www.startdataengineering.com /post/how-to-schedule-a-sql-script-using-apache-airflow-with-an-example/

Scheduling a SQL script, using Apache Airflow, with an example

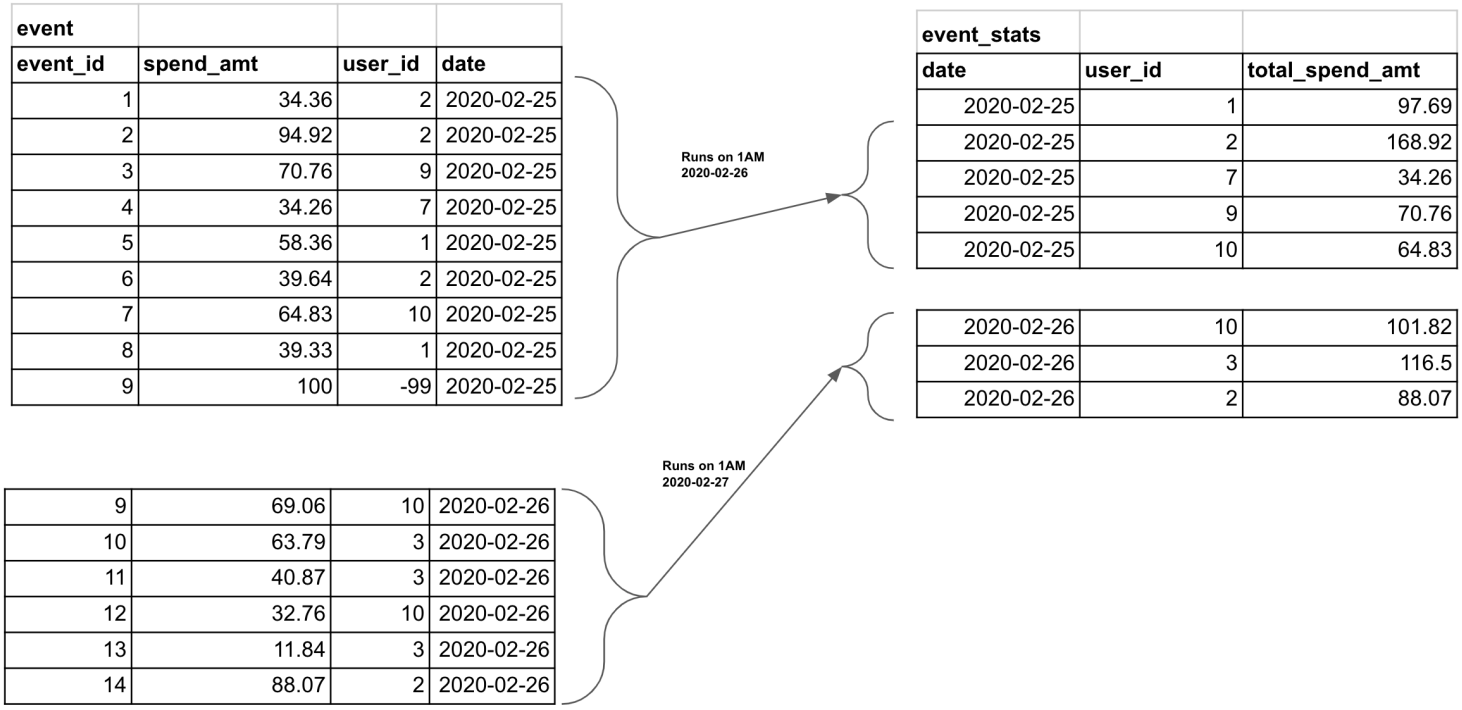
4-5 minutes : 3/29/2020

Mar 29, 2020 · 3 min read

One of the most common use cases for Apache Airflow is to run scheduled SQL scripts. Developers who start with Airflow often ask the following questions

> "How to use airflow to orchestrate sql?"
>
> "How to specify date filters based on schedule intervals in Airflow?"

This post aims to cover the above questions. This post assumes you have a basic understanding of Apache Airflow and SQL. Let's see how we can schedule a SQL script using Airflow, with an example. Let's assume we want to run a SQL script every day at midnight. This SQL script performs data aggregation over the previous day's data from event table and stores this data in another event_stats table.



We can use Airflow to run the SQL script every day. In this example we use MySQL, but airflow provides operators to connect to most databases.

DAG: Directed Acyclic Graph, In Airflow this is used to denote a data pipeline which runs on a scheduled interval. A DAG can be made up of one or more individual tasks.

```
from airflow import DAG
from airflow.operators.mysql_operator import MySqlOperator
```

```
default_arg = {'owner': 'airflow', 'start_date': '2020-02-28'}

dag = DAG('simple-mysql-dag',
          default_args=default_arg,
          schedule_interval='0 0 * * *')

mysql_task = MySqlOperator(dag=dag,
                           mysql_conn_id='mysql_default',
                           task_id='mysql_task'
                           sql='<path>/sample_sql.sql',
                           params={'test_user_id': -99})

mysql_task
```

In the above script

1. `0 0 * * *` is a cron schedule format, denoting that the DAG should be run everyday at midnight, which is denoted by the 0th hour of every day. (note that Airflow by default runs on UTC time) mysql_conn_id is the connection id for your SQL database, you can set this in admin -> connections from airflow UI. There you will set the username and password that Airflow uses to access your database.
2. The SQL script to perform this operation is stored in a separate file sample_sql.sql. This file is read in when the DAG is being run.

```
USE your_database;

DROP TABLE IF EXISTS event_stats_staging;
CREATE TABLE event_stats_staging
AS SELECT date
        , user_id
        , sum(spend_amt) total_spend_amt
     FROM event
    WHERE date = {{ macros.ds }}
      AND user_id <> {{ params.test_user_id }}
    GROUP BY date, user_id;

   INSERT INTO event_stats (
          date
        , user_id
        , total_spend_amt
   )
   SELECT date
        , user_id
        , total_spend_amt
```

```
    FROM event_stats_staging;

DROP TABLE event_stats_staging;
```

The values within {{ }} are called templated parameters. Airflow replaces them with a variable that is passed in through the DAG script at run-time or made available via Airflow metadata macros. This may seem like overkill for our use case. But it becomes very helpful when we have more complex logic and want to dynamically generate parts of the script, such as where clauses, at run time.

There are 2 key concepts in the templated SQL script shown above

1. `Airflow macros`: They provide access to the metadata that is available for each DAG run. We use the execution date as it provides the previous date over which we want to aggregate the data. ref: https://airflow.apache.org/docs/stable/macros.html
2. `Templated parameters`: If we want our SQL script to have some parameters that can be filled at run time from the DAG, we can pass them as parameters to the task. In our example we filter out a specific user_id (-99) before aggregation.

In this post we saw

1. How to schedule SQL scripts using Apache Airflow
2. How Airflow connects to the database using the connection id
3. How to pass in parameters at run-time using input parameters and macros

Hope this gives you an understanding of how to schedule SQL scripts in Airflow and how to use templating. If you have any questions or comments, please let me know below.

Previous Chapter Next Chapter