mysql5.1向5.7数据迁移的步骤（数据量很大的，此次大概有50个G，）：

思路：

1. 创建用户，并授权
2. 搭建主从，做到主从同步，数据目前先保持一致，在切换主从同步切换ip

Mysql5.1已有数据量了，为保证数据的一致性。故需要锁库，等数据导入后再解锁

1. 需要注意从库和主库的引擎，字符集

show variables like 'character\_set%'; --查看当前环境的character\_set

show variables like 'collation%'; --查看当前的collation

1. 数据搬完后做数据校验，校验一致性

采用pt-table-checksum 进行校验用pt-table-sync恢复一致

Zabbix定时监控mysql是否一致，如果不一致，调用脚本自动修复

步骤：

开启二进制文件，做好my.Cnf配置

打开两个窗口，一个登陆上主库的mysql进行锁库：FLUSH TABLES WITH READ LOCK

在另一个窗口进行操作脚本：

A主机数据库导向B主机的数据

mysqldump --host=A\_ip -u用户 -p密码 --opt 库名 | mysql --host=B\_ip -u用户 -p密码 -C 库名

如：

mysqldump --host=10.1.31.198 -uroot -pCfgdcwhpmq2014 --opt software\_scheme | mysql --host=10.1.31.161 -uroot -pCfgdcwhpmq2014 -C software\_scheme

注意：software\_scheme要在B主机的数据库存在，否则会报错

写上脚本，锁表后运行，按照50G的数据量，大概要三个小时左右才能传输完成

传输完成后

解锁unlock tables

show master status 查看mysql\_master 的二进制文件和节点

配置备库的mysql.cnf,写上你想同步的库或表replicate-do-db = gdc-crm

CHANGE MASTER TO

MASTER\_HOST='10.1.31.161', //主服务器IP

MASTER\_USER='vnum', //主服务器用户

MASTER\_PASSWORD='vnum@123', //用户密码

MASTER\_PORT=3306, //端口

MASTER\_LOG\_FILE='mylog.001', //binlog日志文件名称

MASTER\_LOG\_POS=154, //日志位置

MASTER\_CONNECT\_RETRY=10; //如果同步失败的尝试次数

启动slave服务：

mysql > start slave; //关闭：stop slave

mysql > show slave status;

主从配置完成

再主备同步的时候在进行如下操作：

在主库上安装pt-table-checksum

安装：

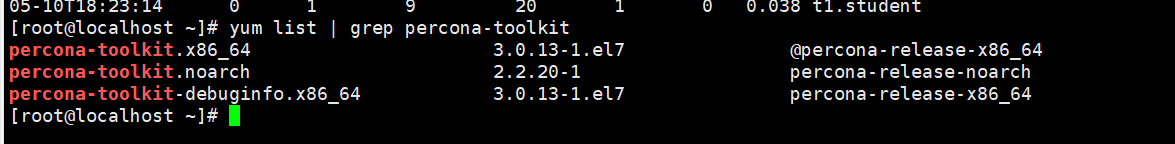
1.安装软件包：

Yum install perl perl-devel perl-Time-HiRes perl-DBI perl-DBD-MySQL

1. 安装工具percona-toolkit

yum install http://www.percona.com/downloads/percona-release/redhat/0.1-4/percona-release-0.1-4.noarch.rpm

yum list | grep percona-toolkit



yum install percona-toolkit

1. 安装pt-table-checksum

 wget [http://www.percona.com/get/percona-toolkit.tar.gz](http://www.percona.com/get/percona-toolkit.tar.gz" \t "https://blog.csdn.net/JSWANGCHANG/article/details/_blank)

1. 解压并安装

tar zxvf percona-toolkit.tar.gz

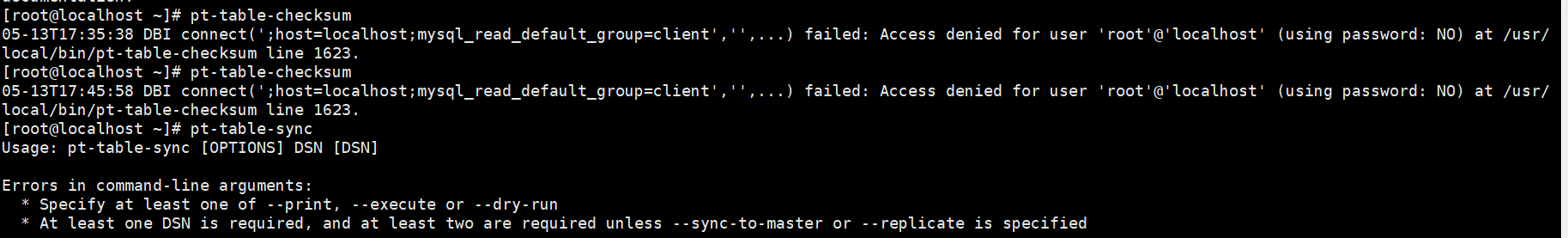
Cd percona-toolkit-3.0.13

perl Makefile.PL  
 make && make install

安装完成后：

pt-table-checksum

pt-table-sync 检查安装是否成功



如果报：

erl: relocation error: /usr/lib64/perl5/vendor\_perl/auto/DBD/mysql/mysql.so

如有报错：  
DBD::mysql::db do failed: Access denied; you need (at least one of) the SUPER privilege(s) for this operation [for Statement "/\*!50105 SET @@binlog\_format="STATEMENT"\*/"] at /usr/local/bin/mk-table-sync line 8568.  
Issuing rollback() due to DESTROY without explicit disconnect() of DBD::mysql::db handle ;host=192.168.1.101;port=3306;mysql\_read\_default\_group=client at /usr/local/bin/mk-table-sync line 8568.

解决办法：安装DBI和DBD-MySQL  
下载地址：  
[http://ftp.cuhk.edu.hk/pub/packages/perl/CPAN/authors/id/T/TI/TIMB/](http://ftp.cuhk.edu.hk/pub/packages/perl/CPAN/authors/id/T/TI/TIMB/" \t "https://blog.csdn.net/weixin_34067102/article/details/_blank)  
[http://search.cpan.org/dist/DBD-mysql/](http://search.cpan.org/dist/DBD-mysql/ " \t "https://blog.csdn.net/weixin_34067102/article/details/_blank)  
或者百度云盘下载：[https://pan.baidu.com/s/1miv0S8g](https://pan.baidu.com/s/1miv0S8g" \t "https://blog.csdn.net/weixin_34067102/article/details/_blank) （提取密码：fh24）

[root@master-server src]# tar -zvxf DBI-1.634.tar.gz   
[root@master-server src]# cd DBI-1.634  
[root@master-server DBI-1.634]# perl Makefile.PL  
[root@master-server DBI-1.634]# make && make install

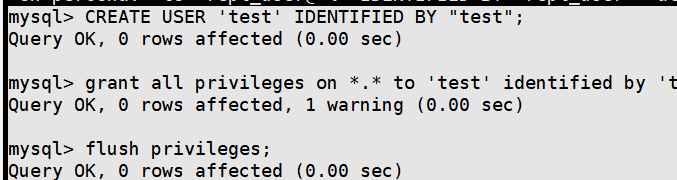
[root@master-server src]# tar -zvxf DBD-mysql-4.041\_01.tar.gz   
[root@master-server src]# cd DBD-mysql-4.041\_01  
[root@master-server DBD-mysql-4.041\_01]# perl Makefile.PL --mysql\_config=/usr/local/mysql/bin/mysql\_config         //主库mysql安装路径/usr/local/mysql  
[root@master-server DBD-mysql-4.041\_01]# make  
[root@master-server DBD-mysql-4.041\_01]# make install

1. 在主数据库master上建立用户并授权

CREATE USER 'test' IDENTIFIED BY "test";

grant all privileges on \*.\* to 'test' identified by 'test';

flush privileges;



1. 检测下test是否可以登录
2. 但是我们检查使用的mysql用户一般是没有 create table 权限的，所以你可能需要先手动创建：

CREATE DATABASE IF NOT EXISTS percona;CREATE TABLE IF NOT EXISTS percona.checksums (

db CHAR(64) NOT NULL,

tbl CHAR(64) NOT NULL,

chunk INT NOT NULL,

chunk\_time FLOAT NULL,

chunk\_index VARCHAR(200) NULL,

lower\_boundary TEXT NULL,

upper\_boundary TEXT NULL,

this\_crc CHAR(40) NOT NULL,

this\_cnt INT NOT NULL,

master\_crc CHAR(40) NULL,

master\_cnt INT NULL,

ts TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

PRIMARY KEY (db,tbl,chunk),

INDEX ts\_db\_tbl(ts,db,tbl)

) ENGINE=InnoDB;

1. 生产环境中数据库用户权限一般都是有严格管理的，假如连接用户是repl\_user（即直接用复制用户来检查），它应该额外赋予对其它库的 SELECT ，LOCK TABLES 权限，如果后续要用 pt-table-sync 就就需要写权限了。对percona库有写权限：

**GRANT** ALL PRIVILEGEES **on** percona.\* **to** repl\_user@'%' **IDENTIFIED** **BY** 'repl\_pass';

**GRANT** **SELECT**,**LOCK** **TABLES**,PROCESS,SUPER **on** \*.\* **to** repl\_user@'%';

注：

1. 为了减少不必要的麻烦，确保你的 repl\_user@'xxx' 用户能同时登陆主库和从库
2. --create-replicate-table 选项会自动创建 percona.checksums 表，但也意味着赋予额外的 CREATE TABLE权限给 percona\_tk@'xxx' 用户。默认yes
3. PROCESS用于自动发现从库信息，SUPER权限用于set binlog\_format。
4. --no-check-replication-filters 表示不需要检查 Master 配置里是否指定了 Filter。 默认会检查，如果配置了 Filter，如 replicate\_do\_db,replicate-wild-ignore-table,binlog\_ignore\_db 等，在从库checksum就与遇到表不存在而报错退出，所以官方默认是yes（--check-replication-filters）但我们实际在检测中时指定--databases=，所以就不存在这个问题，干脆不检测
5. --empty-replicate-table：每个表checksum开始前，清空它之前的检测数据（不影响其它表的checksum数据），默认yes。当然如果使用--resume启动检测数据不会清空。  
   当启用--noempty-replicate-table即不清空时，不计算计算chunk,只计算。
6. --databases=，-d：要检查的数据库，逗号分隔。用脚趾头想也知道 --databases-regex 正则匹配要检测的数据库，--ignore-databases[-regex]忽略检查的库。Filter选项。
7. --tables=，-t：要检查的表，逗号分隔。如果要检查的表分布在不同的db中，可以用--tables=dbname1.table1,dbnamd2.table2的形式。同理有--tables-regex，--ignore-tables，--ignore-tables-regex。--replicate指定的checksum表始终会被过滤。
8. --recursion-method：发现从库的方式。pt-table-checksum 默认可以在主库的 processlist中找到从库复制进程，从而识别出有哪些从库，但如果使用是非标准3306端口，会导致找不到从库信息。此时就会自动采用host方式，但需要提前在从库 my.cnf 里面配置report\_host、report\_port信息，如：

report\_host = MASTER\_HOSt report\_port = 1330

最终极的办法是dsn，dsn指定的是某个表（如 percona.dsns ），表行记录是改主库的（多个）从库的连接信息。适用以下任一情形：

主库不能自动发现从库

不想在从库添加额外配置（因为要重启）

主从检测连接用户信息不一样

多个从库时只想验证指定从库的一致

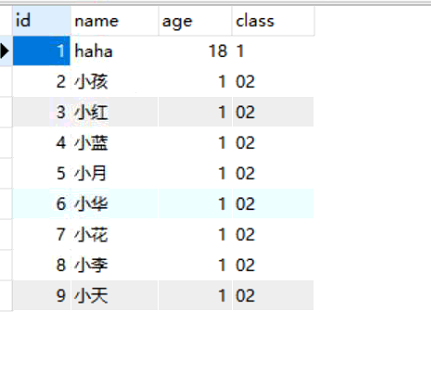
我比较倾向使用DSN的方式。这个dsns表只需要在执行 pt-table-checksum 命令的服务器上能够访问到就行。这里纠正一个认识，网上很多人说 pt-table-checksum 要在主库上执行，其实不是的，我的mysql实例比较多，只需在某一台服务器上安装percona-toolkit，这台服务能够同时访问主库和从库就行了。具体用法见后面实例。

**pt-table-checksum**注意事项：

1. 根据测试，需要一个即能登录主库，也能登录从库的账号；
2. 只能指定一个host，必须为主库的IP；
3. 在检查时会向表加S锁；
4. 如果master和slave的binlog日志不是STATEMENT格式，要用--no-check-binlog-format选项
5. 运行之前需要从库的同步IO和SQL进程是YES状态。
6. 表要有主键索引或唯一键索引
7. 安装完成测试：

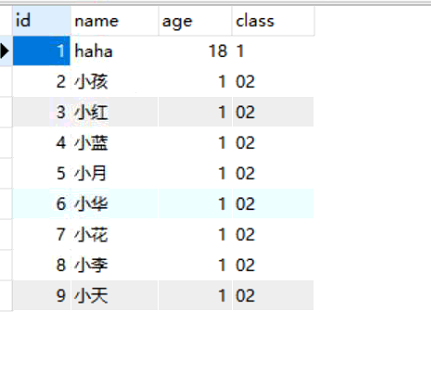
1. 在主库上(master:3306)，如下：

Select \* from t1.student;

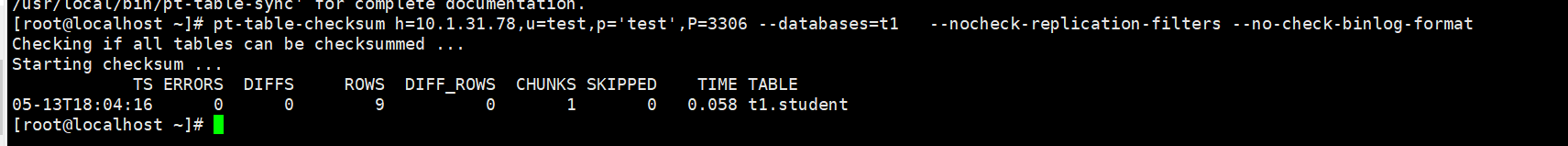


1. 在从库上（slave：3306）,如下：

Select \* from t1.student;



1. pt-table-checksum h=10.1.31.78,u=test,p='test',P=3306 --databases=t1 --nocheck-replication-filters --no-check-binlog-format



TS ：完成检查的时间。

ERRORS ：检查时候发生错误和警告的数量。

DIFFS ：0表示一致，1表示不一致。当指定--no-replicate-check时，会一直为0，当指定--replicate-check-only会显示不同的信息。

ROWS ：表的行数。

CHUNKS ：被划分到表中的块的数目。

SKIPPED ：由于错误或警告或过大，则跳过块的数目。

TIME ：执行的时间。

TABLE ：被检查的表名。

## 常见错误

1. Diffs cannot be detected because no slaves were found  
   不能自动找到从库，确认processlist或host或dsns方式用对了。
2. Cannot connect to h=slave1.\*.com,p=...,u=percona\_user  
   可以在pt-table-checksum命令前加PTDEBUG=1来看详细的执行过程，如端口、用户名、权限错误。
3. Waiting for the --replicate table to replicate to XXX  
   问题出在 percona.checksums 表在从库不存在，根本原因是没有从主库同步过来，所以看一下从库是否延迟严重。
4. Pausing because Threads\_running=25  
   反复打印出类似上面停止检查的信息。这是因为当前数据库正在运行的线程数大于默认25，pt-table-checksum 为了减少对库的压力暂停检查了。等数据库压力过了就好了，或者也可以直接 Ctrl+C 终端，下一次加上--resume继续执行，或者加大--max-load=值。
5. 字符集问题

Error checksumming table Error executing checksum query: DBD::mysql::st execute failed: Illegal mix of collations

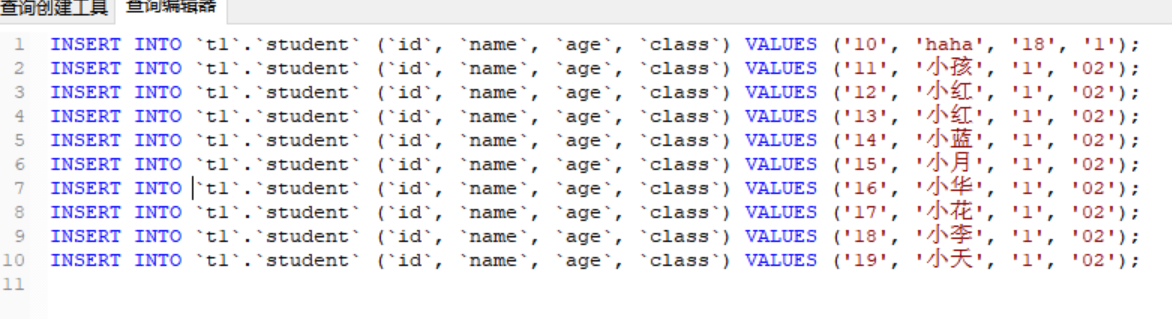
12-17T14:48:04 Error checksumming table d\_ec\_cs.t\_online\_cs: Error executing checksum query:

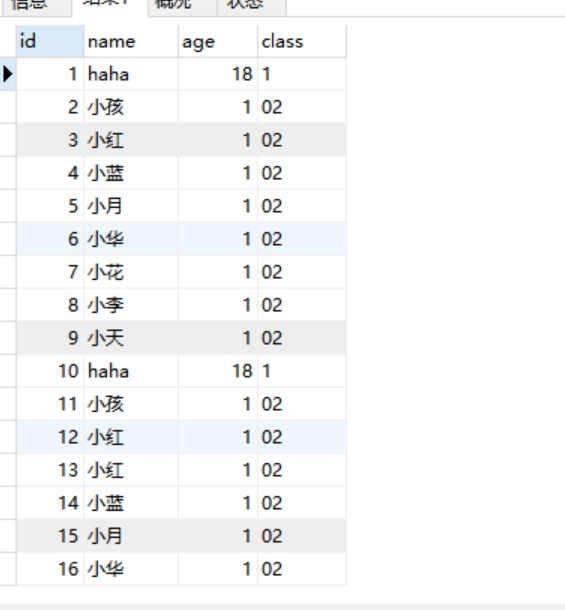
DBD::mysql::st execute failed: Illegal mix of collations for operation 'concat\_ws' [for Statement "REPLACE INTO `percona`.`ali\_checksum` (db, tbl, chunk, chunk\_index, lower\_boundary, upper\_boundary, this\_cnt, this\_crc) SELECT ?, ?, ?, ?, ?, ?, COUNT(\*) AS cnt, COALESCE(LOWER(CONV(BIT\_XOR(CAST(CRC32(CONCAT\_WS('#', `f\_cs\_id`, `f\_corp\_id`, `f\_valid`, `f\_show\_name`, `f\_online\_msg`, `f\_offline\_msg`, `f\_show\_mobile`, `f\_group\_id`, `f\_qq`, `f\_show\_qq`, `f\_msn`, `f\_show\_msn`, `f\_sms\_online`, `f\_scheme`, `f\_tel`, `f\_telno`, `f\_show\_tel`, `f\_contact`, `f\_mobile`, `f\_position`, `f\_other1`, `f\_other2`, `f\_other\_text1`, `f\_other\_text2`, `f\_email`, `f\_qq\_first`, `f\_qq\_first\_type`, `f\_aids\_open`, `f\_aids\_qq`, `f\_aids\_crmqq`, `f\_aids\_yahoo`, `f\_aids\_skype`, `f\_aids\_aliww`, `f\_aids\_msn`, `f\_aids\_alibaba`, `f\_aids\_alitrade`, CONCAT(ISNULL(`f\_show\_name`), ISNULL(`f\_group\_id`), ISNULL(`f\_qq`), ISNULL(`f\_show\_qq`), ISNULL(`f\_sms\_online`), ISNULL(`f\_other\_text1`), ISNULL(`f\_other\_text2`), ISNULL(`f\_email`)) )) AS UNSIGNED)), 10, 16)), 0) AS crc FROM `d\_ec\_cs`.`t\_online\_cs`

/\*checksum table\*/" with ParamValues: 0='d\_ts\_profile', 1='t\_user\_account', 2=1, 3=undef,

没找到解决办法

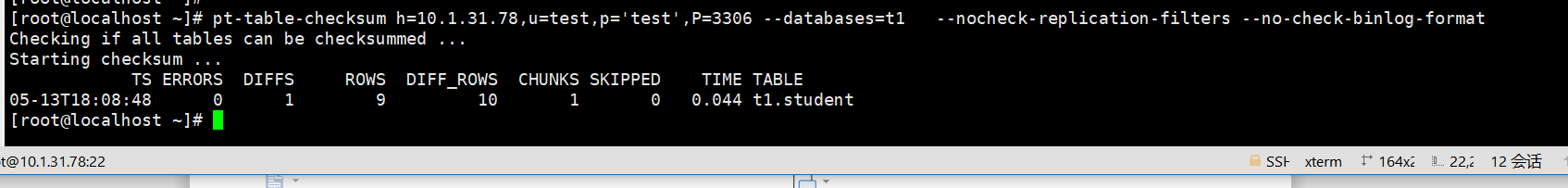
备库插入几笔数据





再次校验：

pt-table-checksum h=10.1.31.78,u=test,p='test',P=3306 --databases=t1 --nocheck-replication-filters --no-check-binlog-format



错10 ROWS

这时用pt-table-sync修复

i.自动消除差异（不推荐）

pt-table-sync --print --execute --sync-to-master h=10.1.31.77,P=3306,u=checksums,p='checksums' --databases=newtables --tables=tb1

ii.打印出sql语句，人工干预到Slave库执行（推荐）

pt-table-sync --print --sync-to-master h=10.1.31.77,P=3306,u=checksums,p='checksums' --databases=newtables --tables=tb1

或

pt-table-sync --print --sync-to-master h=10.1.31.77，P=3306,u=checksums,p='checksums'--replicate pt.checksums

#--sync-to-master : 指定一个DSN，即从的IP，他会通过show processlist或show slave status 去自动的找主。

#--replicate ：指定通过pt-table-checksum得到的表，这2个工具差不多都会一直用。

#--print ：打印，但不执行命令。

#--execute  ：执行命令。

备注：Slave需要授权主库Drop 和Create Temporary Tables权限

8.检验

重新执行一次pt-table-checksum，查看是否还存在差异。

四、注意事项

1.采用replace into来修复主从不一致，必须保证被replace的表上有主键或唯一键，否则replace into退化成insert into，起不到修复的效果。这种情况下pt-table-sync会采用其他校验和修复算法，但是效率非常低，例如对所有列的group by然后求count(\*)(表一定要有主键！)。

2.主从数据不一致需要通过replace into来修复，该sql语句必须是语句级。pt-table-sync会把它发起的所有sql语句都设置为statement格式，而不管全局的binlog\_format值。这在级联A-B-C结构中，也会遇到pt-table-checksum曾经遇到的问题，引起行格式的中继库的从库卡库是必然。不过pt-table-sync默认会无限递归的对从库的binlog格式进行检查并警告。

3.由于pt-table-sync每次只能修复一个表，所以如果修复的是父表，则可能导致子表数据连带被修复，这可能会修复一个不一致而引入另一个不一致；如果表上有触发器，也可能遇到同样问题。所以在有触发器和主外键约束的情况下要慎用。pt-table-sync工具同样也不欢迎主从异构的结构。pt-table-sync工具默认会进行先决条件的检查。

4.pt-table-sync在修复过程中不能容忍从库延迟，这正好与pt-table-checksum相反。如果从库延迟太多，pt-table-sync会长期持有对chunk的for update锁，然后等待从库的master\_pos\_wait执行完毕或超时。从库延迟越大，等待过程就越长，主库加锁的时间就越长，对线上影响就越大。因此要严格设置max-lag。

5.对从库数据的修复通常是在主库执行sql来同步到从库。因此，在有多个从库时，修复某个从库的数据实际会把修复语句同步到所有从库。数据修复的代价取决于从库与主库不一致的程度，如果某从库数据与主库非常不一致，举例说，这个从库只有表结构，那么需要把主库的所有数据重新灌一遍，然后通过binlog同步，同时会传递到所有从库。这会给线上带来很大压力，甚至拖垮集群。正确的做法是，先用pt-table-checksum校验一遍，确定不一致的程度：如果不同步的很少，用pt-table-sync直接修复；否则，用备份先替换它，然后用pt-table-sync修复。 说明： 这实际提供了一种对myisam备份的思路：如果仅有一个myisam的主库，要为其增加从库，则可以：先mysqldump出表结构到从库上，然后启动同步，然后用pt-table-sync来修复数据。