Pivotal GemFire XD 1.3.0 Release Notes

Release Date	18 September 2014
Last Document Update	18 September 2014
Additional Documentation	*All GemFire XD 1.3.0 Documentation*

What's In the Release Notes

The release notes cover the following topics:

- *What's New in Pivotal GemFire XD 1.3*
- *Installing GemFire XD 1.3*
- *Upgrading to GemFire XD 1.3*
- *Basic Features of GemFire XD*
- *Resolved Issues*
- *Known Issues*

What's New in Pivotal GemFire XD 1.3

Pivotal GemFire XD is a memory-optimized, distributed data store that is designed for applications that have demanding scalability and availability requirements. With GemFire XD you can manage data entirely using in-memory tables, or you can persist very large tables to local disk store files or to a Hadoop Distributed File System (HDFS) for big data deployments.

GemFire XD is a new product that builds upon the previous Pivotal SQLFire product. GemFire XD shares many internal features with SQLFire, such as the SQL language extensions, query capabilities, system tables, local disk store persistence, programming APIs, and the gfxd command-line utility (previously sqlf).

In contrast to GemFire XD 1.0, version 1.3 is distributed as a standalone product, and it provides an upgrade path from SQLFire 1.1.1 or 1.1.2. GemFire XD 1.3 also includes support for Windows platforms and for developing Windows clients using ADO.NET.

For Pivotal SQLFire customers, GemFire XD 1.3 introduces these new features:

- **GEMFIRE XD.NET ENTITY FRAMEWORK**

  GemFire XD 1.3 includes an Entity Framework plug-in to generate model objects from an existing schema, or to generate a schema from model objects. You can also issue queries using Language-Integrated Query (LINQ), and then retrieve and manipulate data as strongly-typed objects using C# or VB.NET.

  The Entity FrameWork plug-in is supported with the .NET Framework 4.0 and Microsoft Visual Studio 2010 with Service Pack 1. See *Installing GemFire XD ADO.NET Components* and *Using the GemFire XD.NET Entity Framework Plug-In* for more information.
- **DATA RECOVERY UTILITIES FOR GEMFIRE XD DISK STORES**

  GemFire XD provides a data extractor utility that attempts to recover as much data as possible from available disk store files. The recovered data is stored in multiple comma-separated values (CSV) files, which you can use to load the data into a new GemFire XD system using a separate loader utility.

  The data recovery utilities provide a "best effort" attempt to recover data in the failure scenarios that result in corrupted disk stores, when no disk store backups are available.

  See *Recovering Data from Disk Stores* for more information.

- **HDFS PERSISTANCE**

  GemFire XD provides the capability to store partitioned table data in a Hadoop Distributed File System (HDFS). Using HDFS as a persistent store enables you to capture high rates of table updates in HDFS for later processing in Hadoop. Or, you can choose to manage very large tables in GemFire XD—tables much larger than can be managed in memory.

  GemFire XD provides two models for persisting GemFire XD table data to HDFS. With HDFS write-only persistence all of the DML operations that are applied to a table are persisted to HDFS log files, with no compaction. HDFS write-only persistence enables applications to collect unbounded volumes of data in parallel for future processing using Hadoop tools such as MapReduce, or Hive. You can also use the GemFire XD PXF driver to access the persisted Hadoop data using HAWQ.

  With HDFS read/write persistence, all of the DML operations that are applied to a table are persisted to HDFS log files. GemFire XD periodically compacts the log files into larger files in order to improve read performance. You can access table data in the HDFS log files either by using Hadoop tools or by using SQL queries from clients or peers in the GemFire XD distributed system. In order to preserve high performance for your GemFire XD clients, you define the operational data set that is maintained in memory and choose whether to limit queries to the in-memory data set or scan the persistent HDFS files as necessary.

  See *Persisting Table Data to Hadoop*.
- **MAPREDUCE API**

  GemFire XD extends the Hadoop MapReduce API to support MapReduce jobs that work with data stored in GemFire XD HDFS log files. A custom InputFormat implementation enables MapReduce jobs to access data in HDFS log files without having to start up or connect to a GemFire XD distributed system. Similarly, an OutputFormatter implementation simplifies the task of writing data back into a GemFire XD table using a client connection to a distributed system.

  See *Using MapReduce to Access HDFS Table Data*.
- **PIVOTAL HAWQ INTEGRATION**

  In addition to providing a custom input/output system to support standard Hadoop tools, GemFire XD supports a PXF Driver to enable Pivotal HAWQ to read HDFS table data. (The required PXF driver is installed with HAWQ.) You simply define an external table in HAWQ that references GemFire XD HDFS log files, specifying custom properties to determine which log files are used to provide data. The PXF driver enables deep analytic processing of stored HDFS data without having to connect as a client to the GemFire XD distributed system.

  See *Using HAWQ to Access HDFS Table Data*.
- **OFF-HEAP TABLE STORAGE**

  GemFire XD supports storing table data directly in the JVM heap (the default), or in off-heap memory. Storing table data in off-heap memory enables you to manage very large databases in-memory, while improving CPU utilization and performance by avoiding JVM garbage collection pauses.

  See *Storing Tables in Off-Heap Memory*.
- **ODBC DRIVER**

  GemFire XD now includes an ODBC driver that allows you to execute ODBC API functions on GemFire XD data stores from your client applications. In addition, GemFire XD provides a command-line tool that allow you to easily install and configure the driver using either the unixODBC, iODBC, or Progress ODBC driver manager. The GemFire XD ODBC driver is supported on 32-bit and 64-bit Linux platforms. See *Using ODBC Driver Functions in Client Applications* for more information.

  GemFire XD supports ODBC driver version 3.5 APIs only. For a list of all ODBC function APIs that are supported by GemFire XD, see *ODBC API*. Any ODBC APIs that are not listed in the reference are not supported with GemFire XD.
- **MANAGEMENT FEATURES**

  GemFire XD adds these management features:

  - *Federated JMX Architecture*. GemFire XD uses MBeans to manage and monitor different parts of the software. With federation of the MBeanServers one member, the JMX Manager Node, provides a proxy view of all the MBeans that the MBeanServer hosts, and no separate JMX agent is required. The single JMX manager node can also distribute operations and notifications across the distributed system.
  - *JMX Manager MBeans* and *Managed Node MBeans*. New MBeans are available to provide better statistics and information about the GemFire XD distributed system as well as database objects.
  - *Pulse Monitoring Application*. GemFire XD Pulse is a Web Application that provides a graphical dashboard for monitoring vital, real-time health and performance of GemFire XD clusters, members, and tables. You can use Pulse to examine total memory, CPU, and disk space used by members, uptime statistics, client connections, WAN connections, query statistics, and critical notifications.

    Note: GemFire XD Pulse does not support LDAP authentication.
  - *Enhanced EXPLAIN command*. The EXPLAIN command was updated to provide simpler, customizable output for generated query plans.
- **AUTOMATIC RECONNECT FOR DISTRIBUTED SYSTEM MEMBERS**

  A GemFire XD member that is forcibly disconnected from a distributed system automatically shuts down and then restarts into a "reconnecting" state while it periodically attempts to rejoin the system. Callback methods are provided so you can perform custom actions during the reconnect process, or you can cancel the reconnection if necessary. See *Handling Forced Disconnection*.
- **WAN CONFLICT DETECTION**

  GemFire XD improves the performance of WAN replication. Also, when WAN updates cannot be applied to a cluster due to conflicts or other errors, GemFire XD stores the failed DML operations in XMLs to be applied at a later time. You can program custom error handlers to manage conflict errors as they occur. User-defined error handlers can be used to implement eventual consistency for WAN updates between GemFire XD systems. See *Resolving WAN Errors*.
- **IMPROVED RECOVERY PERFORMANCE**

  When starting distributed system members, GemFire XD can speed the recovery of persisted data for partitioned table by distributing only the data necessary to bring each member up-to-date, rather than performing a full recovery.

  In addition, GemFire XD now persists table indexes (in `.irf` files) to provide faster index recovery at startup. See *Disk Store File Names and Extensions*.

Installing GemFire XD 1.3

Note: Pivotal GemFire XD 1.3.0 will be the last GemFire XD release to support Java SE 6.

This release of GemFire XD is provided as a standalone product install. GemFire XD is also provided as a Pivotal HD 2.1 installable component, for use with the Pivotal Command Center CLI (Command Line Interface) installer. Note that GemFire XD requires an installation of Pivotal HD 2.1 in order to support HDFS table persistence and HAWQ integration features, regardless of whether you install GemFire XD as a standalone product or as a component of Pivotal HD.

To install GemFire XD on a Linux platform, use the instructions in *RHEL: Install GemFire XD from an RPM* or *Ubuntu: Install GemFire XD from a Debian Package*.

To install GemFire XD on a Windows platform, use the instructions in *Windows/Linux: Install GemFire XD from a ZIP File*.

If you purchased GemFire XD as a component of Pivotal HD Enterprise, or you intend to install GemFire XD and Pivotal HD Enterprise on the same systems, see the *Pivotal HD Installation and Administration Guide* for instructions to install GemFire XD.

Upgrading to GemFire XD 1.3

> **Note:** If you purchased GemFire XD as a component of Pivotal HD Enterprise, see the *Pivotal HD Installation and Administration Guide* for instructions to upgrade GemFire XD.

> **Note:** Pivotal GemFire XD 1.3.0 will be the last GemFire XD release to support Java SE 6.

The standalone GemFire XD 1.3 product supports upgrading from SQLFire version 1.1.1 or 1.1.2, with the following limitations:

- Mixing SQLFire and GemFire XD members in the same distributed system is not supported. You are required to stop the entire SQLFire distributed system before upgrading individual members to GemFire XD.
- JDBC thin client and peer client applications must use the new JAR file names, gemfirexd-client.jar and gemfirexd.jar, to access an upgraded distributed system.
- The JDBC connection string prefix has changed from jdbc:sqlfire to jdbc:gemfirexd.
- After you upgrade the GemFire XD software, all client applications, stored procedures, and listener implementations that used the SQLFire public API must be recompiled to use the new GemFire XD package names and libraries. Java package names beginning with com.vmware.sqlfire or com.pivotal.sqlfire were changed to com.pivotal.gemfirexd (see the Javadocs). ADO.NET application interfaces were changed to begin with Pivotal.Data.GemFireXD.
- The ADO.NET client DLL is now installed using an MSI file. See *Installing GemFire XD ADO.NET Components*.
- GemFire XD member startup scripts no longer support the -initial-heap and -max-heap properties. Modify any sqlf startup scripts to use the single -heap-size property instead.
- The "sqlfire." prefix that was required for certain configuration properties has changed to "gemfirexd." You will need to update any properties files or start scripts that use the old prefix.
- GemFire XD now uses a JMX Manager node to provide access to a federated MBean architecture. See *Using Java Management Extensions (JMX)* for information about the new architecture MBean names.
- The --SQLFIRE-PROPERTIES query hint has changed to --GEMFIREXD-PROPERTIES. You will need to modify any scripts that use the older hint.

See the post-upgrade instructions in the *Pivotal GemFire XD User's Guide* for more information.

> **Note:** GemFire XD is compatible with disk stores that were created with SQLFire 1.1.1 or later. GemFire XD automatically loads and upgrades these disk stores during the software upgrade procedure. GemFire XD is not compatible with persistence files (disk stores) that were created in versions of SQLFire earlier than version 1.1.1. You must upgrade these disk store files to version 1.1.1 or 1.1.2 before you can upgrade them to GemFire XD. See the *Pivotal SQLFire documentation* for upgrade instructions.

> **Note:** Thoroughly test your systems with the new 1.3 version before moving into production. Downgrading from 1.3 to a previous version of GemFire XD is not supported

Basic Features of GemFire XD

In addition to the new features highlighted above, GemFire XD provides these basic features:

- *Tables that are replicated or partitioned across the GemFire XD distributed system*
- *Redundancy for partitioned tables*
- *Server groups to control the placement of database objects, such as table replicas and partitions*
- *Data-aware stored procedures that can execute in parallel on multiple GemFire XD members*
- *JDBC peer client and thin client drivers*
- *Eviction with disk overflow for partitioned or replicated tables*
- *Persistent data dictionary*
- *Automatic rebalancing of data when GemFire XD servers are added*
- *Transactions with isolation levels TRANSACTION_NONE, and TRANSACTION_READ_COMMITTED, and REPEATABLE_READ*

- *Loaders for read-through caching of a datastore*
- *Listeners for responding to DML events on table data*
- *Writers for write-through caching*
- *AsyncEventListeners for asynchronous write-behind caching of a datastore*
- *Tools for monitoring cluster membership and analyzing queries*

Resolved Issues

The following key issues from the Pivotal bug tracking system were resolved in Pivotal SQLFire 1.3 and no longer require workarounds.

ID	Bugnote Title	Bugnote Description	Workaround for Previous Release
#41272	ResultSet.get methods using column names can return an incorrect column value if the same column name appears in more that one table in the projection.	If a ResultSet.getObject(), or other getter methods such as getInt()/getString() are used to pass the column name instead of index, and the column name appears more than once in the projection for more than one table, then the result set returned for one of the tables may not be correct.	If the same column name appears more than once in a projection for different tables, use ResultSet.get() methods and pass the column index in the projection.
#41456	Import with replace not implemented	The SYSCS_UTIL.IMPORT_TABLE system procedure imports data from a file into a table. The input parameter "replace=", which deletes all rows from the table before beginning the import, is not supported. Data rows are always added to the existing rows.	n/a

#41930	gfxd script does not support white spaces in member home directory	The gfxd script included in this version of the product does not support starting GemFire Members in a directory path that contains white spaces. For example:
		`$ mkdir my\ server`
		`$ cd my\ server`
		`$ gfxd server start`
		`Starting GemFireXD Server`
		` using multicast for peer`
		` discovery:`
		`239.192.81.1[10334]`
		`Starting network server for`
		` GemFireXD Server at address`
		`localhost/127.0.0.1[1527]`
		`Logs generated in /opt/`
		`Pivotal_GemFireXD_10_b45952/`
		`bin/my`
		`server/gfxdserver.log`
		`GemFireXD Server pid: 730`
		` status: stopped`
		`Error starting server`
		` process:`
		`IllegalArgumentException:`
		` Unknown argument:`
		`server/start_gfxdserver.log -`
		` See`
		`log file for details.`
			Specify a member directory path that does not include spaces.

ID	Bugnote Title	Bugnote Description	Workaround for Previous Release
#44006	Hang in primary rebalancing.	In rare cases, rebalancing primaries may lead to a hang in BucketAdvisor. becomePrimary on one of the members.	Stop the hung member and restart it.
#47869	Disk store upgrade from SQLFire to GemFireXD does not work	Disk stores created with SQLFire 1. 1.1 and 1.1.2 cannot be loaded into GemFireXD 1.0.x. The "upgrade-disk-store" command is not supported with SQLFire disk stores, and it does not upgrade those disk stores to a compatible format.	n/a
#49002	Errors reading HDFS write-only table log files	Running a MapReduce job or a PXF query against HDFS write-only table data fails if any log file in the table's HDFS store is larger than the HDFS file block size (specified during HDFS configuration as the dfs.block.size parameter). In this case, scanning any HDFS block other than the first block in the file fails.	When you create the HDFS store for the write-only table, ensure that the sum of the MaxWriteOnlyFileSize and BatchSize options is less than the HDFS file block size that you have configured (dfs.block.size setting for HDFS configuration).

#49185	Pulse fails to start for multiple users on the same machine	Trying to start the GemFire XD Pulse application may fail if another user has previously started Pulse on the same machine. This occurs because necessary folders in the /tmp directory remain and are owned by the previous user that started Pulse. The problem appears as the following exception:
		`[severe 2013/10/04 15:42:57.`
		`255 IST  <localhost-`
		`startStop-1>`
		`tid=0x35]`
		`(tid=11 msgId=4)`
		` LifecycleException`
		`java.io.IOException: Failed to`
		` create destination directory`
		` to`
		`copy resources`
		`        at org.apache.`
		`catalina.loader.WebappLoader.`
		`setRepositories(WebappLoader.`
		`java:891)`
		`        ...`
			n/a

ID	Bugnote Title	Bugnote Description	Workaround for Previous Release
#49295	Transactional insert operation may throw TransactionDataNodeHasDepartedException	During member initialization or HA failover, a transactional operation on a table with an HDFS store may throw a TransactionDataNodeHasDepartedException. This occurs because the HDFS queue is not fully initialized before a query is redirected to that node.	Catch this exception and retry the operation if necessary.
#49395	Concurrent delete operations by multiple threads causes "Delete did not happen on index"	When multiple threads perform concurrent delete operations on a table, a "Delete did not happen on index" error may occur. This problem has been observed when threads use DELETE FROM to delete all content from the table.	Use TRUNCATE TABLE instead of DELETE FROM to delete all content from a table.
#49406	HDFS queue limitation causes bottleneck	This version of GemFire XD supports only a single HDFS async event queue per set of colocated tables (or per single, non-colocated table). The single queue can become a bottleneck when numerous client threads perform HDFS operations. Bottlenecks are identified by queues that constantly overflow, which reduces performance.	As a workaround, run 2-3 JVMs on each server host to increase the number of HDFS queues available per each set of colocated tables (or individual, non-colocated tables) in the system.
#49899	AssertionError in versioning code	When executing remote operations to a non-persistent table, recreating a bucket might fail with "Assertion error creating bucket in region com.gemstone.gemfire. InternalGemFireError?: recordVersion invoked for a local version tag that is higher than our local version."	n/a
#49901	Gateway initialization may fail with "IllegalStateException: Value is no longer available" when using off-heap tables	When using off-heap tables with a gateway configuration, you may see a secondary gateway fail during startup with "IllegalStateException: Value is no longer available". This is a transient problem.	Restart the member that failed with this exception.

ID	Bugnote Title	Bugnote Description	Workaround for Previous Release
#49941	Insert operations hang after HDFS namenode and datanodes become unreachable	Network failures may lead to unreachable HDFS namenode and datanode hosts. When this occurs, HDFS clients such as GemFire XD wait for the socket timeout (an HDFS client configuration property) before failing a write operation. Concurrent user insert or read requests fail one by one, after each request fails to establish a connection. If new requests continue to arrive, the time that it takes to report a failure continues to increase.	n/a

ID	Bugnote Title	Bugnote Description	Workaround for Previous Release

#49942	Client reports SQLException after namenode loss	If an HDFS write operation is active during at the time a namenode fails due to a broken network connection, the client closes local file system instances. In this case, a user of GemFire XD sees a SQLException caused by the HDFS error:
		`(java.io.IOException: Filesystem closed). Execute SQL`
		`statement failed with: X0Z30 java.sql.SQLException(X0Z30): An error`
		`was received from HDFS while performing 'handleException'. at`
		`com.pivotal.gemfirexd.internal.iapi.error.StandardException.newException(StandardException.java:202)`
		`at`
		`com.pivotal.gemfirexd.internal.engine.Misc.wrapSQLException(Misc.java:912)`
		`at`
		`com.pivotal.gemfirexd.internal.engine.Misc.wrapRemoteSQLException(Misc.java:897)`
		`at`
		`com.pivotal.gemfirexd.internal.engine.distributed.utils.GemFireXDUtils.processCancelException(GemFireXDUtils.java:1952)`
		`at`
		`com.pivotal.gemfirexd.internal.engine.distributed.ResultHolder.processException(ResultHolder.java:1535)`
		`at`
		`com.pivotal.gemfirexd.internal.engine.distributed.ResultHolder.hasNext(ResultHolder.java:1480)`
		`at`
		`com.pivotal.gemfirexd.internal.engine.sql.execute.GemFireDistributedResultSet$SequentialIterator.(GemFireDistributedResultSet.java:594)`
		`at`
			n/a

ID	Bugnote Title	Bugnote Description	Workaround for Previous Release

		`com.pivotal.gemfirexd.internal.engine.sql.execute.GemFireDistributedResultSet$SequentialIterator.(GemFireDistributedResultSet.java:573)`
		`at com.pivotal.gemfirexd.internal.engine.sql.execute.GemFireDistributedResultSet.getIterator(GemFireDistributedResultSet.java:413)`
		`at com.pivotal.gemfirexd.internal.engine.sql.execute.GemFireDistributedResultSet.setup(GemFireDistributedResultSet.java:202)`
		`at com.pivotal.gemfirexd.internal.engine.distributed.message.GfxdFunctionMessage.executeFunction(GfxdFunctionMessage.java:308)`	

#49943	DiskRegionStats.entriesOnlyOnDisk may show an incorrect value	After persistent recovery from disk, the entriesOnlyOnDisk statistic may indicate that entries are still on disk, even though the entries are actually in memory.	n/a
#49987	HDFSIOException during rebalancing	A query that runs concurrently with a rebalance operation on an HDFS-persistent table may receive an HDFSIOException.	Retry the query after rebalancing is complete.
#49989	Member fails to start with NullPointerException or assertion error when syncing concurrent transactions	A new GemFire XD member will sometimes fail to start with a NullPointerException or assertion error if there are concurrent transactions taking place in the distributed system.	Restart the failed node.

ID	Bugnote Title	Bugnote Description	Workaround for Previous Release
#50116	Cluster restart fails when replaying DDL commands against parent/child tables	If you execute certain ALTER TABLE DDL commands against tables in a parent-child relationship, then restarting the GemFire XD system can fail while replaying the DDL commands. This problem has been observed in testing with the following sequence of events: **1.** Tables "parent" and "child" exist, with "child" having a foreign key to a column in "parent." **2.** A copy of the parent table, "newparent," is created by selecting all columns from "parent" and then using ALTER TABLE on "newparent" to re-creating the same constraints. **3.** ALTER TABLE is used with the "child" table to change foreign key relationships to the "newparent" table. **4.** The original "parent" table is dropped from the schema. **5.** All GemFire XD members are shut down and restarted. With this sequence of events, restarting the cluster can fail during DDL replay when recreating the "child" table because the original "parent" table no longer exists.	When duplicating a parent table in this manner, do not drop the original table to avoid failures in DDL replay

Known Issues

OUT OF MEMORY ERROR WHEN STARTING SERVER MEMBER WITH JMX MANAGER

An out of memory error can occur when using the PermGen heap if you attempt to start a JMX manager with a GemFire XD server member. This occurs because of a mismatch in Java versions between GemFire XD and Hadoop nodes.

To avoid this problem, follow the best practice of starting a JMX manager only on GemFire XD locator members, rather than on server members.

GEMFIRE XD ISSUE SUMMARY

The following key issues have been registered as bugs in the Pivotal bug tracking system:

ID	Bugnote Title	Bugnote Description	Workaround
#39408	The CREATE TABLE statement does not support null columns in unique or primary key multicolumn constraints.	You cannot create a unique or primary key constraint on a column that contains null values.	n/a

ID	Bugnote Title	Bugnote Description	Workaround
#40656	GemFire XD does not support sharing a single connection with multiple threads.	Sharing a connection with multiple threads can lead to different kinds of problems in GemFire XD. In particular if an operation is performed by one thread and a commit or rollback operation is performed by another thread, then a "LockNotHeldException" may be thrown during commit/rollback.	n/a
#40919	Batch conflation can cause data mismatch when synchronizing to an external database or WAN site.	If batch conflation is turned on, operations may be overridden while they are being applied to an external database (with DBSynchronizer) or propagated to a remote WAN site.	Disable BATCH_CONFLATION if you configure an AsyncEventListener or a WAN gateway queue.
#41279	If a DDL is executed from a node and it goes down while sending the DDL commit message to other nodes in the system, the DDL may have been executed on some nodes but not on others.	In rare scenarios, if the node firing a DDL fails while sending a DDL commit message to other nodes, the commit may go to some nodes but not to others. DDL may execute successfully on some nodes but not on others.	You may need to manually restart the nodes that have inconsistent data. If those nodes cannot be easily determined, one option is to leave one locator running and bring down other nodes, so that on restart all nodes will sync the DDLs against that locator.
#41706	HA feature of DBSynchronizer may cause underlying DB to go out of synch due to re-application of DML operations.	A ramification of high availability is that a Data Manipulation Language (DML) operation may get reapplied to the database in case of failover of the node hosting primary DBSynchronizer. When a primary DBSynchronizer node fails while applying the operations of the batch, some DMLs of the batch may have already been applied. On failover, the new primary DBSynchronizer node would apply the failed batch, in the process reapplying some of the DML operations. In such cases, the database may or may not get out of synch with the GemFire XD system, depending upon the nature of the DML operation, how it modifies the columns, and the presence or absence of constraints.	To minimize the occurrence of this issue, have tables with primary key constraint defined.

ID	Bugnote Title	Bugnote Description	Workaround
#41737	DBSynchronizer does not work for "SELECT ... FOR UPDATE"	DBSynchronizer does not work for "SELECT ... FOR UPDATE" statements because the update gets translated into "UPDATE ... WHERE CURRENT OF <cursor>."	n/a
#41738	WHERE CURRENT OF clause not supported.	GemFire XD does not support the WHERE CURRENT OF clause with an UPDATE or DELETE statement to perform positioned updates with updateable cursors.	Construct an explicit UPDATE or DELETE statement using a primary key filter. Or, use SELECT ... FOR UPDATE with the GemFire XD peer driver to obtain an updateable result set.
#42013	Parameter inference might interpret a bind value with different precision.	Parameter type inference in an expression is based on the net possible highest value that can be returned from that expression instead of the type of target column value. This might result in parameter value truncation. For example, consider the following query: `update trade.portfolio set subTotal = ? * qty.` If the qty column is of an Integer type and subTotal is a decimal(30,20) type, then GemFire XD expects the parameter in the expression to be an integer. Performing stmtsetDecimal(1, 10.03) truncates the parameter value to 10.	Use explicit cast operator for the parameter (?) to higher precision: `update trade.portfolio set subTotal = Cast(?) as decimal(30,20) * qty`
#42019	GemFire XD does not support setting or getting procedure parameters by name to CallableStatement.	Trying to get or set parameters using the parameter name throws an unimplemented exception in GemFire XD (SQLState 0A000).	Get or set parameters by number rather than by parameter name.
#42035	On a Windows system, if the home directory of a user is on a share, in some cases the sqlf tool may fail to start and cannot create .sqlf.history in user's home.	This issue has been observed in Windows environments where the sqlf tool is running inside VMware Fusion and using a share as user home directory. When sqlf tries to create the .sqlf.history file in the user home directory, it fails and the sqlf tool fails to start.	Disable the sqlf history by setting ".sqlf.history" Java system property to empty string, or change the home directory to not point to a share.

ID	Bugnote Title	Bugnote Description	Workaround
#42075	DBSynchronizer and WAN sites may go out of synch due to foreign key constraint violation.	When a child table has a foreign key relationship with the parent table, it is possible that a sequence of inserts into the parent and child tables in the GemFire XD system may reach the DBSynchronizer or WAN site in reverse order. This results in the child table record being inserted before the parent record in the WAN site or external DB, which causes a foreign key constraint violation. This situation can occur if insertion into parent and child tables is executed by separate threads. There is a window in which a DML operation executed on the GemFire XD system has not yet been placed into the internal queue of the WAN or DBSynchronizer. The record is guaranteed to be put into the internal queue of the WAN or DBSynchronizer only after the DML operation is complete.	To avoid this situation, the insertion into the parent and child tables should be done by the same thread.
#42307	DDLs do not get relayed to the DBSynchronizer.	DDLs like a truncate table statement are not relayed to DBSynchronizer. Truncate table removes all the rows in the table. When executing a truncate table statement, AsyncEventListener callback will not be invoked.	n/a

ID	Bugnote Title	Bugnote Description	Workaround
#42400	No total ordering guarantee for DML in separate threads.	GemFire XD preserves the order of DML statements applied to the distributed system (and queued to AsyncEventListeners or remote WAN sites) only for a single thread of execution. Updates from multiple threads are preserved in first-in, first-out (FIFO) order. Otherwise, GemFire XD provides no "total ordering" guarantees. This means that if two separate threads concurrently update rows on a parent table and child table, respectively, the order in which GemFire XD queues those updates to an AsyncEventListener or WAN gateway may not match the order in which the tables were updated in the main distributed system. This mismatch can cause a foreign key constraint violation in a backend database (for example, when using DBSynchronizer) or in a remote WAN system that does not occur when the tables are initially updated. These types of "out of order" updates do not occur when multiple threads concurrently update the same key of a partitioned table.	An application should always use a transaction for any operation that updates multiple rows.
#42531	GemFire XD does not prevent a subsequent node with a different setting of sqlfire.sql-authorization from joining the cluster.	A locator booted with sqlfire.sql-authorization property set to true could allow a new node to join the cluster even though its sqlfire.sql-authorization is set to false.	sqlfire.sql-authorization property should be set consistently in the application.
#42538	Presence of triggers that manipulate rows can cause data inconsistency across WAN sites.	If triggers are defined on tables configured for WA N, the DML generated as part of trigger action is also propagated to the remote WAN site. Thus the receiving WAN site will receive DML triggers twice. First the trigger is executed by the incoming user-initiated DML operation. The second trigger is the implicit trigger DML (caused by the trigger on WAN site1) getting executed. An example of a trigger action that can cause a problem is of the form , Col1 = Col1 +1. This will cause the Cl1 value to be incremented twice on the receiving WAN site.	Do not create triggers on tables configured for WAN propagation. Avoid trigger action that modifies the column value relative to its existent value.

ID	Bugnote Title	Bugnote Description	Workaround
#42564	If a user's access right to a database object is continuously granted and revoked, the user may not have the last assigned privilege.	It is possible in some cases a user may not have the last assigned privilege, if the user's access right to a database object is continuously granted and revoked.	Limit the frequency of granting and revoking a user's privilege to a database object.
#42672	Inconsistent behavior of conflict exception (SQLState: X0Z02) thrown by product in transactions for parent and child row updates having foreign key relationship	SQLFire may throw a conflict exception (SQLState: X0Z02) when a parent table entry is being UPDATED in a transaction and a child table is inserting a row having a foreign key reference to that parent row. However, for cases when the primary key is not the same as the partitioning key in the parent table, GemFire XD may not throw a conflict exception.	Applications should not treat conflicts as unexpected during transactions when a parent table is being updated while inserts are in progress on the child table having a foreign key reference to the parent.
#42718	Transactions not closed with CLOSE_ CURSORS_AT_ COMMIT	GemFire XD does not support closing a transaction when the connection holdability is set to CLOSE_CURSORS_ AT_COMMIT.	n/a
#42803	GemFire XD does not support ResultSets with holdability as HOLD_ CURSORS_ OVER_COMMIT .	Creating a JDBC Statement with ResultSet holdability as ResultSet. HOLD_CURSORS_OVER_COMMIT is not supported in GemFire XD although the product does not throw an exception during creation of such a statement.	n/a
#42814	GemFire XD does not support the ESCAPE keyword in the LIKE clause.	The ESCAPE keyword in the LIKE clause is currently unsupported in GemFire XD although it does not throw an exception.	n/a
#43096	GemFire XD primary member failures can lead to data inconsistency issues.	During primary to secondary event propagation, the failure of the primary GemFire XD member can cause table and global index data inconsistency issues.	n/a

ID	Bugnote Title	Bugnote Description	Workaround
#43097	A DML operation that gets Offline Exception X0Z09 may not have failed.	In a system with a persistent partitioned table, if the system's nodes are being brought down (or a node fails) and DML operations are occurring concurrently, the DML operation may throw SQLException with SQLState X0Z09. In such cases there will be uncertainty about the outcome of the operation. The operation may or may not have persisted the data. An application should not assume that the DML operation failed, because it is possible that the operation was persisted before the exception was thrown. In this case, the data would be recovered from the persistent table after the datastore is rebooted.	In the event of graceful shutdown, stop the nodes only when all the DML operations have terminated. In case of SQLException with state X0Z09 with node failures, verify whether the data got persisted successfully, and accordingly the application can re-execute the operation or not.
#43188	Updateable result sets not supported JDBC client driver.	You cannot use updateable result sets when you are connected with the GemFire XD JDBC client driver. Updateable result sets are supported only when using SELECT ... FOR UPDATE with the GemFire XD JDBC peer driver.	n/a
#43214	Unexpected NoDataStoresAvailable exception.	In rare cases, an operation receives a NoDataStoreAvailableException while recovering a failed member with persistent tables, even though the operation should succeed with other running data stores.	n/a
#43232	DDLUtils cannot import tables having circular foreign key dependencies.	Circular foreign key dependencies are not yet supported in GemFire XD.	n/a
#43261	Inconsistency between client driver and peer driver when displaying boolean column values.	The peer driver displays boolean column values as true or false while the client driver shows them as 1 or 0.	n/a
#43863	Query plan sometimes omits peer client member id.	Query plans for statements executed from peer clients can sometimes substitute the statement id for the peer client member id in the portion of the query plan that was executed on the peer client itself.	n/a

ID	Bugnote Title	Bugnote Description	Workaround

#43991	CAST may not be honored in aggregates.	In queries having aggregates, the CAST operator on the aggregated value may not be honored by GemFire XD in some cases. For example:
		`select cid, cast(max(price) as decimal(30,1))`
		`from trade.txhistory  where tid =? GROUP BY cid, type`
		against table txhistory with datatype of price decimal(30,20) gives the second value as decimal(30,20) instead of decimal (30,1).
			n/a

#44074	When using DBSynchronizer, if multiple potentially overlapping DMLs are executed in different transactions executing concurrently, then they may result in potentially different rows being affected in GemFire XD and backend DB.	In a situation where multiple DMLs are executing concurrently that potentially overlap in GemFire XD but do not actually conflict, then replay of those DMLs using DBSynchronizer may lead to different rows being affected in the backend DB as compared to GemFire XD. As an example consider a bulk delete that uses some criteria on column and an update that changes the same column:
		`update foo set col1=val1 where col1=val0;`
		`delete from foo where col1= val1;`
			n/a

#44110	DBSynchronizer and automatic generation of identity columns in third party database	If cached tables require automatic generation of identity column values, then you should use GemFire XD to automatically generate the key values but disable generated keys in the backend database. If the backend database also generates key values, then inserts can fail if GemFire XD has already generated the key.	Disable auto generated keys in the backend database.
#44199	SECMEC_ USRSSBPWD is not supported with the GemFire XD thin client driver.	While connecting via a thin driver, GemFire XD currently does not work properly with Strong Password Substitute DRDA security mechanism.	n/a
#44207	Non-prepared stored procedures perform slower than prepared statements.	Only insert, update, delete, and select statements with constants are considered for pattern matching, and avoid complete compilation. Stored procedures with constant parameters (without a prepare call) may be slow due to complete compilation (parsing and optimization).	Configure the stored procedure with dynamic parameter and bind constant values across multiple executions.

ID	Bugnote Title	Bugnote Description	Workaround
#44261	If a DML operation is cancelled due to low memory with SQLState XCL54, the DML operation may have been successful before the low memory condition is hit.	In some cases it is possible that DML operations fail with low memory condition (SQLState: XCL54), but the operation may still have completed when the exception is thrown. Users should not assume that a low memory exception always implies that the operation was unsuccessful. Check for the changes explicitly, if required.	Use transactions when it is expected that DMLs may take substantial memory and can get cancelled due to low memory conditions.
#44534	Index Creation with Concurrent DML on WAN Causes Incorrect Result Sets	In a WAN configuration, if you attempt to create an index on a table while performing concurrent DML statements, queries against the table return incorrect result sets.	n/a
#44414	Unexpected ConflictingPersistentDataException after recovering from multiple failures	If multiple members crash during initial bucket creation or after recovering from a "gfxd shut down all" command, there is a possibility that those members will throw a ConflictingPersistentDataException on restart.	n/a
#44657	Some MBeans not available in JMX Agent.	Certain GemFire XD MBeans are not served from the GemFire XD JMX Agent in this release. These MBeans include: • JDBCMBean for monitoring the JDBC driver version information. • ManagementMBean for monitoring the management status. • NetworkServerMBean for monitoring statistics associated with a GemFire XD member's Network Server (client) connections. • VersionMBean for monitoring the GemFire XD member version and product information.	To access any of the listed GemFire XD MBeans, you must connect directly to the local GemFire XD process rather than to the GemFire XD Agent. For example, with jconsole you would need to connect directly to the local process id of the GemFire XD member (such as com.vmware.sqlfire.tools.internal .SqlfServerLauncher) rather than connecting to the Agent as a remote process.
#44711	Double byte (DBCS) characters in the gfxd command line display as ??? in Windows	Double byte characters are currently not supported in the gfxd command-line for Windows platforms. Many double byte characters do not work correctly when used in diskstore paths, depending on the OS filename or directory name limitations.	n/a

ID	Bugnote Title	Bugnote Description	Workaround
#44793	Carriage return not recognized in gfxd	A carriage return/line feed character (\n\r) is not recognized in the gfxd command line utility. For example, a comment line is terminated by a new line. But a \n won't be recognized in the prompt. SELECT TABLENAME from -- GEMFIREXD-PROPERTIES statementAlias=My_QUERY \n SYS. SYSTABLES;	n/a
#44804	TotalExecutionTime statistics might show incorrect values.	TotalExecutionTime statistics might show a large value sometimes.	n/a
#44814	java.lang. ClassCast Exception: FormatableBitSet cannot be cast to com.vmware. sqlfire. internal. catalog.types. ReferencedColumns DescriptorImpl	A very rare ClassCastException saying FormatableBitSet or GenericPreparedStatement cannot be cast to com.vmware. sqlfire.internal.catalog.types. ReferencedColumnsDescriptorImpl is seen when preparing a query on remote node. Users will need to re-execute the statement when this exception occurs.	n/a
#44872	No query plans for bulk insert/update/ delete with or without subqueries affecting multiple rows.	Query plan may not be returned for a bulk insert/update/delete operation that gets distributed to one or more members.	n/a
#44878	A system user's credentials may fail to start a new GemFire XD node.	When the first locator is booted, a list of username/password credentials could be set. These credentials could be used to boot new GemFire XD nodes to join in the cluster. When booting new nodes using these credentials, sometimes they may be rejected by GemFire XD.	Application could set only one username/ password pair to start all locators and nodes in the cluster.
#44953	HA failover can cause false duplicate key exception with peer driver	If a GemFire XD member fails when the peer driver is inserting data to that member, the peer driver connection retries the operation with another member. In some cases, GemFire XD throws a duplicate key exception when retrying the operation, even though the key has not been inserted. This problem has been observed with the peer client driver when performing INSERT statements that use a subselect.	n/a

ID	Bugnote Title	Bugnote Description	Workaround
#45071	Scroll Insensitive not supported for 'select for update' statements	Scroll Insensitive result sets should not be used with 'select for update' statements as it is not supported.	n/a

#45401	Transitive uniqueness constraints can throw constraint violations	Transitive uniqueness constraints can throw constraint violations in GemFire XD, as with:
		`CREATE TABLE T1 (COL1 INTEGER, COL2 INTEGER);`
		`CREATE UNIQUE INDEX I1 ON T1(COL1,COL2);`
		`INSERT INTO T1 VALUES (1, 2);`
		`INSERT INTO T1 VALUES (2,1);`
		`UPDATE T1 SET COL1 = COL2, COL2 = COL1;`
			n/a

#45544	Insert from thin client into table with auto generated primary key column may insert multiple rows	An insert that is executed from a thin client can result in multiple rows being inserted if member failures occur, and if the table has a primary key defined with GENERATED BY DEFAULT identity. This problem can occur if the insert was successful in the first attempt, but the member fails before responding to the thin client. In this case, the thin client retries the operation with a new server, and since the primary key is autogenerated it results in a new id generated for the new insert.	n/a
#45696	Insert operation reapplied to table having no primary key or unique constraint	During DBSynchronizer/WAN failover, an insert operation may be reapplied to a table that has no primary key or unique constraint defined.	n/a
#45864	NullPointerException when calculating average of BIGINT values	GemFire XD throws a NullPointerFunction if you use the AVG() function to compute the average of values for a column of BIGINT datatype.	n/a
#45942	Complex join/ view queries not supported	GemFire XD does not support complex join queries that involve views. For example, a query with view and an outer join may fail.	n/a

ID	Bugnote Title	Bugnote Description	Workaround
#46179	Insert with sub-select from SYS. MEMBERS distributes to only one node in the system	When using insert with sub-select from distributed virtual tables like SYS. MEMBERS, the insert runs only on the connected node, and selects member information only for that node instead of for all members in the distributed system. For example: insert into m select kind from sys.members; inserts only the KIND column for the current server on which the insert executes.	Use custom code to iterate over the result of distributed virtual tables like SYS.MEMBERS and insert rows from the result. For example: `select * from sys.members` then iterate over the results and insert individually or in batch into the table as required.
#46311	Incorrect column name in DAP when executing a Query to populate a result set	GemFire XD does not provide the correct column name in a data-aware procedure when executing a Query to populate a result set. Instead, the column names are auto-generated (for example, C1,C2, and so forth).	n/a
#46398	A listener event may have a different "last" value than the table's last updated value	Listeners can receive events in a different order than the order in which they were applied to a table. For example, events that come from remote members may be dispatched to listeners before a concurrent local update event is dispatched. This is the expected behavior for listeners in GemFire XD.	n/a
#46485	Executing queries after a low memory event may still cause out of memory exception	After a low memory event occurs, query execution is canceled and all queries are rejected from executing. However, incoming queries continue to create query objects before being rejected. These query objects can cause an out-of-memory exception.	Tune the critical heap percentage and/or throttle the number of queries, especially after a query has been canceled due to a low memory condition.
#46519	No statement matching for procedure invocations	GemFire XD does not perform statement matching for procedures. Each unprepared statement that is invoked in a procedure creates a separate execution plan, which can potentially slow performance and consume memory in the statement cache.	n/a
#46541	EXPLAIN exception for insert with subselect from same table	The EXPLAIN command does not work for queries such as 'INSERT INTO T1 SELECT * FROM T1' in the case of partitioned tables.	n/a

ID	Bugnote Title	Bugnote Description	Workaround
#46758	IMPORT_TABLE_EX with many threads but few rows returns 'table not found'	When import_table_ex is called on more threads than the total rows in table, then each idle thread that does not get a row to import throws the "table not found" error message.	Specify a number of threads that is less than the total number of rows in the table.

#46792	BucketMovedException with select MAX() over empty, partitioned table	Certain queries that perform a table scan of an empty, partitioned table can fail due to GemFire XD scanning buckets on a related partitioned table in the query. For example, consider the following tables and queries:
		`gfxd>  create table t1 (c1 int, c2 int)`
		`> partition by column(c1);`
		`gfxd>  create table empty (c1 int, c2 int)`
		`> partition by column(c1);`
		`gfxd>  insert into t1 values`
		`> (null,null),(1,1),`
		`(null,null),(2,1),(3,1),`
		`(10,10);`
		`gfxd>  select max((select c1 from empty))`
		`> from t1;`
		The query should return null. Instead, GemFire XD hangs and logs the following exception:
		`[info 2013/09/11 11:20:32.525 EDT SQLF:  tid=0x44]`
		`(XID=`
		`154(UserTransaction)@463505cb;txState=`
		`null;`
		`supportsTX=true),`
		`(SESSIONID = 3), (DATABASE = sqlfire), (DRDAID = null),`
		`Failed Statement is:`
		`select max((select c1 from empty)) from t1`
		`com.gemstone.gemfire.internal.`
		`cache.execute.`
		`BucketMovedException:`
		`    Bucket migrated to another node. Please`
		`    retry.[buckedId=1,region=/`
		`APP/EMPTY]`
		`[...]`
			This type of query is not supported for partitioned tables. The problem does not occur with replicated tables.

#46796	SUM(...) on partitioned region in multi-node can throw exception if one server has only a row of NULLs	A SUM(...) function on a partitioned table in a multinode cluster may throw an exception if one of the members has only null rows.	n/a

ID	Bugnote Title	Bugnote Description	Workaround
#46802	Dropping a partitioned trigger table on a multinode cluster can throw a rollback exception and disconnect if the target trigger table is dropped before the trigger	If the trigger and target trigger tables are partitioned and hosted on a multinode cluster and the target trigger table is dropped before dropping the trigger on trigger table, then subsequent DML operations on the trigger table followed by a drop of trigger table may cause a Rollback exception and disconnection from the system.	Always drop triggers before dropping the target trigger tables.

#46804	Delete on a default partitioned table with a subselect on a partitioned table throws AssertionException	For a case with tables such as:
		`create table t1(c1 int)`
		`partition by column(c1);`
		`create table insert_test(c1`
		`int, c2 int, c3 int);`
		If a delete is fired such as:
		`delete from insert_test where`
		`c1 =`
		`(select 9 from t1 a join t1 b`
		`on a.c1 = b.c1 where a.c1 =`
		`1);`
		The delete statement throws Exception.
			If you want a table such as insert_test to be partitioned, provide an explicit partitioning column. For example the insert_test table should be created as:
			`create table`
			`insert_test`
			`(c1 int, c2`
			`int, c3 int)`
			`partition by`
			`primary key;`

#46811	select sum(distinct) may throw a ClassCastException when used in a subquery	A query of the following format throws a ClassCastException:
		`select col1 from z where col1`
		`in (select`
		`sum(distinct col2)`
		`from x group by col2);`
		The error is:
		`java.lang.ClassCastException:`
		`com.pivotal.gemfirexd.`
		`internal.iapi.types.`
		`SQLInteger cannot`
		`be cast to com.pivotal.`
		`gemfirexd.internal.iapi.`
		`types.UserDataValue`
			Cast the sum(distinct) expression to the appropriate datatype. For example:
			`select col1`
			`from z`
			`where col1 in`
			`(select`
			`cast(sum(distinct`
			`col2)`
			`as int)`
			`from x group`
			`by col2);`

#46816	Dropping a global temporary table throws rollback exception in multi node cluster	When more than one datastore is configured, then dropping a global temporary table throws a rollback exception. This problem does not occur with a single datastore.	n/a

ID	Bugnote Title	Bugnote Description	Workaround

#46818	Subselects in projection and FROM clause throw exception	A query such as:
		`select (select i from t where i = 2) *`
		`  (select s from t where i = 2) from s`
		`  where i > (select i from t where i = 0) -`
		`  (select i from t where i = 0);`
		throws an exception, because the subselect in a projection is not supported.
			Do not use sub-selects in a projection or FROM clause, as this is an unsupported feature.

#46865	Error with left outer joins and views	Left outer joins with views are not supported in GemFire XD, and will throw unsupported or assertion errors.	n/a
#46890	Error with create_ user and NULL password	A NULL password passed to create_user causes exception but also disconnects the existing connection.	n/a

#46981	Aggregate queries can grow out of the valid range of the Long datatype	During the calculation of average value in a query, the sum aggregate of all values in the column may go out of the valid range of the Long datatype. This causes the exception:
		`22003:The resulting value is outside the range`
		`for the data type <datatypeName>`
			If you experience this problem, explicitly cast the Long field.

#47199	Server fails to start with SQLException(x0768)	When multiple add/drop operations are performed on the same index, procedure, or function, a server may fail to start with the error "Caused by: java.sql.SQLException(X0Y68): PROCEDURE 'procedure_name' already exists".	Restart the server.
#47289	ALTER TABLE limitation for foreign key constraint	If a table has already contained data values, GemFire XD does not support adding a foreign key constraint with ALTER TABLE in cases where the reference key has a unique column and is a superset of the partitioning column.	Add foreign key constraints to a table in the original DDL, or before you populate the table with data.
#47387	Unable to execute SYSIBM or internal procedure	GemFire XD restricts access to certain internal procedures and SYSIBM schema procedures that are fully permitted by Apache Derby.	n/a

ID	Bugnote Title	Bugnote Description	Workaround
#47389	Update count incorrect after HA retry	If an update operation is retried due to a member failure, then the update count may return a number that is less than the actual number of updated rows. This occurs because GemFire XD only returns the update count of the second, retried operation; updated rows that completed before the member failure are not included in the update count.	n/a
#47419	Issues with GENERATED BY DEFAULT identity columns	If a generated column value reaches its max value, then by default GemFire XD automatically rolls over and begins using the initial (minimum) value for column values. However, this rollover behavior is not supported for BIGINT columns. Also, if GemFire XD retries an operation due to a member failure, then the identity column may be incremented more than once for the same operation.	n/a
#47435	DBSynchronizer may be slow to drain	Connectivity issues with the underlying database can cause DBSynchronizer to be very slow in applying events.	Check connectivity with the underlying database.
#47449	Single-hop query may hang in WAN environment	When a server goes down, a GemFire XD client read operation may be unable to detect the failure, which results in a hang.	Use the "read-timeout" connection property to fail the connection after the specified amount of time. This enables GemFire XD to retry the connection on a new server automatically.
#47687	A JDBC RowLoader may not be invoked if a query uses a CAST in the projected columns	If a query contains a CAST operator in the projection, then GemFire XD invokes the query as a generic scan rather than as a point lookup, and in this case any configured RowLoader is not invoked.	If a RowLoader must be invoked for the query, perform explicit conversion in the Java code that invokes the query, rather than using CAST in the SQL statement.
#47809	Batch inserts may fail during HA failover	GemFire XD member failures that occur during a batch insert may cause a java.sql.BatchUpdateException to be thrown. In this case, the insert operation(s) fail and it is not possible to determine which inserts completed and which did not.	Apply inserts within the scope of a transaction to ensure that either all rows are inserted or none are inserted.

ID	Bugnote Title	Bugnote Description	Workaround
#47947	Sub-select in INSERT may fail to insert all records	Testing has shown that INSERTs with sub-select statements sometimes fail to insert all records.	Retry the insert operation or execute the insert within the scope of a transaction.
#48166	ClassCastException with com.vmware. sqlfire.internal. catalog.types. ReferencedColumnsDescriptorImpl	In rare circumstances, executing a query may cause: ClassCastException: com. vmware.sqlfire.internal.catalog.types. ReferencedColumnsDescriptorImpl cannot be cast to com.vmware. sqlfire.internal.iapi.store.access. StaticCompiledOpenConglomInfo.	Re-execute the query.
#48701	Procedures may not honor invoker permissions	GemFire XD stored procedures may not honor the permission lists of a user who invokes the procedure.	n/a
#48769	Procedure, triggers, functions don't inherit the rights of user that created them	Procedures, triggers, and functions are executed using the rights of the invoker rather than the rights of the user who defined the procedure, function, or trigger. In other words, the CURRENT_ USER permissions list is checked instead of the permission list of the user who defined the object being executed. This can lead to unexpected problems such as when a USER1 creates a table, "sample," and then USER2 creates an after update trigger of USER1.sample that selects a new column value (col1) from USER1.sample and places it into a new table. In this case, a USER3 may have read access to the trigger itself, but executing the trigger would cause a permission violation if USER3 had no read access to USER1.sample (col1), which the trigger selects after an update.	Grant explicit read/ write access to invoking user(s) for any dependent objects used in the body of a procedure, function, or trigger.
#48824	Import table performance with off-heap tables	Performance testing has shown a degradation in performance for SYSCS_ UTIL.IMPORT_TABLE operations on tables that are configured for off-heap storage.	n/a
#48904	ClassCastException: PARTITION BY RANGE (SYMBOL): Cannot have overlapping ranges	When creating a table that is partitioned by range, if the ranges are not well defined then GemFire XD throws the error ClassCastException: PARTITION BY RANGE (SYMBOL): Cannot have overlapping ranges. This error may appear as an internal issue, but it indicates a failure in the provided partitioning range.	Review and correct the partition range provided in the CREATE TABLE statement.

ID	Bugnote Title	Bugnote Description	Workaround
#49195	PXF Driver performance with multiple HDFS home directories	The GemFire XD loner system that is started by the PXF driver is not optimized to access multiple HDFS store home directories. In HAWQ, performance is negatively impacted when accessing a second HDFS home directory, because the PXF driver restarts the loner instance.	n/a
#49295	TransactionDataNodeHasDeparted Exception during transactional insert	During member initialization in HA scenarios, a transactional insert operation on a table with HDFS store may throw TransactionDataNodeHasDepartedException. This occurs when the HDFS queue is not fully initialized before the operation is redirected to the GemFire XD member.	Catch this exception and retry the operation.
#49325	Multiple DELETE FROM TABLE statements cause ASSERT FAILED error	Executing multiple, concurrent DELETE FROM TABLE statements to delete all content from a table can cause an ASSERT FAILED error with the message: Delete did not happen on index *index-name*.	Use TRUNCATE TABLE to delete all content from a table instead of using DELETE FROM TABLE.
#49399	Bulk insert to table with a large dataset hangs	Sometimes a bulk insert to a large table having a global index may hang due to an internal deadlock.	Set conserve-sockets=false and re-try the insert. Do not use conserve-sockets=true.
#49421	HDFS table row counts are not available in JMX	The JMX and Pulse interfaces cannot estimate the size of an HDFS table. Additionally, HDFS tables may contain data from non-GemFire XD sources. This is an expected limitation for HDSF tables.	n/a
#49611	Offline exception X0Z09 during insert	Inserting a row to a table that has foreign keys (which involve a remote index) and unique constraints may cause offline exception X0Z09.	Use a transaction with REPEATABLE_ READ isolation level to perform the operation
#49667	Exception during rollback	An exception may occur during rollback for a failed delete operation from a thin client connection. This problem has been observed during HA failover scenarios.	This exception may be ignored. Retry the failed operation
#49681	Delete operations may fail to delete data on an member that is starting up	If a GemFire XD member is in the process of starting up in the distributed system, then data may not be deleted if member recover has not begun at the time when the delete operation reaches the member. This issue is only applicable to non-persistent tables.	Bring new members online at a time when the system is not expected to perform delete operations.

ID	Bugnote Title	Bugnote Description	Workaround

#49716	If a commit fails due to a GemFire XD member failure (SQLStates 40XD0 and 40XD2), closing the connection may also fail	If a commit fails due to a GemFire XD member failure (SQLState error 40XD0 or 40XD2), then closing the connection may also fail.	To close a connection that has failed due to node failure in a transaction (SQLStates 40XD0, 40XD2), roll the transaction back and then ignore any exception in close, because the immediate server may have failed. For example:
			`try {`
			` conn.rollback();`
			` conn.close();`
			`} catch (SQLException sqle) {`
			` // log failure if required`
			`}`

#49727	Client may receive node failure exception even if transaction commits	If the connection between a GemFire XD client and server is broken, it is possible that the client may receive a node failure exception even thought the client's transaction successfully commits on the server. There is currently no way for the client to determine the state of the transaction after such a failure.	n/a
#49747	"Delete did not happen on index" or "An index maintenance exception was thrown" during member recovery	Testing has shown that "Delete did not happen on index" or "An index maintenance exception was thrown" errors can sometimes occur while performing index maintenance operations in certain HA failover situations.	n/a

ID	Bugnote Title	Bugnote Description	Workaround
#49748	Query of WAN-replicated table on a receiving site hangs	In a WAN configuration, any WAN-replicated table on a receiving site must have the same schema as the associated table on the sending site. There can be several problem scenarios if two sites attempt to use WAN replication to replicate tables having different schemas. One problem scenario is as follows: **1.** Site-2 has a table with one extra column, compared to an associated table in site-1. **2.** A row is inserted on site-1 which is then replicated to site-2. **3.** A select query of the table on site-2 will hang.	All tables which are enabled for WAN replication must have same schema on all GemFire XD sites.
#49760	Assertion failure for ListOfDeltas during index recovery	The following error can occurs during index recovery: AssertionError: getIndexKey: Unhandled row com.gemstone.gemfire.internal.cache.ListOfDeltas. This error indicates that failures occurred during node member initialization.	This error can be ignored.
#49904	Evicted data could temporarily reappear after member recovery	For HDFS tables that use the EVICT BY CRITERIA clause, it is possible that an evicted table value could temporarily reappear in the distributed system if the following sequence of events occurred: **1.** The member hosting secondary buckets for the table goes offline. **2.** The member hosting primary buckets evicts a table entry. **3.** The secondary member returns online, and recovers the older entry. **4.** The primary member goes offline. In this case, a query against the table would use the secondary member's data, which contains the entry that was previously evicted by the primary member.	This problem has negligible impact, because the recurring value would be evicted during the next configured eviction cycle.

ID	Bugnote Title	Bugnote Description	Workaround
#49936	Row eviction is not performed for 'EVICT INCOMING' even though the row matches eviction criteria	Custom eviction in an HDFS table takes place after indexes are updated and the DML event is replicated to secondaries. If a GemFire XD member after these operations but before eviction takes place, then the entry is not evicted even if the event is automatically retried. (This happens because the secondary would have already seen the event and so it would not put it into its local Map, which is necessary to trigger the eviction.)	Use EVICTION BY CRITERIA with an EVICTION FREQUENCY instead of using EVICT INCOMING if you must ensure that custom eviction occurs during HA scenarios.
#49963	SQLIntegrityConstraintViolationException may observed for a unique constraint on partitioning column	If a unique constraint contains a column on which table is partitioned, then an unexpected SQLIntegrityConstraintViolationException may occur for that constraint.	Choose a partitioning column that is not used in the unique constraint.
#49974	Errors attempting to use slf4j packages	This version of GemFire XD cannot be used with external slf4j packages. If you attempt to use these packages you receive errors such as: java.lang.SecurityException: sealing violation: can't seal package org.slf4j: already loaded.	n/a

<table>
<tr><th>ID</th><th>Bugnote Title</th><th>Bugnote Description</th><th>Workaround</th></tr>
<tr><td>#49984</td><td>When using transactions, UNIQUE constraint violation may be thrown when a unique column is updated or deleted multiple times</td><td>The following is an example where updating unique columns multiple times in a transaction causes constraint violations:
gfxd> create table a (symbol varchar(10),
> exchange varchar(10), constraint sec_uq
> unique (symbol, exchange));
0 rows inserted/updated/deleted
gfxd> set ISOLATION READ COMMITTED;
0 rows inserted/updated/deleted
gfxd> insert into a (symbol, exchange)
> values('a', 'a');
1 row inserted/updated/deleted
gfxd> commit;
gfxd> AUTOCOMMIT off;
gfxd> update a set symbol = 'b' where
> symbol='a' and exchange = 'a';
1 row inserted/updated/deleted
gfxd> select * from a;
SYMBOL |EXCHANGE
----
b |a
1 row selected
gfxd> update a set symbol = 'a' where symbol ='b' and exchange = 'a';
ERROR 23505: SQLSTATE=23505,SEVERITY=-1:
[...]</td><td>n/a</td></tr>
<tr><td>#49994</td><td>Unexpected SQLIntegrityConstraintViolationException after server failover</td><td>In a server failover scenario, it is possible that an update to a table with a UNIQUE constraint succeeds in updating the table data, but does not update associated index data. This results in a SQLIntegrityConstraintViolationException for subsequent operations.</td><td>n/a</td></tr>
<tr><td>#50048</td><td>EXPLAIN output does not display</td><td>It is common for GemFire XD to omit the output from an EXPLAIN command because such output frequently exceeds the default maximum display width of the gfxd utility.</td><td>In gfxd, enter the command "maximumdisplaywidth 10000;" or higher to increase the maximum display width before using the EXPLAIN command.</td></tr>
</table>

ID	Bugnote Title	Bugnote Description	Workaround

#50173	Assertion error when inner projection columns are missing in outer query projection list	A SELECT statement that includes an inner query and an outer query with the ORDER BY clause fails with an assertion error if the outer query omits any columns that are projected in the inner query. For example, the following query fails with an assertion:
		`select innerQuery.nl, innerQuery.valid_until`
		`> from (select b.valid_until, b.nl from A a`
		`> LEFT JOIN B b ON a.cell_id=b.cell_id)`
		`> innerQuery group by innerQuery.nl;`
		Re-writing the query to add the missing column from the inner query (valid_until) to the outer query succeeds:
		`select innerQuery.nl, innerQuery.valid_until`
		`> from (select b.valid_until, b.nl from A a`
		`> LEFT JOIN B b ON a.cell_id=b.cell_id)`
		`> innerQuery group by innerQuery.nl,`
		`> innerQuery.valid_until;`
			If you cannot re-write the query to include all columns in the outer query, then you can instead add the system property gemfirexd.no-column-elimination=true when you start GemFire XD members.

#50252	UPDATE and DELETE statements missing from SYS.SESSIONS	The SYS.SESSIONS table does not display statement details for unprepared UPDATE and DELETE statements. Details are displayed for unprepared SELECT statements as well as for all prepared statements.	n/a
#50286	Index entries not cleaned up after dropping a global index	Disk store recovery files (.if files) may not be cleaned up in all cases when a global index is dropped. This problem can occur when a GemFire XD member is stopped after a global index is created and new entries were added, and the global index is then dropped. In this case, the .if file may never record the drop index operation, and the stopped GemFire XD member recovers the index entries at startup, holding them in a placeholder region.	n/a

ID	Bugnote Title	Bugnote Description	Workaround
#50418	Transactional DML operations fail with incorrect exception	If a DML operation in a transaction fails due to a GemFire XD member failure, GemFire XD throws "SQLException 08006: Database 'gemfirexd' shutdown" instead of the expected SQLState X0Z05, X0Z16, 40XD2, or 40XD0 exception.	Applications should check for the 08006 exception to guard against potential member failures while executing DML in a transaction.

#50526	INSERT into a table fails when the table name is a reserved word	If you create a table that is named using a reserved word, the failure occurs when you try to insert data into the table:
		`CREATE TABLE APP."Order" ( Id int NOT NULL );`
		`INSERT INTO APP."Order"(id) values (1);`
		In this example, the CREATE statement succeeds but the INSERT throws an exception.
			Avoid reserved words when creating database objects.

#50538	ArrayIndexOutOfBoundsException with concurrent SELECT and ALTER TABLE	A select query that is concurrently executed on the same table as an ALTER TABLE statement may receive an ArrayIndexOutOfBoundException	Retry the failed query.
#50548	Boolean functions not supported	GemFire XD does not support creating functions that return a Boolean value.	n/a
#50578	Model-first entity framework workflow does not preserve one-to-one relationships	If you use the Entity Framework plug-in to generate a new GemFire XD schema from model objects, one-to-one relationships between entities are lost if you try to recreate the model from the schema. When you apply the generated DDL file to a GemFire XD system and then generate a new model from that schema, a one-to-one relationship between entities is replaced by a one-to-many relationship in the model.	n/a
#50595	Transactional DML not supported with eviction/expiration	GemFire XD does not support executing DML operations in a transaction for tables that are configured with EVICTION or EXPIRE. Attempting such a transaction fails with: ERROR X0Z16: Transaction state is illegal.	n/a

ID	Bugnote Title	Bugnote Description	Workaround
#50778	dsid() function does not work when used with * to retrieve all table columns	The dsid() function does not work when * is used to retrieve columns of a table. For example, the following statement does not work as expected: `select dsid() as "Member", * from emp;`	As a workaround use table aliases when referencing all columns with dsid(). For example: `select dsid() as "member", e.* from emp e;`
#50926	Authentication failures can leak credential information	When security is enabled (for example with `auth-provider=BUILTIN`) information about user IDs can be inferred from error messages that are returned during authentication. By using brute force authentication attempts, it may be possible to infer valid user names in the system.	Require the use of passwords that are at least 8 characters in length.
#50977, #50978	Exceptions when using single hop with connection pooling	GemFire XD does not support single hop connections (`single-hop-enabled=true` property) when used in combination with JDBC connection pooling options such `dbcp` and `c3po`.	n/a

ID	Bugnote Title	Bugnote Description	Workaround
#51039	Query cancellation limitations can lead to long periods of system unavailability	The GemFire XD query cancellation feature cannot be used to cancel DDL statements (see *Cancelling Long-Running Statements* for a full list of limitations.) This can result in long periods of system unavailability when DDL statements obtain a data dictionary lock. For example, consider the case where a long-running query is being executed and a DDL operation such as TRUNCATE TABLE is executed on the same table. The DDL operation obtains a data dictionary lock and then waits for the table lock that was obtained by the query. The data dictionary lock prevents new connections to the system, so you cannot create a new connection in order to cancel the long-running query. In this case you would need to wait until the DDL statement times out (5 minutes by default) before you could connect to the system and cancel the long-running query.	Consider lowering gemfirexd.max-lock-wait to lower the DDL timeout duration. Or, use the *skip-locks* connection property to force a connection to the system without waiting for the DataDictionary lock to be release, and cancel the long-running query. **Warning:** `skip-locks` is provided only for the purpose of cancelling a long-running query in cases where the query causes a DDL operation to hold a DataDictionary lock, preventing new logins to the system. Using this property outside of its intended purpose can lead to data corruption, especially if DDL is performed while the property is enabled.

ID	Bugnote Title	Bugnote Description	Workaround
#51072	Custom `AsyncEventListener` implementations cannot receive events for DMLs executed with `executeBatch()`	Custom implementations of the `AsyncEventListener` interface cannot receive events for DMLs that are executed using the `executeBatch()` method of a JDBC `PreparedStatement`; the `processEvents()` method of `AsyncEventListener` cannot receive the event that is generated for such DML statements.	To ensure that `processEvents()` receives such events, implement the custom `AsyncEventListener` by extending `DBSynchronizer` instead of directly implementing the `AsyncEventListener` interface.
#51132	Insert as subselect fails if schema differs in nullability	An INSERT operation with a subselect fails if one or more columns of the tables have different restrictions with regard to nullability. For example, if one table has a column defined as "int not null" while the corresponding Int column in a second table allows nullability, then an INSERT with subselect from the two tables results in incorrect data for the differing columns.	n/a
#51134	Some queries that do not satisfy colocation requirements fail with NPE	Certain join queries that use OR clauses or equijoin conditions fail with a NullPointerException (instead of a SQLState error) if the joined tables do not satisfy colocation requirements.	n/a
#51194	Table structure is not reported when its synonym name is used	When an application tries to fetch metadata information for a table using its synonym name, the data returned has no information about the table.	Use the base table name, rather than its synonym, to obtain table metadata.
#51218	Thin client driver failures with CHAR columns that have special characters	When using the client driver, querying a CHAR column that contains special characters can result in errors such as "Exception in thread *thread-name* java.lang.StringIndexOutOfBoundsException: String index out of range" or "ERROR 0A000: The DRDA command parseSQLDIAGSTT is not currently implemented. The connection has been terminated." This problem does not happen with VARCHAR columns, or when using the peer client driver.	Use the peer client driver to avoid this problem.

<table>
<tr><th>ID</th><th>Bugnote Title</th><th>Bugnote Description</th><th>Workaround</th></tr>
<tr><td>#51244</td><td>Running <code>IMPORT_DATA_EX</code> in <code>REPLACE</code> mode ignores the <code>skip-constraints-check=true</code> connection property</td><td>While executing <code>SYSCS_UTIL.IMPORT_DATA_EX</code> to import data, if the <code>REPLACE</code> option is enabled (set to a non-zero value) then GemFire XD does not honor the <code>skip-constraint-checks=true</code> connection property. Any imported rows that fail to satisfy a primary key, foreign key, or unique constraint fail with an associated import error.</td><td>n/a</td></tr>
<tr><td>#51246</td><td>Thin client driver fails to execute CASE expression with CHAR return expressions</td><td>This issue is related to #51218. If you use the CASE expression from a thin client connection and the expression defines return expressions of different character lengths, then the statement can fail with errors such as such "JAVA ERROR: StringIndexOutOfBoundsException: String index out of range" or "ERROR 0A000: The DRDA command parseSQLDIAGSTT is not currently implemented. The connection has been terminated."
To avoid this problem, either use a peer client connection or use the CAST function to explicitly cast the returned expression into the VARCHAR datatype. For example:<pre>
SELECT CAST (
         CASE
           WHEN customer.
id_str = '41658296553177088'
 then 'matched'
           ELSE 'other'
         END
       AS VARCHAR(10))
       FROM customer
 WHERE customer.id >
 41658296553177086
         AND customer.id <
 41658296553177090
</pre></td><td>Either execute the statement from a peer client or use the CAST function to cast the returned expression to VARCHAR.</td></tr>
</table>