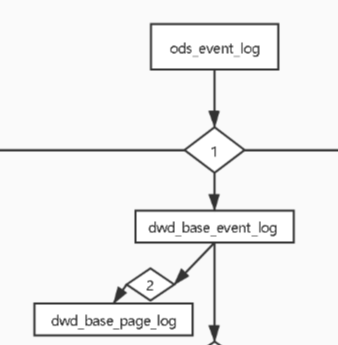
1. **根据我们的spark定时任务进行job的编写**

基本一个spark任务编写一个job

（例外如下

1.我们将25，26，27任务合并成一个job，合并原因是这三个spark任务的实现方法极其相似，我们就在spark任务代码中将他们写到一个scala对象中。该job作用为向 ads\_usr\_dau\_cube、ads\_usr\_wau\_cube、ads\_usr\_mau\_cube 表中导入数据，这三个表的具体执行时间在实现方法中做了判断

2.我们将44，45，46任务合并成一个job，合并原因同上。该job只需要调用这个scala对象就行了。该job作用为向 base\_user\_active\_day、 base\_user\_active\_week、base\_user\_active\_month 表中导入数据，这三个表的具体执行时间在实现方法中做了判断）



以任务1、2为例

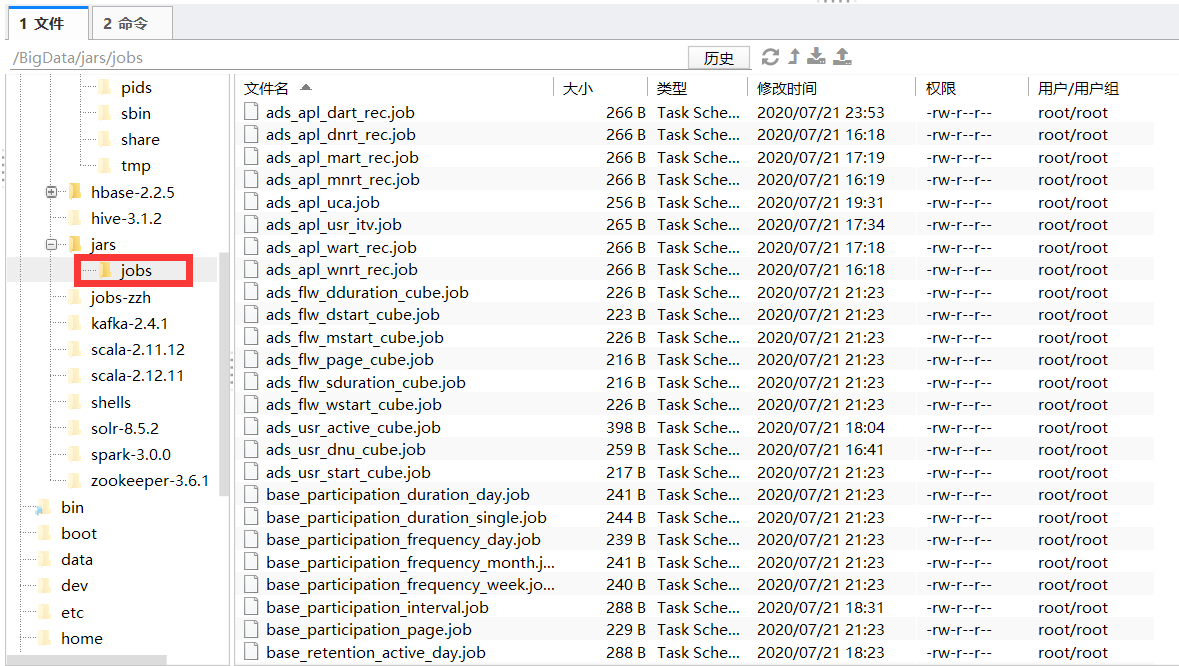
任务1：dwd\_base\_event\_log.job

1. type=command
2. command=spark-submit --master yarn --deploy-mode cluster --class cn.edu.neu.titan.titanSpark.analysis.base.function.JSONParseFunction /BigData/jars/titanSpark-1.0-SNAPSHOT.jar

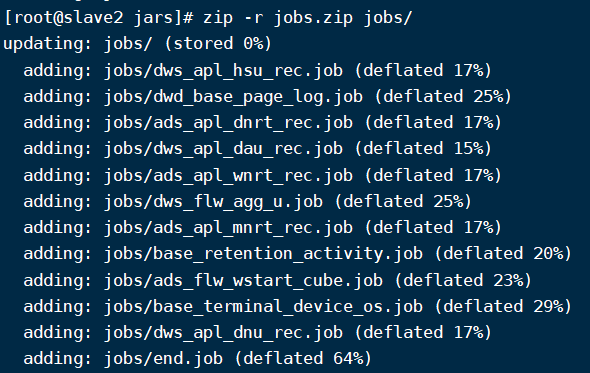
任务2：dwd\_base\_page\_log.job 依赖于dwd\_base\_event\_log.job

1. type=command
2. dependencies=dwd\_base\_event\_log
3. command=spark-submit --master yarn --deploy-mode cluster --class cn.edu.neu.titan.titanSpark.analysis.base.function.JSONParseFunction /BigData/jars/titanSpark-1.0-SNAPSHOT.jar

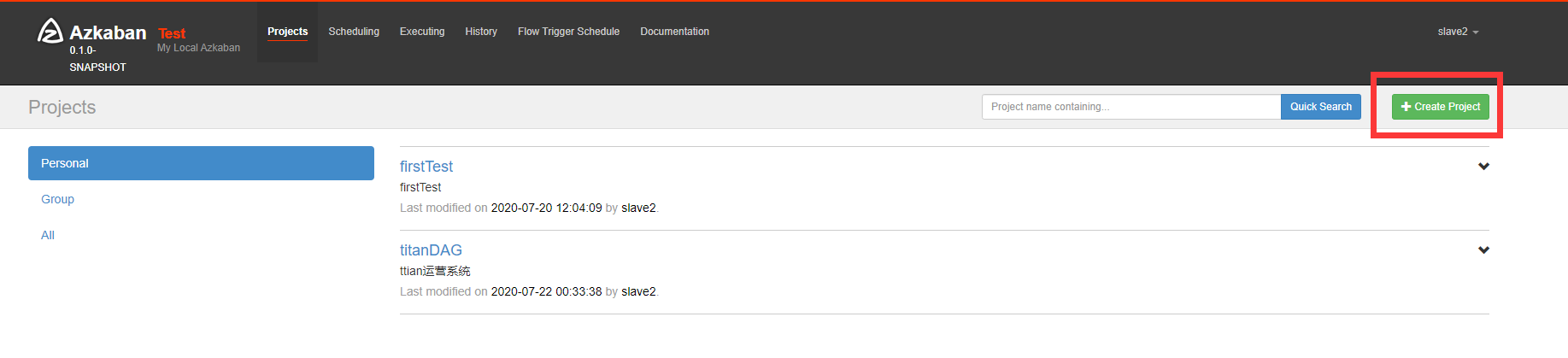
之后一直将所有的任务编写为job，并放到jobs文件夹下面

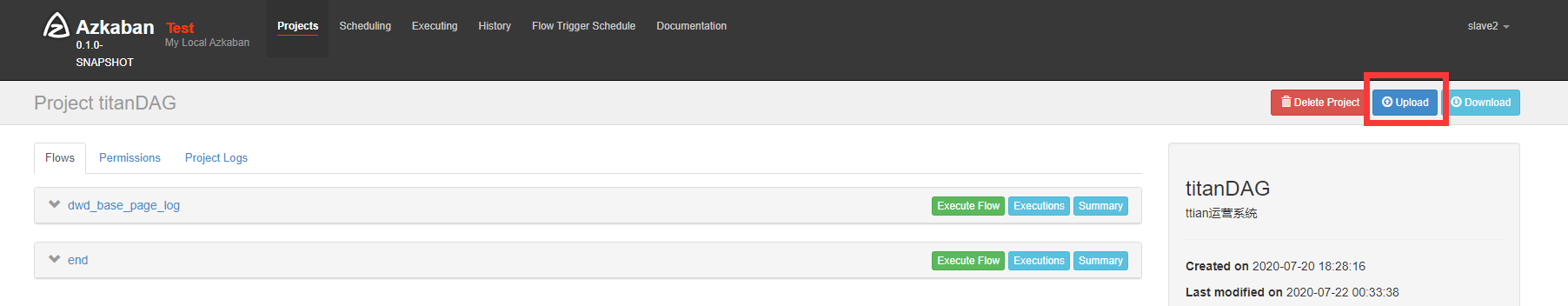


1. **将所有job资源打到一个zip包中**



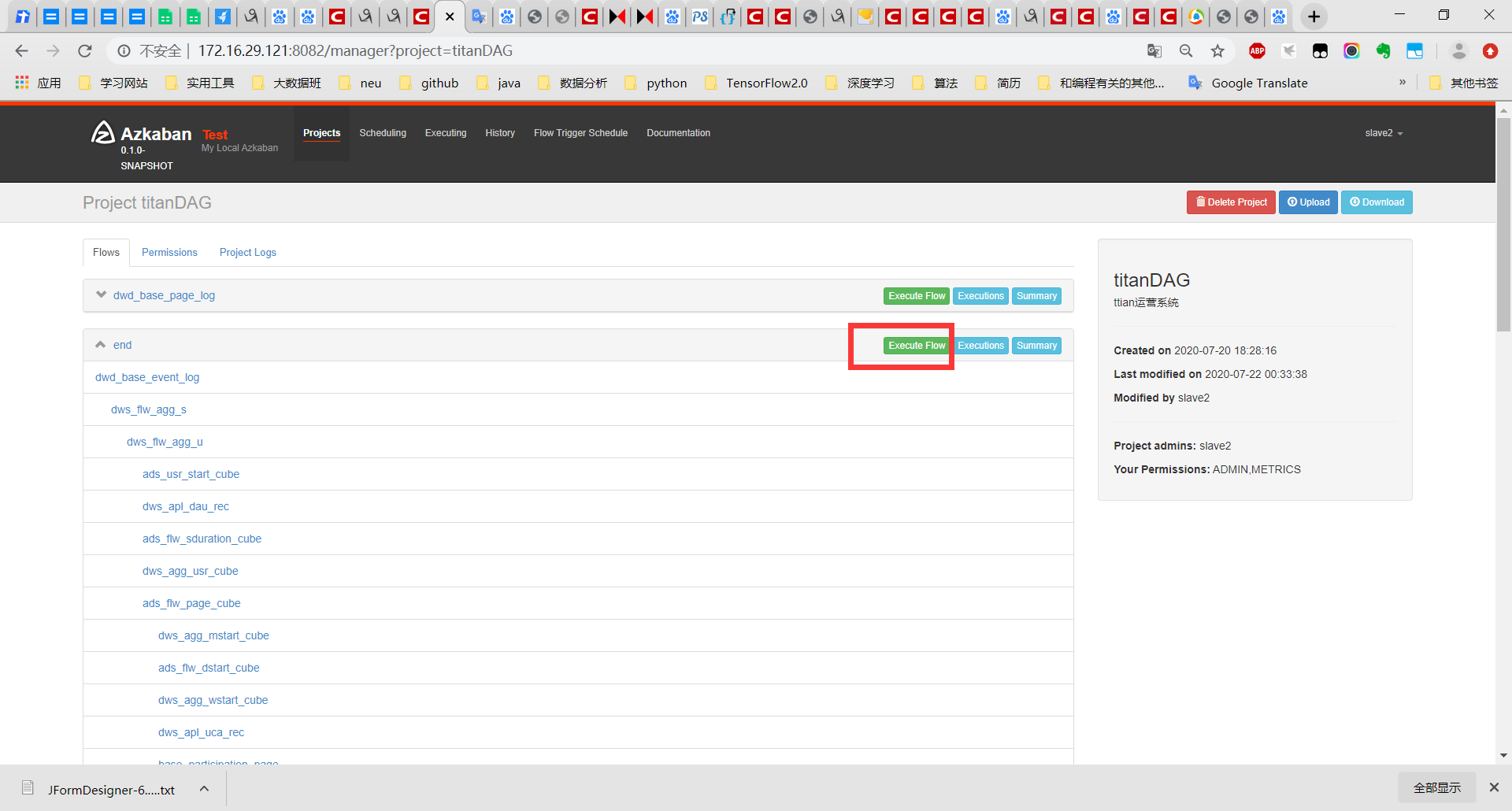
1. **在 azkaban 的 web 管理界面创建工程并上传 zip 包**

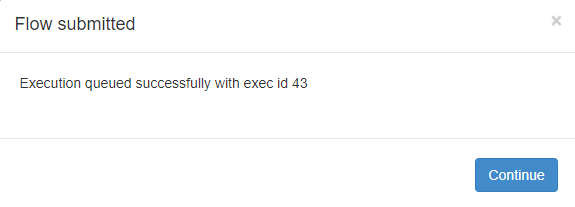
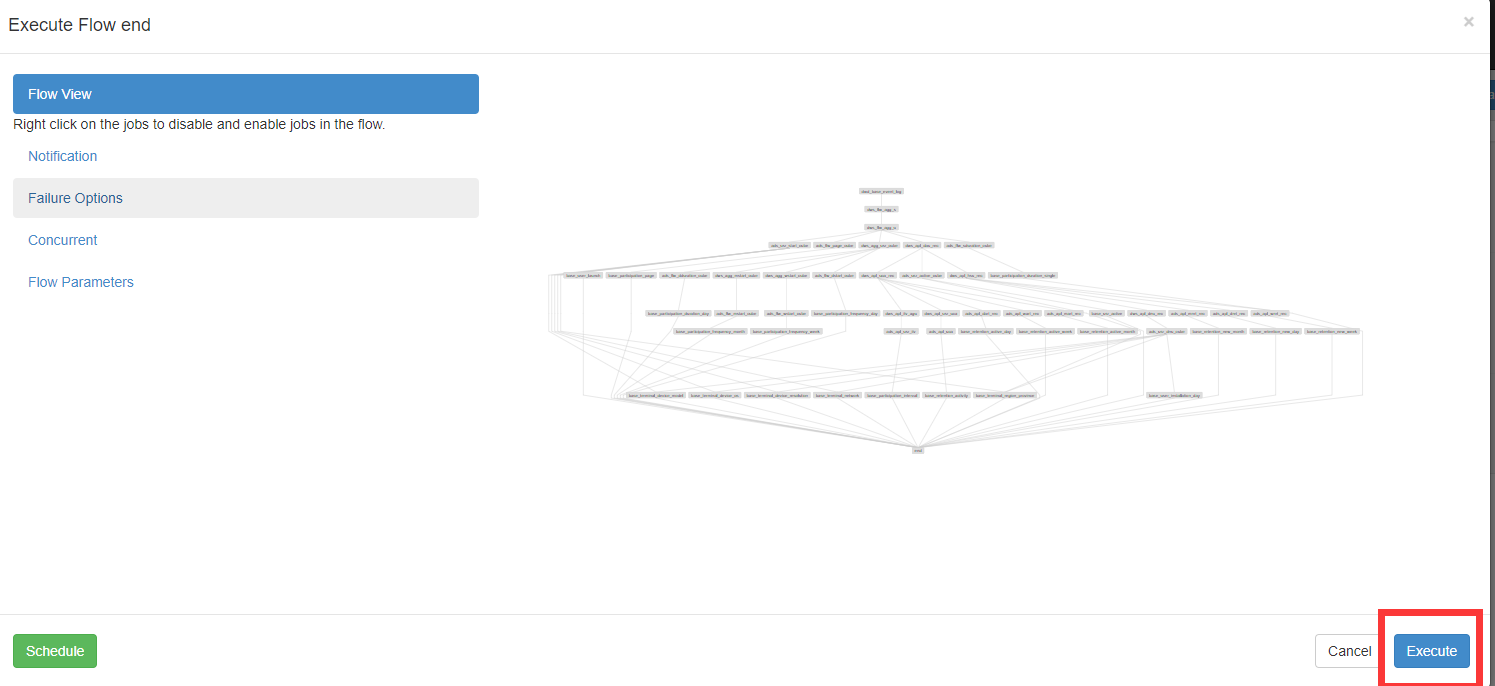




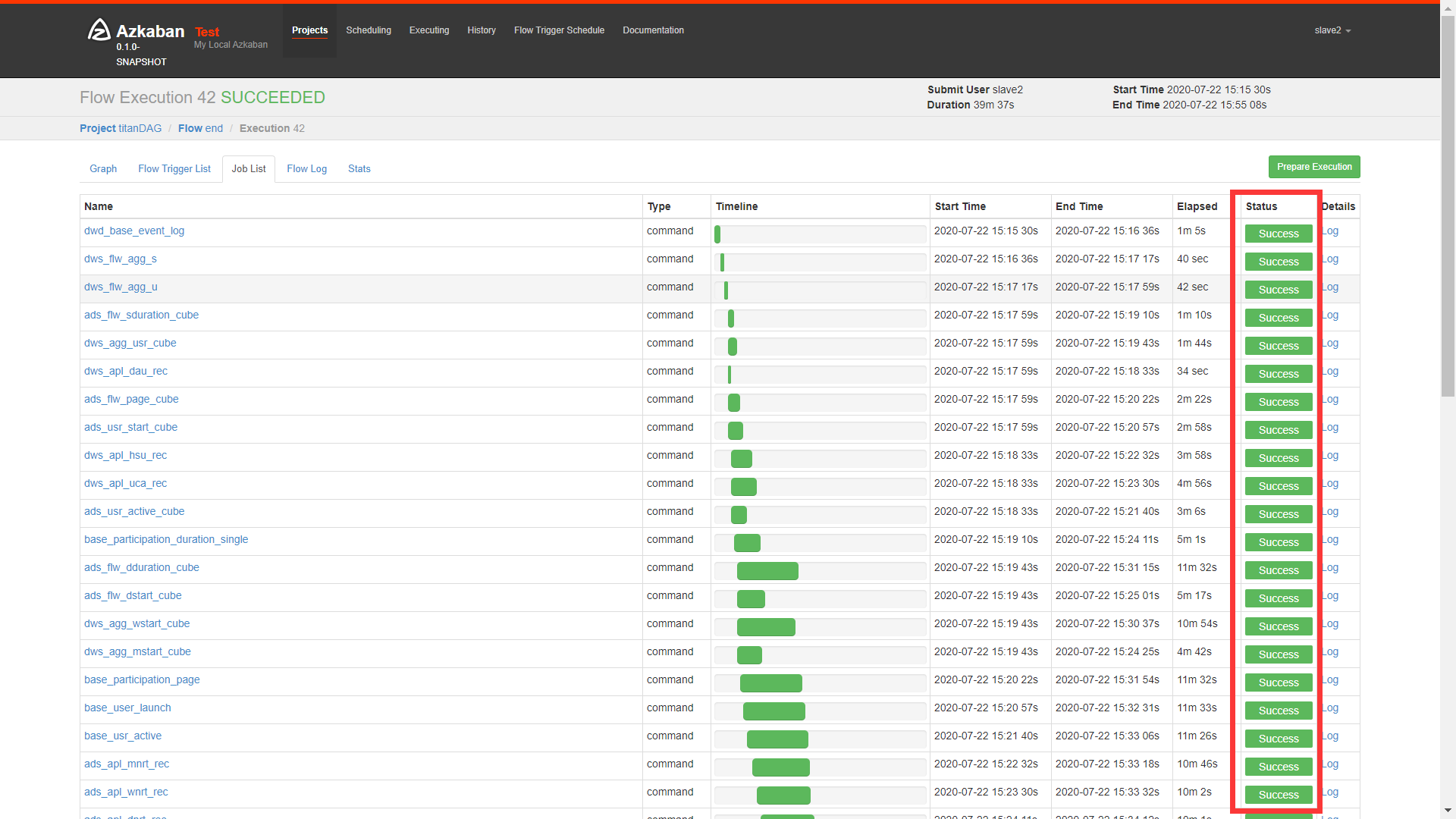
1. **启动执行该Flow**

**点击执行工作流**





1. **Flow执行成功**



1. **点击查看 Flow 日志**

