ARRS2.0指南

- ARRS的全称为：ArteryBase Realtime Replication Server，即：ArteryBase数据库实时复制服务器。
- 2.0版本实现abase数据库中的数据以毫秒级延迟复制到ArteryBase或者sybase ASE中，并提供二次开发接口，以方便复制到其他数据库、大数据平台等多种目标中。

安装约束

- ARRS只支持部署到centos 6.5 64bit 以上版本的服务器。
- 待复制的abase数据库源库支持部署在windows、centos上，推荐部署在centos上。
- ARRS待安装的服务器剩余磁盘空间至少需要100GB。

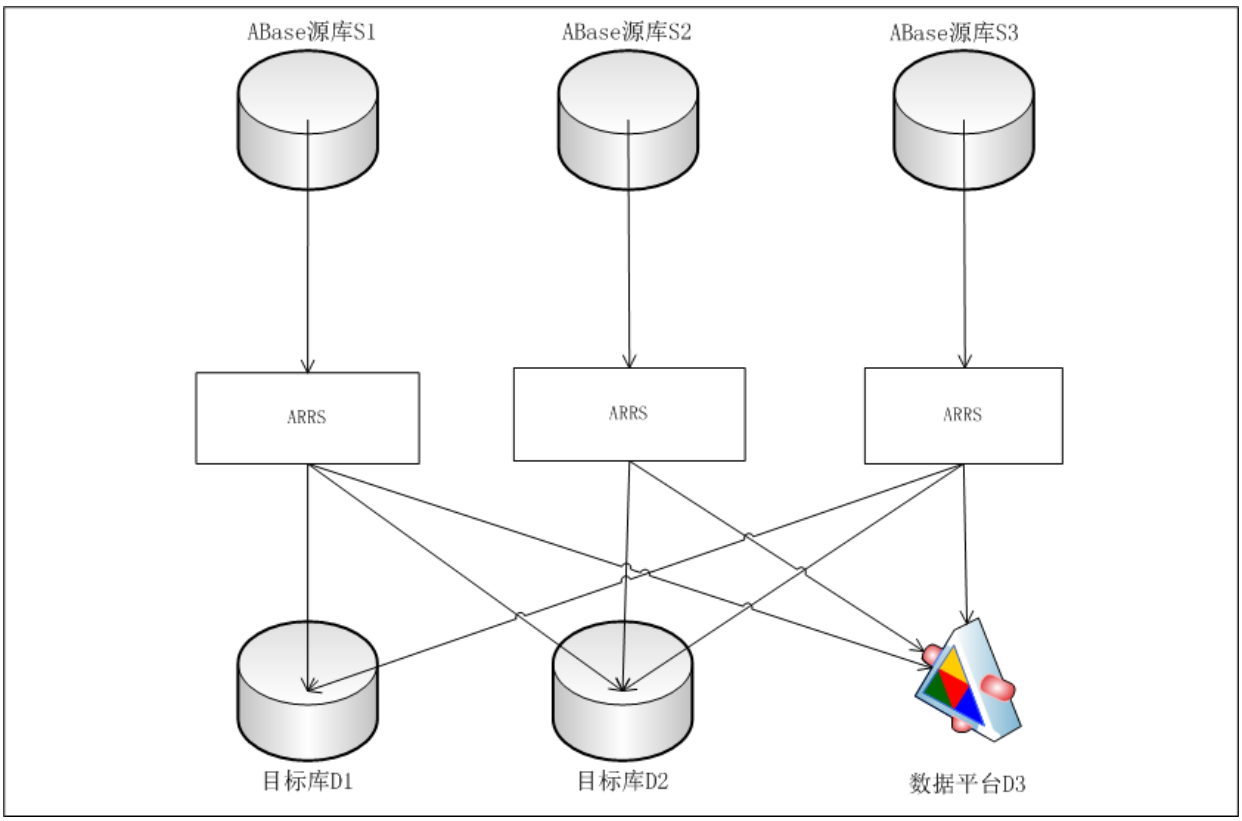
设计约束

- 单条记录不超过10MB。
- 没有主键的表只能复制insert，update和delete不会复制。
- 不支持bit和money类型的复制。
- 不支持DDL操作的复制。
- 如果一个事务涉及的元组更改超过1000条，将以1000条为一个批次向目标库中更新并commit。
- 最多支持8个复制通道。不同的复制通道之间的表应没有关联关系（如主子表关系、外键依赖关系），如一个abase源库中，民事相关的表走一个复制通道，刑事相关的表走一个复制通道，其他表走一个复制通道。

架构简介

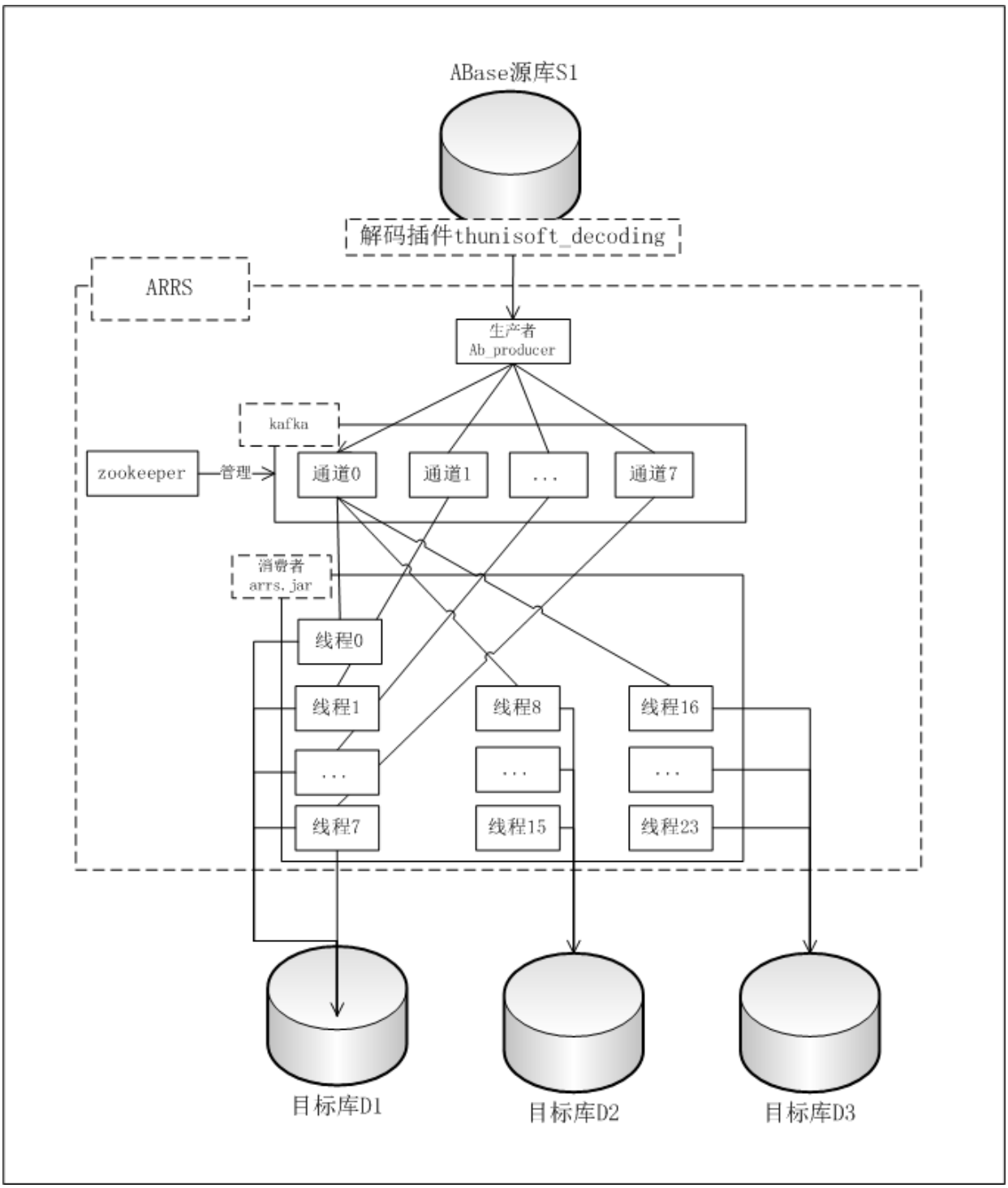
部署

整体上看，在整个复制中，包括abase源库、ARRS、目标库（大数据平台），一个ArteryBase源库需要一个ARRS实例，一个ARRS实例可以对应多个目标库。



组件

ARRS包括6个组件，分别是：解码插件、生产者、zookeeper、kafka、消费者、复制通道，除解码插件需要在源库部署外，其余5个组件作为一个整体部署在centos服务器上。
设置多个复制通道的目的是为了提升数据在目标库的执行速率，多个复制通道可以并行执行数据库更新操作，不同的复制通道的数据放到kafka不同的topic中。
各组件的逻辑关系如下：

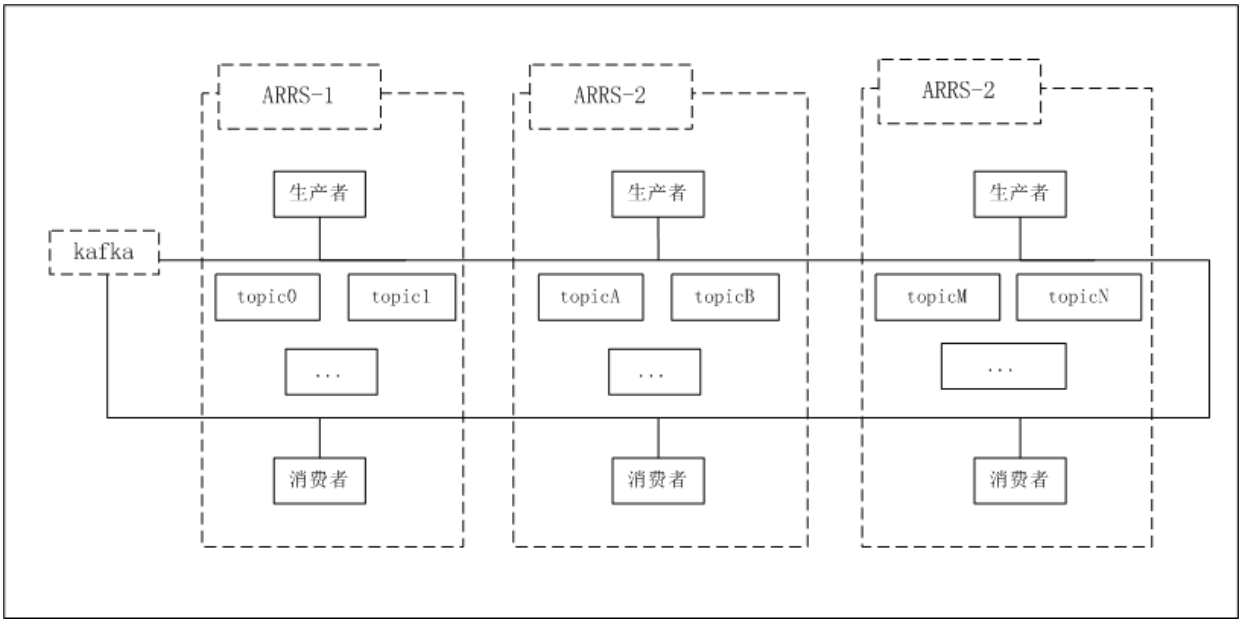


复制流程

当复制启动后，ArteryBase数据库的解码插件`thunisoft_decoding`从源库的wal事务日志中提取数据，由ARRS的生产者`ab_producer`接收处理后并推送到kafka，ARRS的消费者`arrs-2.0-jar-with-dependencies.jar`将kafka中的数据获得后，发送到目标端数据库并执行。

kafka在ARRS中的地位

kafka作为ARRS的消息队列，生产者将产生的消息推送到kafka的不同的topic中（每个复制通道都对应kafka一个单独的topic），消费者从kafka对应的topic中取数据。部署在同一台服务器上的多个ARRS实例可以复用同一个kafka。



安装文件

包括两部分：
arterybase-upgrade-logical.tar.gz：解码插件安装包，用于源库在abase上的解码插件一键安装。如果源库部署在windows操作系统上，请参阅：《windows操作系统安装复制插槽指南》
arrs.tar.gz：ARRS复制工具包，包括arrs复制程序、kafka、zookeeper。

安装步骤

解码插件安装（centos）

1、将`arterybase-upgrade-logical.tar.gz`复制到数据库服务器上，并将该文件的owner改为和数据库实例一样的名称，如`thunisoft`

```
chown thunisoft:thunisoft arterybase-upgrade-logical.tar.gz
```

2、解压

```
su thunisoft
tar -xf arterybase-upgrade-logical.tar.gz
```

3、安装
使用数据库的owner用户安装。进入`arterybase-upgrade-logical`，请根据实际情况修改`arterybase-upgrade-logical.conf`文件：

提示：

- ArteryBase 3.6.1版本的`PGDATA`默认安装路径是`/opt/thunisoft/abdata/3.6/abase1`，不是`/opt/thunisoft/abdata/abase1`。
- 需要根据自己创建的owner用户修改`APPUSER=`的值，有可能不是`thunisoft`用户；
- database_list配置需要复制的源数据库的名称。

然后安装：

```
cd arterybase-upgrade-logical
/bin/bash  arterybase-upgrade-logical.sh
```

安装成功后，提示：数据库实例逻辑复制功能配置完成。
说明安装成功了。
4、验证
启动数据库实例，然后查询复制槽是否创建，如果查询到相应的database上有复制槽，说明安装成功。

```
select * from pg_replication_slots;
 slot_name |       plugin       | slot_type | datoid | database | active
 | active_pid | xmin | catalog_xmin | restart_lsn
-----------+--------------------+-----------+--------+----------------+--
------+------------+------+--------------+-------------
 slot_arrs | thunisoft_decoding | logical   | 203640 | abase | f      |
    23725 |      |       282811 | 34/40016120
```

ARRS复制程序、kafka、zookeeper安装

- 应使用ARRS安装包自带的kafka、zookeeper，不能和其他系统共用kafka。禁止其他应用使用该kafka，以免影响ARRS复制的正常运行。
- 使用root账号安装
- 解压`arrs.tar.gz`

```
tar -xf arrs.tar.gz
```

- 进入arrs目录，修改`arrs-install.conf`
  修改`ROOTDIR`参数，指定arrs程序、kafka、zookeeper及数据目录安装到什么位置，按照技术标准要求，默认安装到`/opt/thunisoft`。
  修改`INSTANCE_NAME`参数，指定要安装几个复制实例（即从几个源库提取数据），多个实例以英文字符逗号分隔。
- 执行`arrs-install.sh`，会将程序、kafka、zookeeper及实例安装完毕。
- 检查：
  正常安装完毕后，程序安装到`/opt/thunisoft/arrs/program/2.0`目录下，在该目录下可看到bin、lib目录；kafka安装到`/opt/thunisoft`目录下；zookeeper、kafka的启停脚本安装到`/opt/thunisoft/arrs/rpdata/2.0`目录下；kafka的数据目录
  在`/opt/thunisoft/kafka-logs`；各复制实例安装

到`/opt/thunisoft/arrs/rpdata/2.0`目录下，在该目录下可看到创建的多个复制实例，各实例目录中包括`config_producer.properties`、`config_consumer_xxx.properties`、以及启停、管理脚本`startup_arrs_xx`、`stop_arrs_xx.sh`、`cmd_arrs_xx.sh`。

使用

保证源库、目标库处于运行中

启动kafka、zookeeper

- root账号，进入`/opt/thunisoft/arrs/rpdata/2.0`，使用`startup_kz.sh`启动kafka、zookeeper。

```
cd /opt/thunisoft/arrs/rpdata/2.0
./startup_kz.sh
```

停止kafka使用`stop_kz.sh`

```
cd /opt/thunisoft/arrs/rpdata/2.0
./stop_kz.sh
```

配置并启动arrs

root账号，进入`/opt/thunisoft/arrs/rpdata/2.0/xx`。

生产者依赖的config_prducer.properties文件

该文件是生产者进程`ab_producer`需要的配置文件。
修改`config_prducer.properties`中配置信息，**注意数据库用户需要是超级管理员用户**：

- `source_hostname`为abase源库的方便识别的名称，请使用英文，不要有空格、冒号、大于、小于号等特殊字符。
- `source_host=172.16.xx.xx，source_port=5432，source_user=sa，source_password=123456，source_database=abase`复制源数据库配置，请根据实际情况自行修改。
- `kafka_server`为kafka服务的地址，如果使用本次安装的kafka，无需修改该参数值。
- `use_regex`表明是否支持正则表达式匹配，根据需要自行修改。
- `blacklist`复制黑名单，如果那些表不想复制，可以在此处添加。

- `tables`白名单，待复制表名，支持穷举及正则表达式，多表用逗号隔开。黑名单优先级高于白名单。
- `topic_group_num`，使用几个复制通道，最多支持8个复制通道。
- `topic_groups`：源库各表根据正则表达式是否匹配进入不同的复制通道。
  如以下配置：

```
topic_groups=(
^regress\.tbl_with_all_type$
^regress\.testblack,^regress\.testblack1.*
^regress\.testtables$
.*
)
```

代表`regress.tbl_with_all_type`的数据进入第一个复制通道，匹配`regress.testblack`或`regress.testblack1.*`的表进入第二个复制通道，`regress.testtables`进入第三个复制通道，其他表进入第四个复制通道。

> **注意：不同的复制通道之间的表应没有关联关系（如主子表关系、外键依赖关系）。**

- gp_dist_key参数的作用是：当目标库是greenplum时，在此标识其分布键，避免update时，更新分布键。如果目标库非greenplum，无需配置该参数。
  如以下配置：

```
gp_dist_key=(
public\.T_AJ:C_BH,C_AJBS
.*:C_AJBS
)
```

代表T_AJ表的分布键是`C_BH,C_AJBS`，生成的sql语句将不含这两个字段的update。其余表的分布键为C_AJBS。

- `repl_decode_format`解码结果格式，可以为sql/json。只支持son格式。
- repl_json_needtype 代表json格式的结果是否需要输出类型，如：`{"col1":{"integer":"123456"}}`
  只支持值为true的情形。
- `slotname=slot_arrs`为源库database复制槽名称，如果有多个复制槽，这个地方请注意改名字。

消费者`arrs-2.0-jar-with-dependencies.jar`依赖的配置文件config_consumer_xx.properties

- 该文件可以创建多个，以方便一份源数据复制到多个目标库，比如我们想将源库数据同时复制到汇总库、大数据分析库，那么，我们创建并命名为config_consumer_hzk.properties、config_consumer_dsj.properties文件，分别代表复制到汇总库和大数据分析库。

config_consumer_xx.properties配置文件中的内容：

- `target_hostname`为目标库名称（用于和其他库区分的名称，如数据汇总库，该值可以配置为hz），该名称和其他的config_consumer_xx.properties文件中的该值必须不同，建议和文件名中的xx保持一致。
- `target_type`为目标库类型配置(目前只支持abase和sybase ASE)，sybase ASE需配置为ase，ABase配置为abase，其他的需要自行实现并写实现类名。

```
target_type=ase
```

- 连接参数配置：分别是目标数据库机器ip、端口、用户名、密码、database。

```
target_host=
target_port=
target_database=
target_user=sa
target_password=
```

- `target_charset`：目标库的字符集
  如果目标库是abase，那么写utf8，如果是sybase ASE，请使用sp_helpsort命令查看，一般是utf8或者cp936
- `schema_mapping`：原库schema与目标库schema（sybase ASE为database）的映射关系若存在多个，用’,’分隔，如：`(public:YWST,regress:DB_PG)`，代表源库的public 模式下的表数据复制到目标库的YWST模式（sybase ASE为database）下的对应表，regress模式下的表数据复制到目标库的DB_PGYWST模式（sybase ASE为database）下。
- 启动
  在各实例目录，如`/opt/thunisoft/arrs/rpdata/2.0/xx`目录，使用`startup_arrs_xx.sh`启动。

```
./startup_arrs_xx.sh
```

- 停止

```
./stop_arrs_xx.sh
```

- 查看运行状态，通过status命令查看。包括：生产者的运行状态，消费者各线程的运行状态等。

```
./cmd_arrs_xx.sh status
```

通过该命令，可以看到生产者和各线程消费者的运行情况，最新的复制进度、最新的报错信息等。

- 跳过出错的事务，通过`skipall/skip -t threadname`来实现。
  skipall的含义为所有暂停的线程都跳过错误的事务，继续运行：

```
./cmd_arrs_xx.sh skipall
```

skip -t threadname的含义为只针对指定的线程跳过错误的事务并继续进行，不影响其他线程：

```
./cmd_arrs_xx.sh skip -t threadname
```

skip执行后，跳过的线程会在以线程名命名的目录下`skip_records.res`中写入信息，并在`history`目录生成以skip掉事务的lsn命名的文件。

- 生产者进程的错误（如kafka宕机恢复后），消费者进程的错误（如缺少字段、服务器宕机等）解决后，可通过`resumeall/resume -t threadname`来继续运行。
  该参数含义和skip保持一致。
- 查看帮助

```
./cmd_arrs_xx.sh --help
```

- 查看运行日志
  进入`/opt/thunisoft/arrs/rpdata/2.0/xx/logs`目录下查看。
  其中以`producer`开头的为生产者日志，`consumer`开头的为消费者日志。根据文件名中的日期查看。

rpdata/xx目录下生成的文件、文件夹说明

- `foo.info`记录arrs复制程序无法确定是否成功提交数据库的事务，一般发生在目标库发生宕机时。需要管理员人工确定运行情况，并删除foo.info后，arrs方可启动。
- command文件，为arrs程序向消费者进程发命令所使用的文件，请勿删除。
- ab_producer.pid、consumer.pid分别为生产者、消费者进程的pid，请勿删除。
- xx_0_to_yy、xx_0_to_zz：源库xx通过某复制通道到yy库的复制进展情况，这些目录及目录下的文件`禁止删除`，否则将导致复制出现严重错误！

运维要求

日常检查要求，每天至少检查一次

- 复制的运行情况，可以使用如下命令：

```
./cmd_arrs_xx.sh status
```

- ARRS所在服务器上磁盘占用情况，避免磁盘空间被占满。
- ARRS各实例的日志，日志在`/opt/thunisoft/arrs/rpdata/2.0/xx/logs`目录下。
- kafka的运行情况，可使用`jps`命令，查看进程列表中是否有kafka。

- zookeeper的运行情况，可使用`netstat -tpnl | grep 2182`查看是否有2182端口。
- 源库`$PGDATA/pg_xlog`目录的文件大小，当超过3GB，就需要特别关注ARRS是否正常。

其他要求

- 修改表结构时，应先修改目标库的表结构，再修改源库的表结构。表结构修改完毕后，应使用以上命令及时关注复制的运行状况。
- 源库临时创建表，表名应包含tmp、temp、backup、bak等前缀或后缀，以免目标库缺少对应的表导致复制阻塞。
- 进行数据校正时，校正完毕后，特别是涉及百万条以上数据的修正，应及时关注复制的运行情况，保证待复制的数据被及时处理。
- 源库、目标库、ARRS、kafka、zookeeper宕机，重启后，应及时关注复制的运行状况。
- abase数据库`pg_xlog`目录下的文件**禁止删除**！！！

通过API接口获得增量数据

ARRS2.0提供了API，以方便开发人员二次开发，以支持不同的数据库、大数据平台、文件系统等。
需实现资源获得、事务开启、数据处理、事务commit、资源释放等接口，同时提供了较多的工具类和基础实现类以减少二次开发编码量，具体可咨询ArteryBase开发小组。

ARRS路线图

后续版本，将实现以下功能，敬请期待。

- DDL语句复制
- 数据重传
- 表过滤
- 行过滤
- 更完善的复制管理