

# shiyanshiyan

## 3.19实验课-使用Flink实时发现最热Github项目

准备工作：

### 1. 创建Flink工作空间

<https://home.console.aliyun.com/home/dashboard/ProductAndService>



点击实时计算Flink版，进入其控制台



上节课已经领取了资源折扣，点击立即购买

Flink 全托管

一键购买

付费模式

按量付费

资源抵扣包

计价说明

地域

华北2 (北京)

华南1 (深圳)

华东2 (上海)

华东1 (杭州)

中国香港

新加坡

英国 (伦敦)

美国 (硅谷)

美国 (弗吉尼亚)

印度尼西亚 (雅加达)

德国 (法兰克福)

日本 (东京)

马来西亚 (吉隆坡)

不同地域的实例之间互不相通，选择靠近您客户的地域，可降低网络时延，提高您客户的访问速度。

如何选择地域

资源抵扣包

☒ 使用资源抵扣包

5000CUH

中国内地公共云类型支持抵扣上海、杭州、北京、深圳、张家口区域的用量

资源包详情

您账户下有 5000CUH 的资源包可用于抵扣中国内地公有云地域按量付费费用，自购买资源包日起有效期三个月。

虚拟交换机可以选择ECS以及RDS上对应的区域

专有网络

vpc-bp12s9xolkmb98ksauq

当前地域支持的可用区为 杭州 可用区 H、I、J。请在当前地域和当前可用区下 创建专有网络

虚拟交换机

	名称 / ID	所属可用区	网段	可用 IP 数
<input checked="" type="checkbox"/>		可用区 H	172.24.144.0/20	4091

您至少需要选择 1 个虚拟交换机，强烈建议您选择 2 个不同可用区的虚拟交换机。请根据 Flink 作业规模，合理规划网段。平台侧计算层和网络层解耦，实际工作空间作业的调度不依赖交换机的选择。  
当前地域支持的可用区为 杭州 可用区 H、I、J。请在当前地域下 创建虚拟交换机

创建完成之后，进入所创建的工作空间的控制台

阿里云

实时计算 Flink 版 / 概览页

实时计算控制台

实时计算 (Alibaba Cloud RealtimeCompute, Powered by Ververica) 是阿里云基于 Apache Flink 构建的企业级实时大数据计算商业产品。实时计算 Flink 由 Apache Flink 创始团队官方出品，拥有全球统一商业化品牌，提供全系列产品信息，完全兼容开源 Flink API，并充分基于强大的阿里云平台提供云原生的 Flink 商业增值能力。

Flink 全托管

提供全托管版 Flink 集群和引擎，无需关注底层，让开发人员能够专注业务、快速交付。适用于不关注集群运维的上云客户。

快速入门

立即购买

Flink 全托管

付费类型

标签选择

工作空间 / 订单实例 ID	标签	工作空间状态	处理引擎架构	已使用 / 已购买 CU	付费类型	时间	操作
exp-319 t-cn-20s46l8pz0m	#	运行中	X86	9.00 / -	按量付费	创建时间: 2025-03-18 20:25 到期时间: -	控制台 更多

最新公告

暂无公告

常见问题

什么是阿里云实时计算 Flink 版

Flink 全托管介绍

Flink 全托管角色授权

Flink 全托管开通流程

Flink 全托管定价文档

Flink 全托管常见问题

## 2. Flink Session集群

根据上节课所讲，创建一个Session，可将Task Managers 数量设为4，增大并发度。

引擎版本 \*

请选择引擎版本

Flink 重启策略配置

请选择一个 Flink 重启策略

其他配置

1

在此设置其他 Flink 配置。例如: `taskmanager.numberOfTaskSlots: 1`

资源配置

Task Manager 数量

4

默认与并行度一致

## 实验一：Github 关注数排行榜

创建一个SQL作业

阿里云实时计算 powered by ververica exp-319-default

新建

概览

数据管理

数据血缘

数据开发

数据接入 Beta

ETL

数据查询

运维中心

作业运维

任务编排 Beta

队列管理

### 作业运维

部署作业 流作业 批量启动 批量停止 名称 搜索... 标签选择 重置

JAR 作业

Python 作业

SQL 作业

YAML 作业

状态 健康分 Beta 业务延时 CPU 内存 修改人

暂无数据



**注意：**其中startTime尽量设置为当前此刻的一周前附近，如果设置的时间太早，前面无效计算时间比较长，不仅耗费资源，而且很久才能加载出计算结果。根据不同的地域设置相应的project和endPoint，在本实验中，实例为上海的服务平台，因此设置'project'='github-events-shanghai'和'endPoint'='https://cn-shanghai-intranet.log.aliyuncs.com'，其他地域如北京、杭州、深圳更改为对应值即可。

```
1  -- 通过DDL语句创建SLS源表，SLS中存放了Github的实时数据。
2  CREATE TEMPORARY TABLE gh_event(
3      id STRING,                                -- 每个事件的唯一ID。
4      created_at BIGINT,                        -- 事件时间，单位秒。
5      created_at_ts as TO_TIMESTAMP(created_at*1000), -- 事件时间戳（当前会话时区下的时间戳，如：Asia/Shanghai）。
6      type STRING,                             -- Github事件类型，如：ForkEvent, WatchEvent, IssuesEvent, CommitCommentEvent等。
7      actor_id STRING,                         -- Github用户ID。
8      actor_login STRING,                      -- Github用户名。
9      repo_id STRING,                          -- Github仓库ID。
10     repo_name STRING,                        -- Github仓库名，如：
        apache/flink, apache/spark, alibaba/fastjson等。
11     org STRING,                             -- Github组织ID。
12     org_login STRING                        -- Github组织名，如：
        apache,google,alibaba等。
13 ) WITH (
14     'connector' = 'sls',                    -- 实时采集的Github事件存放在阿里云SLS中。
15     'project' = 'github-events-shanghai', -- 存放公开数据的SLS项目。例如'github-events-hangzhou'。
16     'endpoint' = 'https://cn-shanghai-intranet.log.aliyuncs.com', -- 公开数据仅限VVP通过私网地址访问。例如'https://cn-hangzhou-
        intranet.log.aliyuncs.com'。
17     'logStore' = 'realtime-github-events', -- 存放公开数据的SLS logStore。
18     'accessId' = 'LTAI5NMF1PBNKQYJr9TKgh', -- 只读账号的AK。
19     'accessKey' = 'F0qsh6fjSakbFxx83TNGH0iQWVTP', -- 只读账号的SK。
20     'batchGetSize' = '500',                -- 批量读取数据，每批最多拉取500条。
21     'startTime' = '2025-03-18 16:00:00'    -- 开始时间，尽量设置到需要计算的时间附近。否则无效计算的时间较长，默认为当前值
22 );
23
24 -- 配置开启mini-batch，每2s处理一次。
25 SET 'table.exec.mini-batch.enabled'='true';
26 SET 'table.exec.mini-batch.allow-latency'='2s';
27 SET 'table.exec.mini-batch.size'='4096';
28
29 -- 作业设置4个并发，算会更快。
30 SET 'parallelism.default' = '4';
31
32
33 -- 查看Github新增star数Top 5仓库。
34 SELECT DATE_FORMAT(created_at_ts, 'yyyy-MM-dd') as 'date', repo_name, COUNT(*) as num
35 FROM gh_event WHERE type = 'WatchEvent'
36 GROUP BY DATE_FORMAT(created_at_ts, 'yyyy-MM-dd'), repo_name
37 ORDER BY num DESC
38 LIMIT 5;
```

```

intranet.log.aliyuncs.com'。
17   'logStore' = 'realtime-github-events',           -- 存放公开数据的SLS
    logStore。
18   'accessId' = 'LTAI5tNF1rP8PKVyYjr9TKgh',       -- 只读账号的AK。
19   'accessKey' = 'FDgsh6fjSmkbFsx083tN6H0iqNVWTP', -- 只读账号的SK。
20   'batchGetSize' = '500',                         -- 批量读取数据，每批最多拉取
    500条。
21   'startTime' = '2025-03-18 14:00:00'           -- 开始时间，尽量设置到需要计
    算的时间附近，否则无效计算的时间较长。默认值为当前值
22 );
23
24 -- 配置开启mini-batch，每2s处理一次。
25 SET 'table.exec.mini-batch.enabled'='true';
26 SET 'table.exec.mini-batch.allow-latency'='2s';
27 SET 'table.exec.mini-batch.size'='4096';
28
29 -- 作业设置4个并发，聚合更快。
30 SET 'parallelism.default' = '4';
31
32 -- 查看Github新增star数Top 5仓库。
33 SELECT DATE_FORMAT(created_at_ts, 'yyyy-MM-dd') as `date`, repo_name,
    COUNT(*) as num
34 FROM gh_event WHERE type = 'WatchEvent'
35 GROUP BY DATE_FORMAT(created_at_ts, 'yyyy-MM-dd'), repo_name
36 ORDER BY num DESC
37 LIMIT 5;

```

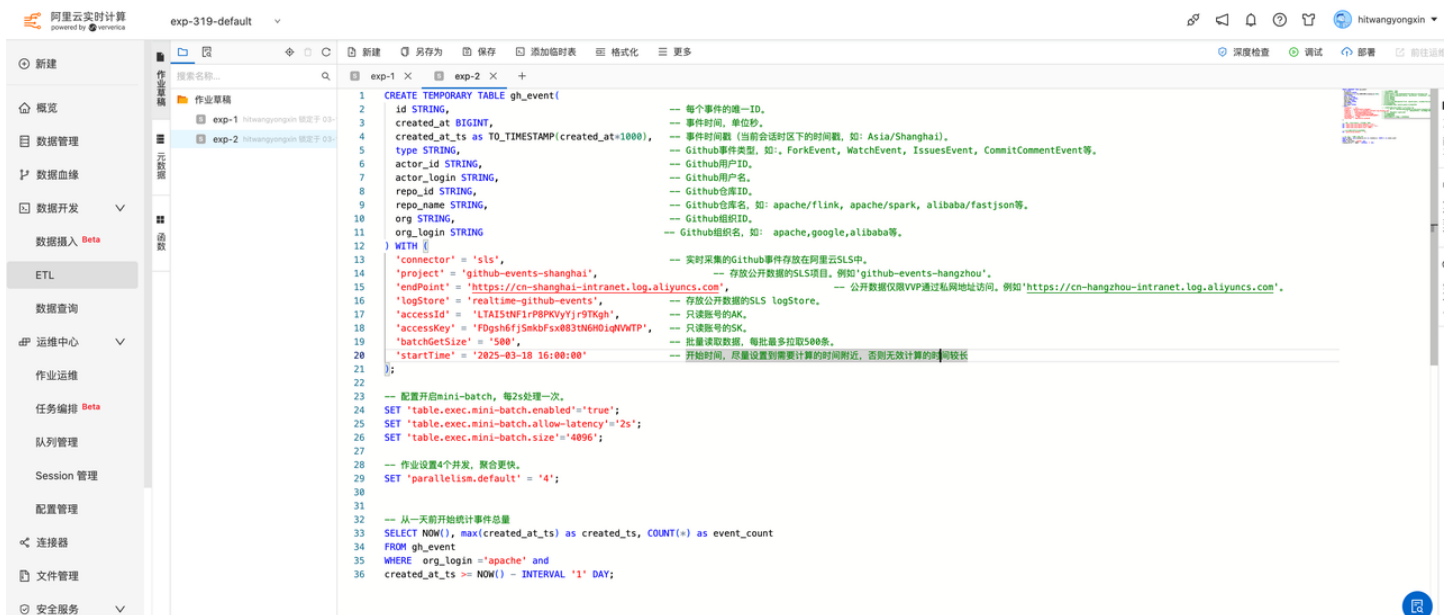


运行之后可以得出类似的结果。

## 实验二：统计组织活跃度变化



新建一个作业，设置保持默认即可。

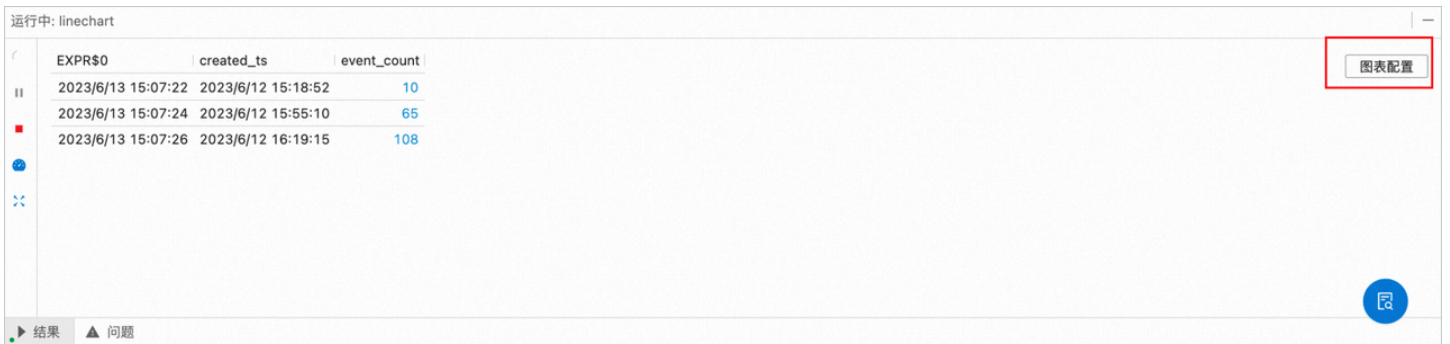


```
1 CREATE TEMPORARY TABLE gh_event(  
2     id STRING, -- 每个事件的唯一ID。  
3     created_at BIGINT, -- 事件时间，单位秒。  
4     created_at_ts as TO_TIMESTAMP(created_at*1000), -- 事件时间戳（当前会话时区下的  
-- 的时间戳，如：Asia/Shanghai）。  
5     type STRING, -- Github事件类型，如：。  
-- ForkEvent, WatchEvent, IssuesEvent, CommitCommentEvent等。  
6     actor_id STRING, -- Github用户ID。  
7     actor_login STRING, -- Github用户名。  
8     repo_id STRING, -- Github仓库ID。  
9     repo_name STRING, -- Github仓库名，如：  
-- apache/flink, apache/spark, alibaba/fastjson等。  
10    org STRING, -- Github组织ID。  
11    org_login STRING -- Github组织名，如：  
-- apache, google, alibaba等。
```

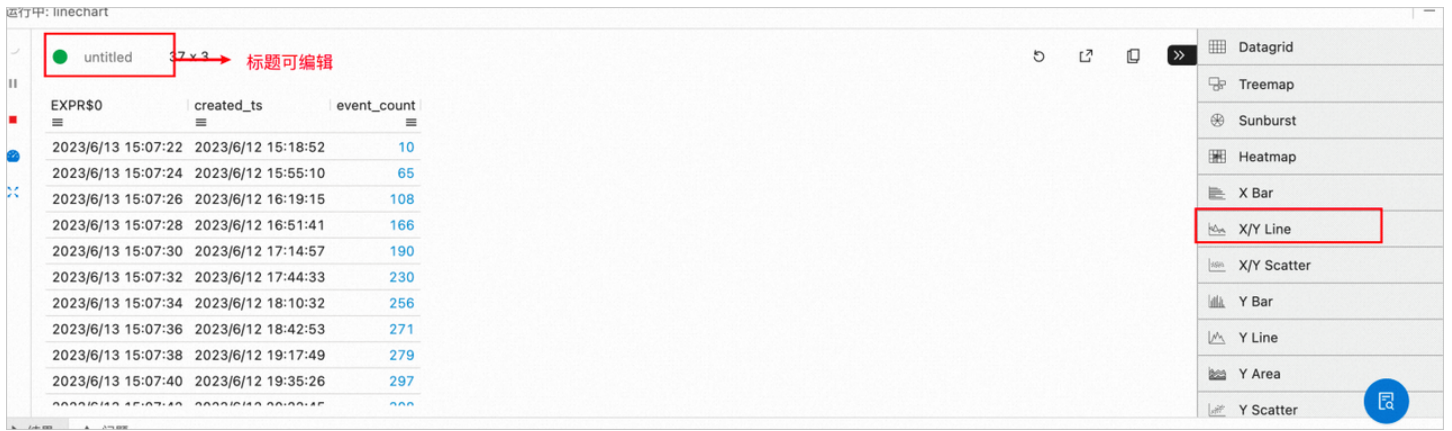
```

12 ) WITH (
13     'connector' = 'sls',                                -- 实时采集的Github事件存放在阿里云SLS中。
14     'project' = 'github-events-shanghai',                -- 存放公开数据的SLS项目。例如'github-events-hangzhou'。
15     'endPoint' = 'https://cn-shanghai-intranet.log.aliyuncs.com',
16     -- 公开数据仅限VVP通过私网地址访问。例如'https://cn-hangzhou-intranet.log.aliyuncs.com'。
17     'logStore' = 'realtime-github-events',                -- 存放公开数据的SLS logStore。
18     'accessId' = 'LTAI5tNF1rP8PKVyYjr9TKgh',            -- 只读账号的AK。
19     'accessKey' = 'FDgsh6fjSmkbFsx083tN6H0iqNVWTP',    -- 只读账号的SK。
20     'batchGetSize' = '500',                              -- 批量读取数据，每批最多拉取500条。
21     'startTime' = '2025-03-18 16:00:00'                 -- 开始时间，尽量设置到需要计算的时间附近，否则无效计算的时间较长
22 );
23 -- 配置开启mini-batch，每2s处理一次。
24 SET 'table.exec.mini-batch.enabled'='true';
25 SET 'table.exec.mini-batch.allow-latency'='2s';
26 SET 'table.exec.mini-batch.size'='4096';
27
28 -- 作业设置4个并发，聚合更快。
29 SET 'parallelism.default' = '4';
30
31 -- 从一天前开始统计事件总量
32 SELECT NOW(), max(created_at_ts) as created_ts, COUNT(*) as event_count
33 FROM gh_event
34 WHERE org_login = 'apache' and
35 created_at_ts >= NOW() - INTERVAL '1' DAY;

```







编辑标题为**apache**，并且选择X/Y Line。

配置X轴为create\_ts,y 轴为event\_count。

### 实验 三: 统计仓库贡献时间分布情况

```
1 CREATE TEMPORARY TABLE gh_event(  
2     id STRING, -- 每个事件的唯一ID。  
3     created_at BIGINT, -- 事件时间，单位秒。  
4     created_at_ts AS TO_TIMESTAMP(created_at*1000), -- 事件时间戳（当前会话时区  
5     type STRING, -- Github事件类型，如：。  
6     actor_id STRING, -- Github用户ID。  
7     actor_login STRING, -- Github用户名。  
8     repo_id STRING, -- Github仓库ID。  
9     repo_name STRING, -- Github仓库名，如：  
10    apache/flink, apache/spark, alibaba/fastjson等。  
11    org STRING, -- Github组织ID。  
12    org_login STRING -- Github组织名，如：  
13    ) WITH ( -- 实时采集的Github事件存放  
14    'connector' = 'sls', -- 在阿里云SLS中。  
15    'project' = 'github-events-shanghai', -- 存放公开数据的SLS  
16    'endPoint' = 'https://cn-shanghai-intranet.log.aliyuncs.com', -- 项目。例如'github-events-hangzhou'。  
17    'logStore' = 'realtime-github-events', -- 公开数据仅限VVP通过私网地址访问。例如'https://cn-hangzhou-  
18    'accessId' = 'LTAI5tNF1rP8PKVyYjr9TKgh', -- intranet.log.aliyuncs.com'。  
19    'accessKey' = 'FDgsh6fjSmkbFsx083tN6H0iqNVWTP', -- 存放公开数据的SLS  
20    'batchGetSize' = '500', -- 只读账号的AK。  
21    'logStore' = 'realtime-github-events', -- 只读账号的SK。  
22    'accessId' = 'LTAI5tNF1rP8PKVyYjr9TKgh', -- 批量读取数据，每批最多拉取  
23    'accessKey' = 'FDgsh6fjSmkbFsx083tN6H0iqNVWTP', -- 500条。  
24    'batchGetSize' = '500',
```



```

20     'startTime' = '2025-03-12 14:00:00' -- 开始时间, 尽量设置到需要计
      算的时间附近, 否则无效计算的时间较长
21 );
22
23 -- 配置开启mini-batch, 每2s处理一次。
24 SET 'table.exec.mini-batch.enabled'='true';
25 SET 'table.exec.mini-batch.allow-latency'='2s';
26 SET 'table.exec.mini-batch.size'='4096';
27
28 -- 作业设置4个并发, 聚合更快。
29 SET 'parallelism.default' = '4';
30
31 -- 统计从上周起的贡献量
32 SELECT DATE_FORMAT(created_at_ts, 'yyyy-MM-dd') as comment_date,
      HOUR(created_at_ts) AS comment_hour ,COUNT(*) AS comment_count
33 FROM gh_event
34 WHERE created_at_ts >= NOW() - INTERVAL '7' DAY
35        AND repo_name = 'apache/flink'
36        AND (type ='CommitCommentEvent' OR
37             type='IssueCommentEvent' or
38             type = 'PullRequestReviewCommentEvent'or
39             type = 'PushEvent' or
40             type = 'PullRequestEvent' or
41             type = 'PullRequestReviewEvent')
42 GROUP BY DATE_FORMAT(created_at_ts, 'yyyy-MM-dd'), HOUR(created_at_ts) ;

```

其中startTime尽量设置为当前此刻的一周前附近, 如果设置的时间太早, 前面无效计算时间比较长, 不仅耗费资源, 而且很久才能加载出计算结果。如果想要统计spark, 改成repo\_name = 'apache/spark' 即可。

选择Heatmap, 设置Group by为comment\_date, Spli By为comment\_hour, Color为Sum(comment\_count), 即X轴为天, Y周为小时, 根据总数量显示颜色深浅。

做做完记得释放资源。

## 作业：

1. 熟练掌握实验工作流程。
2. 完成实验一、二、三, 可对sql语句给出理解, 并对实验结果进行简要分析。