

实验三 基于 flinksql 的流式数据查询

22121630 汪江豪

实验步骤及结果：

1. 创建基于实验二制作的镜像，先下载 java 运行环境和 java 开发工具包
sudo apt-get install openjdk-8-jre openjdk-8-jdk
2. 将 java 添加到环境变量

vim ~/.bashrc

第一行添加

```
export JAVA_HOME=/usr/local/lib/jvm/java-8-openjdk-amd64
```

保存退出后，应用修改

```
source ~/.bashrc
```

3. 输出 java 版号以验证 java 成功安装

```
ubuntu@VM-0-13-ubuntu:~$ java -version
openjdk version "1.8.0_442"
OpenJDK Runtime Environment (build 1.8.0_442-8u442-b06~us1-0ubuntu1~20.04-b06)
OpenJDK 64-Bit Server VM (build 25.442-b06, mixed mode)
```

4. 安装 flink

```
wget https://archive.apache.org/dist/flink/flink-1.16.0/flink-1.16.0-bin-scala\_2.12.tgz
```

解压并改名

```
sudo tar -zxf flink-1.16.0-bin-scala_2.12.tgz -C /usr/local
```

```
cd /usr/local
```

```
sudo mv flink-1.16.0 flink
```

添加 flink 主路径到环境变量并应用

```
vim ~/.bashrc
```

```
export FLINK_HOME=/usr/local/flink
```

```
export PATH=$FLINK_HOME/bin:$PATH
```

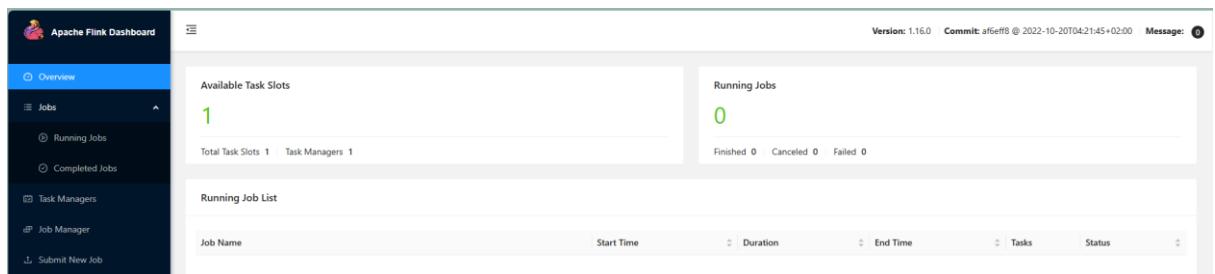
```
source ~/.bashrc
```

5. 到 /usr/local/flink/conf 目录下修改 flink-conf.yaml 文件，将 rest.address 和 rest.bind-address 改为 0.0.0.0

6. 到腾讯云安全组界面放开 8081 端口，启动 flink 集群

start-cluster.sh(路径已添加到环境变量，所以可以这么写)

7. 在任何一个浏览器中输入[服务器的公网 ip]:8081，可进入到 flinksql 的 webui 界面，如下图：



8. 尝试提交一个示例文件，在 webui 的 taskmanager 下查看结果如下

```
flink run /usr/local/flink/examples/batch/WordCount.jar
```

localhost:42083-dcef06					
Path	akka.tcp://flink@localhost:42083/user/pic/taskmanager_0		Free/All Slots		1 / 1
Last Heartbeat	2025-05-13 23:36:54	Data Port	35563	CPU Cores	2
Physical Memory	3.27 GB	JVM Heap Size	512 MB	Flink Managed Memory	512 MB
Metrics	Logs	Stdout	Log List	Thread Dump	
133 (others,1)					
134 (restless,1)					
135 (restless,1)					
136 (scornful,1)					
137 (shocks,1)					
138 (shocks,1)					
139 (spurny,1)					
140 (spurny,1)					
141 (spurny,1)					
142 (spurny,1)					
143 (cowards,1)					
144 (cowards,1)					
145 (cowards,1)					
146 (timid,1)					
147 (timid,1)					
148 (natural,1)					
149 (optimistic,1)					
150 (optimistic,1)					
151 (patient,1)					
152 (patient,1)					
153 (respect,1)					
154 (respect,1)					
155 (thorough,1)					
156 (thorough,1)					
157 (calamity,1)					
158 (calamity,1)					
159 (calamity,1)					
160 (opposing,1)					
161 (opposing,1)					
162 (question,1)					
163 (question,1)					
164 (stirkled,1)					
165 (thousand,1)					
166 (unearthly,1)					
167 (unearthly,1)					
168 (contaminated,1)					
169 (contaminated,1)					
170 (insolence,1)					
171 (oppressor,1)					
172 (oppressive,1)					
173 (oppressive,1)					
174 (traveler,1)					
175 (traveler,1)					
176 (conscience,1)					
177 (resolution,1)					
178 (undiscover,1)					
179 (enterprises,1)					
180 (consuming,1)					
181					

9. 安装 maven 打包编译工具

```
wget https://dlcdn.apache.org/maven/maven-3/3.9.4/binaries/apache-maven-3.9.4-bin.zip
```

也可以将 maven 主目录同样添加到环境变量，便于打包编译 java 项目。

10. 新建 tableapp 文件夹和 tableapp/src/main/java 目录，用于存储项目的作业代码，并写好 pom.xml 配置文件。

11. 导入 medium_relation 数据集后，通过写 java 代码的嵌入式 sql，进行表的创建，结果的输出。

对于 medium_relation 的数据集求相似度的任务，有如下公式：

$$|A| + |B| = |A| \cap |B| + |A| \cup |B|$$

通过先求交集，再通过数学公式求并集，最后交集除以并集，可减少计算量。SQL 主代码如下：

```

package cn.edu.shu;

import org.apache.flink.streaming.api.environment.StreamExecutionEnvironment;
import org.apache.flink.table.api.EnvironmentSettings;
import org.apache.flink.table.api.TableEnvironment;
import org.apache.flink.table.api.bridge.java.StreamTableEnvironment;

public class medium_result {
    public static void main(String[] args) throws Exception {
        // StreamExecutionEnvironment env = StreamExecutionEnvironment.getExecutionEnvironment();
        // StreamTableEnvironment tableEnv = StreamTableEnvironment.create(env);

        // 使用批处理模式
        EnvironmentSettings settings = EnvironmentSettings.newInstance().inBatchMode().build();
        TableEnvironment tableEnv = TableEnvironment.create(settings);

        // env.setParallelism(1);

        String sourceDDL = "CREATE TABLE medium_relation (" +
            + "referrer INT," +
            + "referree INT" +
            + ") WITH (" +
            + " 'connector' = 'filesystem', " +
            + " 'path' = 'file:///home/ubuntu/medium_relation', " +
            + " 'format' = 'csv', " +
            + " 'csv.field-delimiter' = ' ', " +
            + " 'csv.ignore-parse-errors' = 'true' " +
            + ")";
        tableEnv.executeSql(sourceDDL);

        // 修改输出表为文件系统
        tableEnv.executeSql("CREATE TABLE SinkTable (" +
            + "web1 INT, " +
            + "web2 INT, " +
            + "similarity DOUBLE" +
            + ") WITH (" +
            + " 'connector' = 'filesystem', " +
            + " 'path' = 'file:///home/ubuntu/tableapp/output/medium_result', " +
            + " 'format' = 'csv', " +
            + " 'csv.field-delimiter' = ' ', " +
            + " 'sink.rolling-policy.file-size' = '256MB', " +
            + " 'sink.rolling-policy.rollover-interval' = '30 min'" +
            + ")");
    }
}

```

```

// 创建临时视图 ref_count
tableEnv.executeSql("CREATE TEMPORARY VIEW tmp_ref_count AS "
    + "SELECT referer, COUNT(DISTINCT referree) AS web_count "
    + "FROM medium_relation "
    + "GROUP BY referer");

// 创建临时视图 common
tableEnv.executeSql("CREATE TEMPORARY VIEW tmp_common AS "
    + "SELECT a.referrer AS web1, b.referrer AS web2, COUNT(*) AS com_cnt "
    + "FROM medium_relation a "
    + "JOIN medium_relation b ON a.referee = b.referrer "
    + "WHERE a.referrer < b.referrer "
    + "GROUP BY a.referrer, b.referrer");

// 计算相似度并插入结果表
tableEnv.executeSql("INSERT INTO SinkTable "
    + "SELECT "
    + "    common.web1 AS web1, "
    + "    common.web2 AS web2, "
    + "    CASE "
    + "        WHEN r1.web_count + r2.web_count - common.com_cnt > 0 "
    + "        THEN common.com_cnt * 1.0 / (r1.web_count + r2.web_count - common.com_cnt) "
    + "        ELSE 0 "
    + "    END AS similarity "
    + "FROM tmp_common common "
    + "JOIN tmp_ref_count r1 ON common.web1 = r1.referrer "
    + "JOIN tmp_ref_count r2 ON common.web2 = r2.referrer");
}

```

由于输出数据量共六千四百多万行，通过 webui 输出结果容易卡死崩溃，这里将结果输出到了本地的`~/tableapp/output/medium_result`目录下，格式为 csv，列属性为网站 a, 网站 b, 相似度。

12. 保存好 java 作业代码后，由于已经将 maven 的 bin 目录添加到了环境变量，所以可直接在含有 pom.xml 的目录下，输入如下命令对项目进行打包编译：

`mvn package`

13. 然后提交 flink 作业

`flink run`

如下图，使用如下命令：

Head -n 50 [文件名]

对于结果文件，输出了前 50 行结果

```
ubuntu@master:~/tableapp/output/medium_result$ head -n 50 part-1c46a35f-1810-41a8-a43f-b7976276780c-task-0-file-0
18,19,0.2102803738317757
18,20,0.01265822784810127
18,26,0.218023258139535
18,53,0.0213903743315568
18,113,0.04166666666666667
18,196,0.00636942675159236
18,238,0.02762438939226519
18,252,0.03144654088056314
18,263,0.01273885358318471
18,297,0.10465116279069767
18,301,0.12149532719280374
18,386,0.0119047619847619
18,415,0.03067484662576687
18,420,0.02259887005649718
18,424,0.08188168108108189
18,428,0.11494252873563218
18,534,0.01881881801801802
18,579,0.00581395348837289
18,592,0.17983938131804367
18,639,0.01212121212121212
18,654,0.0887573964979414
18,752,0.01030927835851546
18,766,0.00628938817610863
18,767,0.01657458563535912
18,771,0.02586266896551724
18,773,0.07766998291262137
18,791,0.0782122985627933
18,827,0.00966183574879227
18,855,0.01234567981234568
18,891,0.1530054644887432
18,903,0.0051020481632653
18,928,0.00888888888888889
18,935,0.0124223662484472
18,945,0.01052631578947368
18,995,0.1589958158995816
18,1003,0.00628938817610863
18,1010,0.04191616766467866
18,1081,0.0975689756897561
18,1087,0.14634146341463414
18,1090,0.05263157894736842
18,1189,0.00613496932515337
18,1192,0.028108858251256281
18,1198,0.0284090909090909
18,1384,0.16595744688851863
18,1591,0.01734104046242775
18,1972,0.0654761984761984
18,2044,0.01204819277108434
18,2118,0.01219512195121951
18,2178,0.05605381165919283
18,2193,0.01142857142857143
```

```
ubuntu@master:~/tableapp/output/medium_result$ wc -l medium_output
64195979 medium_output
```

为了确认结果文件的数据量的大小，改名后，使用如下命令查看结果文件的行数：

wc -l medium_output

文件行数，共 64195979 行，数量正确，实验结果输出无误，实验成功。