**Kettle快速入门手册**

**Version：**V1.0

**Date：**2014/09/18

**Author:** Agile.Q

目录

[一、 Jre安装部署 2](#_Toc519244728)

[1.1 安装Jre 2](#_Toc519244729)

[1.2 配置java环境变量 3](#_Toc519244730)

[二、 Kettle安装部署 4](#_Toc519244731)

[2.1 安装Kettle 4](#_Toc519244732)

[三、 Kettle实例演示 6](#_Toc519244733)

[3.1 运行kettle 6](#_Toc519244734)

[3.2 任务信息说明 9](#_Toc519244735)

[3.3 创建转换（Transformation） 10](#_Toc519244736)

[3.3.1 新建Transformation 10](#_Toc519244737)

[3.3.2 创建数据库连接 10](#_Toc519244738)

[3.3.3 创建转换关系 13](#_Toc519244739)

[3.3.4 配置表输入信息 14](#_Toc519244740)

[3.3.5 配置表插入/更新信息 16](#_Toc519244741)

[3.3.6 运行转换 18](#_Toc519244742)

[3.4 创建定时任务Job 20](#_Toc519244743)

[3.4.1 kettle自带的‘Star’定时功能 21](#_Toc519244744)

[3.4.2 Linux的crontab脚本定时 27](#_Toc519244745)

# Jre安装部署

## 安装Jre

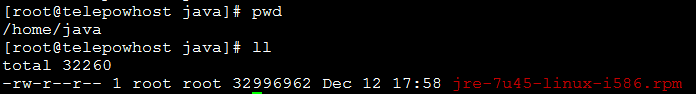
在网站<http://www.oracle.com/technetwork/java/javase/downloads/java-archive-downloads-javase7-521261.html>

下载适合的jre

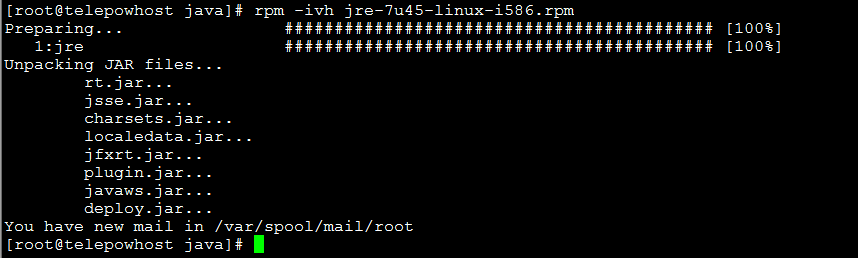
**//\*\*有两种可选版本：rpm安装包，.gz自解压安装包**

这里选择rpm格式安装，将安装包（jre-7u45-linux-i586.rpm）上传到root用户目录/home/java下。

**//\*\*在root下安装可为全用户提供java环境支持，也可根据实际情况为某一用户安装）**



终端输入：rpm -ivh jre-7u45-linux-i586.rpm 开始安装



此时java已成功安装！

**//\*\*rpm命令安装的java默认安装路径是/usr/java/ jre1.7.0\_45**

## 配置java环境变量

修改文件/etc/profile：

进入profile,敲Ctrl +F到文件末尾，写入以下内容：

JAVA\_HOME=/usr/java/jre1.7.0\_45

PATH=$JAVA\_HOME/bin:$PATH

CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar

export JAVA\_HOME

export PATH

export CLASSPATH

▲其中JAVA\_HOME是java的安装路径。

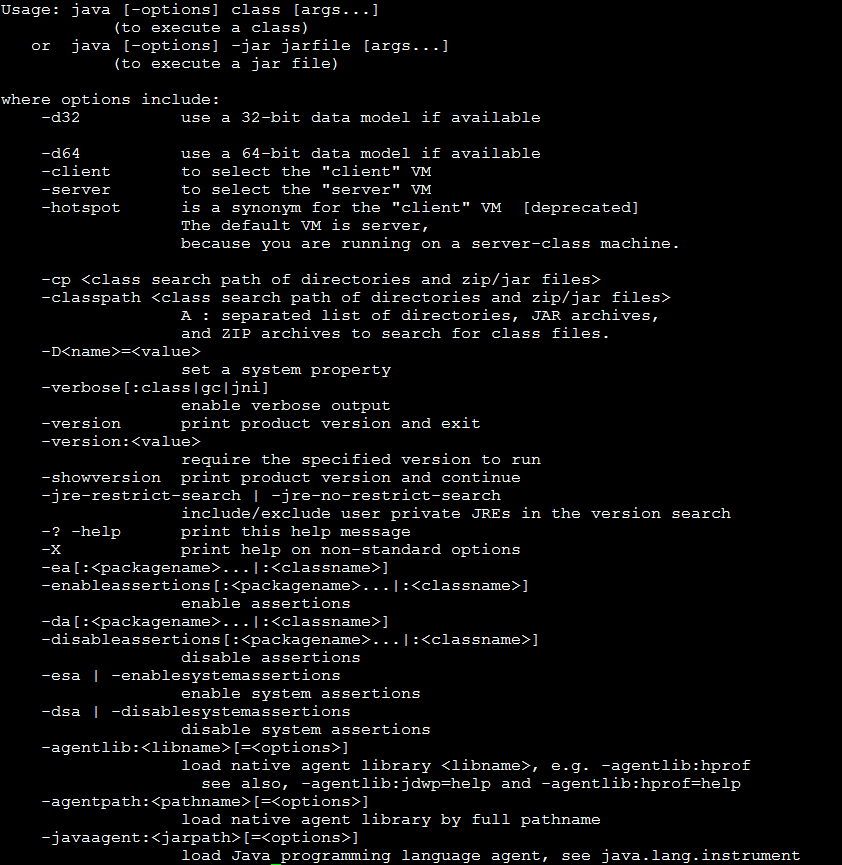
**//\*\*此处要注意环境变量的java路径是安装路径，不是java的执行路径；**

**Java路径查询：whereis java**

**Java执行路径查询：which java**

保存退出!

在终端键入java，如果显示帮助信息，则表示jre环境配置完成：

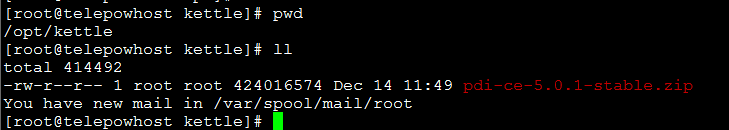


# Kettle安装部署

## 安装Kettle

在网站 <https://github.com/pentaho/pentaho-kettle/releases>下载所需的kettle版本

进root用户将安装包（pdi-ce-5.0.1-stable-R.zip）上传到/opt/kettle目录下



因为要将kettle安装在用户telepow下（也可根据需求为所需用户安装），则需改变kettle文件的属主及属组：

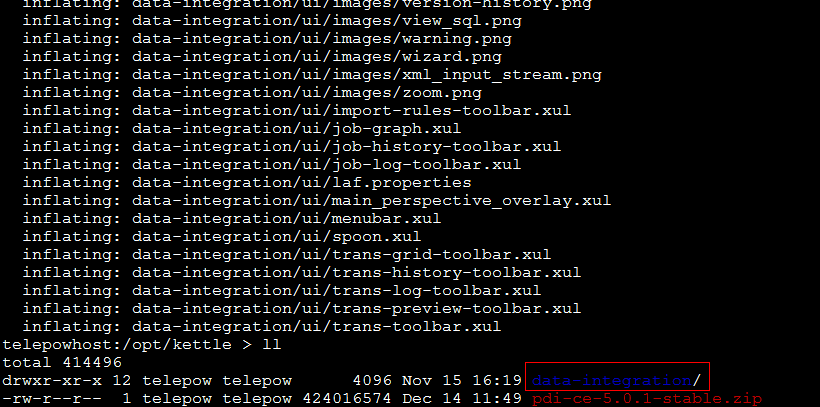
chown -R telepow:telepow kettle

切换到telepow用户下，开始安装，因为kettle安装包格式为.zip，所以运行：

unzip pdi-ce-5.0.1-stable.zip

安装到当前目录。

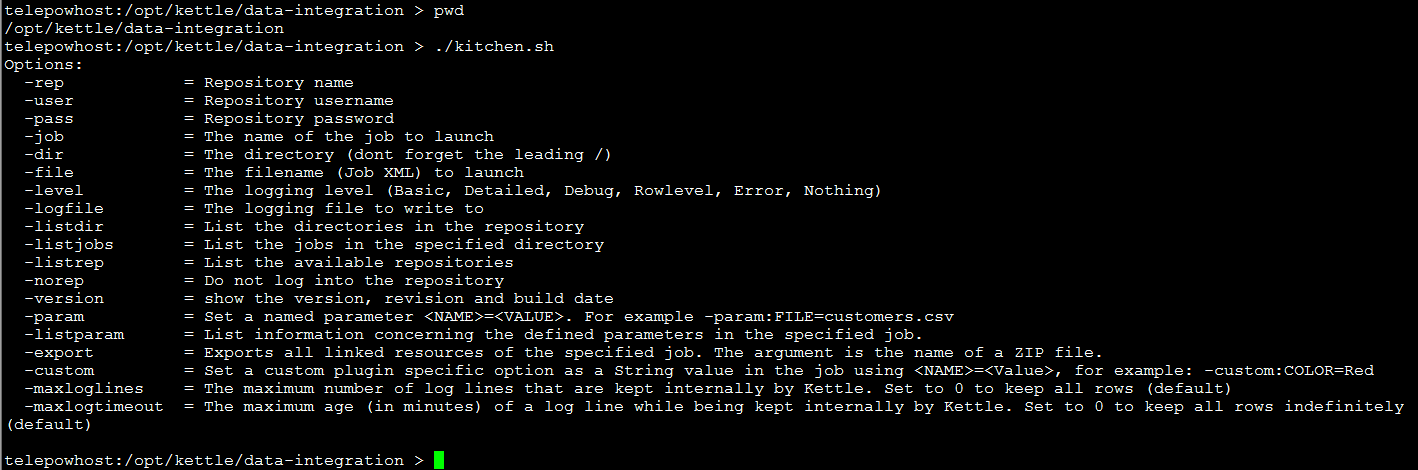
安装完成后，可以看到生成一个新的文件夹data-integration，则/opt/kettle/data-integration既是kettle的安装和执行目录



进入该目录下，为所有.sh文件赋予执行权限：

chmod +x \*.sh

然后在终端键入./kitchen.sh，如果出现帮助信息，说明kettle安装部署完成



# Kettle实例演示

Java和kettle安装部署完成之后，就可以使用kettle工具了！

下面将以实现定时从一台PC上的数据库中抽取数据到另一台PC数据库为例，讲解kettle的使用。

## 运行kettle

在kettle安装目录下运行：

./spoon.sh

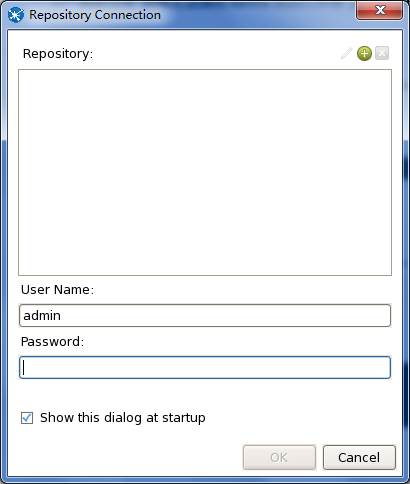
即可弹出kettle操作界面。

**//\*\*此处注意，如果无法弹出kettle的运行界面，提示缺少图像插件的话，需要为linux的系统安装图形运行工具；**

**将图形工具安装包（libXp-1.0.0-8.1.el5.i386.rpm）上传到root下临时目录，安装：**

**rpm –ivh libXp-1.0.0-8.1.el5.i386.rpm**

接着就是登陆界面



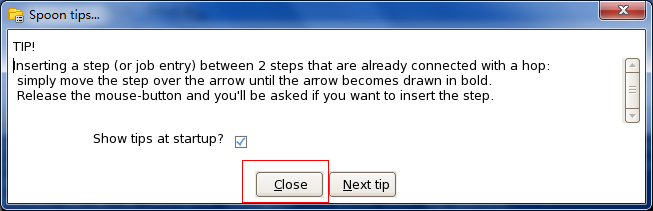
Kettle为用户提供两种存储用户编写的kettle脚本的方式：

一是kettle工具自身的资源库（相当于一个小型专属数据库）；

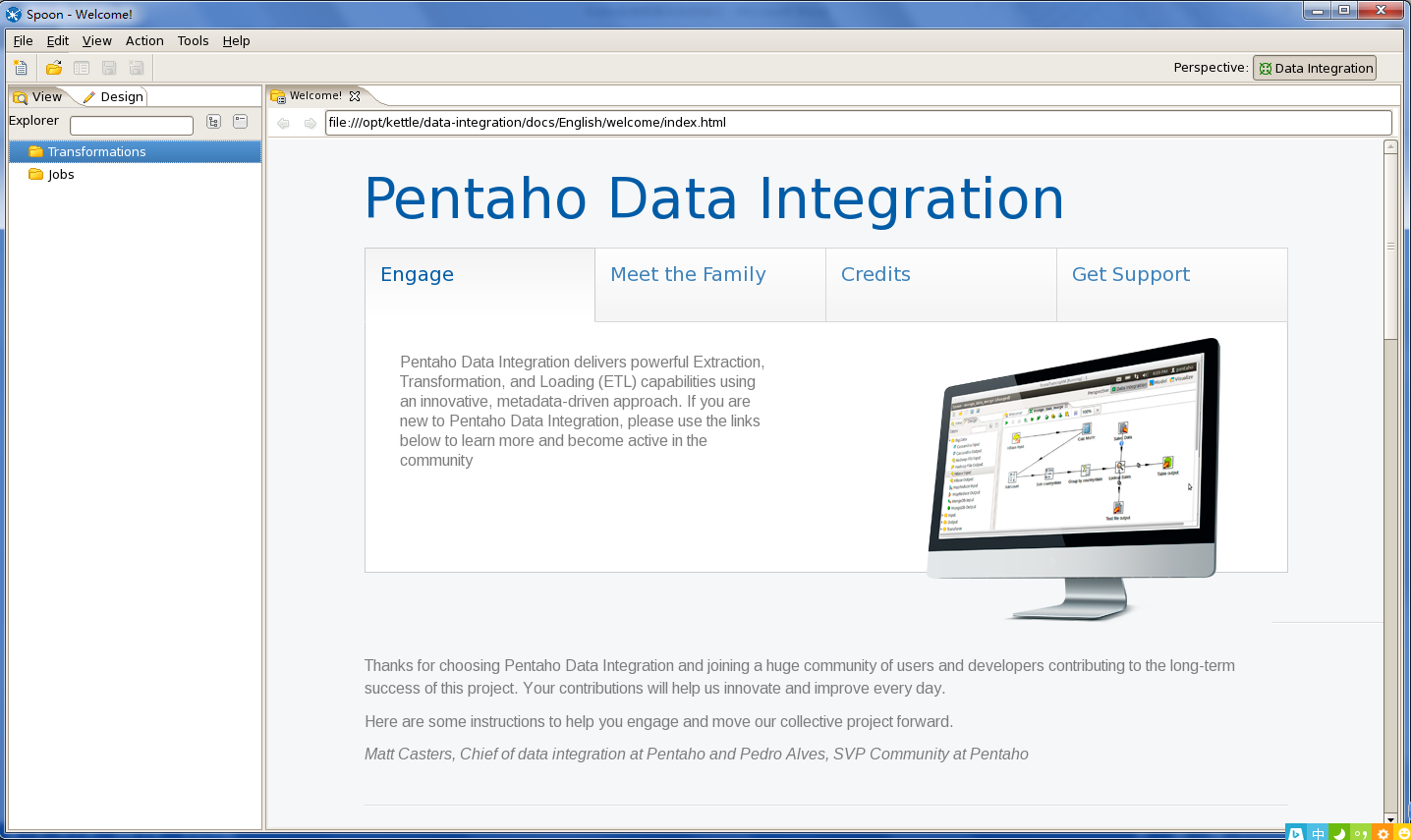
二是不使用资源库，将脚本信息直接存储在自建目录中；

此处我们不使用资源库登陆，点击‘Cancel’

弹出的帮助信息点击‘Close’



紧接着就可以看到kettle的使用界面



## 任务信息说明

此演示工作涉及到的信息是：

**PC01**

IP：192.168.193.130

端口号：8000

数据库：sybase

用户数据库：dynamicdb

数据库用户：telepow

数据库用户密码：123456

任务表：SWITCHPOWDEV

**PC02**

IP：134.81.5.86

端口号：8000

数据库：sybase

用户数据库：dynamicdb

数据库用户：telepow

数据库用户密码：uo23hoS982\_Ow\_db

任务表：V\_SWITCHPOWDEV

需要做的就是将PC02的视图表V\_SWITCHPOWDEV中数据定时抽取到

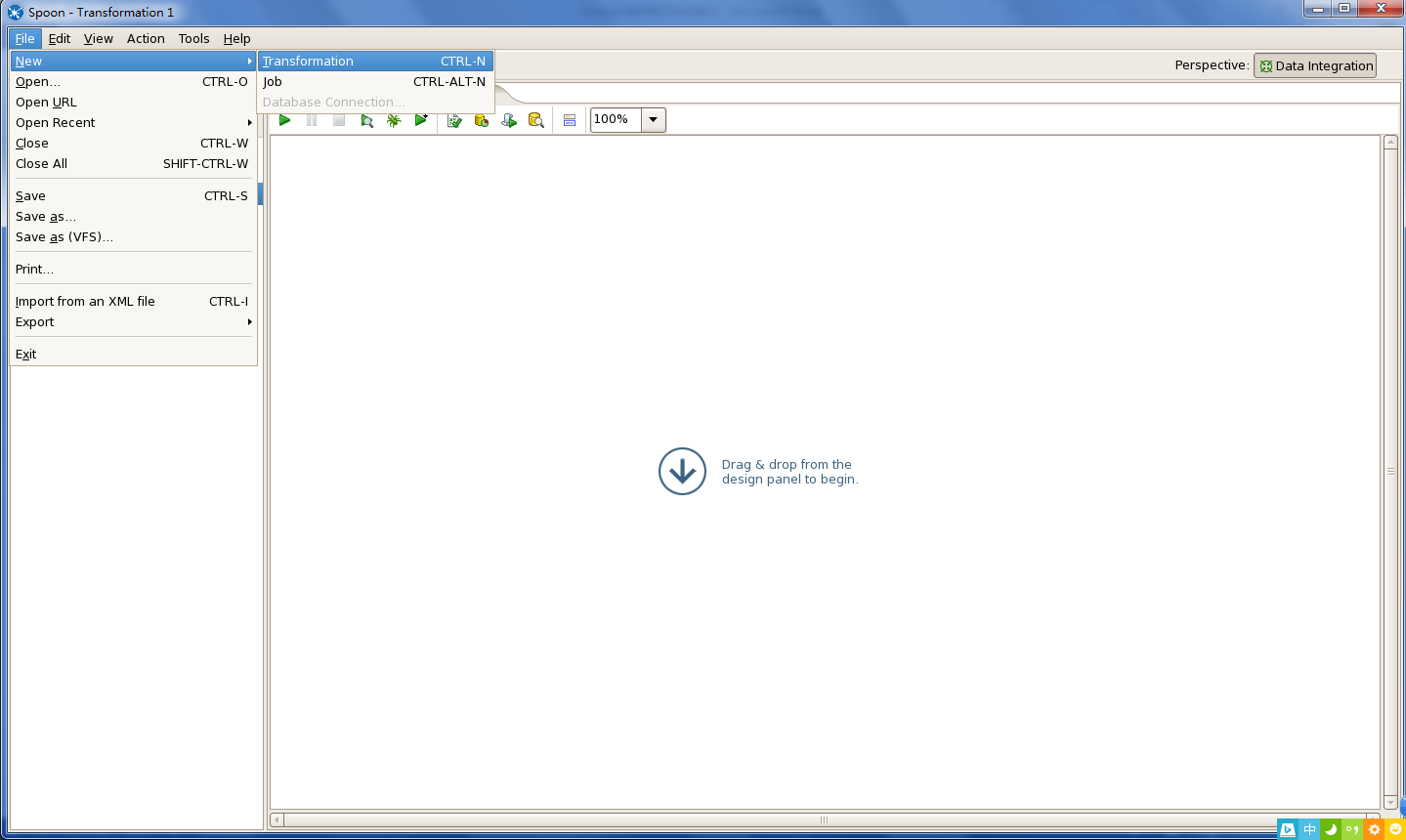
PC01的表SWITCHPOWDEV之中

## 创建转换（Transformation）

### 新建Transformation

打开‘File’—‘New’---‘Transformation’

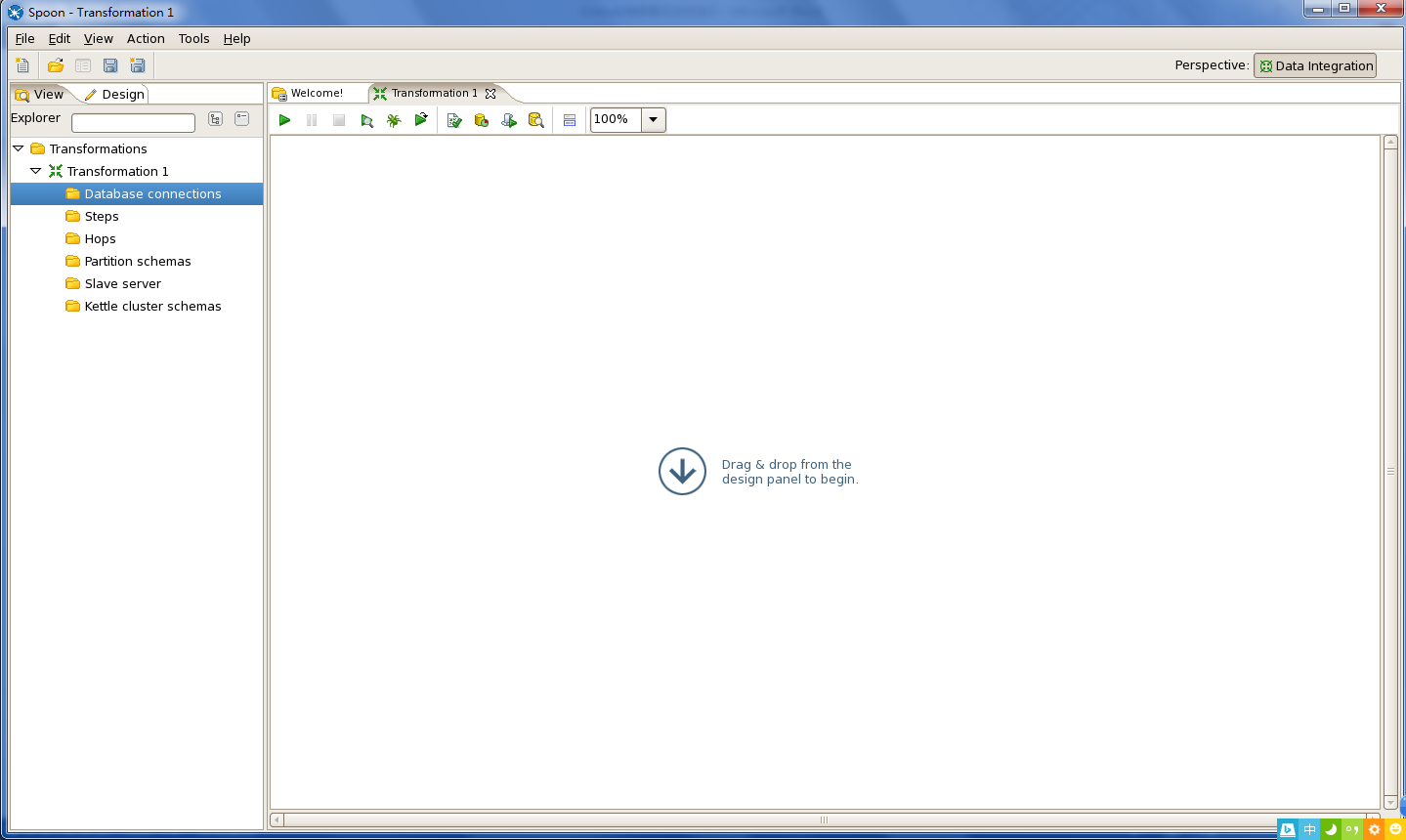
创建一个转换Transformation01

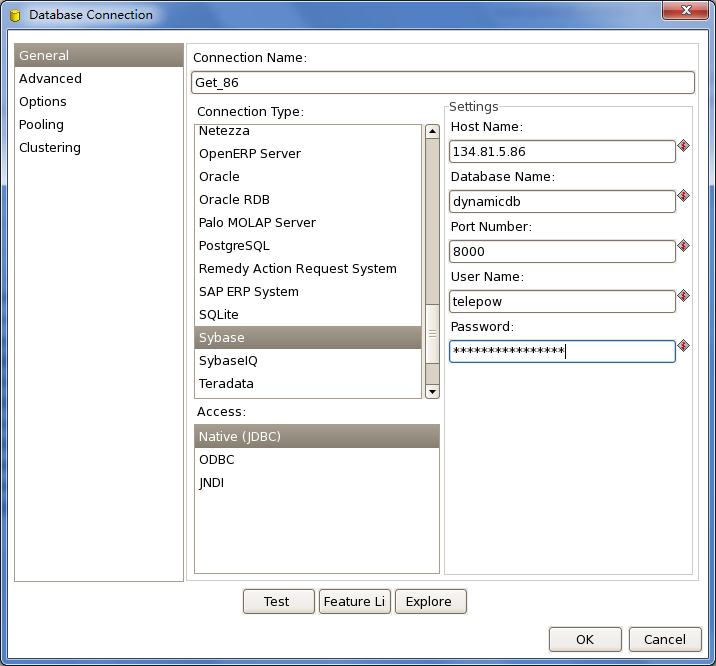


### 创建数据库连接

在转换视图窗口双击‘Database Connections’

为此转换建立一个数据库连接





**Connection Name：**建立的数据库连接名称--Get\_86

**Connection Type：**连接的数据库类型—Sybase

**Access：**连接方式—JDBC

**Host Name：**连接主机的IP地址—127.0.0.1

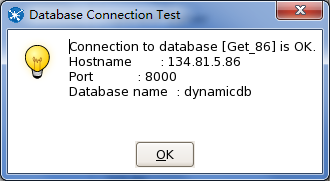
**Database Name：**连接的用户数据库—dynamicdb

**Port Number：**数据库端口号—8000

**User Nmae：**数据库用户名—\*\*\*\*\*\*

**Password：**数据库用户密码--\*\*\*\*\*\*

配置完成之后点击左下角‘Test’，测试数据库连接状态

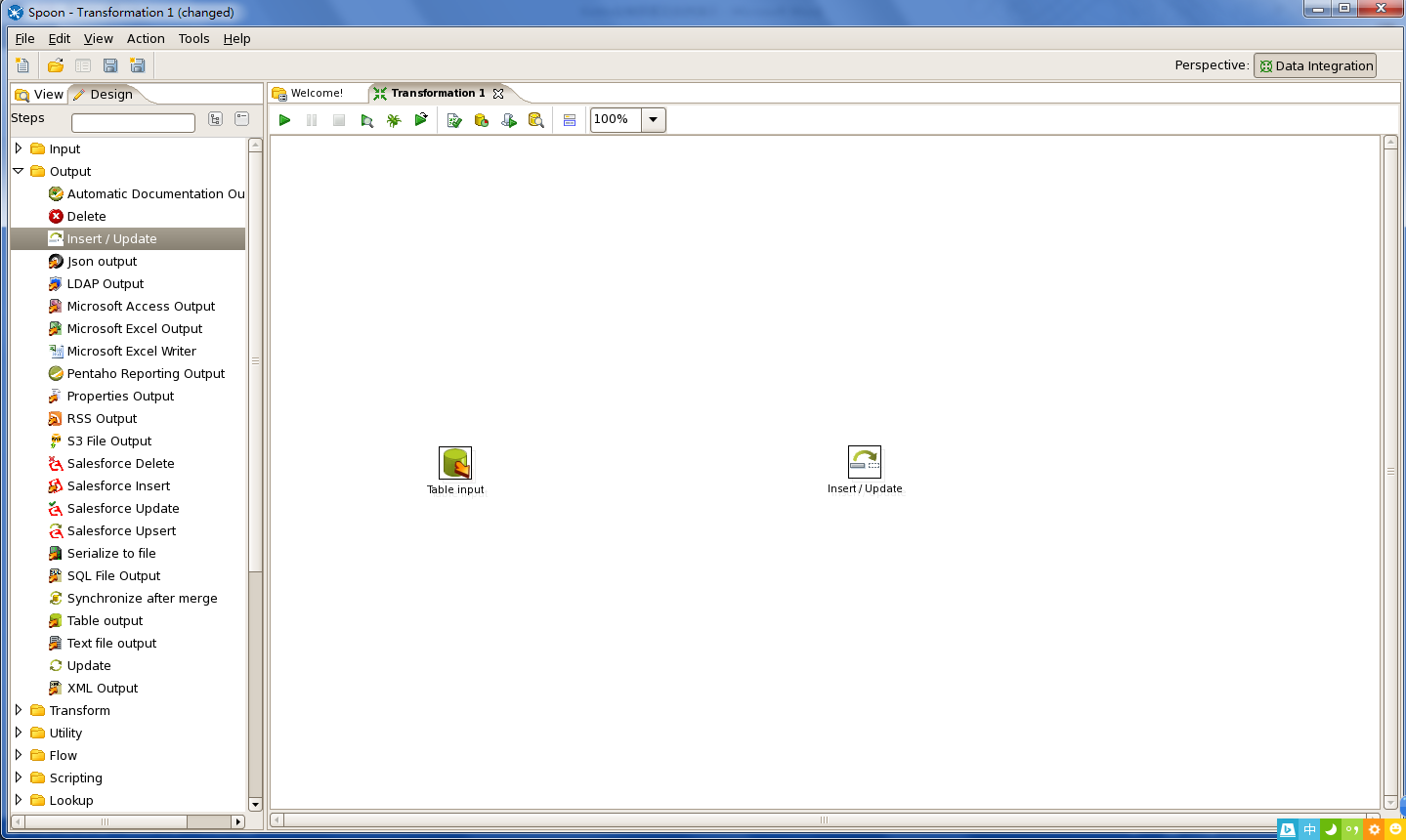


出现截图信息，说明配置正确，数据库连接正常！

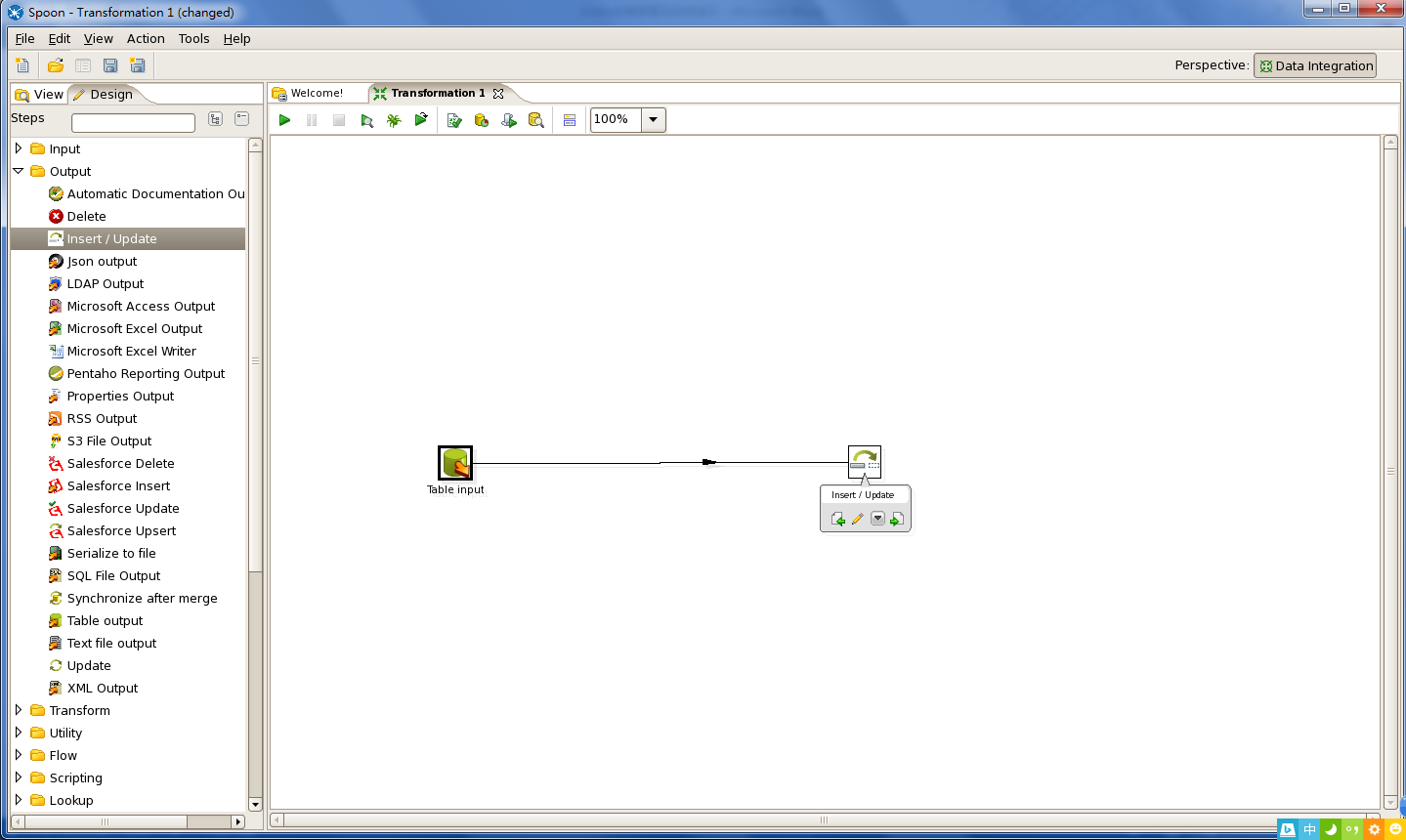
点解OK 保存退出!

### 创建转换关系

点击转换窗口的‘Design’，将‘Input’中的‘Table input’表输入图标拖入右侧窗口，然后再将‘Output’中的‘Insert/update’插入/更新图标拖入右侧窗口



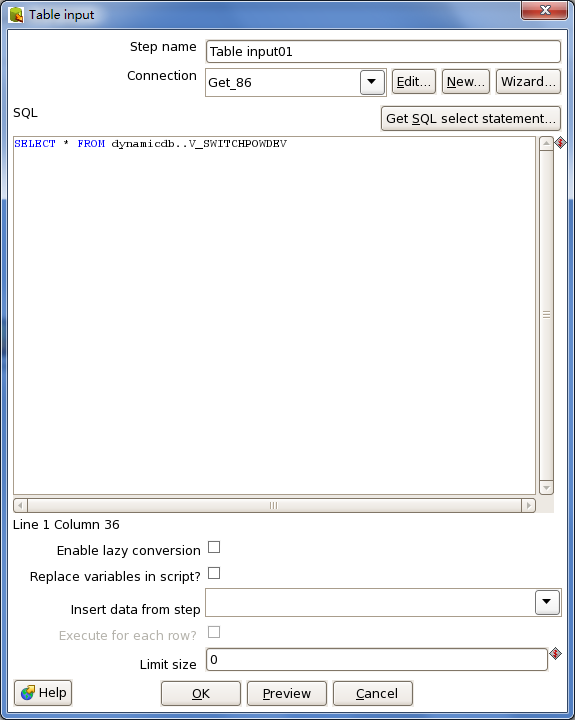
按住‘Shift’键鼠标左击右侧窗口中的‘Table Input’图标，点住不放托至右侧‘Insert/Update’图标上，发现有一条带箭头的连接线将两个图标连接起来。



这样两个图标之间就建立了数据抽取与数据存储的关系

### 配置表输入信息

双击‘Table Input’图标，弹出表输入配置信息，为抽取PC02中的数据配置相关信息。



**Step name：**步骤名称--Table Input01

**Connection：**所属数据库连接—Get\_86，即刚刚建立的连接PC02的数据库连接

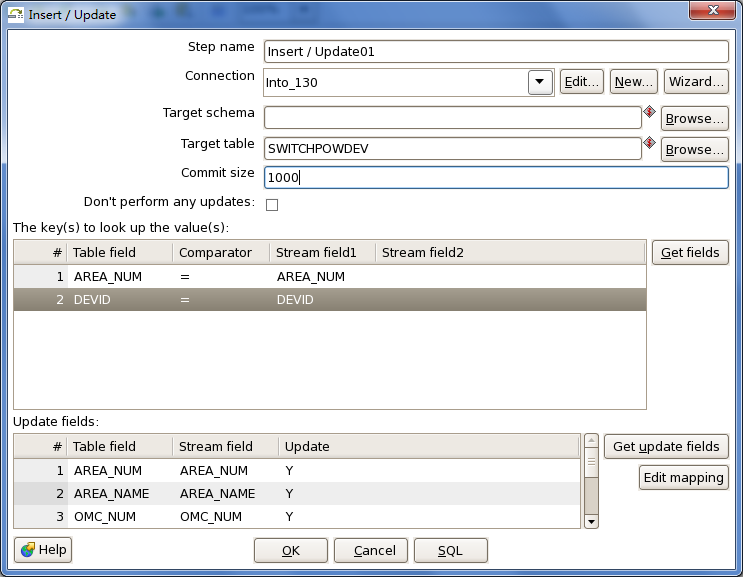
**SQL：**需要对PC02中任务表做的操作—

SELECT \* FROM dynamicdb..V\_SWITCHPOWDEV

点击OK保存退出！

### 配置表插入/更新信息

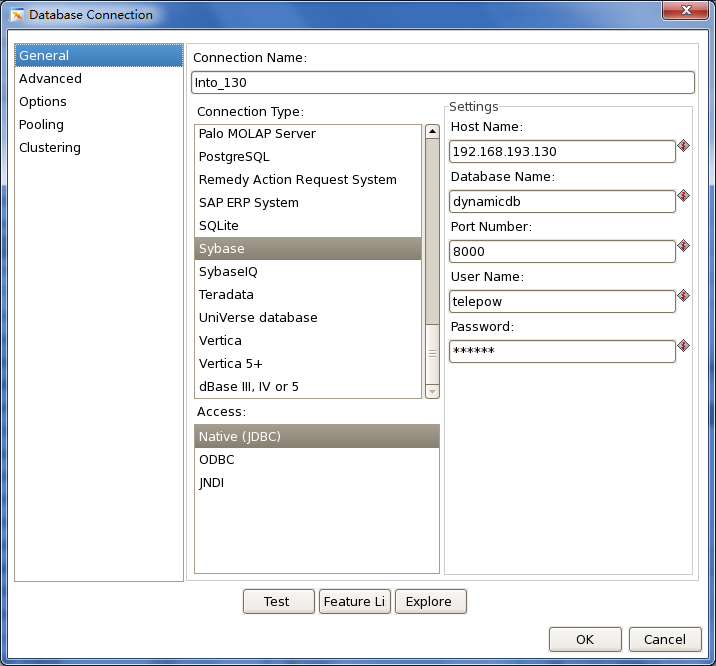
同样双击‘Insert/Update’图标，弹出配置界面，为向PC01中插入数据配置相关信息



**Step name：**步骤名称--insert/Update01

**Connection：**所属数据库连接—Into\_130，此处注意，因为要向PC01中插入数

据，所以点击‘Connection’右侧的‘New’新建按钮，创建新的数据库连接，此连接指向PC01



**Target table：**目标表—SWITCHPOWDEV，即是要把数据插入PC01中的这张表

**Conmmit size：**数据量限制—设置的数值最好大于每次需插入表中总的记录数

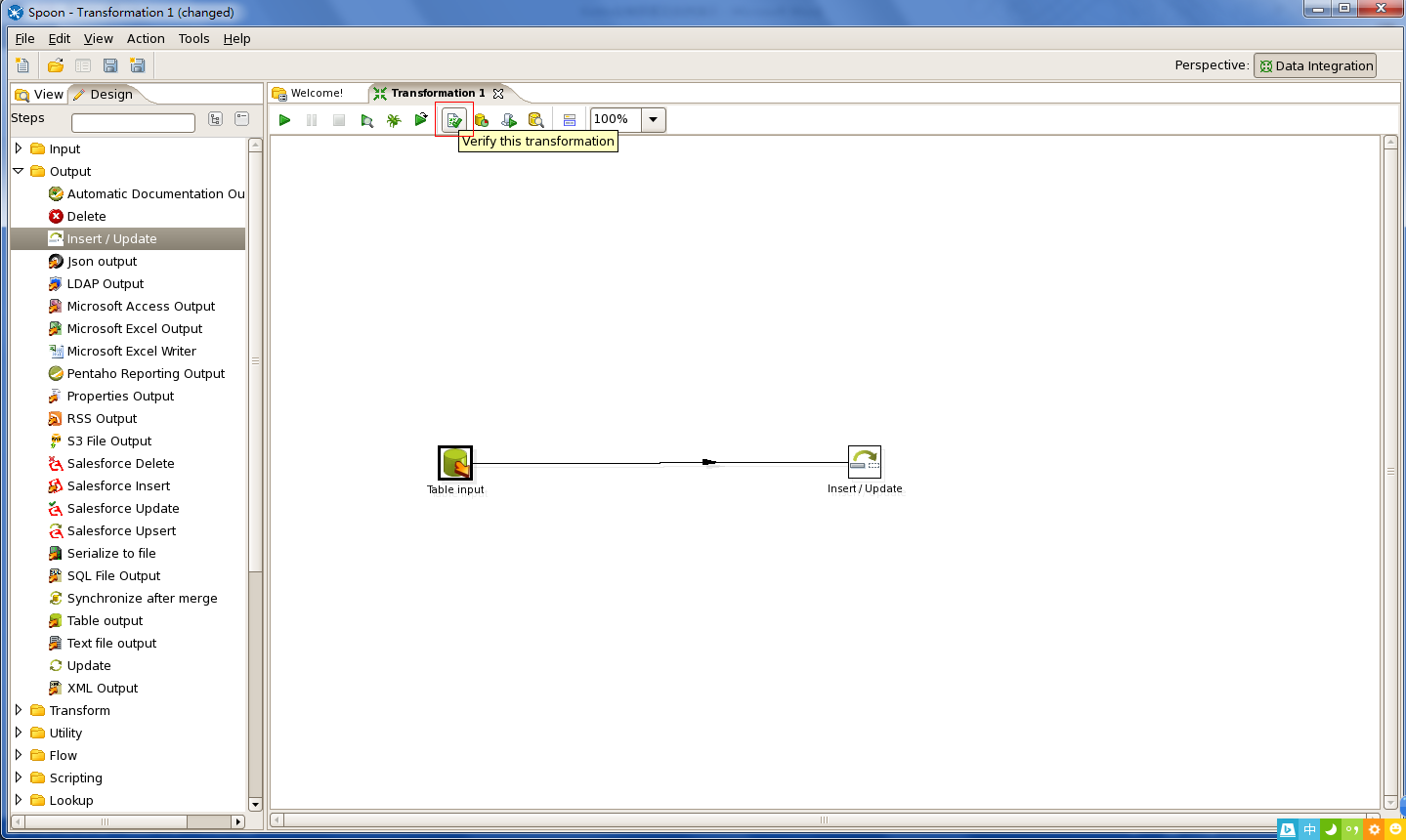
**The Key(s) to look up the value(s)：**查询关键字—抽取PC02中V\_SWITCHPOWDEV的数据时筛选所需关键字，点击右侧‘Get Files’获取所有字段，将多余字段删除

**Update files：**更新的字段—对PC01中SWITCHPOWDEV表需要更新的字段，点击右侧‘Get update files’获取所有字段

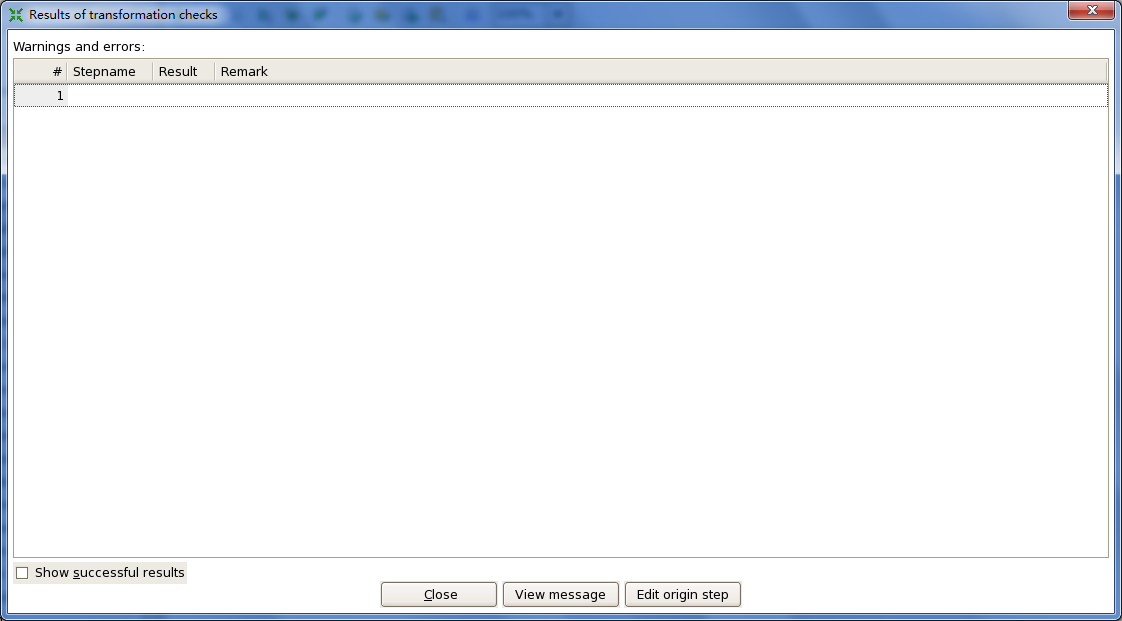
点击OK保存退出！

### 运行转换

转换信息配置完成之后，先对编写好的转换进行调试，验证有无Bug，点击菜单栏图标，

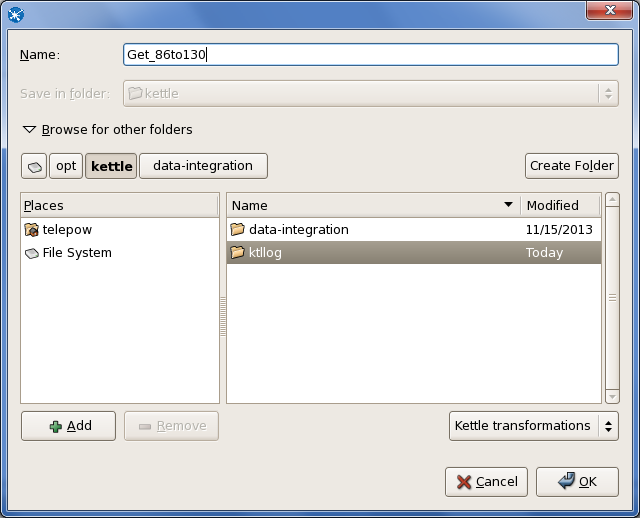


弹出调试结果



可以看出没有错误，最后运行此连接，实现数据的抽取，点击菜单栏图标

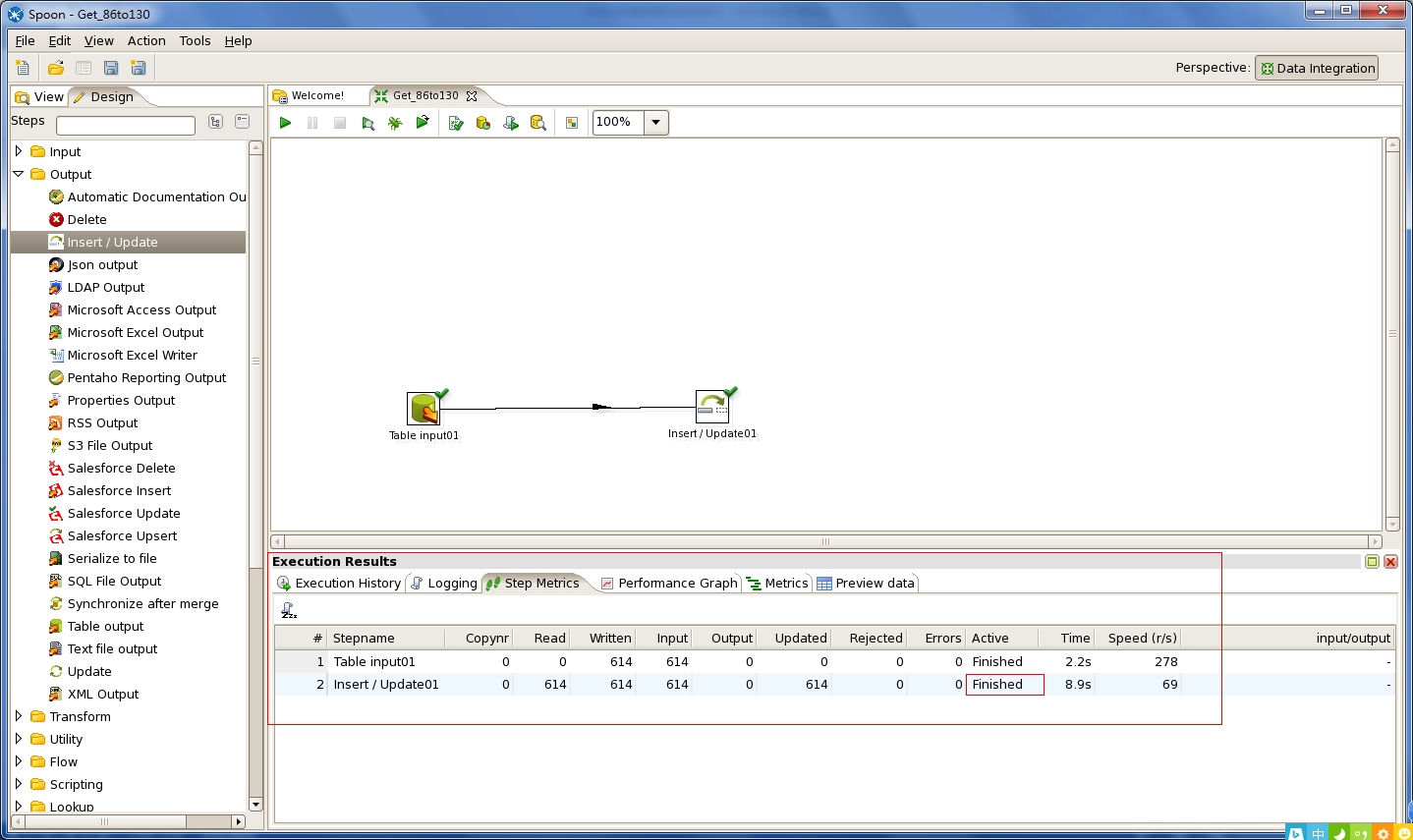
弹出窗口，点击‘Lanuch’----‘Yes’进行保存



**Name：**保存的转换名称—Get\_86to130

**Browser for other folders：**存储路径—/opt/kettle/ktllog，将转换保存在自己建立的路径中，方便后续查找调用

点击OK开始运行，下方会打印出动态日志，完成之后可以看到日志中的‘Finished’字样



至此一个转换已经建立完毕！

## 创建定时任务Job

转换完成之后，开始建立定时任务，已实现定时抽取数据的需求，定时任务有两种途径可以实现：

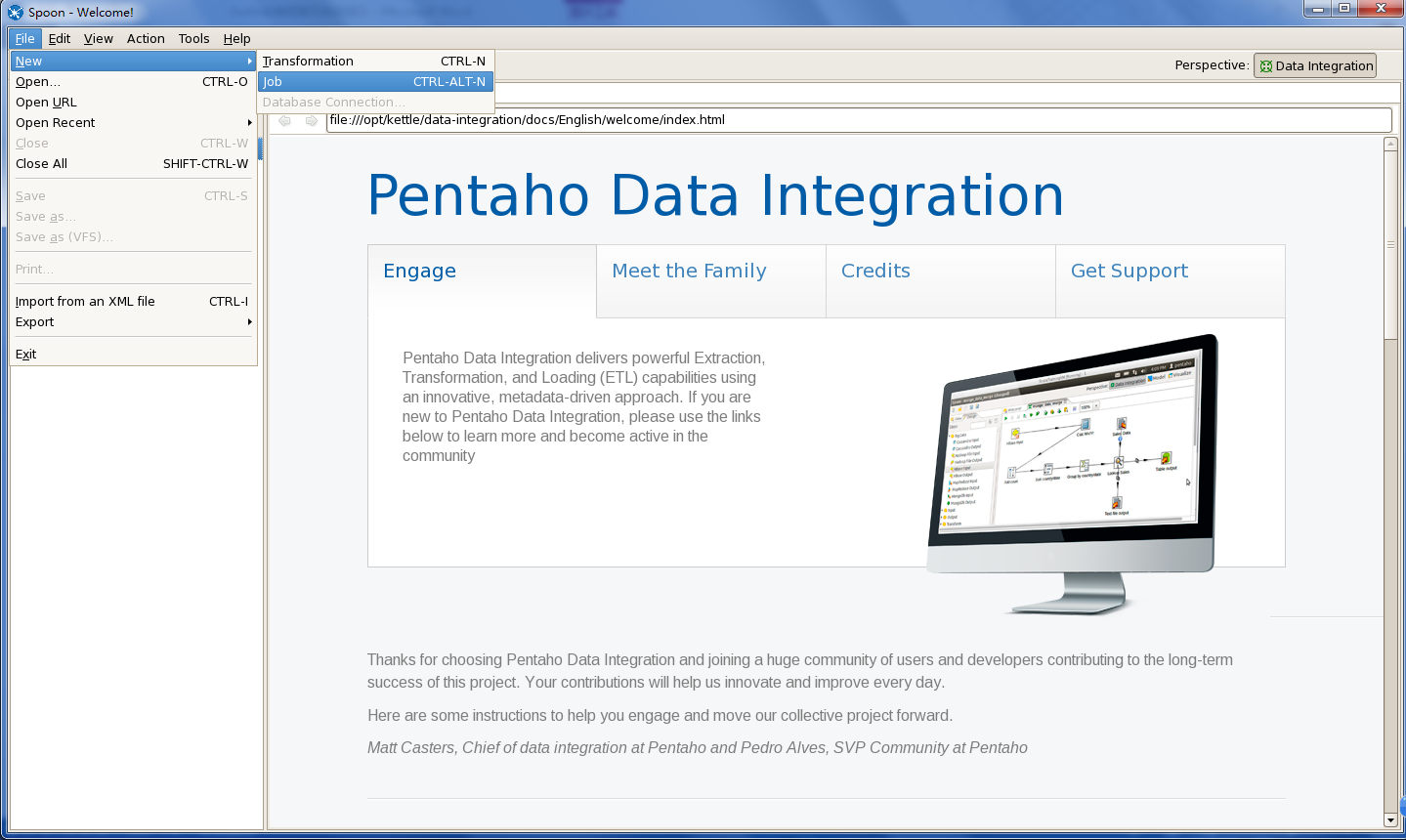
一是利用kettle自身的定时工具‘Start’；

二是利用linux中的crontab功能；

### kettle自带的‘Star’定时功能

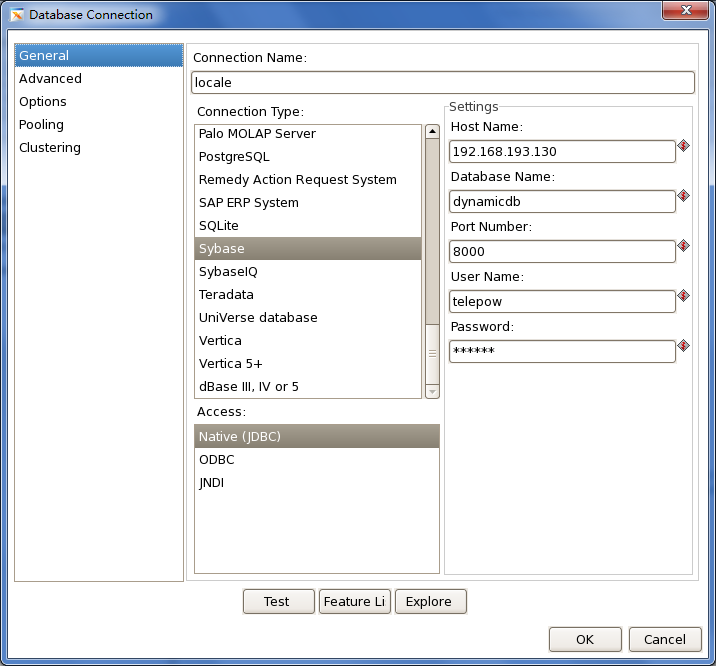
#### 新建Job

打开‘File’—‘New’—‘Job’，新建一个任务Job



#### 创建数据库连接

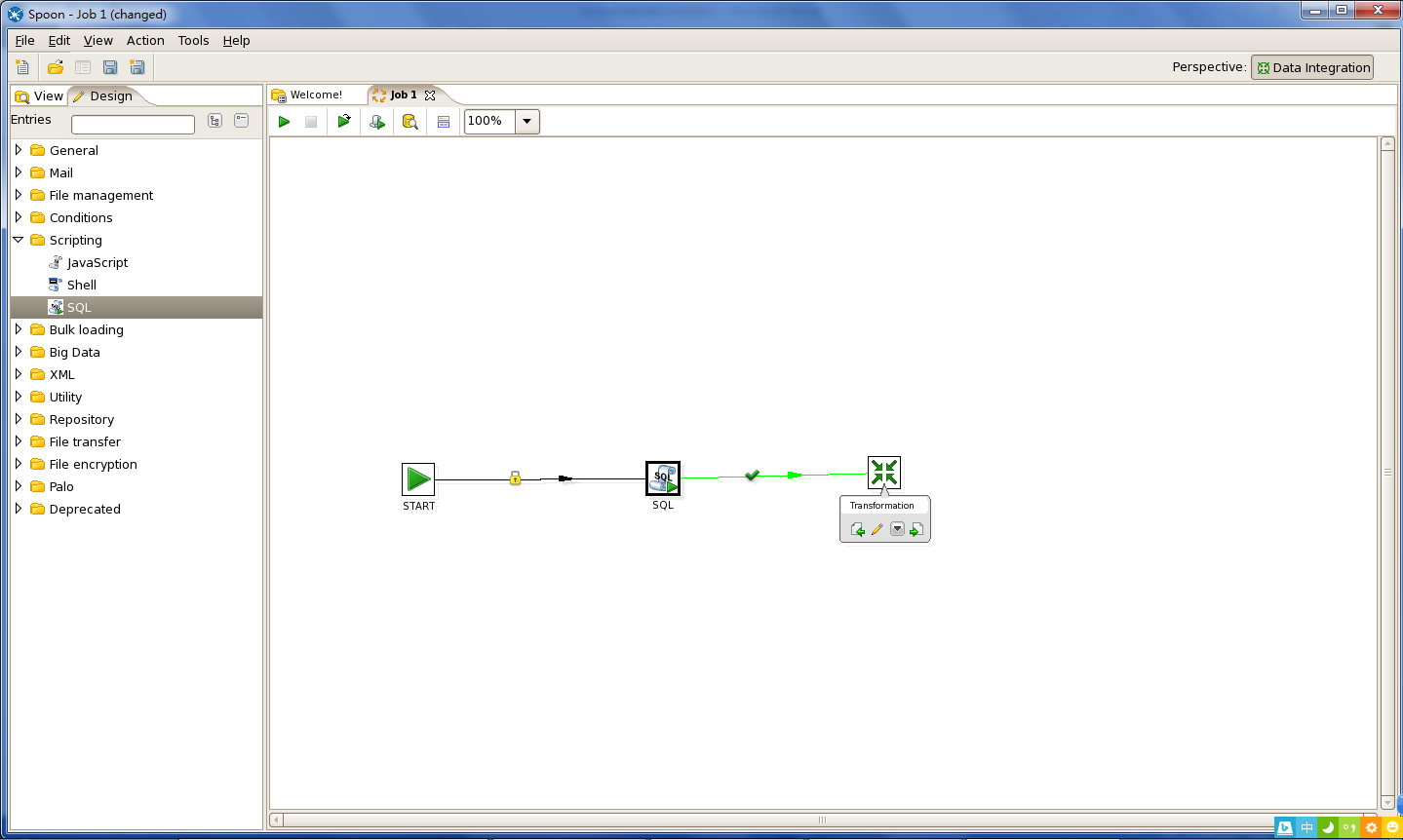
为任务Job创建一个数据库连接即指向PC01的连接，双击Job视图窗口的‘Database Connection’弹出配置界面



配置好之后，保存退出！

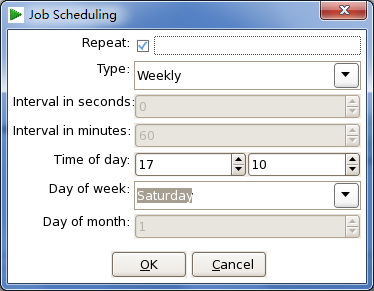
#### 创建任务关系

打开任务设计窗口‘General’将‘START’和’Transformation’图标拖入右侧窗口，再将‘Scripting’中‘SQL’拖入右侧窗口，并将图标连接起来。



#### 配置START信息

双击图标‘START’，弹出配置窗口



**Repeat：**重复—将此项勾选，表示按下面配置的周期重复

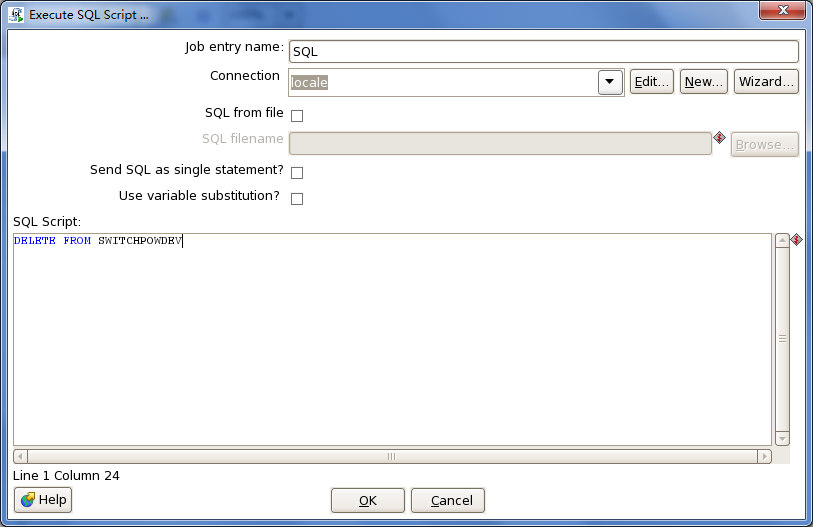
**Type：**周期类型—Weekly,按周重复

**Time of day：**时分设置—当天的17：10

**Day of week：**星期选择—Saturday，定在每周六执行

#### 编写SQL语句

因为上述编写过的转换只能实现将PC02的数据抽取到PC01中（即将PC01与PC02中共有的数据更新和PC01中没有而PC02中有的数据插入PC01中），而不能实现保持PC01与PC02数据完全相同（即不能将PC02中已经没有的数据从PC01中删除），所以在执行转换前要编写个SQL语句，在插入PC02数据之前将PC01中的数据先清空，双击‘SQL’图标，弹出配置界面



**Job entry name：**SQL脚本的名称—SQL

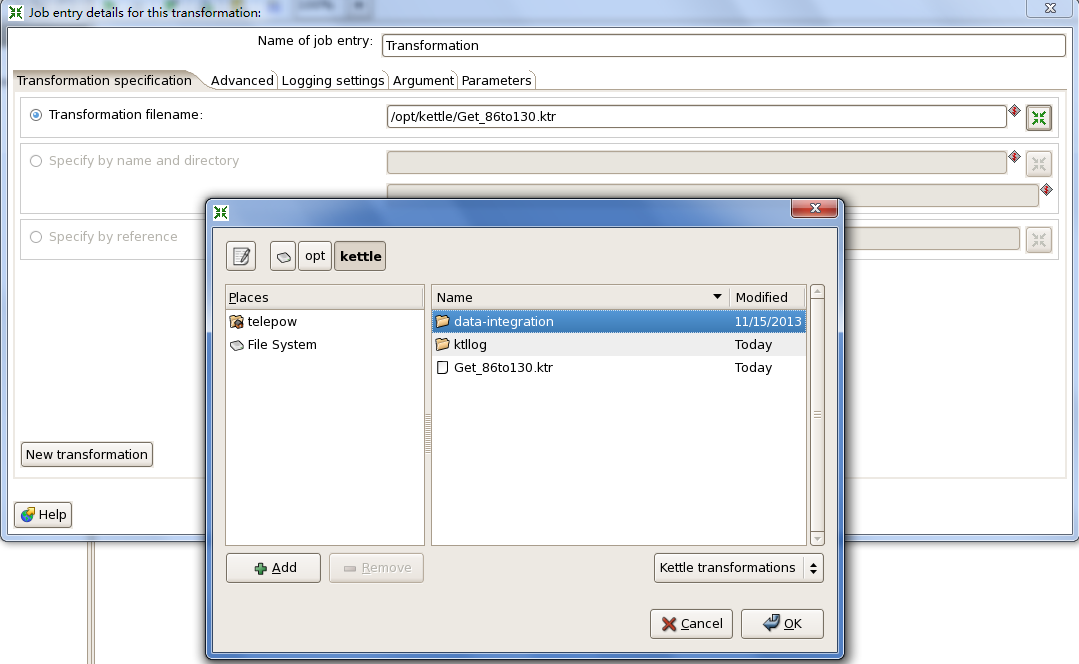
**Connection：**所属数据库连接—locale，此处是要对PC01中的SWITCHPOWDEV表进行操作，所以选择locale连接

**SQL：**需执行的SQL语句-- DELETE FROM SWITCHPOWDEV

保存退出！

#### 导入转换

双击‘Transformation’图标，将已经编写好的转换导入进来



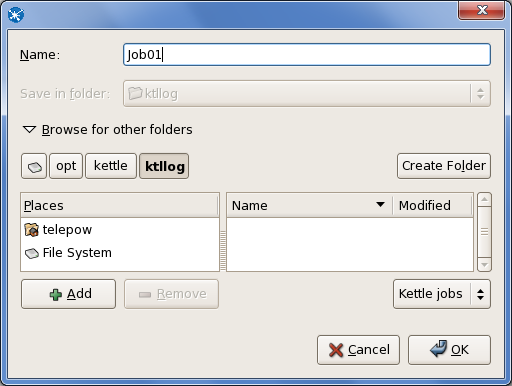
**Name of job entry：**Job步骤名称—Transformation

**Transformation filename：**已编写转换的路径-- /opt/kettle/Get\_86to130.ktr，通过点击右侧的图标，选择转换

保存退出！

#### 运行任务

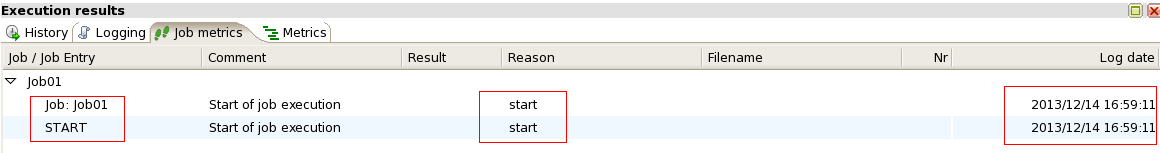
点击菜单栏执行Job，弹出窗口，点击‘Lanuch’



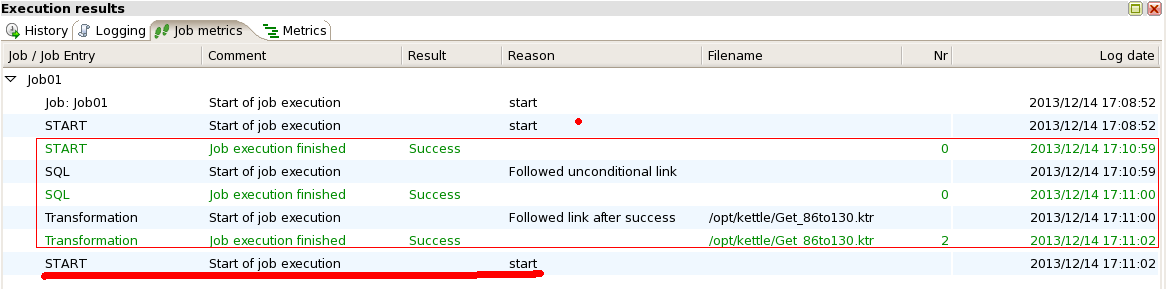
**Name：**保存Job的名称—Job01

**Browse for other folders：**保存Job的路径

Job开始运行，下方会打印动态日志，开始时会停在‘start’状态，因为此时还未到设置的周期时间



等时间来到周六的17：10，任务继续往下走



从日志可以看出本周的任务执行完毕，最后的‘start’状态表示进入下一个周期等待中…….

至此kettle自带的定时任务设置演示完成！

下面讲解通过linux的crontab实现定时的功能

### Linux的crontab脚本定时

因为已经将需要的kettle脚本编写完毕，现在只需配置crontab所需信息，直接调用即可

#### 创建sh脚本

以telepow用户进入目录/opt/kettle/ktlsh，编写Get\_86.sh脚本，写入以下内容：

#!/bin/sh

ROOT\_TOPDIR=/opt/kettle

export ROOT\_TOPDIR

date >> $ROOT\_TOPDIR/ktllog/log01

echo "-------------------------------------------------------" >> $ROOT\_TOPDIR/ktllog/log01

JAVA\_HOME=/usr/java/jre1.7.0\_45

PATH=$JAVA\_HOME/bin:$PATH

CLASSPATH=.:$JAVA\_HOME/lib/dt.jar:$JAVA\_HOME/lib/tools.jar

export JAVA\_HOME

export PATH

export CLASSPATH

$ROOT\_TOPDIR/data-integration/kitchen.sh -file=$ROOT\_TOPDIR/ktllog/Job01.kjb >> $ROOT\_TOPDIR/ktllog/log01

echo "-------------------------------------------------------" >> $ROOT\_TOPDIR/ktllog/log01

**$ROOT\_TOPDIR/ktllog/log01**为打印日志的存储位置

**kitchen.sh -file=$ROOT\_TOPDIR/ktllog/Job01.kjb**为调用kettle自身的**kitchen.sh**命令执行任务Job01.kjb，此处的任务‘START’中不设置任何周期

**JAVA\_HOME=/usr/java/jre1.7.0\_45**此处需要说明的是，虽然之前已经配置好了java环境，但是crontab命令不属于任何用户，调用kettle自身的kitchen.sh命令时，它依然找不到java环境，所以在调用时必须指定java环境，否则sh脚本无法运行

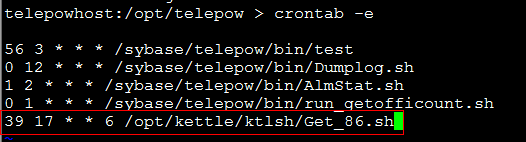
保存退出！

然后为Get\_86.sh脚本赋予执行权限

Chmod +x Get\_86.sh

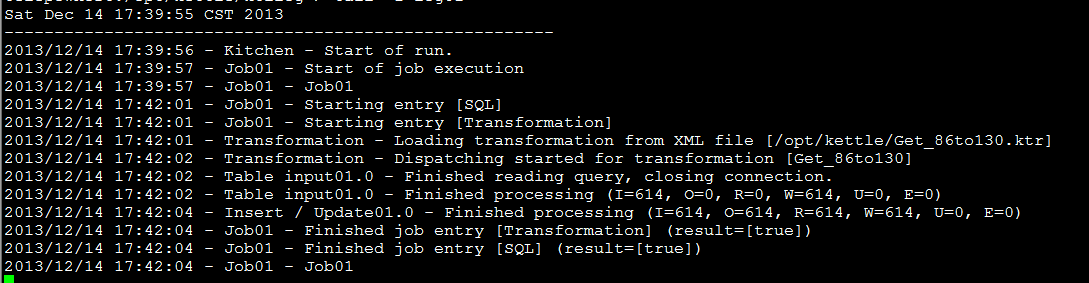
#### 设置crontab

在终端键入crontab –e，编写crontab



定在每周六的17：39执行脚本Get\_86.sh

等时间来到周六的17：39，crontab 脚本运行,通过日志文件可以看到运行的过程



至此linux中crontab命令实现定时任务演示已完成！