Wednesday May 29th 5:40pm-6:05pm

Texas 7

MySQL Shell: The Best DBA tool?

How to Use the MySQL Shell as a Framework for DBAs

Frédéric Descamps
@lefred

ORACLE®

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purpose only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied up in making purchasing decisions. The development, release and timing of any features or functionality described for Oracle´s product remains at the sole discretion of Oracle.

about me - http://about.me/lefred

Who am I ?

Frédéric Descamps

- @lefred
- MySQL Evangelist
- Hacking MySQL since 3.23
- devops believer
- living in Belgium 🇧🇪
- http://lefred.be

a new tool

MySQL Shell

MySQL Shell

The MySQL Shell is an interactive Javascript, Python, or SQL interface supporting development and administration for the MySQL Server and is a component of the MySQL Server. You can use the MySQL Shell to perform data queries and updates as well as various administration operations.

MySQL Shell (2)

The MySQL Shell provides:

MySQL Shell (2)

The MySQL Shell provides:

- Both Interactive and Batch operations

MySQL Shell (2)

The MySQL Shell provides:

- Both Interactive and Batch operations
- Document and Relational Models

MySQL Shell (2)

The MySQL Shell provides:

- Both Interactive and Batch operations
- Document and Relational Models
- CRUD Document and Relational APIs via scripting

MySQL Shell (2)

The MySQL Shell provides:

- Both Interactive and Batch operations
- Document and Relational Models
- CRUD Document and Relational APIs via scripting
- Traditional Table, JSON, Tab Separated output results formats

MySQL Shell (2)

The MySQL Shell provides:

- Both Interactive and Batch operations
- Document and Relational Models
- CRUD Document and Relational APIs via scripting
- Traditional Table, JSON, Tab Separated output results formats
- MySQL Standard and X Protocols

MySQL Shell (2)

The MySQL Shell provides:

- Both Interactive and Batch operations
- Document and Relational Models
- CRUD Document and Relational APIs via scripting
- Traditional Table, JSON, Tab Separated output results formats
- MySQL Standard and X Protocols
- and more...

MySQL Shell and Python

When using the python mode in the Shell, it's possible to use system modules (local).

```
[fred@imac2 workspace] $ mysqlsh
MySQL Shell 8.0.13

Copyright (c) 2016, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type '\help' or '\?' for help; '\quit' to exit.


MySQL  JS  \py
Switching to Python mode...

MySQL  Py  from datetime import datetime
MySQL  Py  datetime.now()
datetime.datetime(2018, 12, 16, 19, 54, 28, 222011)
```

MySQL Shell and Python (2)

Of course this can be any type of modules:

```
MySQL  Py  from isbntools.app import *
MySQL  Py  doc=meta('9781260135442')
MySQL  Py  doc
{
    "Authors": [
        "David Stokes"
    ],
    "ISBN-13": "9781260135442",
    "Language": "en",
    "Publisher": "McGraw-Hill Education",
    "Title": "MySQL And JSON: A Practical Programming Guide",
    "Year": "2018"
}
```

MySQL Shell and Python (2)

Of course this can be any type of modules:

```
MySQL  Py  from isbntools.app import *
MySQL  Py  doc=meta('9781260135442')
MySQL  Py  doc
{
    "Authors": [
        "David Stokes"
    ],
    "ISBN-13": "9781260135442",
    "Language": "en",
    "Publisher": "McGraw-Hill Education",
    "Title": "MySQL And JSON: A Practical Programming Guide",
    "Year": "2018"
}
```

we want more !

Extending MySQL Shell

Extending MySQL Shell

Since 8.0.16, you have two different ways to extend the MySQL Shell:

- using the new Reporting Framework (>= 8.0.16)
- create your own modules to extend MySQL Shell

MySQL Shell User-Defined Reports

You can create reports that can be called ( ) one time or constantly refreshed ( ).

Example for checking the :

```
MySQL 8.0.16  localhost:33060+  2019-05-16 15:03:04
JS  \show
Available reports: gr_info, gr_recovery_progress, locks_info, query.

MySQL 8.0.16  localhost:33060+  2019-05-16 15:03:07
JS  \show locks_info
+--------+-----------+-------------------+-----------+---------------+-------------+-----------------------+
| trx_id | thread_id | table             | lock_type | lock_mode     | lock_status | lock_data             |
+--------+-----------+-------------------+-----------+---------------+-------------+-----------------------+
| 17425  | 47        | james.user_events | TABLE     | IX            | GRANTED     | NULL                  |
| 17425  | 47        | james.user_events | RECORD    | X             | GRANTED     | supremum pseudo-record |
| 17425  | 47        | james.user_events | RECORD    | X             | GRANTED     | 8                     |
| 17422  | 48        | james.user_events | TABLE     | IX            | GRANTED     | NULL                  |
| 17422  | 48        | james.user_events | RECORD    | X,REC_NOT_GAP | GRANTED     | 7                     |
+--------+-----------+-------------------+-----------+---------------+-------------+-----------------------+
```

MySQL Shell User-Defined Reports (2)

This is a Python file ( ) installed in :

It contains a definition and a registration:

MySQL Shell User-Defined Reports (3)

More info:

- https://lefred.be/content/using-the-new-mysql-shell-reporting-framework-to-monitor-innodb-cluster/
- https://lefred.be/content/mysql-innodb-cluster-recovery-process-monitoring-with-the-mysql-shell-reporting-framework/

Sources of Examples:

- https://github.com/lefred/mysql-shell-udr
- Pull Requests welcome !

Create your own modules for MySQL Shell

For calling some long statements or group of operations or sometimes to replace a missing functionality.

Create your own modules for MySQL Shell

For calling some long statements or group of operations or sometimes to replace a missing functionality.

Recently, somebody pointed out that since the new DD it was not anymore possible to delete all routines for a specific schema.

Create your own modules for MySQL Shell

For calling some long statements or group of operations or sometimes to replace a missing functionality.

Recently, somebody pointed out that since the new DD it was not anymore possible to delete all routines for a specific schema.

Jesper explained recently how the MySQL Shell could help here see https://mysql.wisborg.dk/2018/12/02/mysql-8-drop-several-stored-events-procedures-or-functions/

Extending MySQL Shell

```
MySQL  127.0.0.1:33060+  Py  mydba.getProcedures('sys','FUNCTION')
FUNCTION `sys`.`extract_schema_from_file_name`
FUNCTION `sys`.`extract_table_from_file_name`
FUNCTION `sys`.`format_bytes`
FUNCTION `sys`.`format_path`
FUNCTION `sys`.`format_statement`
FUNCTION `sys`.`format_time`
FUNCTION `sys`.`list_add`
FUNCTION `sys`.`list_drop`
FUNCTION `sys`.`ps_is_account_enabled`
FUNCTION `sys`.`ps_is_consumer_enabled`
FUNCTION `sys`.`ps_is_instrument_default_enabled`
FUNCTION `sys`.`ps_is_instrument_default_timed`
FUNCTION `sys`.`ps_is_thread_instrumented`
FUNCTION `sys`.`ps_thread_account`
FUNCTION `sys`.`ps_thread_id`
FUNCTION `sys`.`ps_thread_stack`
FUNCTION `sys`.`ps_thread_trx_info`
FUNCTION `sys`.`quote_identifier`
FUNCTION `sys`.`sys_get_config`
FUNCTION `sys`.`version_major`
FUNCTION `sys`.`version_minor`
FUNCTION `sys`.`version_patch`
Total: 22
```

Extending MySQL Shell

```
[fred@imac2 functions] $ mysqlsh --python root@127.0.0.1
Creating a session to 'root@127.0.0.1'
Fetching schema names for autocompletion... Press ^C to stop.
Your MySQL connection id is 184 (X protocol)
Server version: 8.0.13 MySQL Community Server - GPL
No default schema selected; type \use <schema> to set one.
MySQL Shell 8.0.13

Copyright (c) 2016, 2018, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type '\help' or '\?' for help; '\quit' to exit.


 MySQL  127.0.0.1:33060+  Py  mydba.getProcedures('test')
PROCEDURE `test`.`helloworld`
Total: 1

 MySQL  127.0.0.1:33060+  Py  mydba.deleteProcedures('test')
DROP PROCEDURE `test`.`helloworld`
Total dropped: 1
```

Extending MySQL Shell

Or for example, retrieve the expiration period of passwords (see https://lefred.be/content/mysql-when-will-the-password-of-my-users-expire/):

```
MySQL  127.0.0.1:33060+  Py  mydba.getPasswordExpiration()
+--------------------------------+--------------------------------+-----------------+
| User                           | Password last change           | Expires in      |
+--------------------------------+--------------------------------+-----------------+
| `fred`@`%`                     | 2018-11-12 21:59:56            | expired         |
| `test`@`%`                     | 2018-12-17 09:58:32            | 20 days         |
| `test2`@`%`                    | 2018-11-10 13:16:44            | expired         |
| `test3`@`%`                    | 2018-10-10 13:16:44            | expired         |
| `root`@`localhost`             | 2018-11-16 23:10:41            | expired         |
+--------------------------------+--------------------------------+-----------------+

MySQL  127.0.0.1:33060+  Py  mydba.getPasswordExpiration(False)
+--------------------------------+--------------------------------+-----------------+
| User                           | Password last change           | Expires in      |
+--------------------------------+--------------------------------+-----------------+
| `test`@`%`                     | 2018-12-17 09:58:32            | 20 days         |
+--------------------------------+--------------------------------+-----------------+
```

Extending MySQL Shell

Another example, retrieve the tables potentially fragmented (see https://lefred.be/content/overview-of-fragmented-mysql-innodb-tables/):

```
MySQL  localhost:33060+  Py  mydba.getFragmentedTables()
+-----------------------------+------------+---------+------------+------------+------------+------------------+
| Table                       | Engine     | # Rows  | Data Size  | Idx Size   | Total Size | Data Free        |
+-----------------------------+------------+---------+------------+------------+------------+------------------+
| sbtest.sbtest1              | InnoDB     |   1.54M |      0.38G |      0.03G |      0.41G | 48.00MB (11.54%) |
+-----------------------------+------------+---------+------------+------------+------------+------------------+
```

Extending MySQL Shell

Recently somebody complained about the complexity of knowing what are the default values of columns when expressions are used (https://forums.mysql.com/read.php?101,670682,670682):

```
MySQL  localhost:33060+  Py  mydba.getDefaults('test','default_test')
+-------------------------------+-----------------+-------------------------------------------------------+---------------------------+
| ColumnName                    | Type            | Default                                               | Example                   |
+-------------------------------+-----------------+-------------------------------------------------------+---------------------------+
| bi_col_exp                    | bigint          | (8 * 8)                                               | 64                        |
| d_col                         | date            | curdate()                                             | 2018-12-27                |
| d_col_exp                     | date            | (curdate() + 8)                                       | 20181235                  |
| dt_col                        | datetime        | CURRENT_TIMESTAMP                                     | 2018-12-27 09:53:11       |
| enum_col                      | enum            | value1                                                | value1                    |
| int_col                       | int             | 44                                                    | 44                        |
| t_col                         | time            | curtime()                                             | 09:53:11                  |
| vc_col                        | varchar         | test                                                  | test                      |
| vc_col_exp                    | varchar         | concat(_utf8mb4\'test\',_utf8mb4\'test\')             | testtest                  |
+-------------------------------+-----------------+-------------------------------------------------------+---------------------------+
Total: 9
```

Contribute to MySQL Shell DBA Toolkit ?

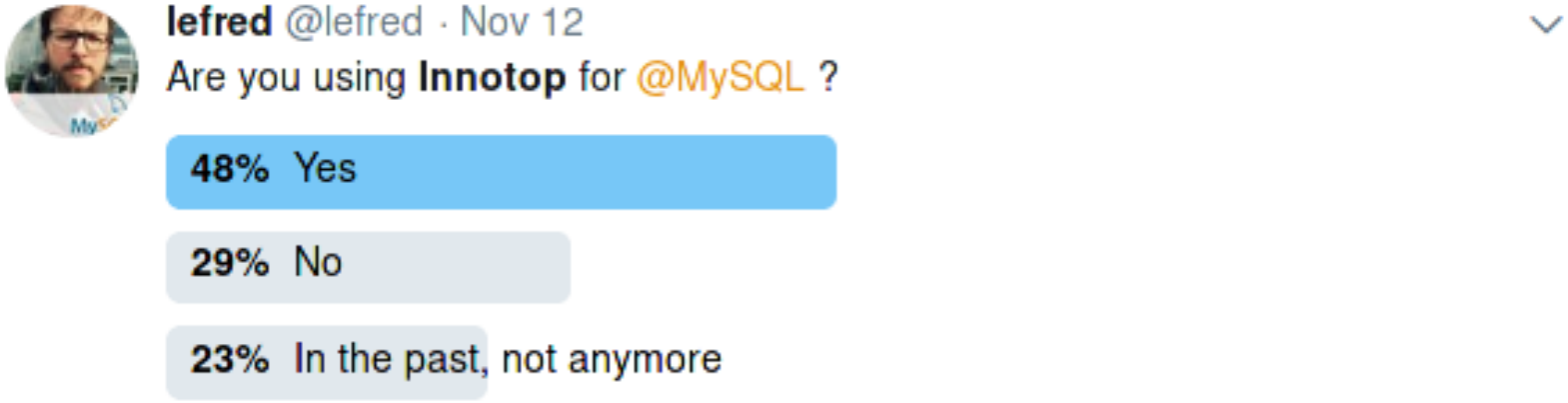
Get the code from https://github.com/lefred/mysql-shell-mydba and Pull Requests are welcome !

we want even more !

Innotop

Innotop

As maintainer of Innotop, after doing so delayed maintenance, I started a poll on Twitter:



Innotop (2)

So Innotop is not dead... but it's very complex to maintain... Perl !

Innotop (2)

So Innotop is not dead... but it's very complex to maintain... Perl !

This is maybe the reason there is less and less contributors...

Innotop (2)

So Innotop is not dead... but it's very complex to maintain... Perl !

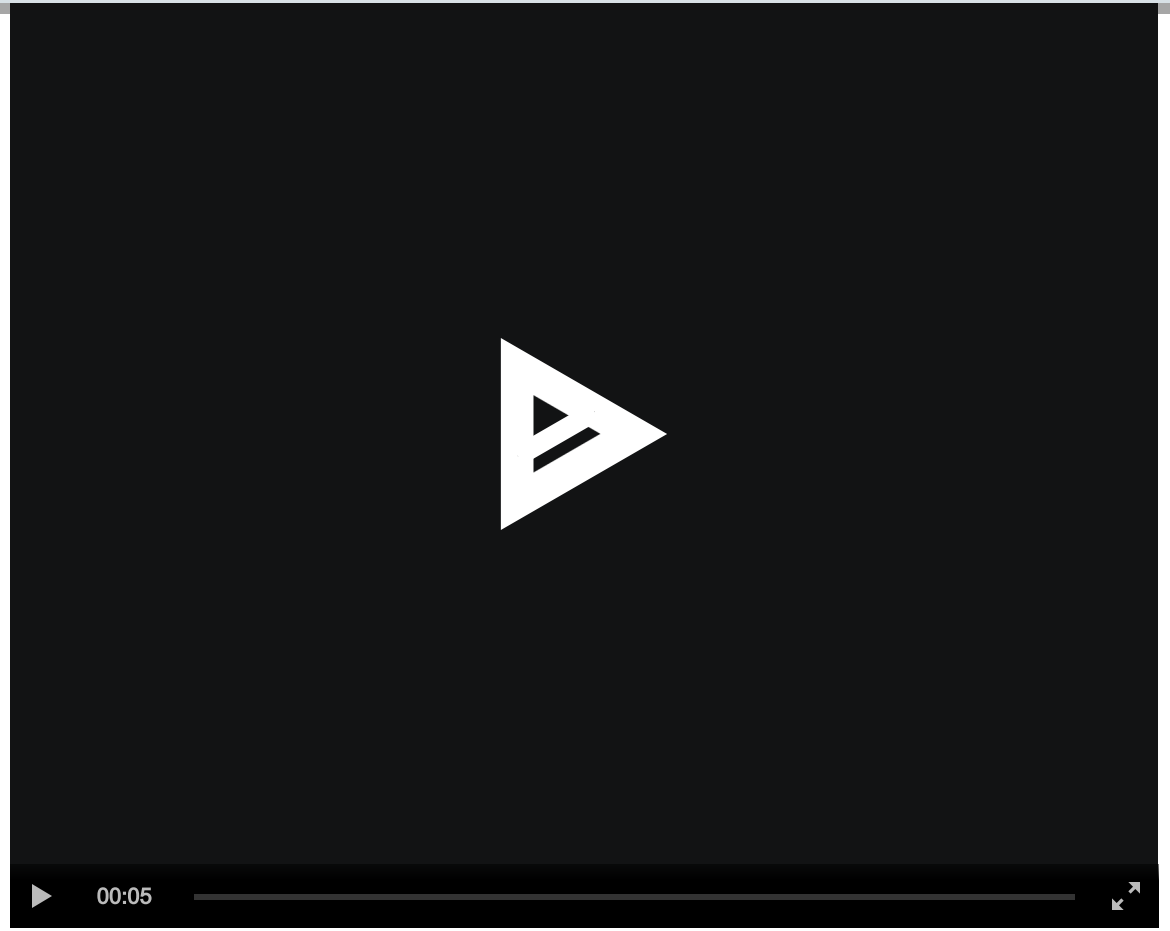
This is maybe the reason there is less and less contributors...

... so and MySQL Shell then ?

Innotop in MySQL Shell

```
MySQL Shell | MySQL Community Server - GPL 8.0.13                                    Sunday 16 December 22:04:53

Cmd       Thd  Conn   Pid State          User              Db          Time       Lock Time  Query
Query      47     9 9297  User sleep      root@localhost    sbtest      13.37 s    0 ps       select sleep(120)
Query     171   143 None  Creating sort index mysqlx/worker     sys          41.15 ms   392.00 us  select `pps`.`PROCESSLIST_COM
Execute   181   142 17461 Sending data    root@localhost    sbtest      2.64 ms    12.00 us   SELECT count(k)          FROM ..
Execute   175   136 17461 Sending data    root@localhost    sbtest      2.25 ms    12.00 us   SELECT count(k)          FROM ..
Execute   176   137 17461 statistics      root@localhost    sbtest      2.14 ms    23.00 us   SELECT count(k)          FROM ..
Execute   174   135 17461 init            root@localhost    sbtest      2.03 ms    0 ps       SELECT count(k)          FROM ..
Execute   178   140 17461 statistics      root@localhost    sbtest      1.93 ms    18.00 us   SELECT count(k)          FROM ..
Execute   173   134 17461 statistics      root@localhost    sbtest      1.86 ms    17.00 us   SELECT count(k)          FROM ..
Execute   180   141 17461 Sending data    root@localhost    sbtest      1.70 ms    14.00 us   SELECT count(k)          FROM ..
Execute   177   138 17461 statistics      root@localhost    sbtest      1.68 ms    17.00 us   SELECT count(k)          FROM ..
. 8 OR k BETWEEN 15000 AND 1500545048
```



Innotop in MySQL Shell

How to use it ?

How to use this module in MySQL Shell ?

The module is available on github: https://github.com/lefred/mysql-shell-innotop

How to use this module in MySQL Shell ?

The module is available on github: https://github.com/lefred/mysql-shell-innotop

How to use this module in MySQL Shell ?

The module is available on github: https://github.com/lefred/mysql-shell-innotop

Add in *~/.mysqlsh/mysqlshrc.py*:

does it work everywhere ?

Limitations

Limitations

- requires ncurses

Limitations

- requires ncurses
- works almost exclusively on Gnu/Linux

I want to contribute !

How to write your own extension ?

Creating you own extension

All extensions are a single file located in the **modules** folder:

Skeleton of an extension

Every extension starts the same way and requires some mandatory modules:

Skeleton of an extension

Every extension starts the same way and requires some mandatory modules:

Skeleton of an extension

Every extension starts the same way and requires some mandatory modules:

Then you need to specify the shortcuts that calls the functions:

Skeleton of an extension

Every extension starts the same way and requires some mandatory modules:

Then you need to specify the shortcuts that calls the functions:

Skeleton of an extension

Every extension starts the same way and requires some mandatory modules:

Then you need to specify the shortcuts that calls the functions:

return: name of the module *(dummy.py)*
stdscr: use the same screen *(and pass it to the function)*

Skeleton of an extension - run()

Thank you !

Any Questions ?

share your 💕 for MySQL on social media using