Airflow详细搭建过程（亲测 + 总结）-CSDN博客

blog.csdn.net/OldDirverHelpMe/article/details/107279735

Airflow 专栏收录该内容

2 篇文章 0 订阅

订阅专栏

Airflow详细搭建过程（亲测+总结）

官网:Apache airflow
Airflow是社区创建的一个平台，用于以编程方式编写，安排和监视工作流。
半个月前搭了一次，现在做一个记录，不逼逼，开始搭建了，全程多图：

环境准备

系统: cent os 7
conda版本: 4.8.2
airflow版本 1.10.11

开始搭建

我这边会用一个`conda`创建一个`apache airflow`的环境:

```
conda create -n airflow_env python=3.7
```

切换到当前的这个环境：

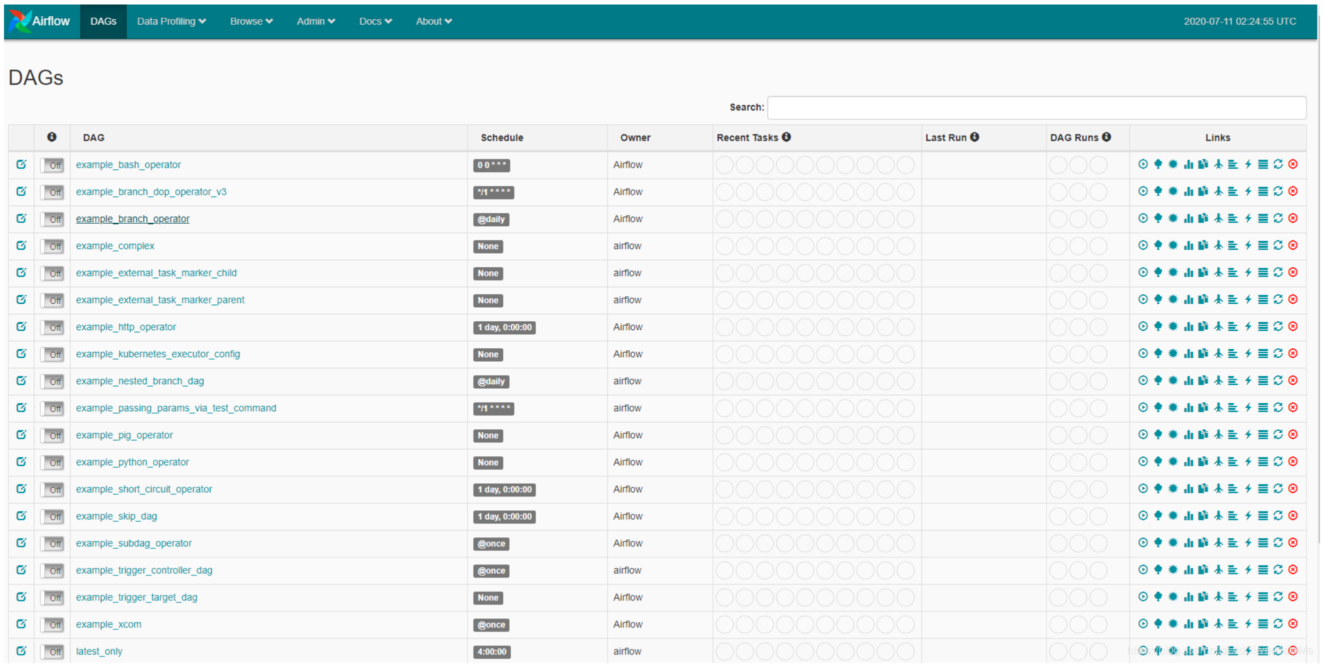
```
conda activate airflow_env
```

搭建airflow

搭建airflow的话，官网有一套详细的文档Airflow 搭建
接下来就按照这个方式来

```
# airflow needs a home, ~/airflow is the default,
# but you can lay foundation somewhere else if you prefer
# (optional)
export AIRFLOW_HOME=~/airflow

# install from pypi using pip
pip install apache-airflow

# initialize the database
airflow initdb

# start the web server, default port is 8080
# airflow webserver -p 8080    这里做个修改，后面加上-D参数让它后台运行
airflow webserver -p 8080 -D

# start the scheduler
airflow scheduler

# visit localhost:8080 in the browser and enable the example dag in the home page
```

经过上述的步骤的话，可以在浏览器中打开这个页面:



DAGs

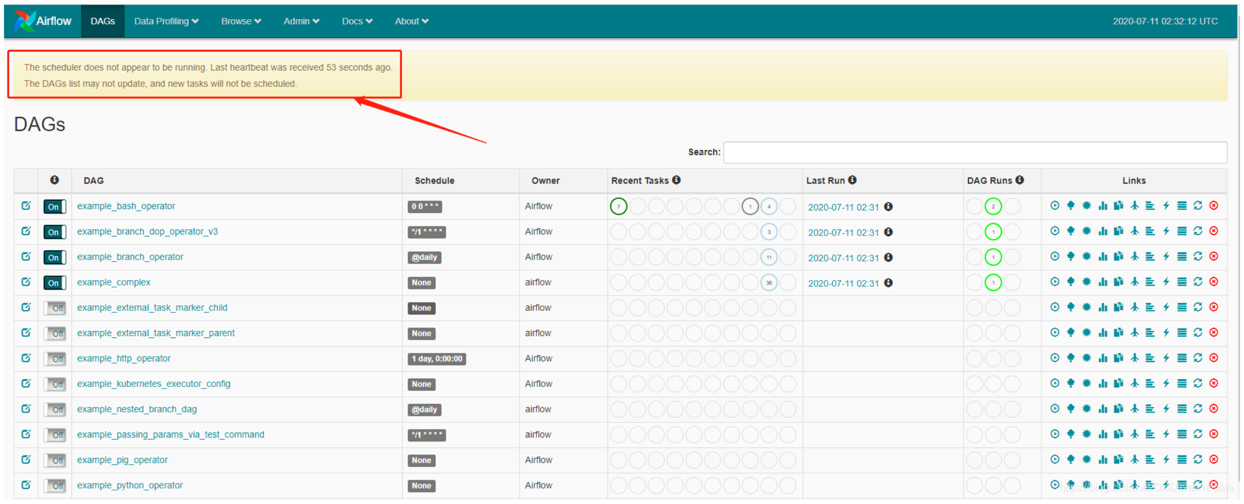
Search:

		DAG	Schedule	Owner	Recent Tasks	Last Run	DAG Runs	Links
	Off	example_bash_operator	0 0 * * *	Airflow				
	Off	example_branch_dop_operator_v3	*/1 * * * *	Airflow				
	Off	example_branch_operator	@daily	Airflow				
	Off	example_complex	None	airflow				
	Off	example_external_task_marker_child	None	airflow				
	Off	example_external_task_marker_parent	None	airflow				
	Off	example_http_operator	1 day, 0:00:00	Airflow				
	Off	example_kubernetes_executor_config	None	Airflow				
	Off	example_nested_branch_dag	@daily	airflow				
	Off	example_passing_params_via_test_command	*/1 * * * *	airflow				
	Off	example_pig_operator	None	Airflow				
	Off	example_python_operator	None	Airflow				
	Off	example_short_circuit_operator	1 day, 0:00:00	Airflow				
	Off	example_skip_dag	1 day, 0:00:00	Airflow				
	Off	example_subdag_operator	@once	Airflow				
	Off	example_trigger_controller_dag	@once	airflow				
	Off	example_trigger_target_dag	None	Airflow				
	Off	example_xcom	@once	Airflow				
	Off	latest_only	4:00:00	airflow				

搭建到这里，可以看到一个基础的样子，但是从目前而言，我能发现的有以下几个问题：

- airflow的元数据存储信息默认是使用`sqlite`进行存储。
- 页面的右上角的时间是比正常的时间晚了8个小时的

- airflow默认是用单线程调度任务的，如下图（之后就知道了）



The scheduler does not appear to be running. Last heartbeat was received 53 seconds ago.
The DAGs list may not update, and new tasks will not be scheduled.

DAGs

Search:

	ℹ	DAG	Schedule	Owner	Recent Tasks ℹ	Last Run ℹ	DAG Runs ℹ	Links
	On	example_bash_operator	0 0 * * *	Airflow	7 1 4	2020-07-11 02:31 ℹ	2	
	On	example_branch_dop_operator_v3	*/1 * * * *	Airflow	3	2020-07-11 02:31 ℹ	1	
	On	example_branch_operator	@daily	Airflow	11	2020-07-11 02:31 ℹ	1	
	On	example_complex	None	airflow	36	2020-07-11 02:31 ℹ	1	
	Off	example_external_task_marker_child	None	airflow				
	Off	example_external_task_marker_parent	None	airflow				
	Off	example_http_operator	1 day, 0:00:00	Airflow				
	Off	example_kubernetes_executor_config	None	Airflow				
	Off	example_nested_branch_dag	@daily	airflow				
	Off	example_passing_params_via_test_command	*/1 * * * *	airflow				
	Off	example_pig_operator	None	Airflow				
	Off	example_python_operator	None	Airflow				

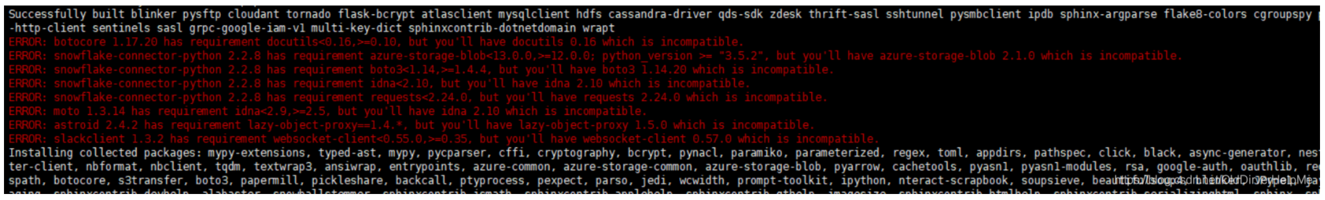
现在问题已经暴露了，那么就开始处理吧。

airflow的元数据存储默认用`sqlite`，现在切换为`mysql`

`sqlite`不支持多线程，所以我打算切换成`mysql`,`mysql`的安装可以参考:Linux centos安装mysql
`airflow`中可选很多其它的选项，那么在`airflow`里面我们可以选择一些所需的组件，例如现在需要利用`mysql`存储`airflow`的相关信息，可选的功能列表在此处airlfow的其它选项
就需要`pip install 'apache-airflow[mysql]'`
这里，我就不挑了，直接来全套插件,这样就会遇到更多的坑：

```
yum install mysql-devel gcc gcc-devel python-devel krb5-devel.x86_64 cyrus-sasl-devel
-y
pip install 'apache-airflow[all]' -i https://pypi.tuna.tsinghua.edu.cn/simple/
```
- 1
- 2

这个时候可能会遇到很多的错误，例如下面的:

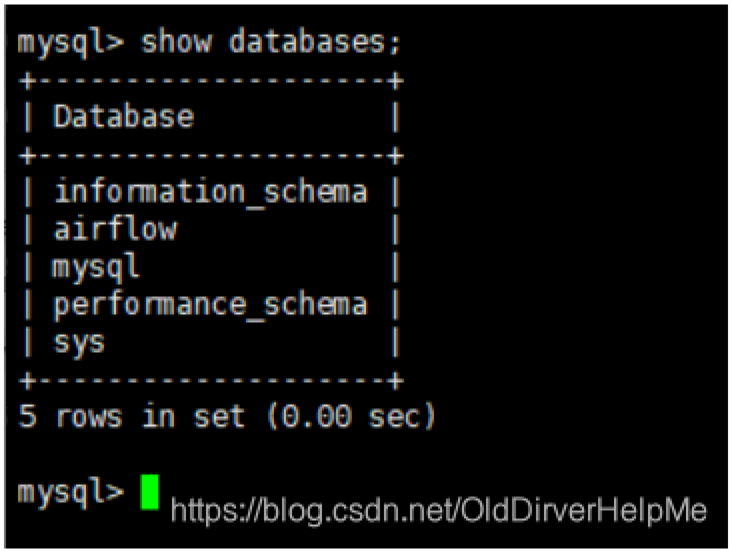


```
Successfully built blinker pysftp cloudant tornado flask-bcrypt atlasclient mysqlclient hdfs cassandra-driver qds-sdk zdesk thrift-sasl sshtunnel pysmbclient ipdb sphinx-argparse flake8-colors cgroupspy p
-http-client sentinels sasl grpc-google-iam-v1 multi-key-dict sphinxcontrib-dotnetdomain wrapt
ERROR: botocore 1.17.20 has requirement docutils<0.16,>=0.10, but you'll have docutils 0.16 which is incompatible.
ERROR: snowflake-connector-python 2.2.8 has requirement azure-storage-blob<13.0.0,>=12.0.0; python_version >= "3.5.2", but you'll have azure-storage-blob 2.1.0 which is incompatible.
ERROR: snowflake-connector-python 2.2.8 has requirement boto3<1.14,>=1.4.4, but you'll have boto3 1.14.20 which is incompatible.
ERROR: snowflake-connector-python 2.2.8 has requirement idna<2.10, but you'll have idna 2.10 which is incompatible.
ERROR: snowflake-connector-python 2.2.8 has requirement requests<2.24.0, but you'll have requests 2.24.0 which is incompatible.
ERROR: moto 1.3.14 has requirement idna<2.9,>=2.5, but you'll have idna 2.10 which is incompatible.
ERROR: astroid 2.4.2 has requirement lazy-object-proxy==1.4.*, but you'll have lazy-object-proxy 1.5.0 which is incompatible.
ERROR: slackclient 1.3.2 has requirement websocket-client<0.55.0,>=0.35, but you'll have websocket-client 0.57.0 which is incompatible.
Installing collected packages: mypy-extensions, typed-ast, mypy, pycparser, cffi, cryptography, bcrypt, pynacl, paramiko, parameterized, regex, toml, appdirs, pathspec, click, black, async-generator, nes
ter-client, nbformat, nbclient, tqdm, textwrap3, ansiwrap, entrypoints, azure-common, azure-storage-common, azure-storage-blob, pyarrow, cachetools, pyasn1, pyasn1-modules, rsa, google-auth, oauthlib, re
spath, botocore, s3transfer, boto3, papermill, pickleshare, backcall, ptyprocess, pexpect, parso, jedi, wcwidth, prompt-toolkit, ipython, nteract-scrapbook, soupsieve, beau
```

这里面的错误要有点耐心去处理一下，在这里面一般都是版本上的错误，将红字里面出现的版本卸载掉，重新指定版本号安装就好了。

配置airflow

- 配置mysql作为airflow的元数据信息的存储：
    - vim /etc/my.cnf 在[mysqld]下面添加explicit_defaults_for_timestamp=1
    - systemctl restart mysqld重启mysql
    - 在mysql中创建一个数据库，我这边的数据库的名字叫做airflow



```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| airflow            |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql>
```

    - 进入airflow的文件夹下，修改airflow.cfg。

```
# The SqlAlchemy connection string to the metadata database.
# SqlAlchemy supports many different database engine, more information
# their website
# sql_alchemy_conn = sqlite:////root/airflow/airflow.db
sql_alchemy_conn = mysql://192.168.100.120/airflow?user=root&password=12345678
```

- 解决airflow一次只能执行一个任务的问题:

```
# The executor class that airflow should use. Choices include
# SequentialExecutor, LocalExecutor, CeleryExecutor, DaskExecutor, KubernetesExecutor
# executor = SequentialExecutor
executor = LocalExecutor
```

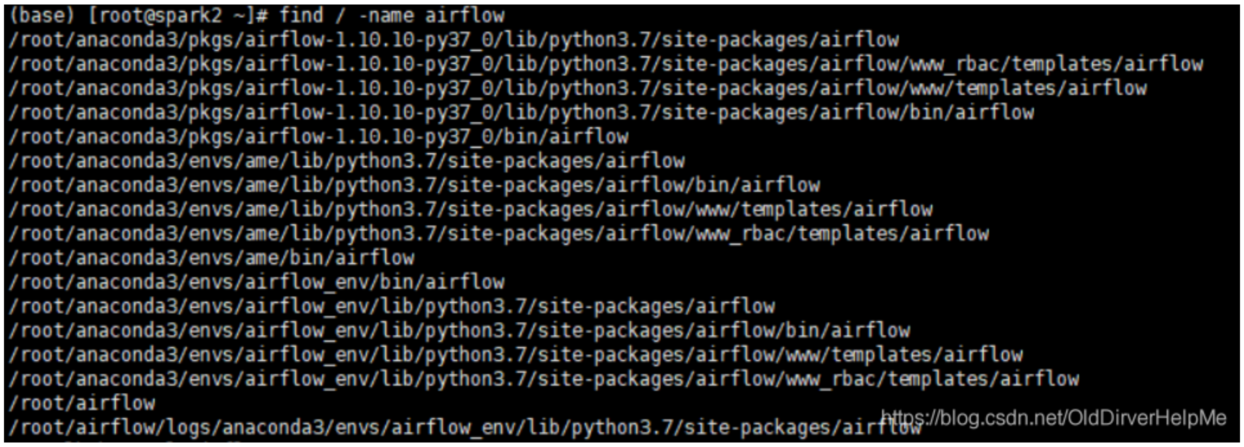
相关的依据请参考airflow 执行者

- 解决airflow时间晚了八个小时的问题
  修改airflow.cfg的内容:

```
# Default timezone in case supplied date times are naive
# can be utc (default), system, or any IANA timezone string (e.g. Europe/Amsterdam)
# default_timezone = utc
default_timezone = Asia/Shanghai
```

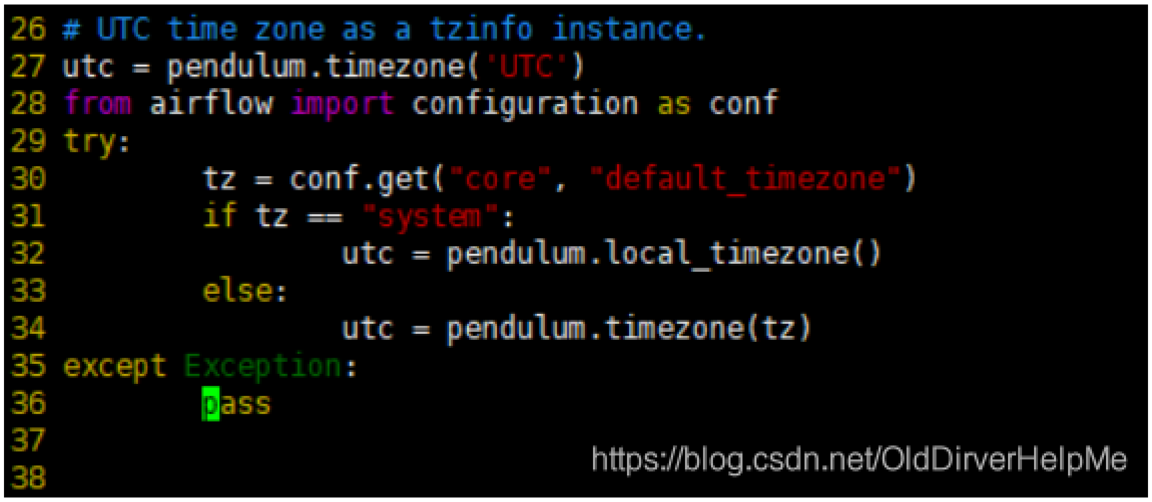
这部分的内容就要参考airflow 修改源码
具体的内容如下:

```
#找到airflow的安装位置
find / -name airflow
```
  - 1
  - 2



```
(base) [root@spark2 ~]# find / -name airflow
/root/anaconda3/pkgs/airflow-1.10.10-py37_0/lib/python3.7/site-packages/airflow
/root/anaconda3/pkgs/airflow-1.10.10-py37_0/lib/python3.7/site-packages/airflow/www_rbac/templates/airflow
/root/anaconda3/pkgs/airflow-1.10.10-py37_0/lib/python3.7/site-packages/airflow/www/templates/airflow
/root/anaconda3/pkgs/airflow-1.10.10-py37_0/lib/python3.7/site-packages/airflow/bin/airflow
/root/anaconda3/pkgs/airflow-1.10.10-py37_0/bin/airflow
/root/anaconda3/envs/ame/lib/python3.7/site-packages/airflow
/root/anaconda3/envs/ame/lib/python3.7/site-packages/airflow/bin/airflow
/root/anaconda3/envs/ame/lib/python3.7/site-packages/airflow/www/templates/airflow
/root/anaconda3/envs/ame/lib/python3.7/site-packages/airflow/www_rbac/templates/airflow
/root/anaconda3/envs/ame/bin/airflow
/root/anaconda3/envs/airflow_env/bin/airflow
/root/anaconda3/envs/airflow_env/lib/python3.7/site-packages/airflow
/root/anaconda3/envs/airflow_env/lib/python3.7/site-packages/airflow/bin/airflow
/root/anaconda3/envs/airflow_env/lib/python3.7/site-packages/airflow/www/templates/airflow
/root/anaconda3/envs/airflow_env/lib/python3.7/site-packages/airflow/www_rbac/templates/airflow
/root/airflow
/root/airflow/logs/anaconda3/envs/airflow_env/lib/python3.7/site-packages/airflow
```

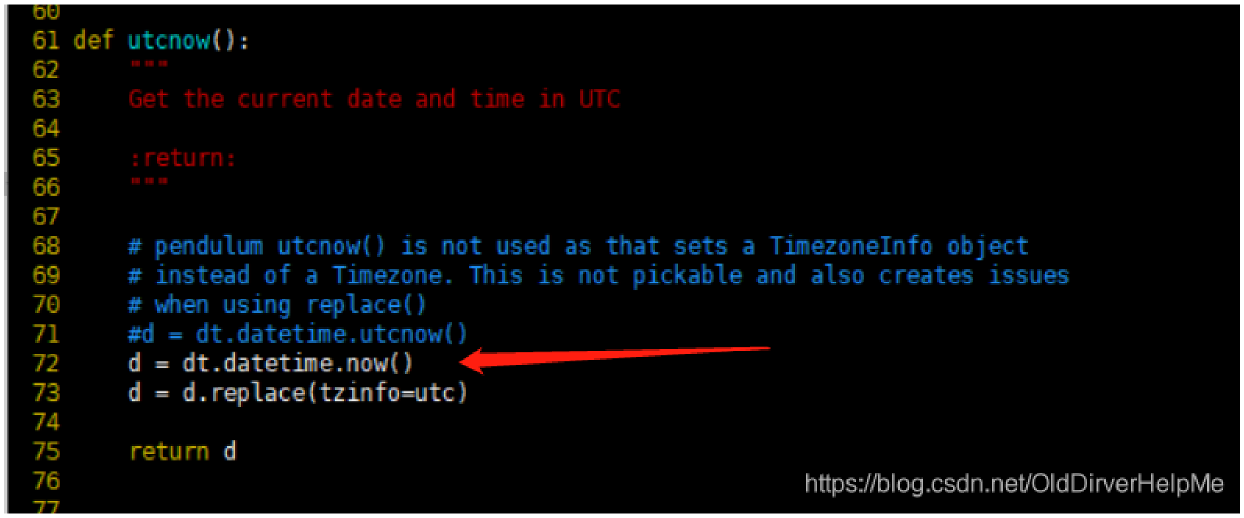
1. 修改`/root/anaconda3/envs/airflow_env/lib/python3.7/site-packages/airflow/utils/timezone.py`
   在第27行下面添加:



```
# UTC time zone as a tzinfo instance.
utc = pendulum.timezone('UTC')
from airflow import configuration as conf
try:
        tz = conf.get("core", "default_timezone")
        if tz == "system":
                utc = pendulum.local_timezone()
        else:
                utc = pendulum.timezone(tz)
except Exception:
        pass
```

```
from airflow import configuration as conf
try:
        tz = conf.get("core", "default_timezone")
        if tz == "system":
                utc = pendulum.local_timezone()
        else:
                utc = pendulum.timezone(tz)
except Exception:
        pass
```

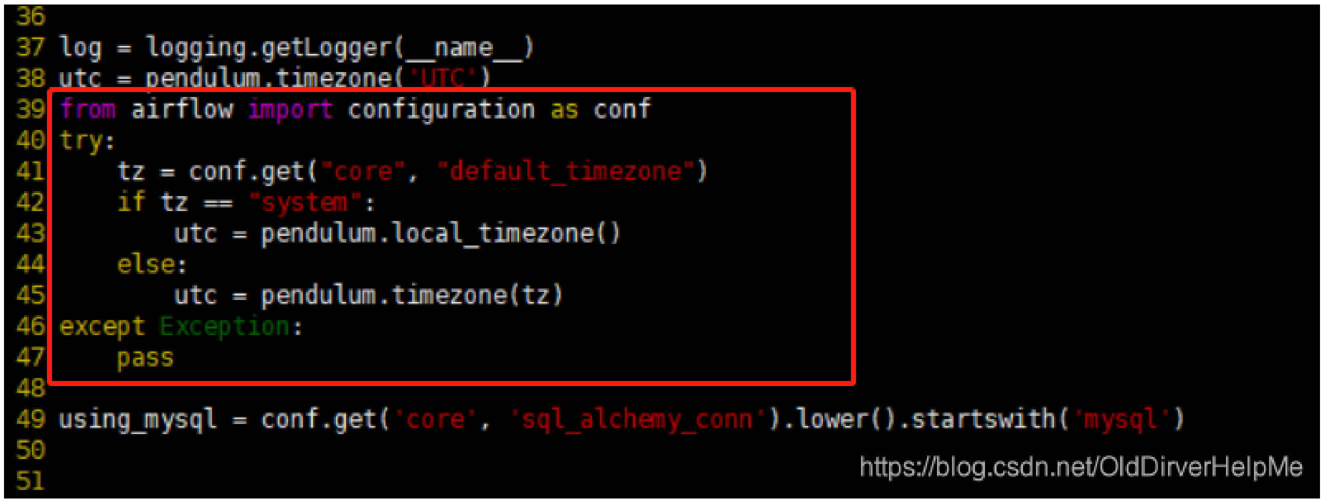
   修改 utcnow() 函数:



```
def utcnow():
    """
    Get the current date and time in UTC

    :return:
    """

    # pendulum utcnow() is not used as that sets a TimezoneInfo object
    # instead of a Timezone. This is not pickable and also creates issues
    # when using replace()
    #d = dt.datetime.utcnow()
    d = dt.datetime.now()
    d = d.replace(tzinfo=utc)

    return d
```

2. 修改`/root/anaconda3/envs/airflow_env/lib/python3.7/site-packages/airflow/utils/sqlalchemy.py`
   在第38行下面添加: `utc = pendulum.timezone('UTC')`

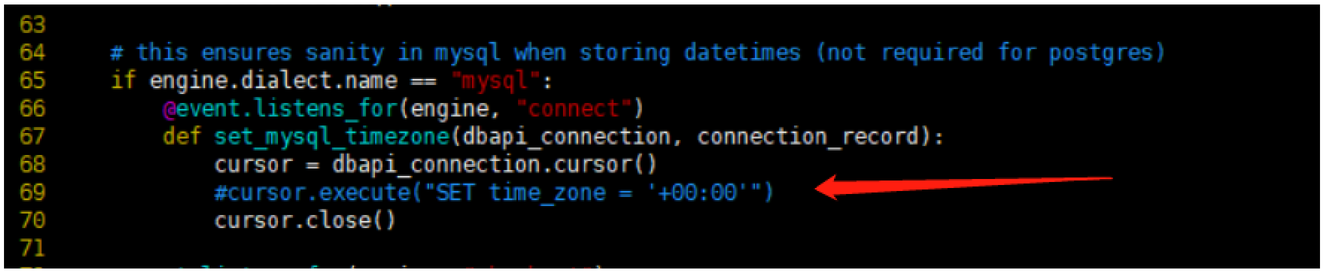
```
from airflow import configuration as conf
try:
        tz = conf.get("core", "default_timezone")
        if tz == "system":
                utc = pendulum.local_timezone()
        else:
                utc = pendulum.timezone(tz)
except Exception:
        pass
```



```

log = logging.getLogger(__name__)
utc = pendulum.timezone('UTC')
from airflow import configuration as conf
try:
    tz = conf.get("core", "default_timezone")
    if tz == "system":
        utc = pendulum.local_timezone()
    else:
        utc = pendulum.timezone(tz)
except Exception:
    pass

using_mysql = conf.get('core', 'sql_alchemy_conn').lower().startswith('mysql')

```

https://blog.csdn.net/OldDirverHelpMe

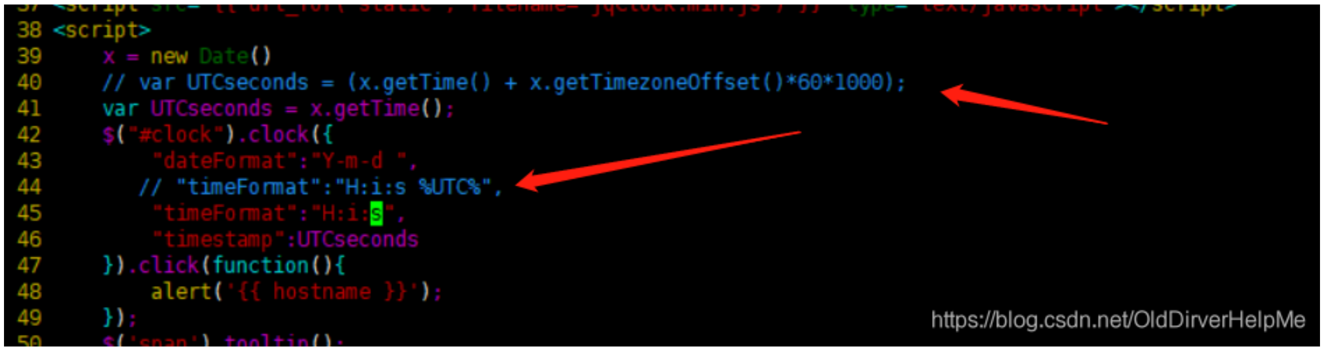
注释掉这个:



```

    # this ensures sanity in mysql when storing datetimes (not required for postgres)
    if engine.dialect.name == "mysql":
        @event.listens_for(engine, "connect")
        def set_mysql_timezone(dbapi_connection, connection_record):
            cursor = dbapi_connection.cursor()
            #cursor.execute("SET time_zone = '+00:00'")
            cursor.close()

```

3. 修改`/root/anaconda3/envs/airflow_env/lib/python3.7/site-packages/airflow/www/templates/admin/master.html`

修改下箭头所指的行，修改的内容在注释的下面



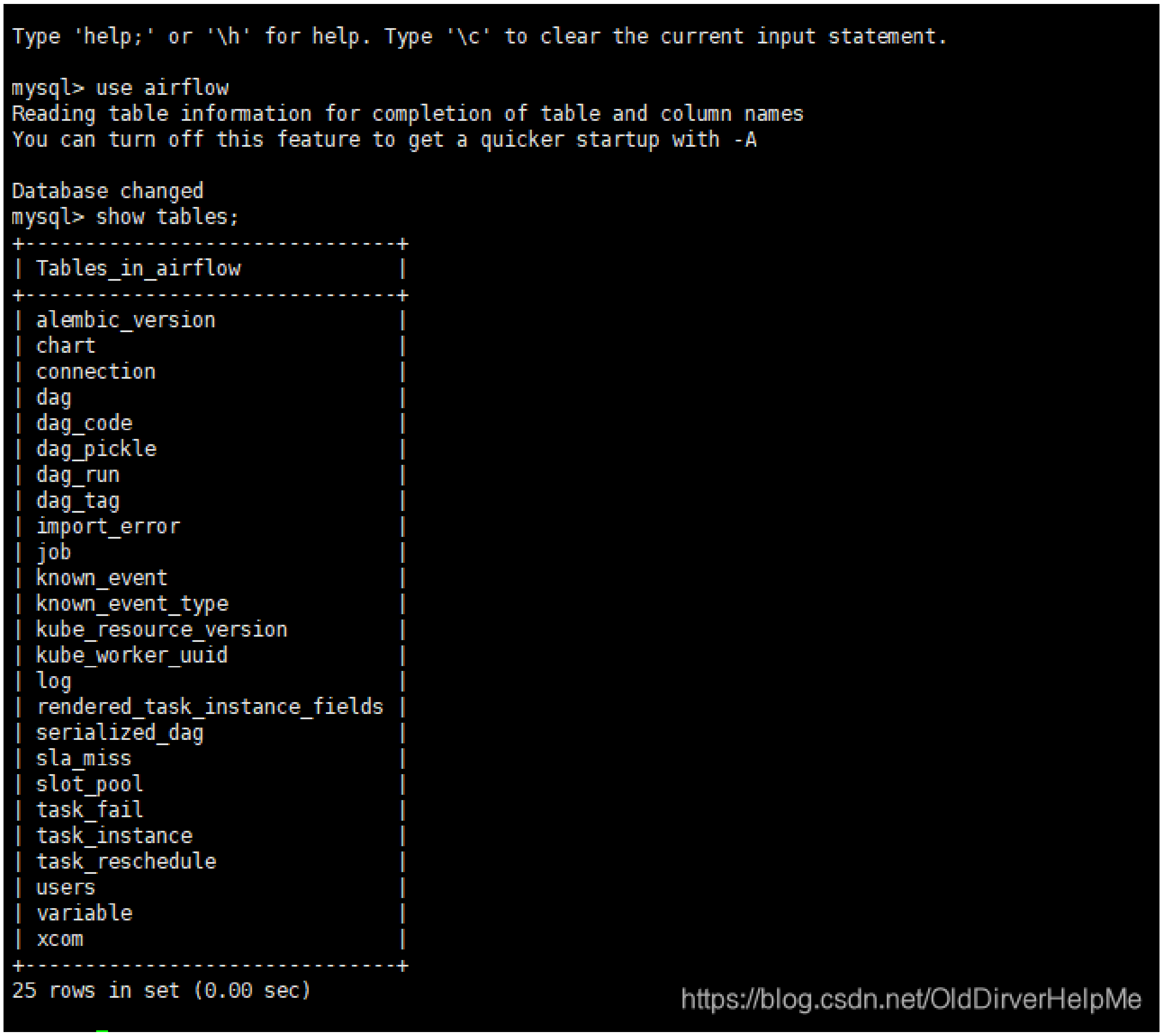
```
<script>
    x = new Date()
    // var UTCseconds = (x.getTime() + x.getTimezoneOffset()*60*1000);
    var UTCseconds = x.getTime();
    $("#clock").clock({
        "dateFormat":"Y-m-d ",
        // "timeFormat":"H:i:s %UTC%",
        "timeFormat":"H:i:s",
        "timestamp":UTCseconds
    }).click(function(){
        alert('{{ hostname }}');
    });
```

https://blog.csdn.net/OldDirverHelpMe

此时修改完毕了，这个时候重启一下 `airflow`就好了。修改源码的操作参考:airflow 修改中国时区(改airflow源码)

最后的结果如下：

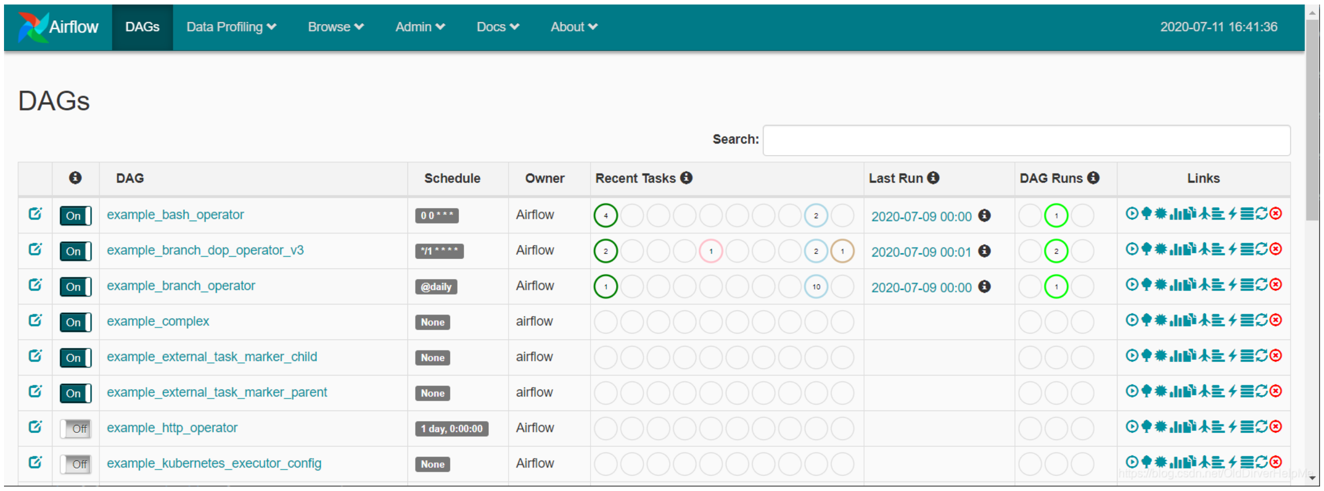
使用了`mysql`作为元数据信息的存储，在`mysql`里面可以看到:



```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use airflow
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-------------------------------+
| Tables_in_airflow             |
+-------------------------------+
| alembic_version               |
| chart                         |
| connection                    |
| dag                           |
| dag_code                      |
| dag_pickle                    |
| dag_run                       |
| dag_tag                       |
| import_error                  |
| job                           |
| known_event                   |
| known_event_type              |
| kube_resource_version         |
| kube_worker_uuid              |
| log                           |
| rendered_task_instance_fields |
| serialized_dag                |
| sla_miss                      |
| slot_pool                     |
| task_fail                     |
| task_instance                 |
| task_reschedule               |
| users                         |
| variable                      |
| xcom                          |
+-------------------------------+
25 rows in set (0.00 sec)
```

https://blog.csdn.net/OldDirverHelpMe

登录`airflow`的时候:



Airflow DAGs Data Profiling Browse Admin Docs About 2020-07-11 16:41:36

DAGs

Search:

		DAG	Schedule	Owner	Recent Tasks	Last Run	DAG Runs	Links
	On	example_bash_operator	0 0 * * *	Airflow	4 2	2020-07-09 00:00	1	
	On	example_branch_dop_operator_v3	*/1 * * * *	Airflow	2 1 2 1	2020-07-09 00:01	2	
	On	example_branch_operator	@daily	Airflow	1 10	2020-07-09 00:00	1	
	On	example_complex	None	airflow				
	On	example_external_task_marker_child	None	airflow				
	On	example_external_task_marker_parent	None	airflow				
	Off	example_http_operator	1 day, 0:00:00	Airflow				
	Off	example_kubernetes_executor_config	None	Airflow				

解决了之前提到的三个问题