MySQL 9.1 Release Notes

Abstract

This document contains release notes for the changes in MySQL 9.1. For information about changes in a different version of MySQL, see the release notes for that version.

For additional MySQL 9.1 documentation, see the MySQL 9.1 Reference Manual, which includes an overview of features added in MySQL 9.1 (What Is New in MySQL 9.1), 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: 2025-04-16 (revision: 29972)

Table of Contents

Preface and Legal Notices

This document contains release notes for the changes in MySQL 9.1.

Legal Notices

or distribute this documentation in any form or on any media, except if you distribute the documentation in a manner similar to how Oracle disseminates it (that is, electronically for download on a Web site with the software) or on a CD-ROM or similar medium, provided however that the documentation is disseminated together with the software on the same medium. Any other use, such as any dissemination of printed copies or use of this documentation, in whole or in part, in another publication, requires the prior written consent from an authorized representative of Oracle. Oracle and/or its affiliates reserve any and all rights to this documentation not expressly granted above.

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 9.1.0 (2024-10-15, Innovation Release)

- Atomic DDL Notes

- Audit Log Notes

- Authentication Notes

- C API Notes

- Compilation Notes

- Component Notes

- Configuration Notes

- Deprecation and Removal Notes

- SQL Function and Operator Notes

- JavaScript Programs

- Keyring Notes

- MySQL Enterprise Notes

- Performance Schema Notes

- Pluggable Authentication

- SQL Syntax Notes

- Thread Pool Notes

- Trigger Notes

- [Vector Data Type](#)

- [Functionality Added or Changed](#)

- [Bugs Fixed](#)

Atomic DDL Notes

- Previously, `CREATE DATABASE` and `DROP DATABASE` were atomic but were not fully crash-safe. This could lead to the following issues:

  - When `CREATE DATABASE` failed after the database directory was created but before the operation was actually committed, the data dictionary had no record of the database's existence, but the database directory was still present in the file system, necessitating manual cleanup.

  - When removal of the database directory as part of `DROP DATABASE` was unsuccessful due to a file system error or an unexpected shutdown, the transaction containing `DROP DATABASE` was not always rolled back; manual intervention was required in such cases to rectify the problem.

  These statements are now fully atomic and crash-safe, as long as all tables in the affected database use a storage engine (such as `InnoDB`) that supports atomic data definition statements.

  For more information, see [Atomic Data Definition Statement Support](#). (WL #16232)

Audit Log Notes

- The audit log treated a user whose name did not begin with an alphanumeric character (such as `'$foo'`) as invalid, despite the fact that the MySQL server allows such users to be created.

  The same issue also affected MySQL Enterprise Firewall. (Bug #36778917)

Authentication Notes

- **Microsoft Windows:** MySQL `webauthn` plugins now support Windows Hello authentication with Windows 11 and later using the MySQL `authentication_webauthn` plugin; the Windows Hello passkey store is now accepted as as a software "device" (in addition to supported hardware devices). The server plugin is available with MySQL Enterprise Edition only; the client plugin is available with MySQL Enterprise Edition and MySQL Community Edition.

  It is possible both to generate and to authenticate against passkeys stored in the Windows OS passkey store using Windows Hello, and is otherwise compatible with previous MySQL releases.

  A new `mysql` client option `--plugin-authentication-webauthn-device=#` can be used to set the device number plugin option for the `webauthn` client plugin when multiple devices are available. If no such option is specified the first one (`0`) is used by default. The client raises an error if the device specified does not exist.

  Only keys stored in the Windows passkey store are supported by this feature. Other possible backends are not supported with Windows Hello.

  Stored passkey deletion from the windows passkey store must be accomplished using the tools provided by the system; it is not possible using the `mysql` client to delete a passkey.

  For more information, see [WebAuthn Pluggable Authentication](#). (WL #16293)

- Added OpenID Connect support for MySQL Enterprise Edition using the `authentication_openid_connect` server-side authentication plugin.

For additional details, see OpenID Connect Pluggable Authentication. (WL #16269)

C API Notes

- The asynchronous interface used unsafe static local variables to store state information. (Bug #115703, Bug #36891894)

Compilation Notes

- MySQL did not compile on Fedora 41. (Bug #37046924)

- MySQL did not compile on Ubuntu 24.10. (Bug #37042308)

- Added `CONTRIBUTING.md` and `SECURITY.md` files to the MySQL sources to conform to Oracle's Open Source guidelines. (Bug #36998165)

- Aligned `CMAKE_MINIMUM_REQUIRED` with the correct required CMake version (3.14.6) as well as CMake policies for third-party libraries used by MySQL. (Bug #36978193)

- Binary packages that include curl rather than linking to the system curl library have been upgraded to use curl 8.9.1. Important issues fixed in curl version 8.9.1 are described at https://curl.se/docs/security.html. (Bug #36967379, Bug #36955197)

- The included `zlib` library has been upgraded from version 1.2.13 to version 1.3.1. (Bug #36950863)

- CMake options `BUILD_SHARED_LIBS` and `CMAKE_SKIP_INSTALL_ALL_DEPENDENCY` are now set to `OFF` in the top level of the build. (Bug #36930664)

- Pulling in the most recent CMake packages for gRPC and Protobuf caused the builds for these libraries to fail. Fixed by rewriting the package lookup to work for both older and newer CMake packages. (Bug #36905657)

- Upgraded the bundled `libcurl` library to version 8.9.0. (Bug #36886877)

- The bundled `lz4` library was upgraded to version 1.10.0. (Bug #36886747)

- Added CMake code for building and linking with the system Tesseract OCR library. (Bug #36872432)

- Replaced the custom CMake function `MY_TARGET_LINK_OPTIONS` with the builtin function `TARGET_LINK_OPTIONS`. (Bug #36850490)

- The `-DWITHOUT_SERVER` option for CMake did not work on Enterprise Linux 7. (Bug #36824515)

- When compiling MySQL, we no longer check for Fedora versions 28 or 34, since these have reached end of life and are no longer supported. (Bug #36819263)

- Linking with `mold` did not work on Enterprise Linux 9. (Bug #36818902)

  References: See also: Bug #34099162.

- Added the `DISABLE_PERFSCHEMA` build option. When enabled, this has the effect of setting all `DISABLE_PSI_*` options to `ON`. The default for `DISABLE_PERFSCHEMA` is `OFF`. (Bug #36778698)

- The version of `libfido` used with MySQL was upgraded to 1.15.0. (Bug #36752604)

- The RE2 regular expression library is now bundled with the MySQL sources. (Bug #36729026)

- The bundled version of `opentelemetry-cpp` was upgraded to version 1.15.0. (Bug #36708755)

- The version of `clang-format` used for the MySQL codebase was upgraded from 10 to 15. (Bug #36500268)

- `SQL_I_list` uses a member field `next` to track the pointer field to the next object, which is initialized as the address of the member field `first`; this is valid only with the list object constructed with regular constructors. The default assignment operator used memberwise assignment, which is not correct for an empty list.

  This is fixed by supplying a customized move constructor and operator `=` function. In addition, we also change the implementation of `save_and_clear()` to use `std::move()` rather than assignment.

  Our thanks to Xingyu Yang and the Tencent team for the contribution. (Bug #115712, Bug #36891161)

- The bundled version of TI-RPC was upgraded to 1.3.5. (Bug #115698, Bug #36886602)

- It was not possible to build MySQL using Protobuf 22 or later.

  Our thanks to Gordon Wang for the contribution. (Bug #115163, Bug #36678092)

Component Notes

- **Group Replication:** The component `component_group_replication_flow_control_stats`, which provides statistics about transactions throttled by Group Replication flow control, is implemented in this release as part of MySQL Enterprise Edition. Installation of this component requires that the Group Replication plugin already be installed (see Configuring an Instance for Group Replication).

  Flow control throttling statistics are made available as the global status variables `Gr_flow_control_throttle_active_count`, `Gr_flow_control_throttle_count`, `Gr_flow_control_throttle_last_throttle_timestamp`, and `Gr_flow_control_throttle_time_sum`.

  These variables can be observed in `performance_schema.global_status` and the output of `SHOW GLOBAL STATUS`; their values are specific to each group member.

  See Group Replication Flow Control Statistics Component, for more information. (WL #16239)

- The `SHA3-224` algorithm was used when `SHA3-256` was specified as the encryption method. (Bug #36980306)

- The header file `mysql/plugin.h` was included with some logging components, although it is not actually needed by any component. (Bug #34772742)

- The Option Tracker component (`component_option_tracker`) added in this release implements two tables that supply information about options (that is, named features) present in the MySQL server as well as those in components and plugins.

  - The read-only `performance_schema.mysql_option` table contains information about all options installed on the server, including name, component name (or `MySQL Server`), and whether the option is currently enabled.

  - The read/write `mysql_option.option_usage` table shows usage information for each option on the system in `JSON` format. If the server is part of a Group Replication cluster, its server ID and cluster ID are also shown in this table.

  For a given option, the values of the `OPTION_NAME` column of the `mysql_option` table and the column having the same name in the `option_usage` table are the same; you can use these as the join columns in queries against both tables together, like this:

```
mysql> SELECT * FROM performance_schema.mysql_option o
    -> JOIN mysql_option.option_usage u
    -> ON o.OPTION_NAME=u.OPTION_NAME\G
*************************** 1. row ***************************
     OPTION_NAME: Enterprise AUDIT
  OPTION_ENABLED: TRUE
OPTION_CONTAINER: audit_log plugin
      CLUSTER_ID:
       SERVER_ID:
     OPTION_NAME: Enterprise AUDIT
      USAGE_DATA: {"used": true, "usedDate": "2024-10-16T09:14:41Z"}
*************************** 2. row ***************************
     OPTION_NAME: JavaScript Stored Program
  OPTION_ENABLED: TRUE
OPTION_CONTAINER: component:mle
      CLUSTER_ID:
       SERVER_ID:
     OPTION_NAME: JavaScript Stored Program
      USAGE_DATA: {"used": false, "usedDate": "2024-10-15T13:40:03Z"}
```

Several status variables are added by the Option Tracker component; see Option Tracker Status Variables. The component also provides functions for working with Option Tracker usage data, which are described in Option Tracker Functions; you should be aware that the use of these functions requires the `OPTION_TRACKER_UPDATER` privilege, which is also implemented by the component.

The Option Tracker component is available as part of MySQL Enterprise Edition, a commercial offering.

For additional general information about the Option Tracker and the tables described previously, see Option Tracker Component. For a list of components and plugins which support the Option Tracker, see Option Tracker Supported Components. See also The mysql_option Table. (WL #16217, WL #16403, WL #16503)

• This release introduces an Amazon Web Services keyring component (`component_keyring_aws`); this component supersedes the AWS keyring plugin (`keyring_aws`), which is now deprecated and thus subject to removal in a future version of MySQL.

For more information about the AWS keyring component, see Using the component_keyring_aws AWS Keyring Component. For help with migration, see Migration from AWS keyring plugin. For an overview comparing MySQL keyring components with keyring plugins, see Keyring Components Versus Keyring Plugins. (WL #16348)

Configuration Notes

• **Microsoft Windows:** On Windows, *MySQL Configurator* revert button functionality did not always revert to the original value. (Bug #36702176)

• **Microsoft Windows:** On Windows, the *MySQL Configurator* data directory file browsing mechanism required the `ProgramData` folder but can now function outside of that folder. (Bug #36702072)

• **Microsoft Windows:** On Windows, *MySQL Configurator* no longer performs unnecessary background operations after clicking **Next** if the selected options did not change. (Bug #36395605)

• **Microsoft Windows:** On Windows, MySQL Configurator described an empty password as strong rather than weak. (Bug #35533726)

Deprecation and Removal Notes

• **NDB Cluster APIs:** Support for `Node.js`, which was deprecated in NDB Cluster 9.0, is removed in this release. (WL #16246)

- The `keyring_aws` plugin is now deprecated, and subject to removal in a future version of MySQL. Users should migrate to the AWS keyring component, which is introduced in this release.

  For more information about the AWS keyring component, see Using the component_keyring_aws AWS Keyring Component. For help with migration, see Migration from AWS keyring plugin. (WL #16348)

SQL Function and Operator Notes

- The output from `DATABASE()` was truncated when this function was used as part of a `UNION` query. (Bug #36871927)

- `SUM()` yielded a different result in some cases for the same value when `DISTINCT` was also used. For example:

```
mysql> SELECT SUM(b'1100'), SUM(DISTINCT b'1100');
+--------------+-----------------------+
| sum(b'1100') | sum(DISTINCT b'1100') |
+--------------+-----------------------+
|           12 |                     9 |
+--------------+-----------------------+
```

  This happened because, when using `DISTINCT`, a temporary table is used to hold the values so that in the end only the unique values are returned to the user. When creating such table while setting up `DISTINCT`, we did not take into consideration that the data type and length had been determined, and use these, recalculating them instead. Now we use the data type and length already determined. (Bug #115476, Bug #36796197)

JavaScript Programs

- A JavaScript stored program always uses the `utf8mb4` character set, but the value shown in the `COLLATION_CONNECTION` column of `INFORMATION_SCHEMA.ROUTINES` for such a stored program was shown as `latin1`; now it is always set to `utf8mb4_0900_ai_ci` (the default collation for `utfmb4`) instead. This overrides any collation set by the user.

  In most cases, this change does not have any visible effect on the result since the collation used for comparisons is determined based on the columns involved. An exception to this occurs when comparing literal values; `COLLATION_CONNECTION` is used in such cases. For example, if the user has previously set `collation_connection` to `latin1_danish_ci`, the result of `SELECT 'å' < 'ø'` when executed from within a JavaScript procedure may not match the user's expectations. A workaround is to provide the collation explicitly, like this: `SELECT 'å' < 'ø' COLLATE utf8mb4_da_0900_ai_ci`. (Bug #36925906)

- When a prepared statement returned a blob value inside a stored routine, the blob value became invalid after the prepared statement was deallocated. (Bug #36902680)

- Employing a user variable in a DML statement within a stored procedure that used the Statement handle interface sometimes led to an unplanned server exit when any statement was executed following execution of such a stored procedure. To address this issue, we now reset the memory used to the store the user variable and its value to the statement execution memory root after executing a sub-statement from the Statement handle interface. (Bug #36892945)

- `SqlResult` did not support iterators.

  Now it is possible to iterate through a result set in the manner shown here:

```
let result = session.runSql("SELECT * FROM t")

for (let row of result) {
```

```
console.log(row.c1+row.c3);
}
```

(Bug #36835486)

- The Multilingual Engine Component (MLE) now emits Telemetry metrics.

  See mysql.mle Metrics.

  Additionally, SQL queries executed in stored JavaScript procedures are now observable in Performance Schema query digests and SQL statements executed in stored JavaScript procedures are now observable in `events_statements_history_long`. (Bug #36216014, WL #16354)

- The `VECTOR` type is now fully supported by JavaScript stored programs. Vectors can now be used as input arguments, output arguments, prepared statement `bind()` parameters, and return values.

  For more information, see JavaScript Stored Program Data Types and Argument Handling. (WL #16405)

Keyring Notes

- Support for AES ECB, which is insecure, has been removed from all MySQL keyring-related components. (Bug #36749277)

MySQL Enterprise Notes

- **Replication:** This release adds the MySQL Replication Applier Metrics component, which provides users with statistical information about replication formerly logged in the error log. The component adds two tables containing this information to the MySQL Performance Schema: `replication_applier_metrics` provides replication applier metrics for a given replication channel, and `replication_applier_progress_by_worker`: This table provides similar metrics for a specific worker.

  This enhances observability of replication by gathering statistics from the entire replication pipeline, and unifying their presentation. As part of this work, some metrics which were not especially helpful have been replaced with more useful ones.

  For more information about this component, see Replication Applier Metrics Component. (WL #15620)

  References: See also: Bug #32587480.

Performance Schema Notes

- Fixed issues relating to the OTLP exporter. (Bug #36792180, Bug #36783070)

- It is now possible to enable or disable Telemetry meters from the command line or configuration file using the `performance-schema-meter` parameter, on server startup.

  See Server Meters. (Bug #36698082)

- Fixed an erroneous comment in `storage/perfschema/table_host.h`.

  Our thanks to Sho Nakazono for the contribution. (Bug #115844, Bug #36954266)

- Running `SELECT * FROM sys.innodb_lock_waits;` on an instance which was under heavy load affected the performance of the server.

  As of this release, `SELECT * FROM sys.innodb_lock_waits;` fetches only 2 locks for each wait, instead of scanning all locks twice for each wait.

As part of this fix, primary keys were added to `DATA_LOCKS` and `DATA_LOCK_WAITS`. (Bug #100537, Bug #31763497)

- MySQL's OpenTelemetry Logging enables you to export telemetry logs from your MySQL Server to OpenTelemetry backends for analysis. This feature is implemented in the following ways:

  - Telemetry Logging Component: (MySQL Enterprise Edition and MySQL HeatWave, only) collects instrumented log events from the server, formats it in OpenTelemetry's OTLP format, and exports the logs to the defined endpoint using the OpenTelemetry OTLP network protocol. The process listening at the endpoint can be an OpenTelemetry collector or any other OpenTelemtry-compatible backend. If you want to export to multiple backends, you must use an OpenTelemetry collector.

    See Installing OpenTelemetry Support.

  - Telemetry Logging Interface: (MySQL Community Server, Enterprise Edition, and MySQL HeatWave) an API which enables you to define and integrate your own OpenTelemetry Logging components. This interface makes it possible to discover the available logging instrumentation, enable loggers, generate records, and extract the associated trace contexts.

    The interface does not provide logging. You must use MySQL Enterprise Edition, MySQL HeatWave, or develop your own component to provide logging.

    For information on the interface and sample component code, see the *Server telemetry logs service* sections of the MySQL Server Doxygen Documentation.

  (WL #15201, WL #15202)

Pluggable Authentication

- `SET PERSIST authentication_ldap_simple_bind_root_pwd=password` did not actually save the password. (Bug #35478833)

SQL Syntax Notes

- A query such as `SELECT test.dt.a FROM (SELECT 1 AS a) AS dt` uses an identifier chain of the form `db_name.tbl_name.col_name` to reference a column in a derived table, although this behavior contradicts the SQL standard. Such references were also accepted (also wrongly) in table expressions. We now disallow such column references in these cases. (Bug #36704815)

  References: See also: Bug #36542023.

- Support is added in this release for an `IF NOT EXISTS` clause for use with the `CREATE VIEW` statement. `IF NOT EXISTS` has the following effects:

  - If the view named in the statement does not exist, it is created as usual, exactly as if the `IF NOT EXISTS` clause had not been used.

  - If the view already exists, the statement appears to succeed but does not change the view definition, and yields a warning, as shown here:

    ```
    mysql> CREATE VIEW v1 AS SELECT c1, c3 FROM t1;
    Query OK, 0 rows affected (0.01 sec)

    mysql> CREATE VIEW v1 AS SELECT c2 FROM t1;
    ERROR 1050 (42S01): Table 'v1' already exists
    mysql> CREATE VIEW IF NOT EXISTS v1 AS SELECT c2 FROM t1;
    ```

```
Query OK, 0 rows affected, 1 warning (0.01 sec)

mysql> SHOW WARNINGS;
+-------+------+--------------------------+
| Level | Code | Message                  |
+-------+------+--------------------------+
| Note  | 1050 | Table 'v1' already exists |
+-------+------+--------------------------+
1 row in set (0.00 sec)

mysql> SHOW CREATE VIEW v1\G
*************************** 1. row ***************************
                View: v1
         Create View: CREATE ALGORITHM=UNDEFINED DEFINER=`vuser`@`localhost` SQL
SECURITY DEFINER VIEW `v1` AS select `t1`.`c1` AS `c1`,`t1`.`c3` AS `c3` from `t1`
character_set_client: utf8mb4
collation_connection: utf8mb4_0900_ai_ci
1 row in set (0.00 sec)
```

IF NOT EXISTS is not compatible with OR REPLACE, and these two clauses cannot be used together in the same CREATE VIEW statement. Attempting to do so causes the statement to be rejected with a syntax error.

For more information, see CREATE VIEW Statement. (WL #14774)

Thread Pool Notes

- Some concurrent connection attempts were not handled correctly. (Bug #36625082)

Trigger Notes

- **Performance:** Previously, for a table having one or more triggers, the triggers were fully parsed and loaded into memory every time the table was accessed. This often resulted in unexpectedly high resource usage and unreasonably long execution times in scenarios involving many tables, complex triggers, or both, in which no data was changed. It also led to unnecessary resource usage on read-only replicas.

  We solve this problem by dividing the parsing and loading of triggers into two separate phases: In the first phase, we read only trigger metadata, since this can be stored once and shared between different instances of the same trigger. Actual parsing and execution of the trigger is done in the second phase, which is performed only for operations that modify table data; this saves resources because we no longer parse and cache triggers which are not used when executing, for example, a SELECT statement.

  We implement this enhancement by tracking tables having triggers in the table cache separately from tables which do not have triggers. The maximum number of open tables which have fully loaded triggers and which are held in the table cache at any given time is determined by the value of the server system variable table_open_cache_triggers, which is added in this release. This variable is similar to the existing table_open_cache system variable, but controls an additional eviction mechanism, introduced in this release, specific to tables with fully loaded triggers. The default value for table_open_cache_triggers is the same as its maximum value (524288). Setting table_open_cache_triggers to a lower value activates the eviction logic specific to tables with fully loaded triggers. Leaving table_open_cache_triggers at its default values means that tables are not evicted from the cache based on whether they have any fully loaded triggers; this is the same behavior as in previous versions of MySQL.

  A number of server status variables added in this release track cache usage for tables having triggers: Table_open_cache_triggers_hits provides the number of cache hits for open tables with triggers; Table_open_cache_triggers_misses shows the number of cache misses for open tables with

triggers; `Table_open_cache_triggers_overflows` provides the number of cache overflows for open tables with triggers.

In addition, memory usage is also improved by replacing static buffers for trigger errors with variably-sized buffers which are allocated on demand.

Also as part of this work, `SHOW CREATE TRIGGER` (which reads but does not modify data) now displays the complete definition of all triggers, even in cases where trigger bodies cannot be parsed by the server.

For more information, see the descriptions of the server variables mentioned previously. For general information about the table cache, see How MySQL Opens and Closes Tables.

Our thanks to Dmitry Lenev for the contribution. (Bug #44625, Bug #86821, Bug #11753220, Bug #26366333, WL #16455)

Vector Data Type

- The `STRING_TO_VECTOR()` function now allows trailing whitespaces in string representations of `VECTOR` arguments. The parser now trims whitespace characters in such values after a number, before a square bracket, after a square bracket, or any combination of these. Some examples of previously unsupported values which are now permitted are shown here:

  - `"[1 ,2]"`

  - `"[1,2 ]"`

  - `" [1,2]"`

  - `"[1,2] "`

  (Bug #37009633)

Functionality Added or Changed

- **Important Change; InnoDB:** The default value of `innodb_log_writer_threads` is now `OFF` on systems with fewer than 32 logical processors. (WL #16396)

- **Important Change:** For platforms on which OpenSSL libraries are bundled, the linked OpenSSL library for MySQL Server has been updated to version 3.0.15. For more information, see OpenSSL 3.0 Series Release Notes and OpenSSL Security Advisory [3rd September 2024]. (Bug #37021075)

- **Performance; Replication:** The data structure used in tracking binary log transaction dependencies has been changed from `Tree` to `ankerl::unordered_dense::map`, which uses approximately 60% less space, and which should thus contribute to better dependency tracking performance. (Bug #37008442, Bug #37529256)

- **Group Replication:** Added new `INFO` level logging messages in the following cases:

  - When an action begins on all nodes

  - When sending of messages is blocked while waiting for the action to complete

  - When messages are received and the internal message counter is decremented

  (Bug #32929873)

  References: See also: Bug #32762677.

- For consistency of output, spaces have been added before and after the `=` sign in the condition shown for `lookup_condition` by `EXPLAIN FORMAT=JSON` when `explain_json_format_version=2`, as well as those shown for `operation` when using `EXPLAIN FORMAT=TREE`. (Bug #36825515)

- Output from `EXPLAIN FORMAT=TREE` now includes information about the semijoin strategy used, if any. (Bug #36773414)

- Added `"multi_range_read": true` to `EXPLAIN FORMAT=JSON` output, when `explain_json_format_version` is set to `2`, whenever MRR is used for MRR and index range scan access paths. In addition, `(Multi-Range Read)` now appears in the `operation` field in the `EXPLAIN FORMAT=TREE` output for index range scans using MRR. (Bug #36614948)

- Added the `--system-command` option for the `mysql` client, which enables or disables the `system` client command.

  This option is disabled by default, which means that the `system` command is rejected with an error. To enable it, use `--system-command=ON`. (Bug #36377685, WL #16482)

  References: See also: Bug #36248967.

- This release implements the following two status variables for tracking internal temporary table conversions from in-memory to disk:

  - `TempTable_count_hit_max_ram` counts the number of in-memory to disk table conversions due to reaching the `temptable_max_ram` limit with the `TempTable` storage engine. This is only a global status variable.

  - `Count_hit_tmp_table_size` counts the number of in-memory to disk conversions due to reaching the `tmp_table_size` limit with the `TempTable` storage engine, or from reaching the smaller value of `tmp_table_size` and `max_heap_table_size` with the `MEMORY` storage engine. This is both a global and session status variable.

  (WL #16377)

- The maximum `TOTAL_ROW_VERSIONS` value changed from 64 to 255. This value is incremented by executing `ADD COLUMN` and `DROP COLUMN` operations using the `INSTANT` ALGORITHM. (WL #16369)

- Added the following server system and status variables for use in tracking memory usage by queries when `global_connection_memory_tracking` is enabled; the first two variables in the list shown here apply globally, and the second two to individual user connections:

  - `global_connection_memory_status_limit`: This system variable determines the maximum total amount of memory that can be consumed by all user connections to the server. When the total amount of memory used by all queries exceeds this amount, this causes `Count_hit_query_past_global_connection_memory_status_limit` to be incremented.

  - `Count_hit_query_past_global_connection_memory_status_limit`: This status variable tracks the number of times that queries using any connection have caused total memory consumption to exceed `global_connection_memory_limit`. This is incremented only when a query brings the total memory consumption from less than `global_connection_memory_limit` to greater than this amount.

  - `connection_memory_status_limit`: This system variable determines the maximum amount of memory that can be consumed by a given user connection before incrementing `Count_hit_query_past_connection_memory_status_limit`.

- **Count_hit_query_past_connection_memory_status_limit**: This status variable tracks the number of times that queries using the current connection have caused total memory consumption to exceed `connection_memory_limit`. This is incremented, for a query executed within the connection, only when such a query brings the total memory consumption from less than `connection_memory_limit` to greater than this amount.

These variables are intended for testing configurations prior to putting them into production, and do not cause queries to be rejected as when exceeding `connection_memory_limit`. (WL #16097)

Bugs Fixed

- **Performance; InnoDB:** Several functions internal to `InnoDB`, which were defined as inline in MySQL 8.0.28, were found to be no longer inline in MySQL 8.0.33, due in part to refactoring which accompanied improvements made in MySQL 8.0.30 to improve the `InnoDB` adaptive hash index. This had an adverse effect on queries using joins on `InnoDB` tables. (Bug #111538, Bug #35531293)

  References: This issue is a regression of: Bug #81814, Bug #16739204, Bug #23584861.

- **InnoDB:** Removed the `DEBUG_SYNC_C_IF_THD` macro in favor of `DEBUG_SYNC`. (Bug #36870544)

- **InnoDB:** InnoDB did not allow updating a `REDUNDANT` table column that was altered with `NULL` as the default value using the `INSTANT` algorithm. Instead, MySQL unexpectedly halted. (Bug #36840107)

- **InnoDB:** Improved error handling for bulk load operations with tables containing a mix of `BLOB` and other data types. Previously, the error type was reported as an empty string. (Bug #36764710)

- **InnoDB:** When redo log capacity was reduced with `log_files_truncate`, under rare circumstances the `file.end_lsn` equaled `log_sys->write_lsn` thus disallowing future redo log writes. (Bug #36730830)

- **InnoDB:** It was possible for the MySQL server to halt unexpectedly when executing a `DELETE` or `UPDATE` statement after a column was dropped using the `INSTANT` algorithm. (Bug #36723117)

- **InnoDB:** FTS index optimization would sometimes not function correctly with tokens equal in collation order but different in terms of bytes. (Bug #36652127)

- **InnoDB:** The log index size calculation now accounts for column order changes. (Bug #36526369)

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

- **InnoDB:** Replaced `std::this_thread::sleep_for(std::chrono::seconds(0))` usage with `std::this_thread::yield()` to prevent it from being optimized out while compiled. (Bug #36522343)

- **InnoDB:** The upgrade process unexpectedly halted when the database contained a full-text index created in MySQL 5.7 with a user-created `FTS_DOC_ID` column as the primary key.

  Our thanks to Huaxiong Song and the team at Alibaba for contributing to this fix. (Bug #36496164)

- **InnoDB:** Disabled the optimizer for indexes with range conditions on multiple-value virtual columns. (Bug #36341532)

- **InnoDB:** Redesigned the performance schema `data_locks` and `data_lock_waits` tables so that querying them does not require an exclusive global mutex on the transaction or lock system. It now iterates over buckets of hash tables that hold the locks to only latch the actively processed shard, when previously it iterated over the transactions. This also improves the iteration logic complexity in terms of speed and memory to decrease the impact of these queries on the rest of the system.

Note that the query result might show an incomplete list of transaction locks if it committed, started, or otherwise changed the set of owned locks in-between visiting two buckets. This differs from previous behavior which always showed a consistent snapshot of locks held by individual transactions, although two different transactions could have been presented at different moments. In other words, the new approach gives a consistent view of a single wait queue to show conflicting locks with a waiting lock because they are always in the same bucket, while the old approach could miss some of them because they belonged to other transactions. The old approach would always show all the other locks held by a reported transaction but could miss locks of other transactions even if they were conflicting. (Bug #36302624)

- **InnoDB:** Fixed a memory leak in the bulk loader. (Bug #35988311)

- **InnoDB:** For tables created with an index on a column that was too wide for the redundant row format (allowed before MySQL 5.7.35), an in-place upgrade silently imported the table but it was not accessible, which interfered with making backups. Now all operations that involve using the invalid index are rejected with `ER_INDEX_CORRUPT` until the index is dropped. An `ER_IB_INDEX_PART_TOO_LONG` error is also reported in the error log. (Bug #35869747)

  References: See also: Bug #34826861.

- **InnoDB:** A schema mismatch was possible when a table was imported with a different `sql_mode` than when it was created, because strict mode influences the number of nullable columns in a table's secondary index. Now the secondary index trees are also checked for corruption after a tablespace is imported. (Bug #35799038)

- **InnoDB:** Given a table with an FTS index, when its tablespace is discarded the corresponding FTS tables are also dropped. When performing an operation that cleared out these tables, the code incorrectly assumed that the FTS tables must also exist if the FTS indexes were present in the table metadata. (Bug #35343458)

- **InnoDB:** An uninitialized buffer was being written to a temporary file when checking if the system supports a different sector size for FusionIO. This check is made when `innodb_flush_method` is set to `O_DIRECT` or `O_DIRECT_NO_FSYNC`. (Bug #115229, Bug #36705034)

- **InnoDB:** Table rebuild operations involving secondary indexes required greater file I/O for `InnoDB` temporary files as compared with MySQL 8.0.26, which degraded query performance. (Bug #114465, Bug #36444172)

- **InnoDB:** Adding indexes with the parallel index builder was much slower with the Performance Schema enabled due to issues in the `Alter_stage` class. (Bug #113505, Bug #36163502)

- **InnoDB:** Fixed an `innodb.parallel_read_kill` related unit test to better account for recent optimizer changes, and fixed an `innodb.ddl_kill` unit test and its associated macro usage.

  Our thanks to Dmitry Lenev and the team at Percona for contributing to these fixes. (Bug #113002, Bug #115416, Bug #35992036, Bug #36764973)

  References: See also: Bug #112767.

- **Partitioning:** `ALTER TABLE` did not always work correctly with partitioned tables. (Bug #36677952)

- **Group Replication:** Under certain conditions, all secondaries shut down unexpectedly following a brief period of network inactivity on the primary host. (Bug #35642087)

  References: See also: Bug #32673639, Bug #34565855.

- **Group Replication:** Memory aggregated by threads as reported did not account for all `memory/sql/Gtid_set::Interval_chunk` freed by other threads from what was allocated, leading to a incorrect ever-increasing consumption of resources by Group Replication thread `thread/group_rpl/THD_applier_module_receiver`.

  Global memory as reported by `performance_schema.memory_summary_global_by_event_name` was not affected by this issue. (Bug #34819861)

- **Group Replication:** Removed a potential memory leak from `xcom\network\xcom_network_provider_native_lib.cc`. (Bug #115162, Bug #36673883)

- The server did not always handle connections correctly when running with both the thread pool and audit log plugins. (Bug #37039422)

- Packages for Debian-based systems are now built with `WITH_ZLIB=system`, and disable `MYSQL_MAINTAINER_MODE` for debug builds. (Bug #37038213)

- Updated the Kerberos library bundled with commercial builds to version 1.21.3. For more information, see the Kerberos 5 1.21.x Release Notes. (Bug #37034600)

- For a subquery which uses window functions in its `SELECT` list, an `EXISTS()` transformation cannot be used; this means that, for such a subquery, the transformation used is always materialization, but this is true only when the subquery uses one of `IN`, `ALL`, `ANY`, or `SOME`. The check for whether materialization is forced (`IsForcedMaterialization()`) for a subquery returned true when the query had an `IN` subquery whose `WHERE` clause used an `EXISTS` subquery containing window functions, due to the fact that the innermost subquery having window functions always returned its strategy as materialization, even though the `EXISTS` strategy had already been chosen for the `IN` subquery. As a result, some checks failed when this subquery was considered for materialization.

  We fix this by adding a check for a quantified comparison predicate in `IsForcedMaterialization()`, so that the subquery with a window function is not looked into. (Bug #37008930)

- Removed a heap-use-after-free warning in `regexp::Regexp_facade::~Regexp_facade()`. (Bug #36867806)

- Some combinations of `EXCEPT` and `INTERSECT` led to an error in `Item::convert_charset()`. (Bug #36838373)

- While dumping tablespaces, `mysqldump` did not properly escape certain SQL statements in its output. In addition, the dump now encloses the following identifiers within backticks: `LOGFILE GROUP`, `TABLESPACE`, and `ENGINE`. (Bug #36816986)

- Eliminated unnecessary copying in `StoreLinkedImmutableStringFromTableBuffers()`, improving the performance of some hash joins by 5 to 10 percent. (Bug #36805260)

- A previous fix for an issue in `sql/sql_executor.cc` checked for `const_item()` rather than `const_for_execution()`. (Bug #36804785)

  References: This issue is a regression of: Bug #34951115.

- When calculating `Last_query_cost`, the optimizer did not always take the costs of all subqueries into account. (Bug #36790906)

- When executing an index range scan using `IndexRangeScanIterator` the record buffer was never set due to the fact that the data type used for `IndexRangeScanIterator::m_expected_rows` had been changed from double to boolean during refactoring. This unintended change has been reverted.

  In addition, when the record buffer was enabled for index range scans a problem arose with multi-valued indexes used for covering these scans. (This is possible only when they are used as part of index merge scans, since index merge scans force covering index scans.) The source of the problem was that the implementation of `Field_typed_array::key_cmp()` needs the value of the generated column for the indexed expression, and this column is not available in the multi-valued index, so the storage engine cannot safely evaluate the end range condition when filling the record buffer for a covering scan. To fix this, we now disable the record buffer when multi-valued indexes are used for covering index range scans. (Bug #36775910)

  References: See also: Bug #36341532.

- Certain triggers and stored procedures were not handled correctly. (Bug #36775910)

- Some errors raised when referencing external tables were not handled correctly. (Bug #36758378)

- When the argument to `COUNT()` used as a window function was a `GROUP BY` expression and `WITH ROLLUP` was used, `COUNT()` returned `NULL` instead of `0`. (Bug #36750571)

- On Debian, stopped stripping `libmysqlclient.a` to increase LTO build compatibility. (Bug #36737581)

- The internal function `my_convert_internal()` sometimes returned a pointer rather than the intended value. (Bug #36684463)

  References: This issue is a regression of: Bug #36479091.

- Removed a memory issue in the server component. (Bug #36600205)

- Fixed a memory leak in the `mysql` client. (Bug #36600203)

- The `AES_ENCRYPT()` function did not always return a valid result. (Bug #36593265)

- In some cases a join involving a very large number of rows and many tables was not processed correctly. (Bug #36562979)

- Preparation of table value constructors did not track the number of hidden columns added for expressions in an `ORDER BY` clause. This could cause problems later in the resolution process, possibly leading to an unplanned server exit.

  We fix this by counting the number of hidden items in table value constructors in the same way as this is done in other query blocks. (Bug #36560156)

  References: This issue is a regression of: Bug #35785452.

- `DROP INDEX` with the addition of a `FULLTEXT` index in the same transaction sometimes led to an unplanned server exit. (Bug #36559642)

- Fixed an issue related to the handling of FTS indexes during an upgrade. (Bug #36526587)

- Some combinations of optimizer hints did not function correctly. (Bug #36492114)

- Following the fix for a previous issue, a `const` item that is part of a `GROUP BY` and is not found in the select list is not added as a hidden item to the field list, but this was not taken into consideration while checking for replacements for expressions involving `ROLLUP` operations while creating a temporary

table. We fix this by checking whether the item that is not found in the field list is a `const` item, and if so, we use the same item as a replacement. (Bug #36444257)

References: This issue is a regression of: Bug #34951115.

- Removed assertions found during testing of the data masking functions `gen_rnd_pan()` and `mask_ssn()`. (Bug #36397869, Bug #36398272)

  References: See also: Bug #36398221.

- If an expression involved in condition pushdown had one view reference which was also an outer reference and one local reference, then dependency consistency checks failed while cloning the condition.

  We solve this by traversing the item tree prefix and postfix. While traversing the prefix, we set the `Item_ref` object to be used for setting dependency and context information for the underlying columns of the reference object; while traversing the postfix, we clear this object. (Bug #36368181)

- Some prepared statements were not reprepared correctly. (Bug #36267792, Bug #35712413)

  References: This issue is a regression of: Bug #34929930.

- Fixed an issue relating to FTS comparisons.

  Our thanks to Shaohua Wang and the team at Alibaba for the contribution. (Bug #36210202)

- RelWithDebInfo builds are no longer compiled with the "-g1" flag, thus increasing the available debugging information and generated file sizes. (Bug #36111629)

  References: See also: Bug #33664929.

- Corrected potential misbehavior when the server was run with `--skip-grant-tables`. (Bug #36043213)

- When converting a string to a `MYSQL_TIME` struct, we did not always verify that the day number was less than or equal to 31. (Bug #35884337, Bug #36633426)

- Fixed a regression in an earlier fix for a problem with references to columns from tables of outer query blocks in the `ON` condition of a join. (Bug #35854686)

  References: This issue is a regression of: Bug #96946, Bug #30350696.

- In certain rare cases, a `CREATE TABLE` statement involving the creation of a table with foreign keys was not processed correctly. (Bug #35553557)

- Fixed an issue relating to `SHOW INDEX` and generated columns. (Bug #35497623)

- Removed a memory leak from the `FEDERATED` storage engine that was discovered during testing of queries using index merge plans. (Bug #35362984)

- In some cases, table DDL in prepared statements was not handled correctly. (Bug #35221658)

- A statement which called a stored routine containing an aggregate function, whose result was stored in a variable local to the routine, raised an assert in `sql/sql_lex.h`. (Bug #35102083)

- In a `UNION`, it was sometimes possible for one of the `SELECT` statements involved to reference a column in an `ORDER BY` clause when this column was not among those accessed by that `SELECT`. (Bug #35058815)

- Running two concurrent `OPTIMIZE TABLE` statements on the same table with fulltext indexes and `innodb_optimize_fulltext_only` enabled sometimes caused the server to exit. (Bug #34929814)

- A left join with an impossible condition as part of an `ON` clause was not optimized as in MySQL 5.7, so that, in later versions, the query executed more quickly without the impossible condition than with it. An example of such a query, impossible condition included, is `SELECT * FROM t1 JOIN t2 ON t1.c1=t2.c1 AND 1=2`. (Bug #34668756)

- The XML function `ExtractValue()` did not handle arguments containing accented characters correctly. (Bug #117778, Bug #36236440)

- `NOT` worked correctly with a quantified comparison such as `< ALL`, `> ANY`, and so on, but `NOT NOT` did not. (Bug #115962, Bug #37004689)

- A `GROUP BY` query containing a `LATERAL` subquery raised a spurious `Duplicate column` error. (Bug #115696, Bug #36885503)

- Changed the SLES 15 target platform from openSUSE 15.5 to 15.6, which also means using GCC 13 instead of GCC 12 and building against the system's OpenSSL 3.x instead of OpenSSL 1.1.x. (Bug #115535, Bug #36934913)

- Added `gcc-toolset-12` to the EL8 and EL9 build specification file to allow building the MySQL 8.0 `mysql-community-libs-compat` RPM.

  Our thanks to Simon Mudd for the contribution. (Bug #115484, Bug #36796229)

- Fixed a typo in the Debian package description.

  Our thanks to Henning Pöttker for the contribution. (Bug #115363, Bug #36749142)

- `MYSQL_TYPE_JSON` was missing from the description of binary resultsets in the MySQL source code documentation.

  Our thanks to Daniël van Eeden for the contribution. (Bug #115360, Bug #36740656)

- Certain complex `UNION` queries triggered an assertion in debug builds, and caused release builds to hang. (Bug #115346, Bug #36739383)

- A query such as `SELECT CASE WHEN 1 THEN NOW() ELSE x.x END FROM (SELECT NULL) x(x)` led an assert. This occurred because a generated column description used a temporal type with a fractional seconds precision of 31, which is an invalid precision value. The invalid precision came from type aggregation, where one of the underlying items to the `CASE` expression was a `NULL` expression, which has a decimal precision equal 31, meaning an undefined precision.

  The problem is fixed by not aggregating types from expressions that yield `NULL`.

  This issue did not occur with numeric and string arguments, since they cannot be aggregated directly into a temporal type.

  This issue did not occur with MySQL 8.0 or with MySQL 8.4 release binaries. (Bug #115233, Bug #36705061)

- A query, which normally used an index range scan, used a less efficient index lookup when run as a prepared statement. (Bug #115227, Bug #36705030)

- A singly-nested derived table was wrongly merged rather than materialized. (Bug #115215, Bug #36695371)

- A `ROLLUP` query did not take account a summary `NULL`. Consider the following statements:

```
CREATE TABLE t(a INT);

SELECT a, COUNT(*) FROM t GROUP BY a WITH ROLLUP;
```

  According to the SQL standard, the `SELECT` statement should be equivalent to a `UNION` query which produces the result `(0, NULL)`. This is because, even with an empty result set, `ROLLUP` should give us a single grouping row with `NULL` entries for all `GROUP BY` expressions as well as a grand total for any aggregates, in this case `0` for `COUNT(*)`, but MySQL returned an empty set instead.

  Now we return the grouping row in such cases. (Bug #114638, Bug #36514339)

- Incorrect results were sometimes obtained when using `DISTINCT` and `ORDER BY` with a derived table. (Bug #114589, Bug #36496160)

- A `CREATE TABLE` statement with a `CHECK` constraint that involved a `CAST` with an `AT TIME ZONE` clause failed with the confusing error `Missing time zone function`. This was because the text of the `CHECK` constraint was regenerated internally and subsequently misinterpreted.

  We fix this by ensuring that the correct `CHECK` constraint text is stored in the data dictionary along with the rest of the `CREATE TABLE` definition. (Bug #114404, Bug #36423829)

- With `prefer_ordering_index=off`, a query with no reference key reverted to scanning and sorting the full table even though it should have been possible to avoid the sort.

  Our thanks to Daniel Nichter for the contribution. (Bug #113699, Bug #36213938)

- A join on subqueries which themselves used subqueries in their `HAVING` and `FROM` clauses led to an assert in `sql/item.h`.

  The problem occurred when removing a scalar subquery with a `HAVING` clause after having decided that the subquery was part of a predicate that could be eliminated. The `HAVING` clause pointed to an aggregation object (`Item_sum_min`), indirectly accessed through an `Item_aggregate_ref` that was added by `Item::split_sum_func2()`. When removing the subquery, the reference count for the `Item_aggregate_ref` was decremented without ever having been incremented.

  We fix this by incrementing the object's reference count when it is first created. (Bug #112615, Bug #35877063)

  References: See also: Bug #111492, Bug #35517962.

- Queries using `CASE (WHEN TRUE AND EXISTS (`*`subquery`*`))` did not always return the same result as with `CASE (EXISTS (`*`subquery`*`))`. (Bug #112557, Bug #35855294)

  References: This issue is a regression of: Bug #32644631, Bug #32802301.

- A character string literal value selected through a derived table and matched with a column with a different character set through a `UNION` operation sometimes raised `ER_CANT_AGGREGATE_2COLLATIONS` (`Illegal mix of collations...`). (Bug #108627, Bug #34646522)

  References: See also: Bug #36812010, Bug #36844420.

- A query using a greater-than (`>`) or less-than (`<`) comparison with a multi-valued index executed much more slowly than the same query using an equality (`=`) comparison with the same index. (Bug #104897, Bug #33334911)