**MySQL版本升级方案**

编 写 人：李建军

部 门：数据支持

完成时间：2017.06

Email：[lijianjun@meilele.com](mailto:lijianjun@meilele.com)

版本信息

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 时间 | 提交人 | 描述 |
| 1.0 | 2017.06 | 李建军 | 创建文档 |

## 1MySQL5.6新增版本特性

## 1.1InnoDB增强

1.1.1增加全文索引（where match(column) agains (‘key’)）

1.1.2 支持简单在线DDL 如创建、删除索引等。

1.1.3 独立表空间下的表数据文件能制定路径(create table t(id int) data directory = ‘/another/directory’)

1.1.4 通过copy方式比mysqldump更加快速的移动和导出数据文件。

1.1.5 能制定innodb page size

1.1.6 undo表空间能独立到共享表空间之外(innodb\_undo\_logs和innodb\_undo\_tablespace)

## 1.2分区新能提高

1.2.1 分区个数扩大到8192个

1.2.2 支持显式分区查询语句select id from t partition(p0,p1) where id < 5

## 1.3复制性能改善

1.3.1 支持GTID（全局事务ID）复制

1.3.2 新增参数[binlog\_row\_image](https://dev.mysql.com/doc/refman/5.6/en/replication-options-binary-log.html#sysvar_binlog_row_image)来控制基于ROW模式下的记录方式，节约日志空间

1.3.3 master.info和replay-log.info支持保存在表中

1.3.4 支持延时复制 在change\_master\_to 中增加MATER\_DELAY

1.3.5 支持基于schema的多线程复制。slave\_parallel\_workers=4。如果只有单库还是单线程。

## 1.4数据类型变化

1.4.1 time、datetime、timestamp支持微妙

1.4.2 datetime 支持default current\_timestamp和update on current\_timestamp

1.4.3 datetime存储结构改变，更节约空间。

## 2MySQL5.6参数移除和变更

2.1—log和log\_file移除替换为 genery\_log genery\_log\_file

2.2 log\_slow\_queries 替换为 slow\_query\_log

2.3 sql\_max\_join\_size替换为max\_join\_size

2.4 max\_long\_data\_size替换为max\_allowed\_packet

## 3MySQL升级方式选择

### 3.1 In-Place upgrade

3.1.1如果使用的innodb，set global innodb\_fast\_shutdown = 0 关闭快速关闭。此时innodb会在关闭前刷新日志，缓存保证数据完全写入到磁盘。

3.1.2关闭数据库

3.1.3新的文件替换久版本文件

3.1.4启动数据库 mysqld\_safe --user=mysql --datadir=/path/to/existing-datadir

3.1.5执行mysqld\_upgrade：mysqld\_upgrade -uroot -p 这里注意应先关闭gtid\_mode

3.1.6重启数据库

mysqladmin -uroot -p shutdown;

mysqld\_safe --user=mysql --datadir=/path/to/existing-datadir

### 3.2 Logic Upgrade

3.2.1导出原数据：

Mysqldum --add-drop-table --routines --events --all-databases >data-for-upgrade.sql\_big\_tables

3.2.2安装新版本数据库

3.2.3导入开始的备份文件。

3.2.4配置主从服务器，新服务器加入现有集群，提供灰度使用并监控数据。

## 4 信息收集

4.1 收集现有集群服务器配置、硬件、软件版本等信息。

4.2 数据访问模式，是否通过代理，中间件特性等。

4.3 高可用的架构以及高可用切换功能是否可用。

## 5 具体实施

5.1 实施方式有2种，一是在线更改，二是根据资源使用情况选择新申请服务器或者停掉其中一台从库，进行停机更改维护再加入到环境中。结合我们的实际情况，我们采用第2种方式进行变更。（新服务器的规格需要不低于现有服务器配置。）

5.2 备份现有数据库信息。并上传至用来变更的服务器。

5.3 新服务器上安装并设置好my.cnf相关信息，诸如 server\_id、innodb\_open\_file、innodb\_open\_tables、innodb\_file\_io\_threads等只读参数。有一部分诸如innodb\_sort\_buffer等可动态更改的参数，我们可以后期动态调整。由于这里我们采用的是逐步，灰度升级的方式，基于GTID的特性，我们此处不能开启GTID。

5.4 在新服务器上还原备份文件，加入测试环境。进行测试和压测。其中测试主要是验证引擎转换后业务正常性测试；压测主要是用于测试配置参数对业务环境的性能。

5.5 基于5.4的结果，我们把新服务器加入生产环境验证。如无问题逐步替换从库并观察数据库状态。如果此时出现异常，我们可以踢掉新加的服务器，从新开始5.4这一步。因为是新加服务器，所以我们此时踢掉新服务器并不会影响业务。

5.6 切换主从，最后我们更改主库。完成全部变更。