# Canal

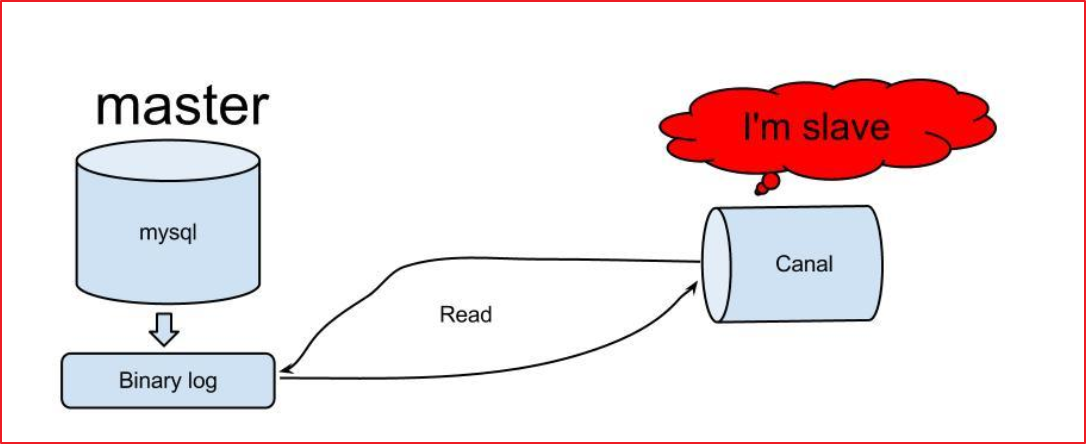
## 一、canal

canal可以用来监控数据库数据的变化，从而获得新增数据，或者修改的数据。

canal是应阿里巴巴存在杭州和美国的双机房部署，存在跨机房同步的业务需求而提出的。

阿里系公司开始逐步的尝试基于数据库的日志解析，获取增量变更进行同步，由此衍生出了增量订阅&消费的业务。

### 1.1 Canal工作原理



原理相对比较简单：

1. canal模拟mysql slave的交互协议，伪装自己为mysql slave，向mysql master发送dump协议
2. mysql master收到dump请求，开始推送binary log给slave(也就是canal)
3. canal解析binary log对象(原始为byte流)

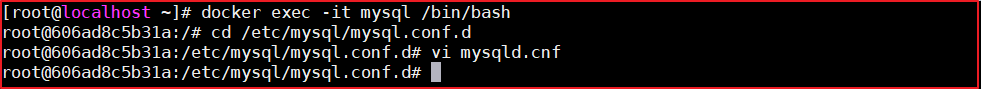
canal需要使用到mysql，我们需要先安装mysql,给大家发的虚拟机中已经安装了mysql容器，但canal是基于mysql的主从模式实现的，所以必须先开启binlog.

### 1.2 开启binlog模式

linux上安装mysql容器.此处不在演示。

(1) 修改/etc/my.cnf 需要开启主 从模式，开启binlog模式。

执行如下命令，编辑mysql配置文件



命令行如下：

docker exec -it mysql /bin/bash  
cd /etc/mysql/mysql.conf.d  
vi mysqld.cnf

修改mysqld.cnf配置文件，添加如下配置：



上图配置如下：

log-bin=/var/lib/mysql/mysql-bin  
server-id=12345

(2) 创建账号 用于测试使用,

使用root账号创建用户并授予权限

create user canal@'%' IDENTIFIED by 'canal';  
GRANT SELECT, REPLICATION SLAVE, REPLICATION CLIENT,SUPER ON \*.\* TO 'canal'@'%';  
FLUSH PRIVILEGES;

(3)重启mysql容器

docker restart mysql

### 1.3 canal容器安装

下载镜像：

docker pull canal/canal-server

容器安装

docker run -p 11111:11111 --name canal -d canal/canal-server

进入容器,修改核心配置canal.properties 和instance.properties，canal.properties 是canal自身的配置，instance.properties是需要同步数据的数据库连接配置。

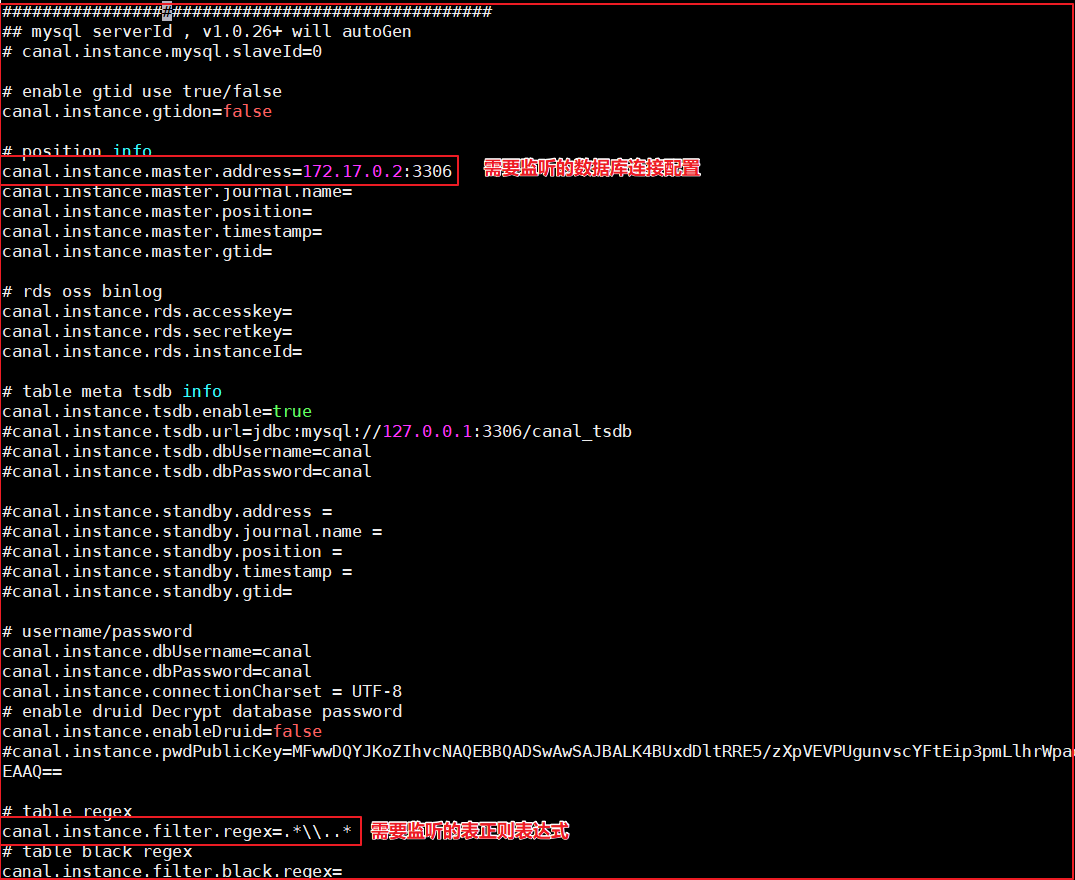
执行代码如下:

docker exec -it canal /bin/bash  
cd canal-server/conf/  
vi canal.properties  
cd example/  
vi instance.properties

修改canal.properties的id，不能和mysql的server-id重复，如下图：



修改instance.properties,配置数据库连接地址:



这里的canal.instance.filter.regex有多种配置，如下：

mysql 数据解析关注的表，Perl正则表达式.  
多个正则之间以逗号(,)分隔，转义符需要双斜杠(\\)   
常见例子：

|  |
| --- |
| 1. 所有表：.\* or .\*\\..\* 2. canal schema下所有表： canal\\..\* 3. canal下的以canal打头的表：canal\\.canal.\* 4. canal schema下的一张表：canal.test1 5. 多个规则组合使用：canal\\..\*,mysql.test1,mysql.test2 (逗号分隔) 注意：此过滤条件只针对row模式的数据有效(ps. mixed/statement因为不解析sql，所以无法准确提取tableName进行过滤) |

配置完成后，设置开机启动，并记得重启canal。

docker update --restart=always canal  
docker restart canal

### 1.4 canal微服务搭建

当用户执行 数据库的操作的时候，binlog 日志会被canal捕获到，并解析出数据。

(1)安装辅助jar包

在canal\spring-boot-starter-canal-master中有一个工程starter-canal，它主要提供了SpringBoot环境下canal的支持，我们需要先安装该工程，在starter-canal目录下执行mvn install

(2)canal微服务工程搭建

创建canal工程，并引入相关配置。

pom.xml  
 <dependencies>  
 <dependency>  
 <groupId>com.xpand</groupId>  
 <artifactId>starter-canal</artifactId>  
 <version>0.0.1-SNAPSHOT</version>  
 </dependency>  
 </dependencies>  
</project>

application.yml配置

server:  
 port: 9009  
spring:  
 application:  
 name: canal  
#canal配置  
canal:  
 client:  
 instances:  
 example:  
 host: 192.168.200.128  
 port: 11111

(3)监听创建

创建一个CanalDataEventListener类，实现对表增删改操作的监听，代码如下：

package com.example.canatest.config;  
  
import com.alibaba.otter.canal.protocol.CanalEntry;  
import com.xpand.starter.canal.annotation.\*;  
  
  
@CanalEventListener  
public class MyEventListener {  
  
 @InsertListenPoint  
 public void onEvent(CanalEntry.EventType eventType, CanalEntry.RowData rowData) {  
 rowData.getAfterColumnsList().forEach((c) -> System.err.println("By--Annotation: " + c.getName() + " :: " + c.getValue()));  
 }  
  
 @UpdateListenPoint  
 public void onEvent1(CanalEntry.RowData rowData) {  
 System.err.println("UpdateListenPoint");  
 rowData.getAfterColumnsList().forEach((c) -> System.err.println("By--Annotation: " + c.getName() + " :: " + c.getValue()));  
 }  
  
 @DeleteListenPoint  
 public void onEvent3(CanalEntry.EventType eventType) {  
 System.err.println("DeleteListenPoint");  
 }  
  
 @ListenPoint(destination = "example", schema = "canal-test", table = {"t\_user", "test\_table"}, eventType = CanalEntry.EventType.UPDATE)  
 public void onEvent4(CanalEntry.EventType eventType, CanalEntry.RowData rowData) {  
 System.err.println("DeleteListenPoint");  
 rowData.getAfterColumnsList().forEach((c) -> System.err.println("By--Annotation: " + c.getName() + " :: " + c.getValue()));  
 }  
}

(4)启动类创建  
@SpringBootApplication  
@EnableCanalClient  
public class CanaTestApplication {  
  
 public static void main(String[] args) {  
 SpringApplication.run(CanaTestApplication.class, args);  
 }  
}

(5)测试

启动canal微服务，然后修改任意数据库的表数据，canal微服务后台输出如下：

