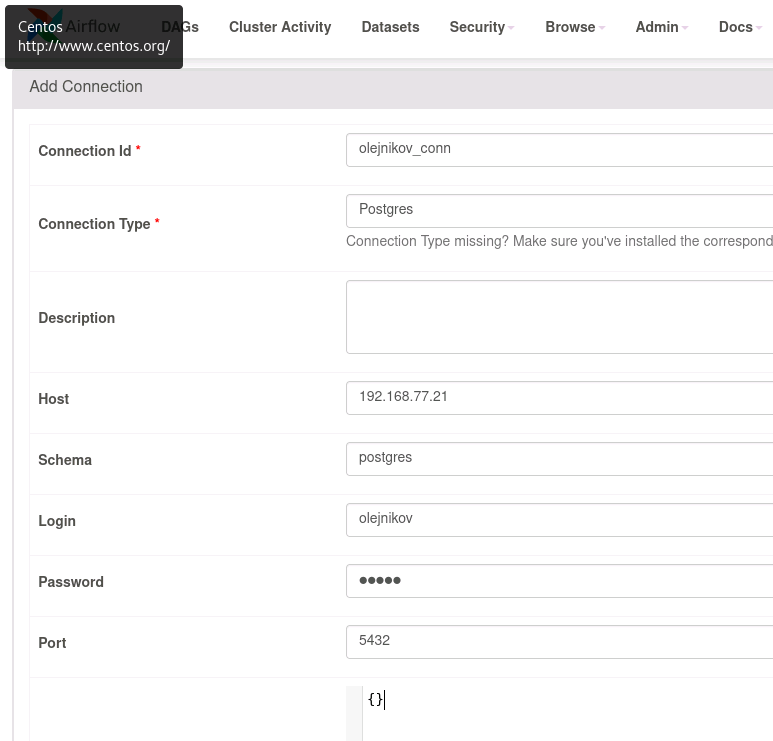
**Лабораторная работа №10**

**Олейников Михаил Николаевич (olejnikov)**

В текущей работе вам потребуется использовать продукт RT.WareHouse и RT.Streaming, который развернут в рамках нашего курса. Если появятся вопросы, пожалуйста напишите в общий учебный чат или личным сообщением преподавателю

**Цель задания**: попрактиковаться с инструментом Apache Airflow и созданием DAG файлов.

1. Используйте интерфейс Apache Airflow (http://vm-airflow.test.local:8080) и ваш логин / пароль, который вам был предоставлен на курсе.
2. Создайте ваш собственный объект “Connection” (“Соединение”) через меню интерфейса Apache Airflow: Admin -> Connections.



**ВНИМАНИЕ!**

* пожалуйста не удаляйте / не редактируйте другие объекты Connections - это может повредить выполнению работы других ребят.
* создайте свой объект Connection с указанием вашей ФИО (например yakupovas) с конфигурированием доступов к базе данных GreenPlum (ip = 192.168.77.21 port = 5432)

1. Соединитесь с виртуальной машиной Airflow (vm-airflow.test.local), используя Remmina Remote Desktop Client ( который находится на Jump Host) и с указанием вашего логина / пароля в курсе (например yakupov)
2. Перейдите под пользователя root (команда sudo -i) и соединитесь с docker-контейнером используя команду

|  |
| --- |
| docker exec -it airflow-airflow-scheduler-1 bash |

и войдите в директорию dags.

1. Находясь внутри docker-контейнера, напишите Python скрипт (DAG) который будет состоять из одного **PythonOperator** и одного **PostgresOperator**. Цепочка для **DAG PostgresOperator -> PythonOperator**

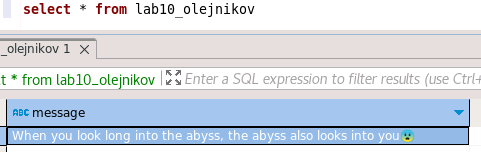
|  |
| --- |
| from datetime import datetime from airflow import DAG from airflow import configuration from airflow.operators.bash import BashOperator from airflow.utils.dates import days\_ago from airflow.operators.postgres\_operator import PostgresOperator from airflow.operators.python\_operator import PythonOperator import requests  DAG\_NAME = 'lab10\_olejnikov\_dag' GP\_CONN\_ID = 'olejnikov\_conn' MESSAGE = 'When you look long into the abyss, the abyss also looks into you' SQL\_STAT = f"insert into lab10\_olejnikov(message) values('{MESSAGE}' ||e'\U0001F628');" args = {'owner': 'olejnikov',  'start\_date': days\_ago(0),  'depends\_on\_past': False}   def send\_message(\*\*kwargs):  send\_text=f'https://api.telegram.org/bot968097013:AAGfYL\_p6CJmfcZctBN81MwEsmgZ4zeENX0/sendMessage?chat\_id=-1001915901409&parse\_mode=Markdown&text={MESSAGE}\U0001F628'  response = requests.get(send\_text)   with DAG(DAG\_NAME, description='lab10\_olejnikov',  schedule\_interval='@once',  catchup=False,  max\_active\_runs=1,  default\_args = args,  params={'labels':{'env': 'prod', 'priority': 'high'}}) as dag:   send\_to\_tg = PythonOperator(  task\_id='send\_to\_tg',  python\_callable = send\_message,  provide\_context=True)   send\_to\_postgres = PostgresOperator(  task\_id = 'send\_to\_postgres',  sql=SQL\_STAT,  postgres\_conn\_id = GP\_CONN\_ID,  autocommit=True)  send\_to\_postgres >> send\_to\_tg |

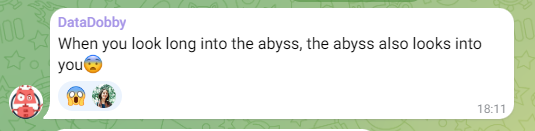
****

**PythonOperator** отправляет в Телеграмм сообщение со следующим текстом: “*ФИО: Доброе пожелание слушателям*” (Например: Якупов АШ: Всем хорошего здоровья и удачи! 🙂 ). Скрипт надо разместить по пути (внутри контейнера): /opt/airflow/dags. Чтобы создать скрипт воспользуйтесь командой nano (например *nano yakupov\_lab10.py*).

**PostgresOperator** сохраняет то же сообщение в вашу таблицу состоящая из одного столбца (message с типом данных **text**) на уровне GreenPlum (имя таблицы - lab\_10\_фамилия - например **lab\_10\_yakupov**)

PS. Добавьте любой символ emoji в сообщение и сохранением в базу данных.





Чтобы отправить сообщение в телеграмм можно воспользоваться общим ботом DataDobby и примером следующего кода.

|  |
| --- |
| import requests ... send\_text='https://api.telegram.org/bot968097013:AAGfY L\_p6CJmfcZctBN81MwEsmgZ4zeENX0/sendMessage?chat\_id=-10 01915901409&text=hello' response = requests.get(send\_text)  ... |

**(Предоставить преподавателю)** скопируйте, пожалуйста, скрипт DAG файла в гугл документ и прикрепите ссылку в анкете. Не забудьте открыть доступ по ссылке. Удачи в выполнении задания!

Пример DAG-скрипта, который мы рассмотрели на лекции:

|  |
| --- |
| from datetime import datetime from datetime import timedelta  from airflow import DAG from airflow import configuration from airflow.operators.postgres\_operator import PostgresOperator from airflow.operators.python\_operator import PythonOperator import requests  DAG\_NAME = 'my\_dag' GP\_CONN\_ID = 'GreenPlum'  SQL\_STAT = 'insert into dag\_test(id) values(100);'  args = {'owner': 'yakupov',  'start\_date': datetime(2023,8,30),  'retries': 3,  'retry\_delay': timedelta(seconds = 600)}  def start\_task(\*\*kwargs):  print('Start')  def finish\_task(\*\*kwargs): send\_text='https://api.telegram.org/bot968097013:AAGfYL\_p6CJmfcZctBN81MwEsmgZ4zeENX0/sen dMessage?chat\_id=-1001915901409&parse\_mode=Markdown&text=hello' response = requests.get(send\_text)  print('Finish')  with DAG(DAG\_NAME, description='My 1st DAG',  schedule\_interval='\* \* \* \* \*',  catchup=False,  max\_active\_runs=1,  default\_args = args,  params={'labels':{'env': 'prod', 'priority': 'high'}}) as dag:  start\_operator=PythonOperator(task\_id='start',  python\_callable = start\_task,  provide\_context=True)  finish\_operator=PythonOperator(task\_id = 'finish',  python\_callable = finish\_task,  provide\_context=True)  sql\_stat = PostgresOperator(task\_id = 'greenplum',  sql=SQL\_STAT,  postgres\_conn\_id = GP\_CONN\_ID,  autocommit=True)  start\_operator >> sql\_stat >> finish\_operator |