2016-07-11 20:29:00,513 ERROR [0x000024f8] CMS <e:\svn\app-financial\trunk\ivms-8120\v1.0\client\psvcdc\fsvcdc\cmservice\dbhelper\DBHelper.h [51] - dbutils::TraceError> <>DB ---- DBHelper caller ExecuteUpsertSQL:UPDATE SoftDongle set nValue = 0, strDescribe = '', dtUpdateTime = now() where nType = 10

INSERT INTO SoftDongle (nType,nValue,strDescribe, dtUpdateTime) values(10, 0, '', now()) fail, error detail:

>DB ---- Code = 80004005

>DB ---- Code meaning = 未指定的错误

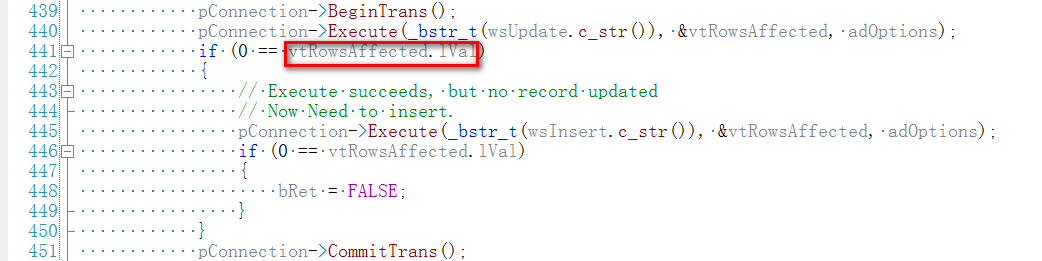
>DB ---- Source = Microsoft OLE DB Provider for ODBC Drivers

>DB ---- Description = [MySQL][ODBC 5.1 Driver][mysqld-5.6.22-log]Cannot execute statement: impossible to write to binary log since BINLOG\_FORMAT = STATEMENT and at least one table uses a storage engine limited to row-based logging. InnoDB is limited to row-logging when transaction isolation level is READ COMMITTED or READ UNCOMMITTED.

**① STATEMENT模式（SBR）**

每一条会修改数据的sql语句会记录到binlog中。优点是并不需要记录每一条sql语句和每一行的数据变化，减少了binlog日志量，节约IO，提高性能。缺点是在某些情况下会导致master-slave中的数据不一致(如sleep()函数， last\_insert\_id()，以及user-defined functions(udf)等会出现问题)

我们下面的代码，应该使用到了类似last\_insert\_id()这些功能了。



mysql中binlog\_format模式与配置详解

mysql复制主要有三种方式：基于SQL语句的复制(statement-based replication, SBR)，基于行的复制(row-based replication, RBR)，混合模式复制(mixed-based replication, MBR)。对应的，binlog的格式也有三种：STATEMENT，ROW，MIXED。

**① STATEMENT模式（SBR）**

每一条会修改数据的sql语句会记录到binlog中。优点是并不需要记录每一条sql语句和每一行的数据变化，减少了binlog日志量，节约IO，提高性能。缺点是在某些情况下会导致master-slave中的数据不一致(如sleep()函数， last\_insert\_id()，以及user-defined functions(udf)等会出现问题)

**② ROW模式（RBR）**

不记录每条sql语句的上下文信息，仅需记录哪条数据被修改了，修改成什么样了。而且不会出现某些特定情况下的存储过程、或function、或trigger的调用和触发无法被正确复制的问题。缺点是会产生大量的日志，尤其是alter table的时候会让日志暴涨。

**③ MIXED模式（MBR）**

以上两种模式的混合使用，一般的复制使用STATEMENT模式保存binlog，对于STATEMENT模式无法复制的操作使用ROW模式保存binlog，MySQL会根据执行的SQL语句选择日志保存方式。  
   
**binlog复制配置**

在mysql的配置文件my.cnf中，可以通过一下选项配置binglog相关

|  |  |
| --- | --- |
| 代码如下 | 复制代码 |
| **binlog\_format**           = MIXED                         //binlog日志格式，mysql默认采用statement，建议使用mixed **log**-**bin**                 = /data/mysql/mysql-**bin**.**log**    //binlog日志文件 expire\_logs\_days        = 7                           //binlog过期清理时间 max\_binlog\_size         = 100m                       //binlog每个日志文件大小 binlog\_cache\_size       = 4m                        //binlog缓存大小 max\_binlog\_cache\_size   = 512m                     //最大binlog缓存大小 | |

**三 MIXED说明**

对于执行的SQL语句中包含now()这样的时间函数，会在日志中产生对应的unix\_timestamp()\*1000的时间字符串，slave在完成同步时，取用的是sqlEvent发生的时间来保证数据的准确性。另外对于一些功能性函数slave能完成相应的数据同步，而对于上面指定的一些类似于UDF函数，导致Slave无法知晓的情况，则会采用ROW格式存储这些Binlog，以保证产生的Binlog可以供Slave完成数据同步。

现在来比较以下 SBR 和 RBR 2中模式各自的优缺点：

**SBR 的优点：**

历史悠久，技术成熟  
binlog文件较小  
binlog中包含了所有[数据库](http://www.111cn.net/database/database.html)更改信息，可以据此来审核数据库的安全等情况  
binlog可以用于实时的还原，而不仅仅用于复制  
主从版本可以不一样，从服务器版本可以比主服务器版本高

**SBR 的缺点：**

不是所有的UPDATE语句都能被复制，尤其是包含不确定操作的时候。  
调用具有不确定因素的 UDF 时复制也可能出问题  
使用以下函数的语句也无法被复制：  
\* LOAD\_FILE()  
\* UUID()  
\* USER()  
\* FOUND\_ROWS()  
\* SYSDATE() (除非启动时启用了 --sysdate-is-now 选项)  
INSERT ... SELECT 会产生比 RBR 更多的行级锁  
复制需要进行全表扫描(WHERE 语句中没有使用到索引)的 UPDATE 时，需要比 RBR 请求更多的行级锁  
对于有 AUTO\_INCREMENT 字段的 InnoDB表而言，INSERT 语句会阻塞其他 INSERT 语句  
对于一些复杂的语句，在从服务器上的耗资源情况会更严重，而 RBR 模式下，只会对那个发生变化的记录产生影响  
存储函数(不是存储过程)在被调用的同时也会执行一次 NOW() 函数，这个可以说是坏事也可能是好事  
确定了的 UDF 也需要在从服务器上执行  
数据表必须几乎和主服务器保持一致才行，否则可能会导致复制出错  
执行复杂语句如果出错的话，会消耗更多资源

**RBR 的优点：**

任何情况都可以被复制，这对复制来说是最安全可靠的  
和其他大多数数据库系统的复制技术一样  
多数情况下，从服务器上的表如果有主键的话，复制就会快了很多  
复制以下几种语句时的行锁更少：  
\* INSERT ... SELECT  
\* 包含 AUTO\_INCREMENT 字段的 INSERT  
\* 没有附带条件或者并没有修改很多记录的 UPDATE 或 DELETE 语句  
执行 INSERT，UPDATE，DELETE 语句时锁更少  
从服务器上采用多线程来执行复制成为可能

**RBR 的缺点：**

binlog 大了很多  
复杂的回滚时 binlog 中会包含大量的数据  
主服务器上执行 UPDATE 语句时，所有发生变化的记录都会写到 binlog 中，而 SBR 只会写一次，这会导致频繁发生 binlog 的并发写问题  
UDF 产生的大 BLOB 值会导致复制变慢  
无法从 binlog 中看到都复制了写什么语句  
当在非事务表上执行一段堆积的SQL语句时，最好采用 SBR 模式，否则很容易导致主从服务器的数据不一致情况发生

另外，针对系统库 mysql 里面的表发生变化时的处理规则如下：  
如果是采用 INSERT，UPDATE，DELETE 直接操作表的情况，则日志格式根据 **binlog\_format** 的设定而记录  
如果是采用 GRANT，REVOKE，SET PASSWORD 等管理语句来做的话，那么无论如何都采用 SBR 模式记录  
注：采用 RBR 模式后，能解决很多原先出现的主键重复问题。