIP RUNDOWN 	MUPIP JOURNAL -ONLINE -ROLLBACK does not run when these commands are running. Analogously, these commands cannot run during a MUPIP JOURNAL -ONLINE -ROLLBACK.

Area Impacted by MUPIP JOURNAL -ONLINE - ROLLBACK	Effect of MUPIP JOURNAL -ONLINE -ROLLBACK
<ul style="list-style-type: none"> • MUPIP RESTORE • MUPIP INTEG -FILE 	
<p>Multi-action state dependent utility commands. For example:</p> <ul style="list-style-type: none"> • MUPIP INTEG -REGION [-ONLINE -NOONLINE] • MUPIP BACKUP • MUPIP LOAD 	<p>Produces a DBROLLEDBACK error.</p> <p>Subsequent MUPIP BACKUP -BYTESTREAM - SINCE=<event> commands may start at transaction 1 if the rollback takes the database state back prior to the event (DATABASE, BYTESTREAM or RECORD).</p>
DSE	DSE cannot start while online rollback is running. If it has already started, its commands work as described in other rows of this table.
<p>Utility commands that do not interact with database files. For example:</p> <ul style="list-style-type: none"> • DSE CLOSE • DSE EVALUATE • DSE EXIT • DSE HELP • DSE OPEN • DSE PAGE • DSE QUIT • DSE SPAWN • DSE VERSION • GDE • LKE CLEAR • LKE SHOW • MUPIP CREATE • MUPIP EXIT • MUPIP FTOK • MUPIP INTRPT • MUPIP QUIT 	<p>DSE and LKE cannot start while online rollback is active, but if active they can continue to operate; MUPIP commands always start from the shell so each command may have different behavior.</p> <p>MUPIP CREATE does create database files but only if they do not already exist and if they do not exist, they can be rolled back.</p> <p>MUPIP INTRPT does not interact with a database, but the target process may and that could prevent it from recognizing the interrupt in a timely fashion.</p>

Area Impacted by MUPIP JOURNAL -ONLINE -ROLLBACK	Effect of MUPIP JOURNAL -ONLINE -ROLLBACK
<ul style="list-style-type: none"> • MUPIP STOP 	
<p>Utility commands that update a database and interlock with online rollback. For example:</p> <ul style="list-style-type: none"> • DSE ADD • DSE ALL -WCINIT • DSE ALL -BUFFER_FLUSH • DSE ALL -SEIZE • DSE ALL -RELEASE • DSE ALL -FREEZE -OVERRIDE • DSE BUFFER_FLUSH • DSE CACHE except -SHOW • DSE CHANGE except -FILEHEADER -NOCRIT • DSE CRITICAL -SEIZE • DSE CRITICAL -RELEASE • DSE CRITICAL -CYCLE • DSE MAPS • DSE OVERWRITE • DSE REMOVE -RECORD • DSE RESTORE • DSE SAVE • DSE SHIFT • DSE WCINIT 	<p>These commands wait for online rollback to complete.</p>
<p>Utility commands that only read database files and display current state information. For example: ·</p> <ul style="list-style-type: none"> • DSE ALL -DUMP • DSE ALL -NOFREEZE • DSE CACHE -SHOW • DSE CRITICAL -ALL 	<p>These commands show the state as of the time they are issued and if issued while there are any concurrent updates, including online rollback may show a transient and out-of-date state.</p>

Area Impacted by MUPIP JOURNAL -ONLINE -ROLLBACK	Effect of MUPIP JOURNAL -ONLINE -ROLLBACK
<ul style="list-style-type: none"> • DSE CRITICAL -CRASH • DSE DUMP -BLOCK • DSE DUMP -RECORD • DSE DUMP -FILE • DSE FIND • DSE INTEG • DSE REMOVE -BLOCK • DSE RANGE 	
<p>Utility commands that interact with a database and are not blocked by online rollback:</p> <ul style="list-style-type: none"> • DSE ALL -REFERENCE • DSE CRITICAL -INIT • DSE CRITICAL -REMOVE • DSE CHANGE -FILEHEADER -NOCRIT 	<p>These commands may disrupt the -online rollback and may cause database damage - they are available only for very unusual situations and should only be used under instructions from your GT.M support channel.</p>