
MySQL Shell 8.0 Release Notes

Abstract

This document contains release notes for the changes in each release of MySQL Shell 8.0.

For additional MySQL Shell documentation, see <http://dev.mysql.com/>.

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 (<http://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 either the [MySQL Forums](#) or [MySQL Mailing Lists](#), where you can discuss your issues with other MySQL users.

Document generated on: 2018-07-14 (revision: 15605)

Table of Contents

Preface and Legal Notices	1
Changes in MySQL Shell 8.0.12 (Not yet released)	3
Changes in MySQL Shell 8.0.11 (2018-04-19, General Availability)	3
Changes in MySQL Shell 8.0.5 - 8.0.10 (Skipped version numbers)	9
Changes in MySQL Shell 8.0.4 (2018-01-25, Release Candidate)	9
Changes in MySQL Shell 8.0.3 (2017-09-29, Development Milestone)	15
Changes in MySQL Shell 8.0.2 (Skipped)	19
Changes in MySQL Shell 8.0.1 (Skipped)	20
Changes in MySQL Shell 8.0.0 (2017-07-14, Development Milestone)	20

Preface and Legal Notices

This document contains release notes for the changes in each release of MySQL Shell 8.0.

Legal Notices

Copyright © 1997, 2018, Oracle and/or its affiliates. All rights reserved.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

This documentation is NOT distributed under a GPL license. Use of this documentation is subject to the following terms:

You may create a printed copy of this documentation solely for your own personal use. Conversion to other formats is allowed as long as the actual content is not altered or edited in any way. You shall not publish 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

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 Shell 8.0.12 (Not yet released)

Version 8.0.12 has no changelog entries, or they have not been published because the product version has not been released.

Changes in MySQL Shell 8.0.11 (2018-04-19, General Availability)

- [Functionality Added or Changed](#)
- [Bugs Fixed](#)

Functionality Added or Changed

- **Important Change; Microsoft Windows:** Before installing MySQL Shell, make sure you have the Visual C++ Redistributable for Visual Studio 2015 (available at the [Microsoft Download Center](#)) installed on your Windows system. This now applies for both Community and Commercial versions of MySQL Shell.
- **Important Change:** The way which document IDs are generated has been changed. Now document IDs are generated by the server, rather than by the client. As a result the `getLastDocumentID()`, `getDocumentId` and `getDocumentIDs()` methods have been removed. To get a list of the document IDs automatically generated by an 8.0.11 and later server, use `Result.getGeneratedIDs()`. The type of column generated for the document ID in a collection has changed from `VARCHAR(32)` to `VARBINARY(32)`. The generated document ID can be overridden manually by including an ID but you must respect the server generated IDs to avoid conflicts. If you are using InnoDB cluster, use the `mysqlx_document_id_unique_prefix` variable to ensure your documents can be moved between replicaset.

Now, if you are adding documents to a collection on a server running a version of MySQL earlier than 8.0.11, you must manually include a document ID as these versions do not add the ID automatically.

- The indexing of collections has been improved to make large collections of documents more efficient to navigate. Now, you can create an index based on one or more fields found in the documents in the collection, using a JSON document which maps fields from the collection's documents to MySQL types. The majority of MySQL types are supported, in addition to spatial indexes and GeoJSON data.
- Support for the `NOWAIT` and `SKIP LOCKED` InnoDB locking modes has been added to the lock operations. Now you can use these locking modes with the `lockShared()` and `lockExclusive()` methods, for example:

- `Table.select().lockShared([LockContention])`
- `Table.select().lockExclusive([LockContention])`
- `Collection.find().lockExclusive([LockContention])`
- `Collection.find().lockExclusive([LockContention])`

where `LockContention` can be one of:

- `DEFAULT` - if the function encounters a row lock it waits until there is no lock
- `NOWAIT` - if the function encounters a row lock it aborts and generates an `ER_LOCK_NOWAIT` error
- `SKIP_LOCKED` - if the function encounters a row lock it skips the row and continues

For more information see [Locking Read Concurrency with NOWAIT and SKIP LOCKED](#).

- MySQL Shell can now connect to a MySQL server with an account that uses the `caching_sha2_password` authentication plugin. Assuming the server is configured for encrypted connections, you can use such accounts over both X Protocol and the classic MySQL protocol. See [Using Encrypted Connections](#).

**Important**

If you are not using encrypted connections, to connect over X Protocol with an account that uses the `caching_sha2_password` authentication plugin, the user's password must be stored in the cache. Currently there is no way to store the password over X Protocol if not using encrypted connections.

When using the classic MySQL protocol for connections with such an account and unencrypted connections, you can configure MySQL for password exchange using an RSA key pair. MySQL Shell supports such connections and the following command options have been added:

- Use the `--server-public-key-path` option to specify the RSA public key file.
- Use the `--get-server-public-key` option to request the public key from the server.

For more information, see [Caching SHA-2 Pluggable Authentication](#).

- InnoDB cluster instances running MySQL 8.0.11 and higher can now be configured for InnoDB cluster usage remotely, without the need to log in to the instance and run `dba.configureLocalInstance()` locally. Use the new `dba.configureInstance()` operation, which enables remote automatic configuration of compatible instances for InnoDB cluster usage and if necessary also supports remote restart of compatible instances after configuration.

Similarly, compatible instances support having any changes to their settings automatically persisted by AdminAPI after any cluster topology changes. Now, when working with remote instances if you create a cluster, add an instance to a cluster, or remove an instance from a cluster supporting this functionality, there is no need to log in to the instance and run `dba.configureLocalInstance()` to persist the changes to the instance's option file. This makes working with production clusters, which are built on networked instances, much easier. This also ensures that instances automatically rejoin a cluster in the case of a restart.

As part of this feature the following improvements were made:

- Improved display of messages.
- Improved detection of loopback hostnames on Debian and Ubuntu, and local instances based on the hostname.
- Improved display of required configuration changes.
- Replaced automatic configuration of instances after issuing `dba.createCluster` and `dba.addInstance` with the configuration check `dba.checkInstance()`, aborting the command if the check fails.
- When specifying an account in interactive mode, if the user's hostname is local, for example localhost, you are given the option to recreate the account configured for remote usage by using the hostname `%`, or to create a new InnoDB cluster administrator account.

References: See also: Bug #27608299, Bug #27112727, Bug #27629803.

- MySQL Shell now has a configuration file which stores configuration changes across sessions. Use the new `\option` MySQL Shell command for querying and changing configuration options. Alternatively use the following methods with the `shell.options` object:

```
shell.options.set_persist(optionName, value)
shell.options.unset_persist(optionName, value)
```

In addition the new `defaultMode` option has been added, which enables you to configure the programming language which MySQL Shell starts up in. You can override the default mode using the command options.

- The `util.checkForServerUpgrade([uri])` operation has been extended to check for the following incompatible features:
 - obsolete `sql_mode`
 - partitioned tables in shared tablespaces
 - removed functions

Bugs Fixed

- When the `\quit` command was used to exit MySQL Shell, this event could be noted in the error log as an aborted connection. The issue has now been fixed. (Bug #27821045, Bug #90281)
- When the MySQL Shell command-line option `--json=raw` was used, the output was actually provided in pretty-printed format, and an empty string was displayed in place of error messages. These issues have now been corrected. (Bug #27733996, Bug #26737357)
- When MySQL Shell commands were executed from a script, interactive prompting for passwords and confirmations was not available. Now, interactive prompting is enabled by default when the commands are used in a script, as it is when they are used on the command line. The `--no-wizard` command line option disables interactive prompting for MySQL Shell commands used in both ways. (Bug #27702250)
- After creating a cluster administrative user using `dba.createCluster()`, MySQL Shell attempted to reconnect using the new user but still using localhost, which failed causing the whole configuration to fail. Now MySQL Shell no longer switches accounts between administrative account creation and instance configuration. (Bug #27673816)
- The changes introduced by Bug#27545850 mean that now it is not possible to modify Group Replication related options while the plugin is starting or stopping. The AdminAPI operations have been modified to ensure that they respect this change, and attempting to issue a statement which would change a Group Replication operation while the plugin is starting results in an error. There is no check for when the plugin is stopping and so you should ensure you do not attempt to configure an instance if there is a chance the plugin is stopping. (Bug #27545850)
- The `util.checkForServerUpgrade()` operation now prompts the user for a password interactively if the required password is not provided along with the command. If the `--no-wizard` option has been used to disable the connection wizard, missing credentials instead result in an error and the function is not executed. (Bug #27514395)
- The `util.checkForServerUpgrade()` operation was requiring the wrong privileges for the user passed to the function. The user now requires `ALL` privileges, and does not require the `GRANT OPTION` privilege. (Bug #27506702)

- The `util.checkForServerUpgrade()` operation was rejecting host names that included the % (percent) symbol or were specified as a numeric IP address. (Bug #27506079, Bug #27513260)
- When rejoining an instance to a cluster the displayed output has been improved. (Bug #27437389)
- The `dba.createCluster()` function fails if used on an instance with `innodb_page_size=4k`, because the `instance_name` column of the instances table in the InnoDB Cluster metadata uses a `VARCHAR` of size 256, which is too large for the `innodb_page_size=4k`. However, the size of this column cannot be changed because it is used to hold hostnames of the cluster members which can have a length up to 255 bytes. This limitation is now validated when using `dba.createCluster()`, `dba.checkInstanceConfiguration()`, `dba.configureInstance()`, and `dba.configureLocalInstance()`. (Bug #27329079)
- The AdminAPI commands which manage sandboxes failed with an error when using a sandbox directory with non-ASCII characters in the path. This occurred when the user name had a non-ASCII character because the default sandbox directory is a subdirectory of the `$HOME` or `%userprofile%` path, which is usually based on the current user name. The fix adds Unicode support to the internal provision tool used by AdminAPI. (Bug #27181177)
- MySQL Shell now sets the value of `group_replication_local_address` based on `port` by calculating the result of `((port * 10) + 1)`. The `port` variable defaults to 3306, in which case MySQL Shell sets `group_replication_local_address` to `((3306 * 10) + 1)`, resulting in port 33061 instead of the previous default of 13306. (Bug #27146799)
- The online help for all AdminAPI commands which require a connection displayed detailed information on connection data, which cluttered up the details of each command. Now, the help displays an overview of connection details and information about where else to get more help. (Bug #27146290)
- When creating a cluster on a Debian type platform with the default system configuration, the `cluster.addInstance()` command failed with an error if the instance hostname was used in the connection parameter. This happened because on these platforms the hostname is resolved to the IP address 127.0.1.1 by default, but this IP address is not supported by the Group Replication Group Communication Service (GCS). The fix adds a validation to the `dba.createCluster()` and `cluster.addInstance()` functions to verify if the connection hostname resolves to 127.0.1.1 and issue an error in that case. (Bug #27095984)
- The `Cluster.forceQuorumUsingPartitionOf()` operation was not working with a non-root user, even if that user had all the required privileges. The fix ensures that the specified user is used to connect to all instances, instead of the root user. (Bug #27089930)
- Issuing `dba.createCluster()` in interactive mode as a user which did not have all the required privileges resulted in an unexpected halt. The error message generated when issuing `dba.configureLocalInstance()` and `dba.checkInstanceConfiguration()` if the account being used did not have the required privileges has also been improved. (Bug #27076753, Bug #27324699)
- When using `adoptFromGR` to convert a Group Replication group into an InnoDB cluster, the output recommended adding instances. This message has now been improved to show that the instances were already added. (Bug #27061615)
- By default, MySQL Shell connections are assumed to require a password, which is requested at the login prompt. A new MySQL Shell command-line option `--no-password` is provided to explicitly specify that no password is used, and to disable the password prompt. The `--no-password` option can be used if socket peer-credential authentication is in use (for Unix socket connections), or for any authentication method where the user has a password-less account (though note that this situation is insecure and not recommended).

The methods previously provided by MySQL Shell for specifying that no password is used for the connection are still valid, and can be used instead of the `--no-password` option. These methods are as follows:

- If you are connecting using a URI type string, place a `:` after the user name in the URI type string but do not specify a password after it.
- If you are connecting using individual parameters, specify the `--password=` option with an empty value.

(Bug #26986360)

- In interactive mode the `cluster.removeInstance()` AdminAPI function was not accepting the second parameter with the options dictionary. This prevented use of the `force` option, which failed with an error. (Bug #26986141)

References: See also: Bug #27572618.

- The `cluster.addInstance()` function was not checking the validity of the `server_uuid` being added to the cluster. Now, the `server_uuid` of an instance joining the cluster is checked to be unique, and if it is not an error is generated and the instance is prevented from joining the cluster. (Bug #26962715)
- When setting up a cluster a replication user is created to enable distributed recovery. If MySQL Shell logging was enabled, the `CREATE USER` statements were not being correctly added to the log. (Bug #26938488)
- When an error occurred while returning a result set, the fetch operation was interrupted but the associated error was not reported. The error is now reported correctly. (Bug #26906527)
- The `dba.checkInstanceConfiguration()` and `dba.configureLocalInstance()` operations were not correctly reporting any errors related to incorrect configuration of the instance's `server_id`. Now, when an error is reported it is displayed in the list of variables not meeting the InnoDB cluster requirements.

In addition, the X Plugin load has been removed from the `my.cnf` created by AdminAPI for sandbox instances running MySQL version 8.0.11 and later because X Plugin is installed by default for those versions. (Bug #26836230)

- When using `dba.configureLocalInstance()` with the `clusterAdmin` option to create a user which can administer the cluster, the created account had too many privileges. (Bug #26737608)
- Support for microseconds has been added to the `mysqlx.dateValue()` function and the `Date` object. (Bug #26429497)
- Argument validation and error messages were improved for the `mysqlx.dateValue()` function. (Bug #26429426, Bug #26429377)
- The `db` global object is not available for connections in SQL mode. This reference has been removed from the message returned by the `\connect` command in that situation. (Bug #26428665)
- When you create a cluster and add instances to it internal users are created, which are required by Group Replication for distributed recovery when instances join the cluster. These automatically generated replication users were not being removed from instances after removing them from the cluster or after dissolving the cluster. (Bug #26395608)

- The handling of SQL wildcard characters was corrected for the X DevAPI `Schema.getTable()`, `Schema.getCollection()` and `Session.getSchema()` functions. (Bug #26392984)
- A shortcut to uninstall MySQL Shell is no longer provided in Microsoft Windows menus in the group of installed MySQL programs. Uninstallation of MySQL Shell should be handled through MySQL Installer. (Bug #26317449)
- If an instance contained InnoDB cluster metadata but was stand alone, in other words it did not belong to a cluster, it was not possible to use `dba.dropMetadataSchema()`. (Bug #26315635)
- The runtime timer for MySQL Shell, which reports the time taken for each query execution, has been refactored to provide increased precision of 4 digits for fractional seconds. (Bug #25976636, Bug #86135)
- In the event of a disconnection, MySQL Shell did not reconnect to the schema that was in use before the connection was lost. The last active schema set by the user is now restored during the automatic reconnection process, and during a manually triggered reconnection using the `\reconnect` command. (Bug #25974003, Bug #86115)
- If you issued `dba.configureLocalInstance()` against an instance which was already valid for InnoDB cluster usage and specified the `myCnf` option, but the `my.cnf` file was not readable, the operation would print that there are issues that need to be fixed. Now when you set the `myCnf` option, using either `dba.configureInstance()` and `dba.configureLocalInstance()`, the operations only use it if necessary. In other words, if the target instance is not valid for InnoDB cluster usage and settings must be changed. (Bug #25702994)
- The `cluster.describe()` and `cluster.status()` AdminAPI methods return JSON objects containing the same information but some fields were identified by different tags. To make the tags consistent, the object returned from `cluster.describe()` has changed so that `instances` has been replaced with `topology`, and `host` with `address`. (Bug #25247515)
- MySQL Shell no longer attempts to reconnect automatically when the connection to the server is lost. A new MySQL Shell command `\reconnect` is provided, which makes MySQL Shell try several reconnection attempts for the current global session with the existing connection parameters. If those attempts are unsuccessful, you can make a fresh connection using the `\connect` command and specifying the connection parameters. (Bug #25105307)
- Creating a cluster from an existing Group Replication group by using the `adoptFromGR:true` option on an instance which already belonged to cluster was not being correctly detected. Now such a situation is detected and generates an error. (Bug #25061891, Bug #25664766)
- MySQL Shell is able to create a session to an IPv6 address, but it failed to create an InnoDB cluster using such connections, reporting an URI-related error. This was related to the encoding of the URI containing an IPv6 address, which has been fixed. The core issue, IPv6 support for AdminAPI is a known issue as Group replication requires an IPv4 network, see [Group Replication Requirements](#). The implementation of InnoDB cluster related operations has been modified in order to check if operations are executed using an IPv4 based session and IPv4 connection data. An exception is generated if these conditions are not met. (Bug #25042407)
- A memory leak was fixed that occurred if MySQL Shell was started and a connection was made to the MySQL server, then the user exited MySQL Shell without executing any commands. (Bug #24794589)
- Attempting to add two different instances with the same label to an InnoDB cluster correctly resulted in an error, however the instance with the duplicated label was being left with a partially initialized configuration. (Bug #24761416)

Changes in MySQL Shell 8.0.5 - 8.0.10 (Skipped version numbers)

There are no release notes for these skipped version numbers.

Changes in MySQL Shell 8.0.4 (2018-01-25, Release Candidate)

- [Functionality Added or Changed](#)
- [Bugs Fixed](#)

Functionality Added or Changed

- When you create clusters or add instances you can now override the default group name, local addresses, and group seeds. This makes it easier to customize your clusters. The following options were added to the `dba.createCluster()` and `cluster.addInstance()` commands:
 - use `groupName` with `dba.createCluster()` to set the name of the cluster
 - use `localAddress` to set the address which an instance provides to communicate with other instances
 - use `groupSeeds` to set the instances used as seeds when instances join the cluster

For more information, see [Customizing InnoDB clusters](#). (Bug #26485254, Bug #26838005)

- Connections to an InnoDB cluster have been simplified. Now, when you issue `dba.getCluster()` and the active Shell session is not a connection to the primary, the cluster is queried for the primary information and a connection to the primary is automatically opened. This ensures that configuration changes can be written to the metadata. As part of this improvement you can now configure the type of MySQL Shell connection to an InnoDB cluster with the following options which have been added:
 - `--redirect-primary`
 - `--redirect-secondary`
 - `--cluster`

Additionally the following changes were made:

- The `dba.resetSession()` method has been removed.
- A new `disconnect()` method has been added to the `cluster` object, which closes all internal sessions opened by the cluster. InnoDB cluster operations on a disconnected cluster object result in an error.
- The output of the `Cluster.status()` method now includes the `groupInformationSourceMember` field, which shows the URI of the internal connection used by the cluster object to obtain information about the cluster.

References: See also: Bug #25091586.

- MySQL Shell now supports autocompletion of text preceding the cursor by pressing the Tab key. Autocompletion is available for:
 - Built-in MySQL Shell commands, for example typing `\con` followed by the Tab key completes to `\connect`.

- SQL, JavaScript and Python language keywords depending on the current MySQL Shell mode.
- Table names, column names, and active schema names in SQL mode, based on the current default schema.

Autocompletion can be configured using the following command options:

- `--name-cache`
- `--no-name-cache`
- A new `patch()` operation has been added to the `modify()` function which enables you to modify a document by merging in a set of changes. For more information see `JSON_MERGE_PATCH()`.
- You can now use Unix sockets for X Protocol connections. Socket file paths in URI type strings should be either percent encoded, such as `root@/home%2Fuser%2Fmysql-sandboxes%2F3310%2Fsandboxdata%2Fmysqlx.sock`, or surrounded by parenthesis such as `root@(/home/user/mysql-sandboxes/3310/sandboxdata/mysqlx.sock)`. The `--socket` option cannot be combined with the `--port` option.
- MySQL Shell performs automatic `_id` generation on collection add operations when no `_id` is specified on the documents being added. The autogenerated `_id` is created using the `UUID()` function. Now, the order of the tokens used has changed from `aaaaaaaa-bbbb-cccc-dddd-eeeeeeeeeeee`, where `eeeeeeeeeeee` is the MAC Address, to `eeeeeeeeeeee-dddd-cccc-bbbb-aaaaaaaa`.
- Sessions created by X DevAPI using either `mysql.getClassicSession(connection_data)` or `mysqlx.getSession(connection_data)` now use `ssl-mode=REQUIRED` as the default if no `ssl-mode` is provided, and neither `ssl-ca` nor `ssl-capath` is provided. If no `ssl-mode` is provided and any of `ssl-ca` or `ssl-capath` is provided, all Sessions created by MySQL Shell now default to `ssl-mode=VERIFY_CA`.
- In addition to the existing CRUD commands which work on documents in a collection by matching a filter, the following operations have been added to enable you to work with single documents:
 - `Collection.replaceOne(string id, Document doc)` updates (or replaces) the document identified by `id` with the provided one, if it exists.
 - `Collection.addOrReplaceOne(string id, Document doc)` add the given document. If the `id` or any other field that has a unique index on it already exists in the collection, the operations updates the matching document instead.
 - `Document Collection.getOne(string id)` returns the document with the given `id`. This is a shortcut for `Collection.find("_id = :id").bind("id", id).execute().fetchOne()`.
 - `Result Collection.removeOne(string id)` removes the document with the given `id`. This is a shortcut for `Collection.remove("_id = :id").bind("id", id).execute()`.

Using these operations you can reference documents by ID, making operations on single documents simpler by following a load, modify and save pattern such as:

```
doc = collection.getOne(id);
doc["address"] = "123 Long Street";
collection.replaceOne(id, doc);
```

- You can now use savepoints with MySQL Shell sessions. The following methods have been added to the Session object:

- Use `setSavepoint()` to generate a savepoint. The server executes the `SAVEPOINT` SQL statement and returns the generated savepoint name.
- Use `setSavepoint(name)` to specify the name used by the `SAVEPOINT` SQL statement.
- Use `releaseSavepoint(name)` to execute the `RELEASE` SQL statement.
- Use `rollbackTo(name)` to execute the `ROLLBACK TO name` SQL statement.

Any names passed to these functions are checked to make sure that the name is not null or an empty string. Names such as `'`, `"`, ```, and so on are not allowed even though they are allowed by the server. For more information, see [SAVEPOINT, ROLLBACK TO SAVEPOINT, and RELEASE SAVEPOINT Syntax](#).

- X DevAPI now supports row-locking for CRUD operations on tables and collections. Use the `lockShared()` method for write locks and the `lockExclusive()` method for read locks, which have been added to `find()` and `select()`. Either method can be called any number of times, including none, in any combination. But the locking type to be used is always the last one called. Once you call a lock method you can only then execute the operation, calling `execute()` or `bind()`. For more information, see [InnoDB Locking](#).
- The X DevAPI drop operations have been improved. Now, the `drop()` methods are at the same level as the `create()` methods and they all return nothing. There is now no need to use `execute()` after the drop method. If a drop method is called on an object which no longer exists in the database there is now no error.

This modifies Session objects so that:

- `dropSchema()` returns Nothing
- `dropTable(String schema, String name)` is removed
- `dropCollection(String schema, String name)` is removed
- `dropView(String schema, String name)` is removed

This modifies Schema objects so that:

- Nothing `dropCollection(String name)` is added

This modifies Collection objects so that:

- `dropIndex(String name)` is a direct operation, meaning `.execute()` is no longer needed
- `dropIndex(String name)` returns Nothing
- The `mysql` module, used with classic sessions for SQL connections to instances, has been streamlined. This has resulted in the following changes:
 - The new `getSession()` function has been added to the `mysql` module to create a classic session. This function works in the same way as the existing `mysql.getClassicSession()` function.
 - The `runSql()` method of `ClassicSession` has been extended to take a list of arguments to replace with placeholders in the query. For example, now you can issue:

```
session.runSql("select * from tbl where name = ? and age > ?", ["Joe", "20"])
```

Additionally a `query()` function has been added to `ClassicSession` as an alias to `runSql()`, making it consistent with `Session.query()`.

- The following functions have been removed from `ClassicSession`:
 - `createSchema()`
 - `getSchema()`
 - `getDefaultSchema()`
 - `getCurrentSchema()`
 - `setCurrentSchema()`
 - `ClassicSchema()`
- The `ClassicTable` object has been removed.
- The following `Table` and View DDL functions have been removed from `Schema` objects:
 - `dropTable()`
 - `dropView()`
- The behavior of the global `db` variable has been modified. When a global connection is created and the connection data includes a default schema, the global `db` variable is set to the corresponding `Schema` object. Now, this is not done for connections through the MySQL protocol, such as those created by `ClassicSession`.
- Use `util.checkForServerUpgrade([uri])` to check if a server can be upgraded from the version it is currently running to the next major series release. For example, you can verify that a server instance running MySQL 5.7 satisfies the upgrade prerequisites for MySQL 8.0, see [Verifying Upgrade Prerequisites for Your MySQL 5.7 Installation](#). To verify a server instance at a URI type string such as `user@example.com:3306` issue:

```
util.checkForServerUpgrade(['user@example.com:3306'])
```

MySQL Shell provides a report on anything in the table space which would cause a problem when upgrading the instance to MySQL 8.0.

- MySQL Shell builds against MySQL server version 8.0.4 and OpenSSL.
- MySQL 8.0.4 uses the `caching_sha2_password` authentication plugin and encrypted connections by default. This means any new accounts created in MySQL use these features. The new authentication method is completely transparent if the connection is made with encrypted connections enabled, which is the default, and the preferred mode of connection. If encrypted connections are explicitly disabled, the following limitations apply:
 - X Protocol sessions require encrypted connections.
 - If the `caching_sha2_password` plugin reports an error such as `Authentication requires a secure connection`, it is not possible to connect to the account with MySQL Shell, until either an encrypted connection is made or the `mysql` client is used to clear the error.

See [Using Encrypted Connections](#) and [Caching SHA-2 Pluggable Authentication](#).

Bugs Fixed

- Attempting a server connection without specifying the port or socket, when using an expired or temporary password, failed with an error requiring a password reset. The default connection data is now set at an appropriate point in connection processing, so the connection succeeds. (Bug #27266648)
- MySQL Shell required the option name `ssl-ciphers` instead of the standard MySQL option name `--ssl-cipher` to specify the list of permitted ciphers for connection encryption. The standard option name `--ssl-cipher` is now required in MySQL Shell. (Bug #27185275)
- The `--show-warnings` option was not working in MySQL Shell. (Bug #27036716)
- The command line options `--classic`, `--node`, `--sqln`, and `--ssl` were stated as deprecated, but actually had been removed. MySQL Shell now handles the options again, but prints a warning message when they are used. The deprecated options are processed as their replacement options, as follows:
 - `--classic` is processed as `--mysql`
 - `--node` is processed as `--mysqlx`
 - `--sqln` is processed as `--sqlx`
 - `--ssl` is processed as `--ssl-mode`(Bug #27012385)
- The output of `\status` when using a Unix socket for connections was showing a relative path. Now the absolute path of the socket is shown. (Bug #26983193)
- Account validation did not work correctly unless the session account existed. Now, validation is done using the account that was authenticated by the server. (Bug #26979375)
- The AdminAPI in MySQL Shell for working with InnoDB cluster only supports TCP connections to server instances. The AdminAPI now checks that a TCP connection is in use before starting an operation that requires database access, instead of attempting the operation with another connection type and not succeeding. (Bug #26970629)
- If you issued `STOP GROUP REPLICATION` on an instance that belonged to a cluster, attempting to rejoin the instance to the cluster failed because the wrong Group Replication seeds were being used. Now, `Cluster.rejoinInstance()` correctly sets `group_replication_group_seeds` based on the `group_replication_local_address` of all currently active instances in the cluster. (Bug #26861636)
- Sometimes the `dba.addInstance()` command failed with an error indicating that the server was in `RECOVERING` state despite being `ONLINE`. The fix ensures the correct instance state is returned. (Bug #26834542)
- If the user running MySQL Shell did not have write permissions to the option file configured by AdminAPI, no error was displayed. (Bug #26830224)
- With the addition of [WL#10470](#), the default value of `server_id` has changed to 1. As the server ID has to be unique for each instance, this caused issues with AdminAPI. Now, server instances with `server_id=1` are correctly identified as incorrectly configured for InnoDB cluster use. (Bug #26818744)
- AdminAPI now supports Python 2.6 in addition to Python 2.7, removing the need to manually install on Oracle Linux 6. (Bug #26809748)

- After removing an instance from a cluster using `Cluster.removeInstance()`, the instance silently rejoined the Group Replication group after it restarted. This happened because `group_replication_start_on_boot` was set to `ON` by default. Now, for instances running MySQL version 8.0.4 and later, the fix sets `group_replication_start_on_boot` to `OFF` in the option file. For instances running a MySQL version earlier than 8.0.4, a warning is issued to tell you to manually edit `group_replication_start_on_boot` in the instance's option file to avoid the issue. (Bug #26796118)
- Using AdminAPI commands on Windows that required SSL resulted in an error due to the Python version being used. (Bug #26636911)
- Creating an InnoDB cluster from an existing Group Replication deployment, by using the `adoptFromGR` option with the `dba.createCluster()` command, would fail with an error stating that the instance was already part of a replication group. The issue was only present in the MySQL Shell default wizard mode. The fix ensures that the interactive layer of the `dba.createCluster()` command allows the use of the `adoptFromGR` option. (Bug #26485316)
- The warnings generated when creating and adding sandbox instances have been improved. (Bug #26393614)
- When working with instances that had `require_secure_transport=ON`, AdminAPI commands that required a connection to the instance failed. (Bug #26248116)
- MySQL Shell now automatically pre-loads the built-in `mysql` and `mysqlx` API modules when it is invoked in batch mode, as well as when it is used in interactive mode. (Bug #26174373)
- The user configuration path for MySQL Shell defaults to `%AppData%\MySQL\mysqlsh\` on Windows, and `~/.mysqlsh/` on Unix. This directory is used for the MySQL Shell history file (`history`), log file (`mysqlsh.log`), and theme file (`prompt.json`). It is also the final location that MySQL Shell searches for startup scripts (`mysqlshrc.js` or `mysqlshrc.py`).

You can now override the user configuration path by specifying an alternative path using the `MYSQLSH_USER_CONFIG_HOME` environment variable. A directory specified by that environment variable is used by MySQL Shell for the user configuration data in place of `%AppData%\MySQL\mysqlsh\` on Windows, and `~/.mysqlsh/` on Unix. (Bug #26025157)

- The `Cluster.dissolve()` command was trying to stop Group Replication on all of the instances registered in the metadata which lead to connection errors if any of those instances were not contactable, in other words with the state (`MISSING`). The fix ensures that only instances which can be contacted, in other words with the state `ONLINE`, are stopped. (Bug #26001653)
- When adding instances to an InnoDB Cluster using the appropriate AdminAPI operations, checks are performed to verify the compatibility of any existing tables. If incompatible tables (for example using `MyISAM`) are detected then an error is issued. However the error message was referring to an option not available for the AdminAPI operations: `--allow-non-compatible-tables`. (Bug #25966731)
- For file processing, MySQL Shell now expands a leading tilde in a file path to the appropriate home directory. MySQL Shell identifies the home directory using a relevant environment variable, or looks it up for the logged-in user. (Bug #25676405)
- MySQL Shell now provides a `program_name` connection attribute to the server at connect time, with the value `mysqlsh`. Connection attributes are displayed in the Performance Schema connection attribute tables. (Bug #24735491, Bug #82771)
- Running `help()` in MySQL Shell in Python mode caused an `AttributeError`. (Bug #24554329, Bug #82767)

- The `cluster.rejoinInstance()` command attempted to rejoin an instance even if it was already part of the cluster. Now, only instances in the `MISSING` state are accepted by `cluster.rejoinInstance()`. Attempting to rejoin an instance in any other state fails with an error. (Bug #87873, Bug #26870329)
- On Unix, if Python 3 was installed AdminAPI commands failed. (Bug #87731, Bug #26785584)
- When using the `dba.checkInstanceConfiguration()` and `dba.configureLocalInstance()` commands, the account being used was not being checked if it had enough privileges to actually execute the command. The fix ensures that account has the required privileges before proceeding. This also required a change of the privileges given to `clusterAdmin` users. (Bug #87300, Bug #26609909)
- Arrays and Objects now accept the `IN` operator. For example:

```
collection.find( "'Fred' IN username" )
```

Changes in MySQL Shell 8.0.3 (2017-09-29, Development Milestone)

MySQL Shell now synchronizes the first digit of its version number with the (highest) MySQL server version it supports. This change makes it easy and intuitive to decide which client version to use for which server version. MySQL Shell now uses the same version number as MySQL Server.

MySQL Shell 8.0.3 is the first release to use the new numbering. It is the successor to MySQL Shell 8.0.0.

- [Functionality Added or Changed](#)
- [Bugs Fixed](#)

Functionality Added or Changed

- With the addition of [WL#10611](#) and [WL#10960](#), it was not possible to add or rejoin instances that belonged to a cluster (or a replication group) because `super_read_only=ON` was being set by Group Replication when stopping. To ensure that AdminAPI supports instances running MySQL 8.0.2 and later, the following functions have been modified:

- `dba.configureLocalInstance()`
- `dba.createCluster()`
- `dba.rebootClusterFromCompleteOutage()`
- `dba.dropMetadataSchema()`

Now, if any of these functions is issued against an instance which has `super_read_only=ON`, in interactive mode you are given the choice to set `super_read_only=OFF`. To force the function to set `super_read_only=OFF` in a script, pass the `clearReadOnly` option set to `true`. For example `dba.configureLocalInstance({clearReadOnly: true})`. For more information see [Super Read-only and Instances](#). (Bug #26422638)

- MySQL Shell now handles user interrupts, such as SIGINT, correctly. For example on Linux pressing Control-C when MySQL Shell is not executing anything exits the application. In SQL mode, interruption sends a `KILL QUERY` statement to the active MySQL Shell session from a new temporary session, resulting in the server interrupting the query and returning an error (or in an early return with no error in some cases, like the `sleep()` function). In JavaScript or Python scripting modes, how interruption behaves depends on the specific function being executed. If what is being executed is language code (such as a while loop and other normal script code), an exception is generated in the active language,

which causes the code to stop executing. The exception may be caught by the script, but if not, the execution control returns to MySQL Shell. (Bug #24757361)

- MySQL Shell now includes a history function that stores the code which you issue. The history can be saved, searched, and filtered. A new mechanism to customize the MySQL Shell prompt has been added. Information such as the current mode (SQL, JavaScript, or Python), session information (host, uri, port and so on), the current active schema and others can be included in the prompt through variables. The customization information is self-contained in JSON theme files, which can be shared between users. MySQL Shell now supports unicode if the terminal used to run MySQL Shell supports it. Similarly if the terminal supports color, MySQL Shell can be configured to use colors in the theme.
- The connection options passed to MySQL Shell, such as `sslMode` and so on, have been changed to use dashes and no longer be case sensitive. The options are now:
 - `sslMode` is now `ssl-mode`
 - `sslCa` is now `ssl-ca`
 - `sslCaPath` is now `ssl-capath`
 - `sslCert` is now `ssl-cert`
 - `sslKey` is now `ssl-key`
 - `sslCrl` is now `ssl-crl`
 - `sslCrlPath` is now `ssl-crlpath`
 - `sslCiphers` is now `ssl-ciphers`
 - `sslTlsVersion` is now `tls-version`
 - `authMethod` is now `auth-method`
- The types of session available have been simplified. `XSession` and `NodeSession` have been consolidated into `Session`. This has caused the following changes:
 - The following command options have been deprecated: `--node`, `--sqln`, `--classic`
 - The following command options have been introduced to replace the deprecated ones: `-ma`, `--mysqlx (-mx)`, `--mysql (-mc)`, `--sqlx`
 - The `\connect` MySQL Shell command no longer supports the arguments `-c`, and `-n`. Now the `\connect` command supports the argument `--mysqlx(-mx)` for creating X Protocol connections, and `--mysql(-mc)` for creating MySQL protocol connections.
- The interpretation of the `document_path` field in operations such as `modify()` has been changed. Now, when the `document_path` is not set, operations apply to the whole document. All operations always preserve a document's `_id` field.

Bugs Fixed

- In MySQL Shell, the `Schema.getCollectionAsTable()` function and the `select()` method could not be used in the same Python statement. (Bug #26552804)
- When using the `clusterAdmin` option, the created account did not have all of the correct privileges. (Bug #26523629)

- MySQL Shell returned some elements of DATE and DATETIME values incorrectly, including month values and fractional seconds. (Bug #26428636)
- The month was incorrectly incremented on insertion of a timestamp in a table using MySQL Shell. (Bug #26423177)
- For columns with the `ZEROFILL` attribute, `NULL` was also returned padded with zeroes. (Bug #26406214)
- The output of the MySQL Shell `\status` command was enhanced with additional information. (Bug #26403909)
- The MySQL Shell help for the `\connect` command indicated that a connection name could be used instead of a URI string, which was incorrect. (Bug #26392676)
- When using the `multiMaster` option with `dba.createCluster()`, the warning displayed in interactive mode was not being logged. (Bug #26385634)
- When making cluster topology or membership changes, AdminAPI was not taking into consideration the value of `group_replication_group_name`, which could lead to incorrect, non-deterministic results in scenarios such as a split brain. Now, the following commands validate the InnoDB cluster Metadata and the corresponding instance's `group_replication_group_name` value:

- `dba.getCluster()`
- `Cluster.rejoinInstance()`
- `Cluster.forceQuorumUsingPartitionOf()`

If the values of `group_replication_group_name` do not match, the commands abort with an error.

`dba.rebootClusterFromCompleteOutage()` was also updated to ensure that the `group_replication_group_name` variable has not been changed before rejoining the instance. (Bug #26159339)

- AdminAPI now always uses the active user value for the current `mysqlsh` session, whether the value was explicitly specified by the user or is the result of an implicit default used by `mysqlsh`. (Bug #26132527)
- The checks performed by the AdminAPI upon issuing `dba.rebootClusterFromCompleteOutage()` were more strict than those required by Group Replication. Now, the AdminAPI considers tables with a Primary Key Equivalent (such as a Non Null Unique Key) as compatible, matching the current requirement for Group Replication. (Bug #25974689)
- The MySQL Shell command `\use` did not attempt to reconnect if the connection to the global session was lost. (Bug #25974014, Bug #86118)
- The short form `-?` can now be used as an alias for the `--help` command-line option in MySQL Shell. (Bug #25813228)
- The MySQL Shell command history displayed the commands that were used to automatically import the `mysql` and `mysqlx` API modules when MySQL Shell started. (Bug #25739185)
- The randomly generated passwords used by internal users were not compatible with instances running the Password Validation plugin. (Bug #25714751)
- The MySQL Shell command history displayed the contents of scripts that were run using the `\source` MySQL Shell command. (Bug #25676495)

- It is no longer possible to use the `adoptFromGr` option with the `multiMaster` option. When adopting an existing group to an InnoDB cluster, the group is adopted based on whether it is running as multi-primary or single-primary. Therefore there is no use for the `multiMaster` option when adapting a group. (Bug #25664700)
- Issuing `configureLocalInstance()` when using a URI that contained a user without the correct privileges resulted in an incorrect new user being created. Now, if the user in `configureLocalInstance()` URI does not have enough privileges to grant all the necessary privileges for the new user chosen during the interactive wizard configuration the user is not created. (Bug #25614855)
- The `mysqlx.getNodeSession()` function in MySQL Shell now returns an error if an unrecognized connection option is provided. (Bug #25552033)
- Issuing `Cluster.rescan()` resulted in non-deterministic behavior which could produce incorrect JSON output, showing an instance that was already part of the cluster as belonging to the `newlyDiscoveredInstances[]` list and to the `unavailableInstances[]` list. This also resulted in MySQL Shell prompting to add or remove the instance from the cluster. (Bug #25534693)
- AdminAPI functions now accept the standard connection parameters as used by `shell.connect`. New validations have been added for when `require_secure_transport` is ON, now it is not possible to create a cluster with `memberSslMode:DISABLED` or to add an instance with `require_secure_transport=ON` to a cluster where `memberSslMode:DISABLED`. (Bug #25532298)
- The parsing of account names, for example when passing the `clusterAdmin` option to `dba.configureLocalInstance()` has been improved. (Bug #25528695)
- The file permissions of option files created by AdminAPI did not match those of options files created by MySQL install. (Bug #25526248)
- Issuing `configureLocalInstance()` twice could fail. (Bug #25519190)
- When passing the `rejoinInstances[]` option to `dba.rebootClusterFromCompleteOutage()`, if no `rejoinInstances[]` option was specified then members were being incorrectly handled during the rebuild. Now, instances that are eligible to be added to the `rejoinInstances[]` list but that are specified in the `removeInstances[]` list are skipped by the interactive wizard that tries to automatically build a `rejoinInstances[]` list if one was not provided. This fix also ensures that both interactive and non-interactive use of MySQL Shell correctly verify the `rejoinInstances[]` list does not contain a unreachable instance. (Bug #25516390)
- The error messages issued when the SSL mode used by the cluster and the one specified when issuing `addInstance()` command do not match have been improved. (Bug #25495056)
- The `--ssl` option has been deprecated, use the `--ssl-mode` option. Now, if you use the `--ssl` option a deprecation warning is generated and the `--ssl-mode` option is set to either `DISABLED` or `REQUIRED` based on the value used with the `--ssl` option. (Bug #25403945)
- When creating a sandbox instance using the `dba.deploySandboxInstance()` function in MySQL Shell, pressing Ctrl + C at the prompt for a MySQL root password for the instance did not cancel the deployment. (Bug #25316811)
- MySQL Shell did not exit gracefully when the user did not have a valid and accessible home directory. (Bug #25298480)
- MySQL Shell created a logger but did not deallocate it on exiting the shell. (Bug #25238576)

- Issuing `removeInstance()` on the last member of a cluster, and particularly the seed member, was resulting in a cluster that could not be dissolved. Now, issuing `removeInstance()` on the last member of a cluster results in an error, and you must use `dissolve()` on that instance to ensure the cluster is correctly dissolved. (Bug #25226130)
 - The output of `cluster.status()` now includes the `ssl` parameter, which shows whether secure connections are required by the cluster or disabled. (Bug #25226117)
 - MySQL Shell could hang when Ctrl + C was used to exit the shell. (Bug #25180850, Bug #84022)
 - Attempting to create a multi-primary cluster in interactive mode failed unless you passed in the `{force: true}` option. Now when you confirm that you understand the impact of using multi-primary mode the command correctly creates a multi-primary cluster. (Bug #25034951)
 - The `removeInstance()` was not working on stopped instances and it was not possible to remove an unavailable instance from the cluster. The fix adds a new option `force` to the `removeInstance()` command to enable you to remove instances from the metadata that are permanently not available, avoiding obsolete data from being kept in the metadata of the cluster. In addition the error message provided when not using the `force` option has been improved and the online help for the `removeInstance()` was also updated accordingly. (Bug #24916064)
 - Unsigned data could be incorrectly read from the database. (Bug #24912358)
 - The parsing of Unix sockets provided as part of a URI has been improved. (Bug #24905066)
 - The error messages generated by issuing `dba.deployLocalInstance()` against an unsuitable or incompatible instance have been improved. (Bug #24598272)
 - The `dba.createCluster()`, `dba.getCluster()`, and `dba.rebootClusterFromCompleteOutage()` functions have been updated to validate the cluster name, using the following rules:
 - Name must start with a letter or the `_` character
 - Name can only contain alphanumeric characters and the `_` character
 - Cannot be longer than 40 characters
 - Cannot be empty
- The `Cluster.addInstance()` function has been updated to validate the label used on an instance in the cluster, using the following rules:
- Label can only contain alphanumerics or the `_` character
 - Cannot be longer than 256 characters
 - Cannot be empty
- (Bug #24565242)
- MySQL Shell now accepts Unicode characters as input. (Bug #23151666, Bug #81176)

Changes in MySQL Shell 8.0.2 (Skipped)

Version 8.0.2 has no changelog entries, or they have not been published because the product version has not been released.

Changes in MySQL Shell 8.0.1 (Skipped)

Version 8.0.1 has no changelog entries, or they have not been published because the product version has not been released.

Changes in MySQL Shell 8.0.0 (2017-07-14, Development Milestone)

- [Functionality Added or Changed](#)
- [Bugs Fixed](#)

Functionality Added or Changed

- Calling the `modify()` or `remove()` function without a parameter caused the function to be executed against the whole collection, which could cause unexpected results such as deleting all rows in a table. To avoid this and make the behavior consistent with `update()` and `delete()`, a client-side exception is now thrown if the `modify()` or `remove()` function is called without a parameter. Now, to execute the `modify()` or `remove()` function against a collection call them with an expression that evaluates to `true`, for example `remove('true')` or `modify('true')`.

Bugs Fixed

- Executing AdminAPI commands on a server with a version of Python lower than 2.7 was failing without the correct error message. (Bug #25975317)
- When using MySQL Shell on Windows any files created or opened, for example those used during `dba.createSandboxInstance()`, could not be deleted. (Bug #25789094)
- The help for `dba.configureLocalInstance(instance[, options])` has been improved to describe the returned JSON object. (Bug #25703028)
- The options in the MySQL Shell options dictionary are now fully documented. (Bug #25701345)
- When using `dba.deploySandboxInstance()` and passing in `sandboxDir`, the specified path must not exceed 89 characters. (Bug #25485035)
- `shell.connect()` did not report an error if an invalid argument was used. An `ArgumentError` is now issued for any invalid argument.

The following mutually exclusive pairs of options are now checked, and an error is issued if both are specified:

- `--password` and `--dbPassword`
 - `--user` and `--dbUser`
 - `--port` and `--socket`
- (Bug #25268670)

References: See also: Bug #24911173.

- `removeInstance()` resulted in unexpected behavior in some cases, for example when an empty password was passed as part of the URI to the instance. (Bug #25111911)
- A number of issues with the output of `shell.help("prompt")` have been corrected. (Bug #25026855, Bug #25242638, Bug #25676343, Bug #25176769)

- MySQL Shell now displays an invalid year as `0000`, matching the behavior of the MySQL prompt, rather than as `0`. (Bug #24912061)
- MySQL Shell did not display fractional seconds for values in DATETIME columns. (Bug #24911885)
- Creating Classic sessions that connect using Unix sockets now uses the correct defaults such as hostname. This resolves the previous limitation of using Unix sockets to connect to InnoDB cluster instances. See [MySQL Shell Connections](#) for information on how the defaults are applied to socket connections. (Bug #24848763, Bug #26036466)

References: See also: Bug #24911068.

- Some issues with the MySQL Shell command line help output were fixed. (Bug #24841749, Bug #24841493, Bug #24910540)
- URIs were incorrectly parsed in MySQL Shell when passwords were hidden. (Bug #24793956)
- `mysqlsh` stopped responding if the `\source` command was given a directory (rather than file) argument. (Bug #23097932, Bug #81060)
- On an instance configured as a multithreaded slave, in other words `slave_parallel_workers` set to greater than 0, and with `slave_parallel_type=DATABASE`, `dba.checkInstanceConfiguration()` was not detecting that the instance was not correctly configured for InnoDB cluster usage.
- If `removeInstance()` failed due to a connection error, an error was reported but the instance was incorrectly removed from the InnoDB cluster metadata, and remained part of the replication group. The fix ensures the metadata is correctly updated according to the result of `removeInstance()`.
- In a situation where a new primary instance was elected, adding a new instance to the cluster resulted in an error due to a failed connection to the previous primary instance.
- The functions that modify server variables, such as `dba.createCluster()` and `dba.validateInstance()` now provide more information in interactive mode output and log output about server variables which are changed when executed.
- Deploying instances to paths with directories that contained spaces was failing without error. Use double backslash to specify such paths, for example `D:\\Cluster\\foo bar`.
- The Cluster object obtained from functions such as `dba.createCluster()` or `dba.getCluster()` became unusable once the Shell session in which the object was created is was connected to a different server. The fix modifies the Cluster object so that:
 - The Cluster object holds an internal reference to the Session from which it was created or retrieved.
 - AdminAPI functions that modify the Cluster are made using the session referenced by the object.

