

基于 DLF 进行脚本开发和作业开发

1. 任务介绍

通过数据湖工厂 DLF 和数据湖探索 DLI 服务对电影评分原始数据进行分析，输出评分最高和最活跃 Top20 电影。用户可以学习到 DLF 脚本编辑、作业编辑、作业调度等功能，以及 DLI 的 SQL 基本语法。

操作流程如下：

1. 准备原始数据，并上传到 OBS 中。
2. 创建 DLF 到 DLI 的数据连接，之后用户可以在 DLF 界面中操作 DLI，例如：创建数据库、创建数据表、分析数据。
3. 创建 DLI SQL 脚本，通过 DLI SQL 脚本可以创建数据表、分析数据。
4. 创建 DLF 作业，通过编排作业和配置作业调度策略，定期执行 DLI SQL 脚本，使得用户可以每五分钟获取到最新的 Top20 电影结果。

2. 任务执行

2.1 环境准备

- 已开通对象存储服务 OBS，并创建桶，例如 “s3a://obs-movies”，用于存放原始数据和分析结果数据。
- 已开通数据湖探索服务 DLI。

2.2 数据准备

1. 获取原始数据（演示数据来自：<https://grouplens.org/datasets/movielens/>），并保存为 csv 格式的文件。原始数据说明如下：

- **movies.csv**

保存电影的基本信息，包含电影 ID、名称、类型。部分数据如表 1 movies.csv 部分数据所示。

表 1 movies.csv 部分数据

movieId	title	genres
1	Toy Story (1995)	Adventure Animation Children Comedy Fantasy
2	Jumanji (1995)	Adventure Children Fantasy
3	Grumpier Old Men (1995)	Comedy Romance
4	Waiting to Exhale (1995)	Comedy Drama Romance
5	Father of the Bride Part II (1995)	Comedy

- **ratings.csv**

保存电影评分，包含用户 ID、电影 ID、评分（0~5）、评分时间。部分数据如 表 2 ratings.csv 部分数据所示。

表 2 ratings.csv 部分数据

userId	movieId	rating	timestamp
1	31	2.5	1260759144
1	1029	3	1260759179
1	1061	3	1260759182
1	1129	2	1260759185
1	1172	4	1260759205

2. 将 movies.csv、ratings.csv 两个原始数据上传至 OBS 桶，例如
“s3a://obs-movies”，后续 DLF 和 DLI 将直接对 OBS 桶中的数据进行
处理。

注意：桶名称可能会被占用，用户根据实际情况创建桶及目录，并在后面脚本中替换对应对象存储服务路径地址。

2.3 创建数据表

用户可以通过 DLI 或 DLF 的编辑器执行 SQL 来创建数据表，本文以使用 DLF 编辑器为例。

步骤 1 创建一个 DLF 到 DLI 的连接，数据连接名称为“dli”。

图 1 创建数据连接



步骤 2 使用 DLF 在 DLI 中创建一个数据库，用于存放数据表，数据库名称为
“movies”。

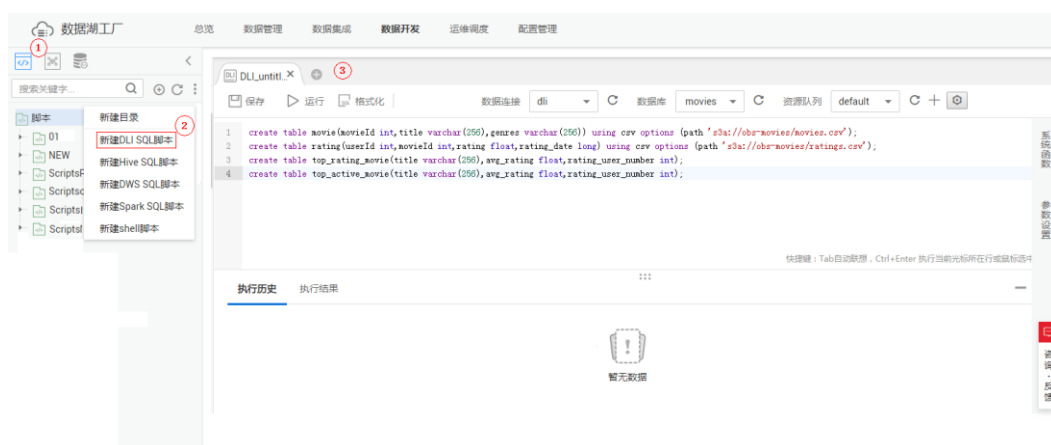
图 2 从数据开发-脚本开发进入下图页面，创建数据库




步骤 3 创建一个 DLI SQL 脚本，通过 SQL 语句来创建数据表。

其中，movie、rating 为 OBS 表，数据存储在 OBS 中，两张表用于存放原始数据。top_rating_movie、top_active_movie 为 DLI 表，两张表用于存放分析结果。

图 3 创建数据表



关键操作说明：

- 图 3 创建数据表中的脚本开发区为临时调试区，关闭脚本页签后，开发区的内容将丢失。如需保留该 SQL 脚本，请单击 ，将脚本保存至指定的目录中。

关键参数说明：

- 数据连接：[步骤 1](#) 创建一个 DLF 到 DLI 的连接，数据连接名称为“dli”。中创建的 DLI 数据连接。
- 数据库：[步骤 2](#) 使用 DLF 在 DLI 中创建一个数据库，用于存放数据表，数据库名称为“movies”。中创建的数据库。
- 资源队列：使用 DLI 提供的默认资源队列“default”。
- SQL 语句：如下所示。

```
create table movie(movieId int,title varchar(256),genres
varchar(256)) using csv options (path 's3a://obs-
movies/movies.csv');
```

```
create table rating(userId int,movieId int,rating
float,rating_date long) using csv options (path 's3a://obs-
movies/ratings.csv');
create table top_rating_movie(title varchar(256),avg_rating
float,rating_user_number int);
create table top_active_movie(title varchar(256),avg_rating
float,rating_user_number int);
```

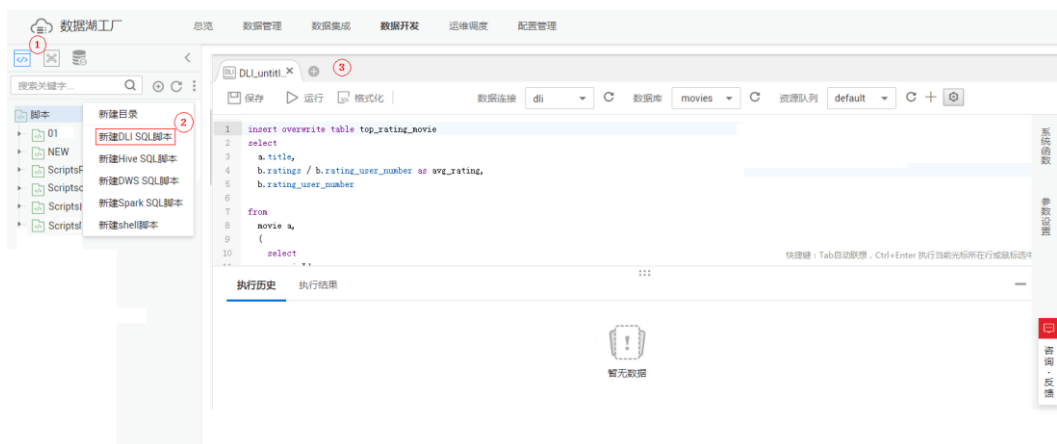
----结束

2.4 分析数据（评分最高 Top20 电影）

评分最高 Top20 电影的计算方法是：先计算出每部电影的总评分和参与评分的用户数，过滤掉参与评分的用户数小于 100 的记录,返回电影名称、平均评分和参与评分用户数。

步骤1 开发一个 DLI SQL 脚本，从 movie 和 rating 表中计算出评分最高的 Top20 电影，将结果存放到 top_rating_movie 表。

图 4 脚本（评分最高 Top20 电影）



关键参数说明：

- 数据连接：[步骤 1](#) 创建一个 DLF 到 DLI 的连接，数据连接名称为“dli”。中创建的 DLI 数据连接。

- 数据库：[步骤 2](#) 使用 DLF 在 DLI 中创建一个数据库，用于存放数据表，数据库名称为 “movies”。中创建的数据库。
- 资源队列：使用 DLI 提供的默认资源队列 “default”。
- SQL 语句：如下所示。

```
insert overwrite table top_rating_movie
select
a.title,
b.ratings / b.rating_user_number as avg_rating,
b.rating_user_number

from
movie a,
(
select
movieId,
sum(rating) ratings,
count(1) as rating_user_number

from
rating
group by
movieId
) b
where
rating_user_number > 100
and a.movieId = b.movieId
order by
avg_rating desc
limit
20
```

步骤2 脚本调试无误后，我们需要保存该脚本，脚本名称为“top_rating_movie”。在后续[创建 DLF 作业](#)引用该脚本。

----结束

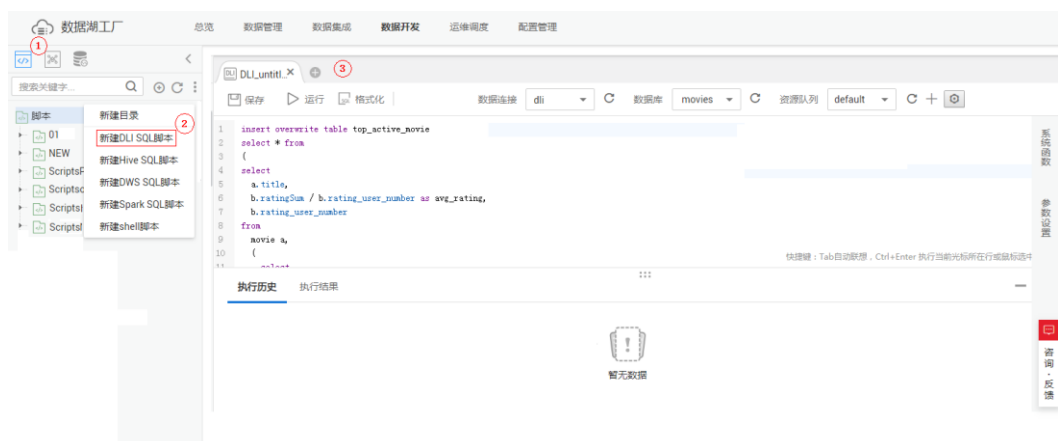
2.5 分析数据（最活跃 Top20 电影）

最活跃 Top20 电影的计算方法是：平均评分大于 3.5 中用户评分数最多的 20 部电影。

步骤1 创建和开发一个 DLI SQL 脚本，从 movie 和 rating 表中计算出最活跃的 Top20 电影，

将结果存放到 top_active_movie 表。

图 5 脚本（分析最活跃 Top20 电影）



关键参数说明：

- 数据连接：[步骤 1](#) 创建一个 DLF 到 DLI 的连接，数据连接名称为“dli”。中创建的 DLI 数据连接。
- 数据库：[步骤 2](#) 使用 DLF 在 DLI 中创建一个数据库，用于存放数据表，数据库名称为“movies”。中创建的数据库。
- 资源队列：使用 DLI 提供的默认资源队列“default”。
- SQL 语句：如下所示。

```
insert overwrite table top_active_movie
select * from
(
select
  a.title,
  b.ratingSum / b.rating_user_number as avg_rating,
  b.rating_user_number
from
  movie a,
  (
    select
      movieId,
      sum(rating) ratingSum,
      count(1) as rating_user_number
    from
      rating
    group by
      movieId
  ) b
where
  a.movieId = b.movieId
) t
where
  t.avg_rating > 3.5
order by
  rating_user_number desc
limit
  20
```

步骤2 脚本调试无误后，我们需要保存该脚本，脚本名称为
“top_active_movie”。在后续[创建 DLF 作业](#)引用该脚本。

-----结束

2.6 创建 DLF 作业

假设 “movie” 和 “rating” 表是实时变动的，我们希望每五分钟更新 Top20 电影，那么这里可以使用 DLF 作业编排和作业调度功能。

步骤1 创建一个 DLF 空作业，作业名称为 “topmovie”。

图 6 创建作业



步骤2 然后进入到**作业开发页面**，拖动 Dummy 和 DLI SQL 节点到画布中，连接并配置节点的属性。

图 7 连接和配置节点属性



关键说明：

- Begin (Dummy 节点)：不执行任何操作，只作为起始点的标识。
- top_rating_movie (DLI SQL 节点)：在节点属性中，关联[分析数据 \(评分最高 Top20 电影 \)](#)中开发完成的 DLI SQL 脚本 “top_rating_movie”。
- top_active_movie (DLI SQL 节点)：在节点属性中，关联[分析数据 \(最活跃 Top20 电影 \)](#)中开发完成的 DLI SQL 脚本 “top_active_movie”。

步骤3 作业编排完成后，单击▶，测试运行作业。



步骤4 如果日志运行正常，单击画布空白处，在右侧的“调度配置”页面，配置作业的调度策略。

图 8 调度配置



说明：

- 2018/10/01 至 2018/10/31，每 5 分钟执行一次作业。

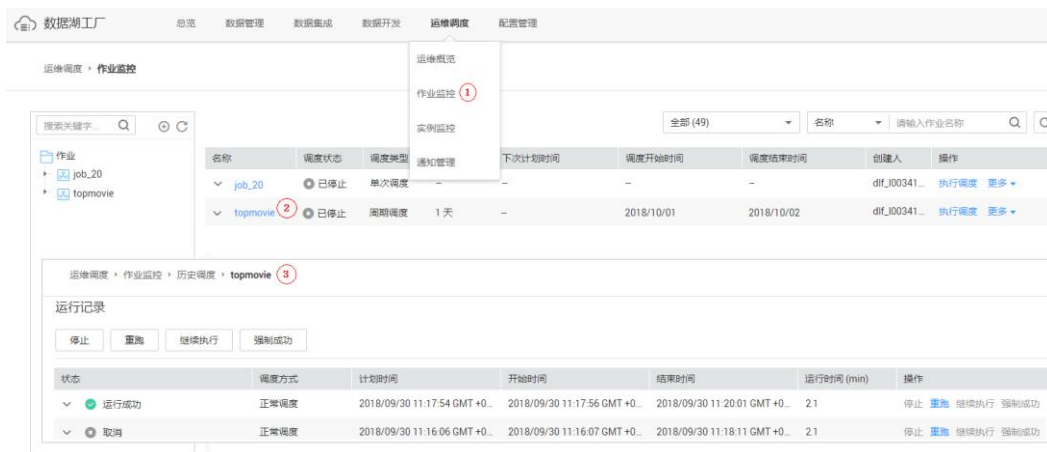
步骤5 最后我们需要保存作业（单击），并执行调度作业（单击）。实现作业每五分钟自动运行，Top20 电影的结果自动保存到 “top_active_movie” 和 “top_rating_movie” 表。

-----结束

2.7 监控作业执行情况

用户如果需要及时了解作业的执行结果是成功还是失败，可以通过 DLF 的监控界面、邮件通知、短信通知进行了解。以下展示如何进入监控界面查看执行结果。

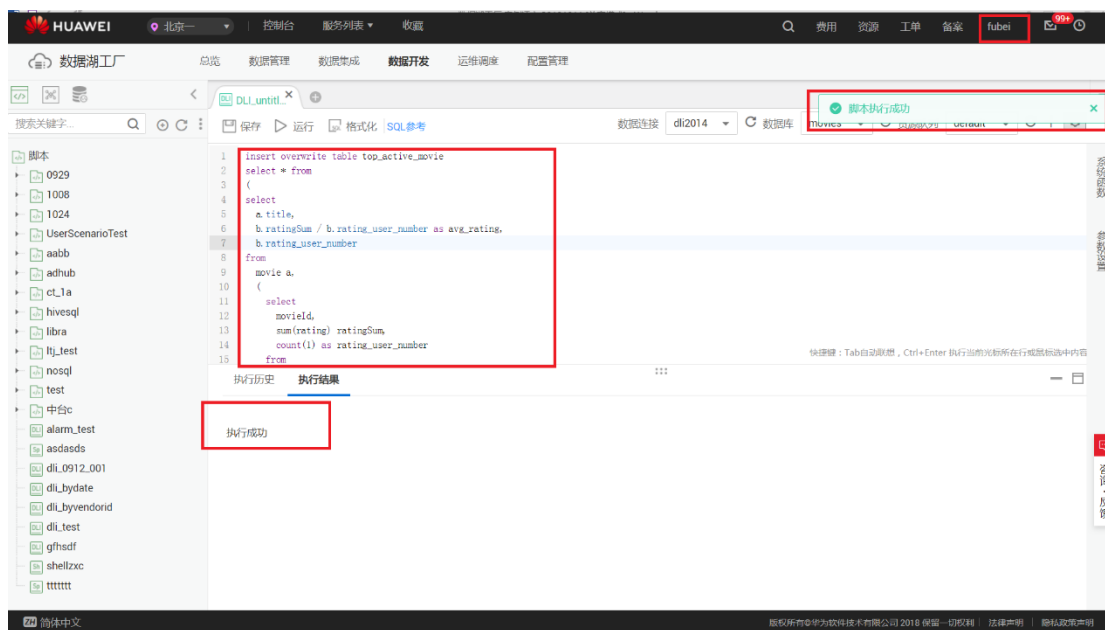
图 9 查看作业执行情况



3. 打卡任务

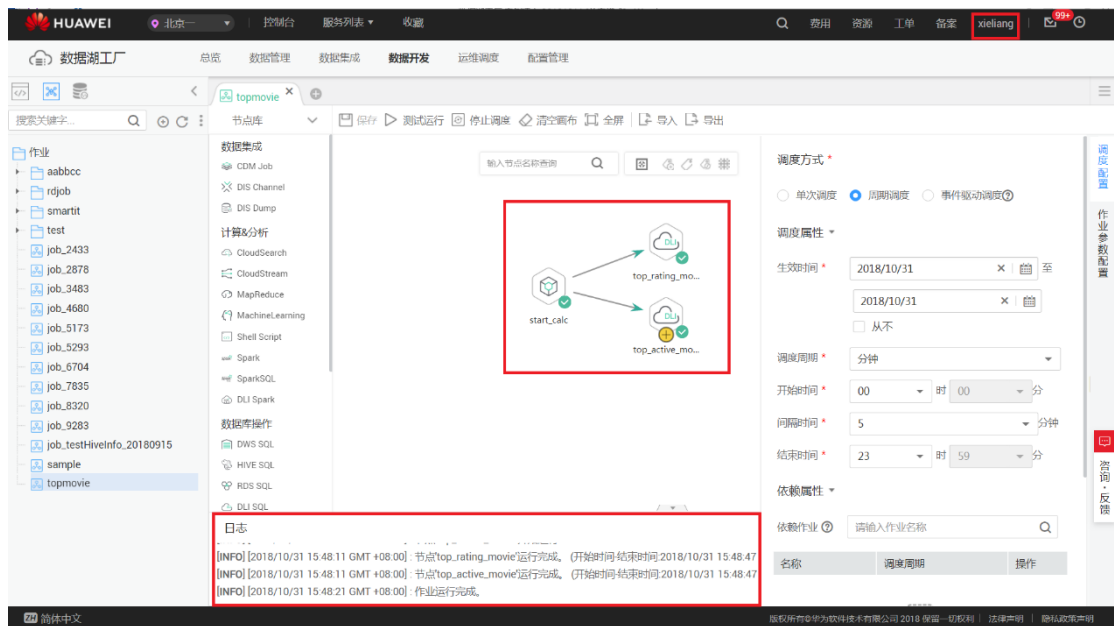
- 1、 生成一个最活跃 Top20 电影的脚本并执行成功，截图包含用户名称。

图 10 脚本运行成功截图



- 2、 点击作业的测试运行按钮，测试运行作业成功。

图 11 作业测试运行成功截图



3、运维调度作业成功运行一次。

图 12 作业调度运行成功截图

