MySQL 9.0 Release Notes

Abstract

This document contains release notes for the changes in MySQL 9.0. For information about changes in a different version of MySQL, see the release notes for that version.

For additional MySQL 9.0 documentation, see the MySQL 9.0 Reference Manual, which includes an overview of features added in MySQL 9.0 (What Is New in MySQL 9.0), and discussion of upgrade issues that you may encounter while upgrading.

MySQL platform support evolves over time; please refer to https://www.mysql.com/support/supportedplatforms/database.html for the latest updates.

Updates to these notes occur as new product features are added, so that everybody can follow the development process. If a recent version is listed here that you cannot find on the download page (https://dev.mysql.com/downloads/), the version has not yet been released.

The documentation included in source and binary distributions may not be fully up to date with respect to release note entries because integration of the documentation occurs at release build time. For the most up-to-date release notes, please refer to the online documentation instead.

For legal information, see the Legal Notices.

For help with using MySQL, please visit the MySQL Forums, where you can discuss your issues with other MySQL users.

Document generated on: 2025-01-24 (revision: 29588)

Table of Contents

Preface and Legal Notices

This document contains release notes for the changes in MySQL 9.0.

Legal Notices

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support for Accessibility

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Changes in MySQL 9.0.1 (2024-07-23, Innovation Release)

Bugs Fixed

- **InnoDB:** In some cases, following the creation of a very large number of tables (8001 or more), the server could not be restarted successfully. (Bug #36808732)

  References: This issue is a regression of: Bug #33398681.

- **InnoDB:** Improved tablespace file scan performance at startup. (Bug #110402, Bug #35200385)

- **Group Replication:** Running a `CREATE TABLE ... SELECT` statement on a source coming from an asynchronous channel to Group Replication led to errors on the replica. (Bug #36784284)

Changes in MySQL 9.0.0 (2024-07-01, Innovation Release)

Important

This release is no longer available for download. It was removed due to a critical issue that could stop the server from restarting following the creation of a very large number of tables (8001 or more). Please upgrade to MySQL 9.0.1 instead.

Audit Log Notes

- The audit log plugin printed the warning `Plugin audit_log reported: 'Cannot process audit log file. File name timestamp value is missing or invalid: 'dbname''` when no audit log file was being processed. (Bug #36400810)
- With large files (200 MB or larger), executing `audit_log_read()` took an excessive amount of time to find the bookmark matching the one specified when invoking the function. (Bug #36281295)
- Audit log pruning did not function after removing or renaming a file from the audit log. Now pruning continues in such cases, but with a warning printed in the error log stating that it was not possible to delete the missing audit log file. (Bug #35902913)
- Improved the `audit_null` example plugin. (Bug #35820314)
- MySQL now calls `plugin->deinit()` with a valid plugin struct as an argument regardless of the plugin's type.

  Our thanks to Martin Alderete for the contribution.

Authentication Notes

- Made improvements to the authentication code. (Bug #35325870)
- Improved log messages to provide clear reasons for `Access denied` errors when using the `authentication_ldap_sasl` plugin without proxying. (Bug #35317691)
- MySQL LDAP SASL authentication, when used with the GSSAPI method to access an OpenLDAP server, was rejected with the MySQL server error `Plugin authentication_ldap_sasl reported: 'LDAP authentication failed or group retrieval failed: LDAP error: Invalid DN syntax'`, because OpenLDAP did not recognize the root DN used. (Bug #32631511)

C API Notes

- C API applications stalled while receiving results for server side prepared statements.

Character Set Support

- When resolving a call to the `REPLACE()` function, the character set and collation of the function result are copied from the first argument. The remaining two arguments, if they are literal values, should be converted to this character set, but only the second argument was converted.

This fix ensures that the third argument is also converted to the first argument's character set and collation. (Bug #114769, Bug #36562972)

- The internal function `my_instr_mb()` assumed incorrectly assumption in several places that byte lengths for input strings could be used to short-cut certain decisions. In the case of multibyte character sets and collations, this cannot be done since, under some collation rules, characters with differing byte lengths can be considered equal. In addition, `Item_func_locate()` used an incorrect byte length.

  Our thanks to Dirkjan Bussink for the contribution. (Bug #113933, Bug #36277823)

- The internal function `Item_func::eq()` erroneously treated the two expressions `CONVERT(a USING latin1)` and `CONVERT(a USING utf8mb4)` as being equal. (Bug #113506, Bug #116762, Bug #36137690, Bug #37323921)

Compilation Notes

- **macOS:** CMake no longer tries to use the native `ctags` on MacOS, and now requires the Homebrew version of it to be installed on the system when building MySQL. (Bug #36590594)
- **macOS:** Enabled use of gRPC when building MySQL on MacOS. (Bug #36537726)
- **macOS:** The Xcode version of `zlib` was removed from the default list of system libraries to use when configuring with `-DWITH_SYSTEM_LIBS=ON`. (Bug #36537593)
- **Microsoft Windows:** The `BUILD_ALL` target did not work when compiling on Windows. (Bug #36424619)
- **Microsoft Windows:** Excessive RAM usage led to disk swapping when compiling MySQL on Windows using Ninja. (Bug #36399256)
- Interface libraries for `librpdserver_shared.so` were missing from the bundled `protobuf/grpc` on Ubuntu 24.04. (Bug #36678790)
- Maintainer mode is now disabled when building the debug version of the server for `.deb` packages. (Bug #36619757)
- Upgraded the bundled `googletest` and `googlemock` sources to version 1.14.0. (Bug #36562482)
- Added a missing dependency on `GenError`. (Bug #36551721)
- When compiling on Fedora 38, `grep -E` is now used in place of `egrep`. (Bug #36507549)
- The version of Boost used for compiling MySQL was upgraded from 1.84.0 to 1.85.0. For more information see the Boost 1.85.0 Release Notes. (Bug #36495694)
- Binaries for Enterprise Linux 8 and 9 are now built using GCC 13. (Bug #36331855)
- Removed linker warnings raised when compiling code that used RapidJSON. (Bug #36322583)
- It is now possible on Linux systems to build MySQL using a bundled `tcmalloc` library that is provided with the source by specifying `-DWITH_TCMALLOC=BUNDLED`. This is supported on Linux only. (Bug #36313839)
- The bundled `tcmalloc()` is now used when building MySQL on Enterprise Linux 8. (Bug #114844, Bug #35674008)
- Removed warnings raised in `sql/statement/ed_connection.cc` when building on Ubuntu 23.04. (Bug #114436, Bug #36428465)
- Linux `aarch64` platform binaries are now built using `patchelf --page-size=65536` for compatibility with systems using either 4k or 64k for the page size. (Bug #114233, Bug #36393794)

Component Notes

- When the server was started with an incorrect `--basedir` option and no `--plugin-dir` option on UNIX platforms with a manifest and configuration file in place, the keyring component loaded but failed to unload.

  Now, when loading keyring components from the manifest, if `--basedir` is specified but `--plugin-dir` is not specified, we derive the plugin directory from the base directory; if neither of these is specified, we derive the plugin directory from the MySQL installation directory. The same handling also now applies to `--datadir`. (Bug #36398484)

- The values for component options set using the `--loose` prefix were not read when the component was installed. (Bug #28341329)

Configuration Notes

- **Microsoft Windows:** On Windows, *MySQL Configurator* was updated to support in-place upgrades as per Upgrade Paths. (Bug #36685422)

- **Microsoft Windows:** On Windows, clicking the **[X]** close button on a *MySQL Configurator* wizard's page now yields a confirmation popup if the wizard is busy executing an operation. (Bug #36671317)

- **Microsoft Windows:** On Windows, *MySQL Configurator* no longer defines a custom `server_type` variable in the generated MySQL Server configuration file. This information is now stored in the `configurator_settings.xml` file. (Bug #36670309)

- **Microsoft Windows:** On Windows, the `Removing Windows Firewall` step in *MySQL Configurator* would fail if the `my.ini` file was missing a `mysqlx_port` definition. (Bug #36666260)

- **Microsoft Windows:** On Windows, if *MySQL Configurator* failed to find a valid `my.ini` or `my.cnf` file from the *MySQL Server Installations* page, then clicking the **Browse** button disabled the **Next** button even when the selected file was valid. (Bug #36395569)

- **Microsoft Windows:** On Windows, *MySQL Configurator* now only shows the removal steps if the associated MySQL Server was previously configured. (Bug #36395417)

- **Microsoft Windows:** On Windows, *MySQL Configurator* now detects if the existing root user is using the `mysql_native_password` authentication plugin (removed in MySQL 9.0.0) and prompts to convert root to use the `caching_sha2_password` authentication plugin before performing a MySQL Server upgrade. (WL #16139)

- **Microsoft Windows:** For MSI installations on Windows, *MySQL Configurator* now automatically upgrades MySQL 8.4 LTS installations without user intervention. (WL #16274)

Connection Management Notes

- The `conn_delay/Waiting in connection_control plugin` stage was not reset after a delay introduced by the connection control plugin which resulted in incorrect monitoring information. (Bug #35205358)

Data Dictionary Notes

- Attempting to upgrade a `MyISAM` table containing a mix of regular columns and generated columns from MySQL 5.7 to 8.0 or later led to table corruption. (Bug #105301, Bug #33503328)

Data Type Notes

- When a string is converted to a numeric value, any non-numeric data trailing the numeric value should cause an error with strict mode and a warning with any other SQL mode, but in some cases, depending on the length and character set of the string, an invalid string did not raise any of the expected errors or warnings. (Bug #36457756)

- In some cases, casting a double to an integer value used rounding, and in others, with truncation, which led to inconsistent results. Now rounding up is used in all such cases. (Bug #114549, Bug #36481397)

- In some cases, `DECIMAL` 0 was treated as less than a `FLOAT` value between 0 and -1. (Bug #114196, Bug #117093, Bug #36361165, Bug #37438582)

Deprecation and Removal Notes

- **Replication:** MySQL 9.0.0 deprecates transactions which update both transactional tables and nontransactional or noncomposable tables. Such a transaction now causes a deprecation warning to be written to both the client and the error log. Only the `InnoDB` and `BLACKHOLE` storage engines are transactional and composable. This means that only the combinations of storage engines shown here do *not* raise the deprecation warning:

  - `InnoDB` and `BLACKHOLE`
  - `MyISAM` and `MERGE`
  - `performance_schema` and any other storage engine
  - `TempTable` and any other storage engine

  

  Note

  `NDBCLUSTER` is transactional but not composable.

  For more information, see Replication and Transactions. (WL #10495)

- The `mysql_native_password` authentication plugin, deprecated in MySQL 8.0, has been removed, and the server now rejects `mysql_native` authentication requests from older client programs which do not have `CLIENT_PLUGIN_AUTH` capability. For backward compatibility, `mysql_native_password` remains available on the client; the client-side built-in authentication plugin has been converted into a dynamically loadable plugin.

  

  Note

  In MySQL 8.0, the default MySQL authentication plugin was changed to `caching_sha2_password` (see Caching SHA-2 Pluggable Authentication).

  These changes also entail the removal of the following server options and variables:

  - The `--mysql-native-password` server option
  - The `--mysql-native-password-proxy-users` server option
  - The `default_authentication_plugin` server system variable

  For more information, see Authentication Plugins. (WL #15930)

- The `MIN_VALUE` and `MAX_VALUE` columns of the Performance Schema `variables_info` table are now deprecated, and subject to removal in a future version of MySQL. Instead, you should use the `MIN_VALUE` and `MAX_VALUE` columns of the `variables_metadata` table, which provide the same information. (WL #15585)

Event Scheduler Notes

- **Important Change:** The following SQL statements relating to events may now be prepared:

  - `CREATE EVENT`

- `ALTER EVENT`
- `DROP EVENT`

Positional parameters (? placeholders) are not supported for these statements; you must assemble the text of the statement to be prepared from some combination of string literals, system variables, and user variables. See PREPARE, EXECUTE, and DEALLOCATE PREPARE Statements, and SQL Syntax Permitted in Prepared Statements, for more information. CREATE EVENT Statement, provides a basic example. (Bug #109309, Bug #34875573, WL #16298)

JavaScript Programs

- MySQL 9.0 Enterprise Edition now includes support for stored programs written in JavaScript, such as this simple example created using the `CREATE FUNCTION` statement and JavaScript code shown here:

```
CREATE FUNCTION gcd(a INT, b INT)
RETURNS INT
NO SQL
LANGUAGE JAVASCRIPT AS
  $mle$
    let x = Math.abs(a)
    let y = Math.abs(b)
    while(y) {
      var t = y
      y = x % y
      x = t
    }
    return x
  $mle$
;
```

JavaScript Stored Program Creation and Management, describes creation and execution of JavaScript stored programs.

JavaScript language support includes both stored procedures and stored functions, and is provided by the Multilingual Engine Component (MLE). For more information about determining whether your distribution includes this component, and enabling it, see Multilingual Engine Component (MLE).

JavaScript language support in MySQL conforms to the ECMAScript 2023 Specification, and uses strict mode by default. Strict mode cannot be disabled. This implementation includes all of the standard ECMAScript library objects such as `Object`, `Function`, `Math`, `Date`, and `String`. `console.log()` and `console.error()` are also supported.

Most MySQL data types are supported for JavaScript stored program input and output arguments, as well as for return data types. Strings must use the `utf8mb4` character set. MySQL `BLOB` and `TEXT` types are supported, as are many MySQL temporal types. `JSON` is also supported. The `VECTOR` type is not supported by the MLE component or by JavaScript stored programs. For more information, see JavaScript Stored Program Data Types and Argument Handling, and JavaScript Stored Program Limitations and Restrictions.

Stored programs written in JavaScript support an SQL and result set API provided by the MLE component. See JavaScript SQL API, and Using the JavaScript SQL API, for more information.

The MLE component provides a number of session information and management functions including `mle_session_state()` and `mle_session_reset()`. You can also view a number of MLE status variables in the output of a statement similar to `SHOW STATUS LIKE 'mle%'`. See also JavaScript Stored Programs—Session Information and Options.

For general information about JavaScript stored programs, see JavaScript Stored Programs. (WL #15605, WL #16129, WL #16172, WL #16226, WL #16272, WL #16276)

Keyring Notes

- Made a number of improvements in the keyring code. (Bug #36401550)

Optimizer Notes

- **Important Change:** `ER_SUBQUERY_NO_1_ROW` has been removed from the list of errors which are ignored by statements which include the `IGNORE` keyword. This has been done for the following reasons:

  - Ignoring such errors sometimes led to insertion of NULL into non-nullable columns (for untransformed subqueries), or of no row at all (subqueries using `subquery_to_derived`).
  - When subqueries were transformed to join with derived tables, the behavior differed from that of untransformed queries.

  Following an upgrade to this release, this change can make an `UPDATE`, `DELETE`, or `INSERT` statement which includes the `IGNORE` keyword raise errors if it contains a `SELECT` statement with a scalar subquery that produces more than one row.

  For more information, see The Effect of IGNORE on Statement Execution. (Bug #110961, Bug #35373406)

Performance Schema Notes

- **Group Replication:** The following tables did not contain data on replication channels which did not have a configured hostname, such as Group Replication recovery channels:

  - `REPLICATION_CONNECTION_STATUS`
  - `REPLICATION_CONNECTION_CONFIGURATION`
  - `REPLICATION_APPLIER_CONFIGURATION`
  - `REPLICATION_APPLIER_STATUS`
  - `REPLICATION_APPLIER_STATUS_BY_COORDINATOR`
  - `REPLICATION_APPLIER_STATUS_BY_WORKER`

  As of this release, these tables contain data for partially configured Group Replication channels. (Bug #36018242)

- Under certain conditions, a race condition could result in the amount of RAM used by `TABLE_HANDLES` increasing to a maximum of 9GB. (Bug #36170903)

- The `PROCESSLIST_INFO` column of `THREADS` was not updated when executing a prepared statement.

  Thanks to Daniel Lenski and Amazon for the contribution. (Bug #104121, Bug #33057164)

- This release adds two tables to the MySQL Performance Schema, listed here:

  - The `variables_metadata` table provides general information about system variables. This information includes the name, scope, type, range (where applicable), and description of each system variable recognized by the MySQL server.

    The `MIN_VALUE` and `MAX_VALUE` columns of this table are intended to replace the deprecated `MIN_VALUE` and `MAX_VALUE` columns of the `variables_info` table.

  - The `global_variable_attributes` table provides information about attribute-value pairs assigned by the server to global system variables.

For more information, see Performance Schema System Variable Tables. (WL #15855)

SQL Syntax Notes

- **JSON:** You can now save the `JSON` output from `EXPLAIN ANALYZE` into a user variable using the syntax shown here:

```
EXPLAIN ANALYZE FORMAT=JSON INTO @variable select_stmt
```

The variable can be used subsequently as a JSON argument to any of MySQL's JSON functions (see JSON Functions). The `INTO` clause is supported only with `FORMAT=JSON`, which must be included explicitly. This form of `EXPLAIN ANALYZE` also supports an optional `FOR SCHEMA` or `FOR DATABASE` clause preceding the `SELECT` statement being analyzed. Statements other than `SELECT` are not supported.

Note

This feature is available only if the `explain_json_format_version` server system variable is set to `2`; otherwise, attempting to make use of it raises `ER_EXPLAIN_ANALYZE_JSON_FORMAT_VERSION_NOT_SUPPORTED`.

For more information and examples, see Obtaining Execution Plan Information. (WL #16216)

- MySQL now accepts and enforces inline foreign key specifications (these were previously accepted by the parser, but ignored). MySQL now also accepts implicit references to parent table primary key columns.

Consider the parent table `person` created by the following statement:

```
CREATE TABLE person (
    id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    name CHAR(60) NOT NULL
);
```

To create a table `shirt` having a foreign key `owner` on `person`, MySQL now accepts and handles correctly any of the `CREATE TABLE` statements shown here, according to the standard:

```
CREATE TABLE shirt (
    id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    style ENUM('tee', 'polo', 'dress') NOT NULL,
    color ENUM('red', 'blue', 'yellow', 'white', 'black') NOT NULL,
    owner SMALLINT UNSIGNED NOT NULL,

    FOREIGN KEY (owner) REFERENCES person (id)
);

CREATE TABLE shirt (
    id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    style ENUM('tee', 'polo', 'dress') NOT NULL,
    color ENUM('red', 'blue', 'yellow', 'white', 'black') NOT NULL,
    owner SMALLINT UNSIGNED NOT NULL REFERENCES person (id)
);

CREATE TABLE shirt (
    id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    style ENUM('tee', 'polo', 'dress') NOT NULL,
    color ENUM('red', 'blue', 'yellow', 'white', 'black') NOT NULL,
    owner SMALLINT UNSIGNED NOT NULL,

    FOREIGN KEY (owner) REFERENCES person
);

CREATE TABLE shirt (
    id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
    style ENUM('tee', 'polo', 'dress') NOT NULL,
    color ENUM('red', 'blue', 'yellow', 'white', 'black') NOT NULL,
```

```
    owner SMALLINT UNSIGNED NOT NULL REFERENCES person
);
```

In previous versions of MySQL, only the first of the four statements just shown had the effect of creating a foreign key; the second was parsed, but the `REFERENCES` clause was ignored. The remaining two statements caused syntax errors.

For more information, see FOREIGN KEY Constraints, as well as FOREIGN KEY Constraint Differences. (Bug #4919, Bug #17943, Bug #102904, Bug #11744902, Bug #11745637, Bug #32613571, WL #16130, WL #16131)

sys Schema Notes

- The performance of the `innodb_lock_waits` view is improved in this release. (Bug #36337708)

Thread Pool Notes

- Connecting to a thread group that had no connection handler threads stalled. We fix this by making sure that connection handler threads terminate only if there is at least one connection thread left. (Bug #36550125)
- Previous refactoring incorrectly removed the connection locking performed when processing incoming connection requests, which ledto a race condition between the thread adding new connections and the connection handler thread processing them. This appeared to cause a situation in which connection requests might be ignored and not processed, so that the connection attempt appeared to hang.

  We fix this by taking the connection before processing the queue, and releasing it before waking or creating new threads. (Bug #36548687)
- The Information Schema tables `TP_THREAD_GROUP_STATE`, `TP_THREAD_GROUP_STATS`, and `TP_THREAD_STATE` were deprecated in MySQL 8.0.14. Accessing any of these tables now produces a warning; you should use the equivalent Performance Schema tables instead.

  For more information, see Performance Schema Thread Pool Tables. (Bug #36359860)
- It was possible to set the `thread_pool_longrun_trx_limit` system variable to values outside its stated range.

  In addition, settings for this variable were not reflected in the output of SHOW VARIABLES or SELECT. (Bug #36347102, Bug #36371145)
- `SET PERSIST_ONLY` did not work correctly with `thread_pool_max_transactions_limit`. (Bug #35019884)
- `KILL CONNECTION` did not work correctly with `thread_pool_max_transactions_limit`. (Bug #34019954)
- The thread ID was not displayed for client connections in the `performance_schema.socket_instances` table when using the Thread Pool plugin. (Bug #24796018)

Vector Data Type

- Support is added in this release for a `VECTOR` column type. A vector is a data structure which consists of a list of entries (4-byte floating-point values) which can be expressed either as a binary string value or a list-formatted string. A `VECTOR` column is declared with a maximum length or number of entries (in parentheses); the default is 2048, and the maximum is 16383.

  You can create `InnoDB` tables with `VECTOR` columns using `CREATE TABLE` as shown here:

```
mysql> CREATE TABLE v1 (c1 VECTOR(5000));
```

```
Query OK, 0 rows affected (0.03 sec)
```

Other storage engines do not support tables with `VECTOR` columns.

Vector columns in this release are subject to restrictions, some of which are listed here:

- A `VECTOR` column cannot be used as any type of key. This includes primary keys, foreign keys, unique keys, and partitioning keys.
- Some types of MySQL functions and operators do not accept vectors as arguments. These include but are not limited to numeric functions and operators, temporal functions, full-text search functions, XML functions, bit functions, and JSON functions.

  Some (but not all) string and encryption functions support `VECTOR` values. For more complete information, see VECTOR Supported and Unsupported Functions.
- A `VECTOR` cannot be compared with any other type, and can be compared with another `VECTOR` only for equality.
- MLE JavaScript programs do not support `VECTOR` columns or values.
- NDB Cluster does not support `VECTOR` columns or values.

Use the `VECTOR_DIM()` function (also added in MySQL 9.0) to obtain the length of a vector. Functions to convert between representations are available. `STRING_TO_VECTOR()` (alias: `TO_VECTOR()`) takes a list-formatted representation of a vector and returns the binary string representation; `VECTOR_TO_STRING()` (alias: `FROM_VECTOR()`) performs the inverse, as shown here:

```
mysql> SELECT STRING_TO_VECTOR('[2, 3, 5, 7]');
+-------------------------------------------------------+
| TO_VECTOR('[2, 3, 5, 7]')                             |
+-------------------------------------------------------+
| 0x0000004000004040000A0400000E040                     |
+-------------------------------------------------------+
1 row in set (0.00 sec)

mysql> SELECT VECTOR_TO_STRING(0x0000004000004040000A0400000E040);
+-------------------------------------------------------+
| VECTOR_TO_STRING(0x0000004000004040000A0400000E040) |
+-------------------------------------------------------+
| [2.00000e+00,3.00000e+00,5.00000e+00,7.00000e+00]     |
+-------------------------------------------------------+
1 row in set (0.00 sec)
```

For more information and examples, see The VECTOR Type, and Vector Functions. (WL #16081)

X Plugin Notes

- The system variable `caching_sha2_password_digest_rounds` could not be set to a non-default value using X Protocol. (Bug #36402455)
- An outdated link to the MySQL documentation in the `mysql_function_names` unit test source file has been updated.

  Our thanks to Minha Jeong for the contribution. (Bug #113500, Bug #36137217)

Functionality Added or Changed

- A correlated subquery can now be optimized as an outer left join on a derived table when the subquery contains a `LIMIT 1` clause (previously, such a subquery could not be transformed in this way if it contained any `LIMIT` clause). The `LIMIT` clause must use a literal `1`; if it uses any other value, or a placeholder (`?`) or variable, the subquery is not eligible for this transformation.

For more information, see Correlated Subqueries. (Bug #36475554, WL #16124)

- Join columns are now included in the output of `EXPLAIN FORMAT=JSON`. (Bug #36230046)

Bugs Fixed

- **Important Change; Replication:** When `replica_parallel_workers` was equal to `1`, incorrect logical timestamps for transactions caused the replica to stop with an error, despite the fact that logical timestamps are not relevant when there is only a single applier thread.

  Now, the replica logs a warning when `replica_parallel_workers = 1`, and raises an error only when the value of this system variable is greater than `1`. In addition, the format of and information contained in the error message has been improved, and the same message is now used in the output of `SHOW REPLICA STATUS` as well as in the Performance Schema `replication_applier_status` table. (Bug #36058442)

- **InnoDB:** MySQL unexpectedly halted on an `UPDATE` after an `ALTER TABLE` operation. (Bug #36571091)

  References: This issue is a regression of: Bug #35183686.

- **InnoDB:** Improved the `InnoDB` recovery logic to reduce pauses between recovery actions. (Bug #36332645)

- **InnoDB:** File system operations performed by `InnoDB` now consistently `fsync` the parent directory when performing directory altering tasks. (Bug #36174938)

- **InnoDB:** In debug builds, setting the `innodb_interpreter_output` debug variable would cause the server to unexpectedly halt. This is now a read-only variable. (Bug #36041032)

- **InnoDB:** Improved `os_innodb_umask` handling, and made it read-only. (Bug #35932118)

  References: This issue is a regression of: Bug #29472125.

- **InnoDB:** Removed functionality specific to the Fusion IO atomic write feature, a product that was last available in 2014. (Bug #35072139)

- **InnoDB:** An `InnoDB` assertion error referencing an invalid column index was triggered when the column index was valid. (Bug #34800754)

- **InnoDB:** With an empty `XA` transaction, shutting the server down after an `XA START` would cause the server to halt unexpectedly. (Bug #32416819)

- **InnoDB:** Shutting down the replication applier or binlog applier while processing an empty XA transaction caused the system to unexpectedly halt. (Bug #32416819)

- **InnoDB:** Removed unnecessary heap usage in the `Validate_files::check()` function.

  Our thanks to Huaxiong Song for the contribution. (Bug #115041, Bug #36626203)

- **InnoDB:** Improved the `notify_about_advanced_write_lsn()` logic to prevent potential log notification delays.

  Our thanks to Zongzhi Chen for the contribution. (Bug #114660, Bug #36528707)

- **InnoDB:** If a partition table was read with `innodb_parallel_read_threads=1`, read performance greatly decreased from any table after 256 reads. InnoDB behaved as if it reached the maximum capacity of parallel read threads despite not using any.

  Our thanks to Ke Yu for the contribution. (Bug #114154, Bug #36347408)

- **InnoDB:** Removed an unnecessary conditional check from `get_next_redo_rseg_from_undo_spaces()`.

Our thanks to Alex Xing for the contribution. (Bug #113640, Bug #36185805)

- **InnoDB:** The result from a spatial index containing a column with a spatial reference identifier (SRID) attribute was empty. In addition, using `FORCE INDEX` to force a covering index scan on a spatial index led to an assertion. (Bug #112676, Bug #114200, Bug #35894664, Bug #36361834)

- **InnoDB:** `SELECT ... GROUP BY` queries were at least twice as slow with the TempTable engine than the Memory engine. (Bug #107700, Bug #34338001)

- **Replication:** If a source contained a stored, generated column populated by a JSON function and `binlog_row_image` was set to `MINIMAL`, any subsequent update or deletion on the underlying column failed with the following error:

  ```
  Invalid JSON text in argument 1 to function json_extract: 'The document is empty.'
  ```

  The replica attempted to re-evaluate the generated column and failed with that error because the underlying column was unavailable. As of this release, stored, generated columns are not re-evaluated when the underlying columns are unavailable. (Bug #36515172)

- **Replication:** When running GTID-based replication with `relay_log_space_limit` enabled, a restart of the auto positioning protocol sometimes resulted in an infinite loop, leading to a deadlock in replication. This was because `relay_log_space_limit` was not honored, not only for transactions whose size exceed this limit, but when the replica could not purge previous logs as well.

  To fix this issue, we make the following changes:

  - The receiver respects `relay_log_space_limit` as set by the user, unless a transaction received by the receiver cannot fit into the purged relay log. Before queuing the received transaction, receiver now checks whether scheduling a full transaction is possible. If not, the receiver performs the following actions:

    - Sets the flag indicating that receiver is waiting
    - Rotates the relay log
    - Waits until it is notified that relay log purge was executed and that the applier has purged all available relay logs; after this, the receiver may queue a transaction without checking the limit again

  - Before moving to the next file, the coordinator checks whether the receiver is waiting for available relay log space. If so, the coordinator forcibly purges the applied logs, including the current relay log file. To purge the current relay log file safely, the coordinator must do the following:

    - Synchronize all of its workers before moving to the next file
    - Forcibly update group positions, which is necessary to allow current purging of the relay log
    - Update the variable read by the receiver which contains relay log filename to which applier was moved

    These operations are allowed because we know that receiver waits at a transaction boundary and rotates the relay log before waiting.

  (Bug #36507020)

- **Replication:** Worker jobs now contain information about the relay log file which initiated the transaction, instead of using the default defined by `relay_log`. (Bug #36395631)

- **Replication:** The column number returned in the error `ER_SERVER_REPLICA_CONVERSION_FAILED` was incorrect. It was one less than the actual value. (Bug #36246205)
- **Replication:** Handling an incident while transactions were being committed to the binary log caused MySQL to wait indefinitely. (Bug #35671897)
- **Replication:** When using row-based replication with `binlog_rows_query_log_events` enabled, SQL statements are logged in a `Rows_query_log_event` which is written before the `Table_map_event`. Since SQL may contain any binary data including embedded nulls (`\0`), when using `strlen()`, `snprintf()`, and other similar C functions which rely on processing up to the null byte, it was possible for the query string to be truncated at the first null byte, which resulted in processing incomplete data.

  We fix this by specifying the length of each `Rows_query_log_event` such that it is no longer necessary to rely on C-style string methods for length calculations; in addition, while processing the query, we now use functions which do not rely on null termination. We also mark the first byte of a `Rows_query_log_event` data body as unused. (Bug #109401, Bug #35336260)
- **Group Replication:** Removed a memory leak from `/xcom/gcs_xcom_networking.cc`. (Bug #36532199)
- **Group Replication:** Under certain circumstances, after successfully setting a new primary, `group_replication_set_as_primary()` waited indefinitely for the operation to complete.

  As of this release, a periodic check is performed to ensure the function does not wait unnecessarily. (Bug #36348650)
- **Group Replication:** Under certain circumstances, if a primary's host experienced network inactivity of 20 seconds or more, the secondaries could stop unexpectedly. (Bug #36306144)
- **Group Replication:** The `MEMBER_ID`, `MEMBER_HOST`, and `MEMBER_PORT` columns of the `REPLICATION_GROUP_MEMBERS` table were not always populated for offline members. (Bug #36290046)
- **Group Replication:** Under certain circumstances, if garbage collection occurred just before a relay log rotation, it could cause the applier to stop applying new transactions on the secondary members.

  This was caused by garbage collection incrementing the relay log's `last_committed` and `sequence_number`, creating a gap in the recorded `sequence_number` after the log rotation. The applier was unaffected if the gap occurred anywhere else in the relay log.

  As of this release, only `last_committed` is updated during garbage collection. (Bug #36280130, Bug #36446250)
- **JSON:** Added missing checks for error handling to `NULLIF()`, `COALESCE()`, and the shift (`>>`) operator. (Bug #113668, Bug #35513196, Bug #36198403)

  References: See also: Bug #31358416.
- **MySQL NDB ClusterJ:** Running the ClusterJ test suite resulted in an error message saying a number of threads did not exist. That was due to some wrong handling of threads and connections, which was corrected by this patch. (Bug #36086735)
- Added a missing error check needed when evaluating the `<=>` operator. (Bug #36570474)
- Added a missing error check needed for evaluating `CASE` operators. (Bug #36570439)
- Averages of certain numbers were not always computed correctly. (Bug #36563773)
- Removed redundant assignments to `Item::m_table_ref` in `find_field_in_tables()` which led to invalid `GROUP BY` results and other errors. (Bug #36556725, Bug #36557029)

References: This issue is a regression of: Bug #36533080.

- The following files in `strings` contained incorrect license information:

  - `mb_wc.h`
  - `ctype-uca.cc`
  - `ctype-ucs2.cc`
  - `ctype-utf8.cc`
  - `dtoa.cc`
  - `strxmov.cc`
  - `strxnmov.cc`

  (Bug #36506181)

- In certain unusual cases, the `UpdateXML()` function did not process all of its arguments correctly. (Bug #36479091)

- With `subquery_to_derived=ON`, an outer reference was not replaced in some cases following transformation to a derived table. (Bug #36475633)

  References: See also: Bug #36314993.

- A missing check for errors relating to `TIME` values sometimes led to an assert in `sql/item.cc`. (Bug #36421511)

- Explaining a query which used `FORCE INDEX` on a spatial index containing a column with SRID attributes led to an unplanned exit. (Bug #36418426)

- Events created within stored programs were not always handled correctly. (Bug #36402968, Bug #35395333)

  References: This issue is a regression of: Bug #17809, Bug #11745618.

- The `InnoDB` OpenTelemetry metrics (`mysql.inno`) were not automatically updated. (Bug #36399090)

- This fix addresses two issues:

  - An item that was not yet fixed when searching for an item placed in the `GROUP BY` list led to an assert in `include/sql_string.h`.
  - The `TIME_FORMAT()` function did not handle `NULL` arguments correctly in all cases.

  (Bug #36367313, Bug #36367776)

- Updated `BuildRequire` rules to align with versions now required for CMake and Bison. (Bug #36343254)

- Removed an unused argument from the internal function `MY_COLLATION_HANDLER::strstr()`. (Bug #36342997)

- An `IN` predicate containing `EXCEPT ALL` set operations yielded the wrong result. (Bug #36332697)

- A query using `WHERE primary_key IN(SELECT constant1 EXCEPT SELECT constant2)` returned a differing number of rows depending on the presence or absence of an `ORDER BY` clause. (Bug #36307622)

- When incrementing the reference count for an expression, underlying expressions within this expression are not looked at. While removing an expression, after decrementing the reference count, even the underlying expressions were examined, which led to unintentional deletion of the underlying expressions. This issue manifested in `Item_ref::real_item()` as well as in an assert in `sql/item.h`. We fix this by not looking at the underlying expression unless the current expression contains the only remaining reference. (Bug #36204344, Bug #36356279)

- Under certain conditions, `EXPLAIN FORMAT=JSON FOR CONNECTION` sometimes led to an unplanned exit. (Bug #36189820)

- In transforming subqueries to derived tables, replacement of a subquery in a `HAVING` condition failed to use an item reference, which led to an assert in production builds and an unplanned exit in debug builds. This occurred because the optimizer did not correctly detect that the subquery was part of a `HAVING` condition. (Bug #36079456)

  References: This issue is a regression of: Bug #35060385.

- It was possible for `MaterializeIterator<Profiler>::load_HF_row_into_hash_map()` to exhaust resources while re-reading rows. (Bug #36075756)

- Some `CREATE USER` statements were not handled correctly. (Bug #36022885)

- In certain cases, a lateral join was not handled correctly. (Bug #35945239)

  References: See also: Bug #107700, Bug #34338001. This issue is a regression of: Bug #32644631.

- For a `SELECT` with `ORDER BY` and `LIMIT`, the optimizer first chose a full table scan with a very expensive cost, then performed another check and used the `perform_order_index` type of path, but this was not reflected by the cost in the optimizer plan. (Bug #35930969)

- Client connections were not alway terminated correctly during shutdown. (Bug #35854919)

- Executing `mysqldump` on a replica would insert the `FLUSH TABLES` operation, an operation that writes to the binary log. Now `FLUSH LOCAL TABLES` is inserted instead to prevent GTID related issues during replication due to these binary log changes.

  The workaround was to set the `--source-data` option to 1 or 2. (Bug #35665076)

  References: This issue is a regression of: Bug #33630199.

- All internal ACL bitmask variables are now explicitly 32 bits (`uint32_t`). (Bug #35507223)

- It was not possible to add a functional index on `FIND_IN_SET()`. (Bug #35352161)

- Removed a memory leak observed while running `authentication_kerberos` under Valgrind. (Bug #34482788, Bug #36570929)

- The `gen_range()` function as implemented by the (deprecated) data masking plugin did not always return the correct result.

  This issue affected the data masking plugin only, and did not affect the data masking component which supersedes it. (Bug #34163992)

- In some cases, a `SELECT constant` from an empty table with `ORDER BY COUNT(*)`, when used in a view, did not return any rows. (Bug #115035, Bug #36625752)

- In some circumstances, such when DDL operations were performed on a very large number of tables, the error log was flooded with warnings from background histogram updates; the offending warning was concerning a failure to acquire metadata locks on a table.

  To remedy this problem we now throttle messages written to the error log from background histogram update operations, the rate being capped at one message per minute, which should suffice for the user to identify potential problems with background histogram updates. In addition, we

downgrade all error events that occur during background histogram updates from errors to warnings. (Bug #114845, Bug #36574298)

- Fixed an erroneous comment in `include/my_command.h`.

  Our thanks to Sho Nakazono for the contribution. (Bug #114507, Bug #36455468)

- The range of error numbers for new errors in MySQL 9 has been designated to begin with 6400. (Bug #114414, Bug #36421351)

- When the character set for arguments to a UDF was specified using component services and the argument values passed did not convert cleanly to the desired character set, the UDF ceased executing and returned SQL `NULL`. (Bug #114409, Bug #36420251)

- It was possible for a deterministic stored function to return an incorrect result when the function used `JOIN ON` inside the `return` statement. If the query needed to be reprepared due to a table metadata caused by, for example, `FLUSH TABLES` between two executions, the `ON` clause was sometimes lost. (Bug #114235, Bug #36379879)

- The server rejected a query containing a subquery which referred to a column of the parent table. (Bug #113887, Bug #36262779)

- A query such as the following:

  ```
  SELECT
    (
      SELECT COUNT(1) AS cnt
      FROM t2
      WHERE t2.a = t1.a
      HAVING cnt > 0
    )
  FROM t1;
  ```

  was transformed to this:

  ```
  SELECT COALESCE(derived_1_2.cnt,0) AS cnt
  FROM t1
  LEFT JOIN
    (
      SELECT COUNT(1) AS cnt,
      t2.a AS a from t2
      GROUP BY t2.a
      HAVING (cnt > 0)
    ) AS derived_1_2
  ON derived_1_2.a = t1.a;
  ```

  The presence of a false `HAVING` condition in the subquery should semantically change the correct result of the scalar subquery from zero to `NULL`, which happened as expected for the original query, but not for the transformed case. (Bug #113319, Bug #36070647)

- `SUM(SUBSTRING())` returned a warning as expected, but `SUM(DISTINCT SUBSTRING())` did not. (Bug #113171, Bug #36035064)

- Added the missing `mysql-community-libs-compat` package for the EL8 and EL9 platforms. (Bug #112949, Bug #35975348)

- `SHOW PARSE_TREE CREATE SCHEMA` caused a server exit in debug builds.

  

  Note

  The `SHOW PARSE_TREE` statement is available in debug builds only.

  (Bug #112883, Bug #35964157)

- A different result was obtained when a column reference argument to the `CHAR()` function was replaced with a `CASE()` expression that was essentially the same as the column reference. This took place when the `CHAR()` function was placed in the `WHERE` clause of an outer join, and the column reference was from the inner table of the outer join. An example of such a query is shown here:

  ```
  SELECT 1 AS c_0
  FROM t0
  LEFT JOIN t1 ON t0.vkey = t1.vkey
  WHERE CHAR(CASE WHEN FALSE THEN t1.vkey ELSE t1.vkey END) NOT LIKE 'X';
  ```

  A wrong value was obtained when the column reference was used directly; the `CHAR()` function in the `WHERE` clause was used to convert the outer join to an inner join, although this is correct only when `NULL` as the argument implies a `NULL` result, which is not true of `CHAR()`. The implementation detail that enforces this conversion is that the function's `not_null_tables()` property function returns the map bit of the table, but when the column reference was replaced with the `CASE()` expression, the `CASE()` did not propagate the `not_null_tables()` value to keep the outer join from being optimized improperly to an inner join.

  We fix this problem by setting the `CHAR()` function's `null_on_null` property to false instead of true, which ensures that `not_null_tables()` returns 0 rather than the table's map bit, so that that the outer join is not converted to an inner join when it should not. (Bug #112397, Bug #36118590)