Python MySQL Replication Documentation

Release 0.7

Julien Duponchelle

December 04, 2015

Contents

Pure Python Implementation of MySQL replication protocol build on top of PyMYSQL. This allow you to receive event like insert, update, delete with their datas and raw SQL queries.

CHAPTER 1

Use cases

- MySQL to NoSQL database replication
- MySQL to search engine replication
- Invalidate cache when something change in database
- Audit
- Real time analytics

CHAPTER 2

Contents

2.1 Installation

Python MySQL Replication is available on PyPi. You can install it with:

```
pip install mysql-replication
```

2.2 Changelog

2.2.1 0.3 07/07/2014

- use NotImplementedEvent instead of raising an Exception
- Python 3 fix
- Add 2006 to Mysql expected error codes

2.2.2 0.2 13/10/2013

- pymysql 0.6 support
- fix smallint24
- fix new decimal support
- TINYINT(1) to bool mapping
- change names of events to V2 from default
- Fix broken “dates” - zero years..
- add support for NULL_EVENT, INTVAR_EVENT and GTID_LOG_EVENT
- Skip invalid packets
- Display log pos inside events dump
- Handle utf8 name for queries

2.2.3 0.1 01/05/2013

First public version

2.3 Limitations

2.3.1 GEOMETRY

GEOMETRY field is not decoded you will get the raw data.

2.3.2 binlog_row_image

Only [binlog_row_image=full](http://dev.mysql.com/doc/refman/5.6/en/replication-options-binary-log.html#sysvar_binlog_row_image) is supported (it's the default value).

2.3.3 BOOLEAN and BOOL

Boolean is returned as TINYINT(1) because it's the reality.

http://dev.mysql.com/doc/refman/5.6/en/numeric-type-overview.html

Our discussion about it: https://github.com/noplay/python-mysql-replication/pull/16

2.4 BinLogStreamReader

class pymysqlreplication.binlogstream.**BinLogStreamReader**(*connection_settings*, *server_id*, *resume_stream=False*, *blocking=False*, *only_events=None*, *log_file=None*, *log_pos=None*, *filter_non_implemented_events=True*, *ignored_events=None*, *auto_position=None*, *only_tables=None*, *only_schemas=None*, *freeze_schema=False*, *skip_to_timestamp=None*)

Connect to replication stream and read event

2.5 Events

class pymysqlreplication.event.**BeginLoadQueryEvent**(*from_packet*, *event_size*, *table_map*, *ctl_connection*, ***kwargs*)

Attributes: file_id block-data

class pymysqlreplication.event.**ExecuteLoadQueryEvent**(*from_packet*, *event_size*, *table_map*, *ctl_connection*, ***kwargs*)

Attributes: slave_proxy_id execution_time schema_length error_code status_vars_length

file_id start_pos end_pos dup_handling_flags

class pymysqlreplication.event.**GtidEvent**(*from_packet*, *event_size*, *table_map*, *ctl_connection*, ***kwargs*)

GTID change in binlog event

gtid

GTID = source_id:transaction_id Eg: 3E11FA47-71CA-11E1-9E33-C80AA9429562:23 See: http://dev.mysql.com/doc/refman/5.6/en/replication-gtids-concepts.html

class pymysqlreplication.event.**QueryEvent**(*from_packet*, *event_size*, *table_map*, *ctl_connection*, ***kwargs*)

This evenement is trigger when a query is run of the database. Only replicated queries are logged.

class pymysqlreplication.event.**RotateEvent**(*from_packet*, *event_size*, *table_map*, *ctl_connection*, ***kwargs*)

Change MySQL bin log file

Attributes: position: Position inside next binlog next_binlog: Name of next binlog file

class pymysqlreplication.event.**XidEvent**(*from_packet*, *event_size*, *table_map*, *ctl_connection*, ***kwargs*)

A COMMIT event

Attributes: xid: Transaction ID for 2PC

2.5.1 Row events

This events are send by MySQL when data are modified.

class pymysqlreplication.row_event.**DeleteRowsEvent**(*from_packet*, *event_size*, *table_map*, *ctl_connection*, ***kwargs*)

This event is trigger when a row in the database is removed

For each row you have a hash with a single key: values which contain the data of the removed line.

class pymysqlreplication.row_event.**TableMapEvent**(*from_packet*, *event_size*, *table_map*, *ctl_connection*, ***kwargs*)

This evenement describe the structure of a table. It's send before a change append on a table. A end user of the lib should have no usage of this

class pymysqlreplication.row_event.**UpdateRowsEvent**(*from_packet*, *event_size*, *table_map*, *ctl_connection*, ***kwargs*)

This event is triggered when a row in the database is changed

For each row you got a hash with two keys:

- before_values
- after_values

Depending of your MySQL configuration the hash can contains the full row or only the changes: http://dev.mysql.com/doc/refman/5.6/en/replication-options-binary-log.html#sysvar_binlog_row_image

class pymysqlreplication.row_event.**WriteRowsEvent**(*from_packet*, *event_size*, *table_map*, *ctl_connection*, ***kwargs*)

This event is triggered when a row in database is added

For each row you have a hash with a single key: values which contain the data of the new line.

2.6 Examples

You can found a list of working examples here: https://github.com/noplay/python-mysql-replication/tree/master/examples

2.7 Support

You can get support and discuss about new features on: https://groups.google.com/d/forum/python-mysql-replication

You can browse and report issues on: https://github.com/noplay/python-mysql-replication/issues

2.8 Developement

2.8.1 Contributions

You can report issues and contribute to the project on: https://github.com/noplay/python-mysql-replication

The standard way to contribute code to the project is to fork the Github project and open a pull request with your changes: https://github.com/noplay/python-mysql-replication

Don't hesitate to open an issue with what you want to changes if you want to discuss about it before coding.

2.8.2 Tests

When it's possible we have an unit test.

pymysqlreplication/tests/ contains the test suite. The test suite use the standard *unittest* Python module.

Be carefull tests will reset the binary log of your MySQL server.

Make sure you have the following configuration set in your mysql config file (usually my.cnf on development env):

```
log-bin=mysql-bin
server-id=1
binlog-format    = row #Very important if you want to receive write, update and delete row events
gtid_mode=ON
log-slave_updates=true
enforce_gtid_consistency
```

To run tests:

```
python setup.py test
```

Each pull request is tested on Travis CI: https://travis-ci.org/noplay/python-mysql-replication

2.8.3 Build the documentation

The documentation is available in docs folder. You can build it using Sphinx:

```
cd docs
pip install sphinx
make html
```

2.9 Licence

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

p