

canal调研

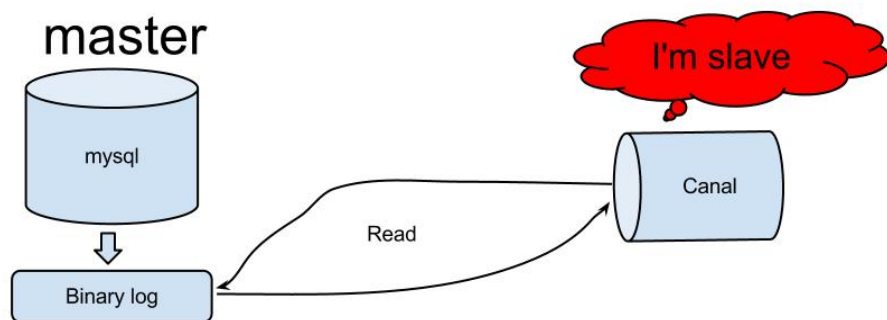
Created by 杨超, last modified on 2019 Apr 15

- 1 简介
- 2 使用
 - 2.0 说明
 - 2.1 canal-server
 - 2.2 canal-example
 - 2.3 canal-adapter-client
- 3 现有问题及解决方式

1 简介

canal (<https://github.com/alibaba/canal>) 是一个阿里的开源项目，主要用来在两种不同数据库直接增量同步数据，阿里的dts/otter等是基于它开发的

调研的目的是为了从rds->es同步实时数据，canal的原理就是模拟成mysql的一个从库获取binlog，然后再通过订阅的形式推送到其他数据库 也可以订阅推送到一个mq队列里，然后自己再另行处理 支持分布式集群





2 使用

最新: <https://github.com/alibaba/canal/releases>

所有的测试都是基于 v1.1.3

2.0 说明

 canal.adapter-1.1.3.tar.gz	96.6 MB
 canal.deployer-1.1.3.tar.gz	48.6 MB
 canal.example-1.1.3.tar.gz	23.3 MB

特别注意 有3个gz

- 1 deployer里面放着server
- 2 example里面放着供测试使用的client 只打印日志 类似echo的功能 建议开启server没报错之后先拿这个example试一下binlog是否通了 – 当然这部不必须
- 3 adapter说白了就是写好了的client 支持es /redis/ mongo等

所以 启动canal需要先开启至少一个server来同步接受mysql的binlog 然后再开启N个client来订阅server接收的binlog然后执行

ps: 其实还可以直接下载源码 然后重新编译 来扩展功能 但是这样比较麻烦 我没有采用这种方案

下面通过演示dev线下数据库在newdev机器上的配置使用来说明

顺序为 2.1 开始server ===》 2.2 开始example测试 ==> 2.3 关闭exmpale 开始es-adapter

2.1 canal-server

测试代码位于:

@newdev/tmp/canal-server

建议配置之前先阅读文档: <https://github.com/alibaba/canal/wiki/QuickStart>

1 下载deployer的代码 放在某个文件夹下

2 conf/canal.properties:

这个文件里配置canal服务器的port和其他 一般来说只要改下port就好

```
canal.port = 11111
```

3 conf/example/instance.properties:

在这里配置名为"example"的实例

这里主要配置rds数据库的地址 和 用户名/密码 !!!!特别注意建议defaultDatabaseName留空, 现在发现过滤好像有问题 建议binlog全量接收

```
canal.instance.master.address=rm-***.mysql.rds.aliyuncs.com:3306
canal.instance.dbUsername=user
canal.instance.dbPassword=passwd
canal.instance.connectionCharset = UTF-8
canal.instance.defaultDatabaseName =
```

4 sh bin/startup.sh

配置完毕之后开启服务

logs/canal/canal.log 里看见 the next step is binlog dump 基本上就ok了

然后logs/example里面会放example实例打印的日志并按天收集 没啥太多有用信息

想看binlog信息 需要后面打开client订阅之后才可以实时看到

```
2019-04-04 17:14:33.910 [destination = example , address = rm-m5ed42yxxg6bx4b3p8fo.mysql.rds.aliyuncs.com/47.105.19.149:3306
, EventParser] WARN c.a.o.c.p.inbound.mysql.rds.RdsBinlogEventParserProxy - ---> begin to find start position, it will be l
ong time for reset or first position
2019-04-04 17:14:33.911 [destination = example , address = rm-m5ed42yxxg6bx4b3p8fo.mysql.rds.aliyuncs.com/47.105.19.149:3306
, EventParser] WARN c.a.o.c.p.inbound.mysql.rds.RdsBinlogEventParserProxy - prepare to find start position just show master
status
2019-04-04 17:14:39.632 [destination = example , address = rm-m5ed42yxxg6bx4b3p8fo.mysql.rds.aliyuncs.com/47.105.19.149:3306
, EventParser] WARN c.a.o.c.p.inbound.mysql.rds.RdsBinlogEventParserProxy - ---> find start position successfully, EntryPos
ition[included=false,journalName=mysql-bin.003143,position=165027,serverId=2869757446,gtid=,timestamp=1554369270000] cost :
5708ms , the next step is binlog dump
```

```
[yangchao@newdev:canal-server]$ ll logs/example/
total 740
drwxrwxr-x 2 yangchao www 4096 Apr 5 17:36 2019-04-04
drwxrwxr-x 2 yangchao www 4096 Apr 6 17:36 2019-04-05
drwxrwxr-x 2 yangchao www 4096 Apr 7 17:36 2019-04-06
drwxrwxr-x 2 yangchao www 4096 Apr 8 17:36 2019-04-07
drwxrwxr-x 2 yangchao www 4096 Apr 9 17:36 2019-04-08
drwxrwxr-x 2 yangchao www 4096 Apr 10 17:36 2019-04-09
drwxrwxr-x 2 yangchao www 4096 Apr 11 17:36 2019-04-10
drwxrwxr-x 2 yangchao www 4096 Apr 12 17:10 2019-04-11
drwxrwxr-x 2 yangchao www 4096 Apr 13 17:36 2019-04-12
drwxrwxr-x 2 yangchao www 4096 Apr 14 17:36 2019-04-13
drwxrwxr-x 2 yangchao www 4096 Apr 15 00:00 2019-04-14
-rw-rw-r-- 1 yangchao www 2248 Apr 14 17:36 example.log
-rw-rw-r-- 1 yangchao www 700486 Apr 15 15:42 meta.log
```

2.2 canal-example

这个工程本质上是一个demo log-client 就是一个只打印全量日志的客户端 类似echo接口一样 用来测试canal-server是否真正正常工作了 有时候也可以作为client参照组, 比如client没有接收到一条binlog消息 可以查看canal-example里面是否收到了 从而确定是server的问题还是client的问题

1 首先下载canal-example的包 解压到一个独立的文件夹下

测试代码位于 @newdev/tmp/canal-example

2 conf/client.properties

这个文件里只要确保一行写对 就是client目标配置写成server里配的实例名"example"即可

```
client.destination=example
```

3 sh bin/startup.sh

启动完成之后 logs/example/entry.log

就可以看到binlog了, 然后对一个表进行 crud变更操作就可以看到log了 比如下面的示例就是对up_user表某条记录的update

```
*****
* Batch Id: [36417], count: [3], memsize: [389], Time: 2019-04-10 15:22:14
* Start: [mysql-bin.003166:1502215:1554880934000(2019-04-10 15:22:14)]
* End: [mysql-bin.003166:1502656:1554880934000(2019-04-10 15:22:14)]
*****

===== binlog[mysql-bin.003166:1502215], executeTime: 1554880934000(2019-04-10 15:22:14), gtid: ○, delay: 255ms
BEGIN ----> Thread id: 464637
----- binlog[mysql-bin.003166:1502376], name[up_user], eventType: UPDATE, executeTime: 1554880934000(2019-04-10 15:22:14), gtid:
○, delay: 255 ms
id: 92 type=int(11) unsigned
userId: 4 type=int(11) unsigned
userName: 崔志浩 type=varchar(36) update=true
email: 12312312@biah.com type=varchar(128)
status: 4 type=int(11)
attr: {"isFirst":false} type=text
firstLoginTime: 2018-11-22 13:40:20 type=datetime
beginTime: 2018-11-22 15:16:10 type=datetime
endTime: 2038-11-17 00:00:00 type=datetime
creationTime: 2018-11-22 13:40:20 type=datetime
updateTime: 2018-12-03 11:15:14 type=datetime
phone: 13051891037 type=varchar(36)
position: type=varchar(36)
fundId: 19 type=varchar(36)
fundName: 测试基金 type=varchar(128)
dt_modtime: 2019-04-10 15:22:14 type=datetime update=true
-----
END ----> transaction id: 6135163
===== binlog[mysql-bin.003166:1502656], executeTime: 1554880934000(2019-04-10 15:22:14), gtid: ○, delay: 255ms
*****
* Batch Id: [36418], count: [4], memsize: [395], Time: 2019-04-10 15:22:19
* Start: [mysql-bin.003166:1502735:1554880939000(2019-04-10 15:22:19)]
* End: [mysql-bin.003166:1503209:1554880939000(2019-04-10 15:22:19)]
*****
```

2.3 canal-adapter-client

顺利通过2.1/2.2的时候 证明server同步ok 能正确获取binlog了

然后下载canal-adapter并解压到一个独立文件夹

测试代码位于 @newdev/tmp/canal-es-client2 里

建议配置之前先读下adapter和toEsAdapter的文档

<https://github.com/alibaba/canal/wiki/ClientAdapter>

<https://github.com/alibaba/canal/wiki/Sync-ES>

1 vi conf/application.yml

```
server:
  port: 20811 #配置一个独立的port
canal.conf:
  srcDataSources:
    defaultDS:
      url: jdbc:mysql://rm-***.mysql.rds.aliyuncs.com:3306/up?useUnicode=true
      username: user
      password: passwd
  canalAdapters:
    - instance: example # canal instance Name or mq topic name
      groups:
        - groupId: g1 #随便
          outerAdapters:
            - name: es
              hosts: newdev.ethercap.com:*** #es地址
              properties:
                cluster.name: docker-cluster #es集群名字 !!! 这个一定要写对, 如果不知道 curl newdev.ethercap.com:port看一下
```

上面提到的集群名字 如果不知道 可以curl es地址:9200看一下, 如下所示, docker里的es的9200端口被我映射到了9527上 所以curl了9527端口

```
[yangchao@newdev:canal-es-client2]$ curl newdev.ethercap.com:9527
{
  "name" : "IHACxEs",
  "cluster_name" : "docker-cluster",
  "cluster_uuid" : "Dsk17hvXTuWRHamTN7nXkg",
  "version" : {
    "number" : "6.7.0",
    "build_flavor" : "default",
    "build_type" : "docker",
    "build_hash" : "8453f77",
    "build_date" : "2019-03-21T15:32:29.844721Z",
    "build_snapshot" : false,
    "lucene_version" : "7.7.0",
    "minimum_wire_compatibility_version" : "5.6.0",
    "minimum_index_compatibility_version" : "5.0.0"
  },
  "tagline" : "You Know, for Search"
}
```

2 在conf/es 下面 新建需要的yml文件

比如我要同步up_user这张表，那么就新建一个up_user.yml文件

- sql那一栏写上select语句并且将对应的字段名重命名好
- es里的index要提前手动建好
- 其他一些高阶用法文档和示例中有 不再赘述

```
dataSourceKey: defaultDS
destination: example
groupId: g1
esMapping:
  _index: up_user
  _type: default
  _id: _id
  upsert: true
  pk: id
  sql: "select a.userId as _id, a.userId as id, a.userName as name, a.updateTime as c_time from up_user a
      "
  etlCondition: ""
  commitBatch: 3000
```

3 开始同步

sh bin/startup.sh

启动后 看less logs/adapter/adapter.log

在mysql里对up_user表进行 update/insert/delete操作 就可以看到收到对应的log并且推送给了es里对应的index 然后自己手动对比一下Mysql 和 es里的数据是否相同

ok 走到这里 就完成了从rds -> es的canal同步

```
2019-04-12 10:15:24.204 [Thread-4] INFO c.a.o.canal.adapter.launcher.loader.CanalAdapterWorker - > Start to subscribe destination: example
2019-04-12 10:15:24.245 [Thread-4] INFO c.a.o.canal.adapter.launcher.loader.CanalAdapterWorker - > Subscribe destination: example succeeded
2019-04-12 10:17:28.906 [pool-2-thread-1] DEBUG c.a.otter.canal.client.adapter.es.service.ESSyncService - DML: {"data":[{"id":123,"userId":157853,"userName":"daxiang","email":"","status":0,"attr":"","firstLoginTime":1552579200000,"beginTime":1552579200000,"endTime":1552579200000,"creationTime":1554947947000,"updateTime":1554947947000,"phone":"","position":"","fundId":"","fundName":"","dt_modtime":1555035448000}], "database":"up","destination":"example","es":1555035448000,"groupId":null,"isDdl":false,"old":[{"userName":"dn","dt_modtime":1554948194000}], "pkNames":["id"],"sql":"","table":"up_user","ts":1555035448689,"type":"UPDATE"}
Affected indexes: up_user
2019-04-12 10:18:16.744 [pool-2-thread-1] DEBUG c.a.otter.canal.client.adapter.es.service.ESSyncService - DML: {"data":[{"id":123,"userId":157853,"userName":"daxiang2","email":"","status":0,"attr":"","firstLoginTime":1552579200000,"beginTime":1552579200000,"endTime":1552579200000,"creationTime":1554947947000,"updateTime":1554947947000,"phone":"","position":"","fundId":"","fundName":"","dt_modtime":1555035496000}], "database":"up","destination":"example","es":1555035496000,"groupId":null,"isDdl":false,"old":[{"userName":"daxiang","dt_modtime":1555035448000}], "pkNames":["id"],"sql":"","table":"up_user","ts":1555035496743,"type":"UPDATE"}
Affected indexes: up_user
2019-04-12 10:18:52.569 [pool-2-thread-1] DEBUG c.a.otter.canal.client.adapter.es.service.ESSyncService - DML: {"data":[{"id":123,"userId":157853,"userName":"dax","email":"","status":0,"attr":"","firstLoginTime":1555035560000,"beginTime":1555035560000,"endTime":1555035560000,"creationTime":1555035560000,"updateTime":1555035560000,"phone":"","position":"","fundId":"","fundName":"","dt_modtime":1555035560000}], "database":"up","destination":"example","es":1555035532000,"groupId":null,"isDdl":false,"old":null,"pkNames":["id"],"sql":"","table":"up_user","ts":1555035532568,"type":"DELETE"}
Affected indexes: up_user
2019-04-12 10:19:20.789 [pool-2-thread-1] DEBUG c.a.otter.canal.client.adapter.es.service.ESSyncService - DML: {"data":[{"id":124,"userId":157853,"userName":"dax","email":"","status":0,"attr":"","firstLoginTime":1555035560000,"beginTime":1555035560000,"endTime":1555035560000,"creationTime":1555035560000,"updateTime":1555035560000,"phone":"","position":"","fundId":"","fundName":"","dt_modtime":1555035560000}], "database":"up","destination":"example","es":1555035560000,"groupId":null,"isDdl":false,"old":null,"pkNames":["id"],"sql":"","table":"up_user","ts":1555035560761,"type":"INSERT"}
Affected indexes: up_user
```

3 现有问题及解决方式

现有最大的问题在于现在的es adapter还不支持es的用户名+密码认证 所以无法推送 本身es-adapter也比较新 2018.11末才释放第一个alpha版本 可能需要等稳定了再说 现在的处理方式就是先改用go写的这个es的工具: <https://github.com/siddontang/go-mysql-elasticsearch>

Like Be the first to like this

No labels

地址：北京市朝阳区建国路86号佳兆业广场北塔6层梦想加空间601室

以太资本由艾普拉斯投资顾问(北京)有限公司运营，提供早期互联网项目的投融资对接服务

©2014-2017 以太资本 京ICP备14028208号