

Curso Data Engineer: Creando un pipeline de datos

Módulo E - Clase 8



Airflow - Dependencia DAGs

Airflow

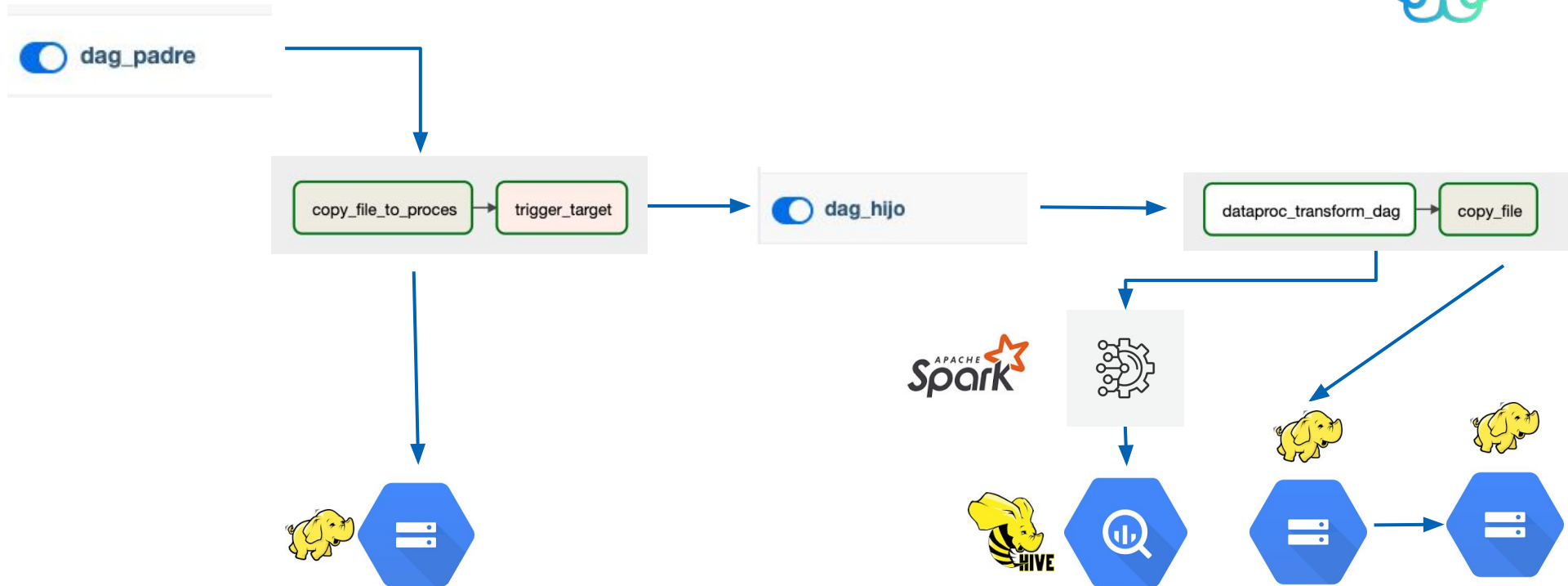


2 Dags



En casos que necesitemos ejecutar varias tareas dependientes en otros DAGs podemos utilizar el operador `TriggerDagRunOperator`, esto nos va a permitir desde un DAG padre o principal correr uno o varios DAGs hijos y esperar a que estos finalicen.

Airflow




Airflow



```
trigger_target = TriggerDagRunOperator(  
    task_id = 'trigger_target',  
    trigger_dag_id = 'dag_hijo',  
    execution_date = '{{ ds }}',  
    reset_dag_run = True,  
    wait_for_completion = True,  
    poke_interval = 30  
)
```

- **reset_dag_run:** borra (clear) la ejecución de DAG existente si ya existe. Esto es útil cuando se hace un backfill o se vuelve a ejecutar una ejecución dag existente.
- **wait_for_completion:** Espera hasta que el dag hijo finalice. (default: False)
- **poke_interval:** Revisa cada x tiempo si el DAG hijo finalizó. (default: 60)

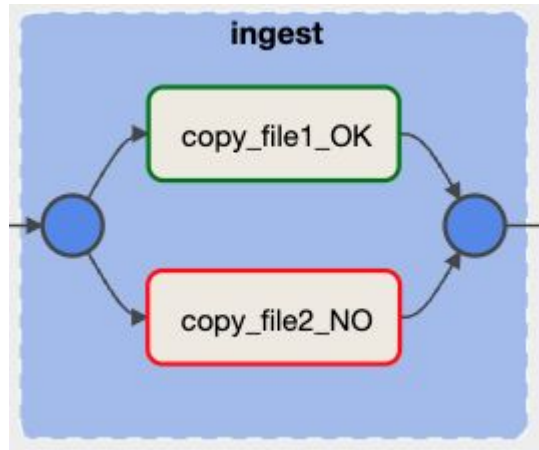


Airflow - Manejo de errores

Airflow



Task Group

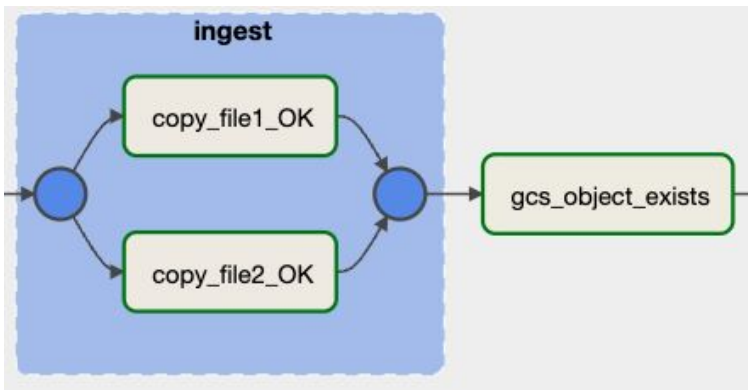


Cuando necesitamos realizar ingesta de varias fuentes en un mismo DAG, una buena práctica es agrupar las tareas, de esta manera podremos independizar los logs para una fácil lectura e identificar fácilmente si hubo un error.

Airflow



Task Group



Para controlar errores podremos hacerlo desde el mismo job, por medio de código o comandos shell.

Otra opción es agregar un sensor luego de este, para determinar si efectivamente pudo obtener las fuentes.

Airflow



DAG: task_groups

failed

Schedule: @daily

Next Run: 2022-12-07, 00:00:00

Tree Graph Calendar Task Duration Task Tries Landing Times Gantt Details <> Code

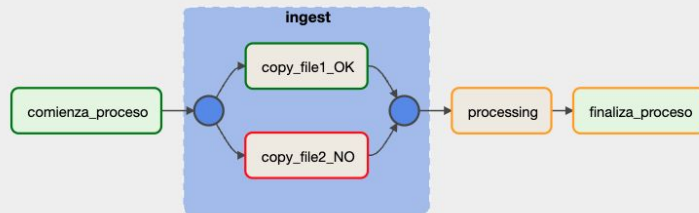
2022-12-06T00:00:01Z Runs 25 Run scheduled__2022-12-06T00:00:00+00:00 Layout Left > Right Update

Find Task...

BashOperator DummyOperator

queued running success failed up_for_retry up_for_reschedule upstream_failed skipped scheduled deferred no_status

Auto-refresh



trigger_rule: all_done

Airflow



DAG: task_groups_sensor

success

Schedule: @daily

Next Run: 2022-12-07, 00:00:00

Tree Graph Calendar Task Duration Task Tries Landing Times Gantt Details < > Code

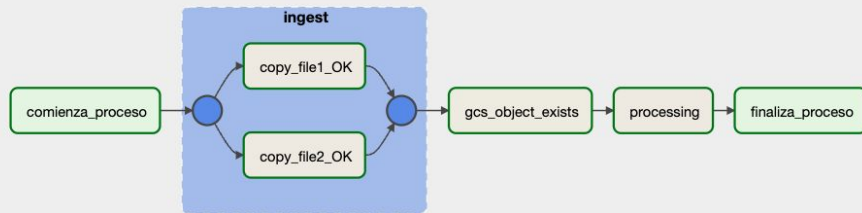
2022-12-06T00:00:01Z Runs 25 Run scheduled__2022-12-06T00:00:00+00:00 Layout Left > Right Update

Find Task...

BashOperator DummyOperator GCObjectExistenceSensor

queued running success failed up_for_retry up_for_reschedule upstream_failed skipped scheduled deferred no_status

Auto-refresh

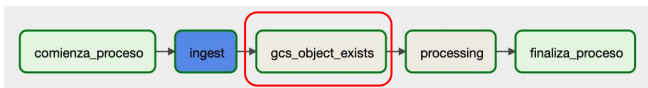


Airflow



File Sensor

Al agregar el sensor de files podemos determinar fehacientemente si los archivos ya están listos para ser procesados .



```
from airflow.contrib.sensors.file_sensor import FileSensor
```

```
from airflow.providers.google.cloud.sensors.gcs import  
GCSObjectExistenceSensor
```

```
from airflow.providers.apache.hdfs.sensors.hdfs import  
HdfsSensor
```

Airflow - envío de alertas por correo



```
1  [smtp]
2  # If you want airflow to send emails on retries, failure, and you want to use
3  # the airflow.utils.email.send_email_smtp function, you have to configure an
4  # smtp server here
5  smtp_host = your-smtp-host.com
6  smtp_starttls = True
7  smtp_ssl = False
8  # Uncomment and set the user/pass settings if you want to use SMTP AUTH
9  # smtp_user =
10 # smtp_password =
11 smtp_port = 587
12 smtp_mail_from = envio_alertas@mi_dominio.com
```

Configurar smtp en el archivo de configuración airflow.cfg

Airflow - envío de alertas por correo



```
from datetime import datetime
from airflow import DAG

default_args = {
    'owner': 'airflow',
    'start_date': datetime(2022, 1, 1),
    'email': ['alerts@mi_dominio.com'],
    'email_on_failure': True
}

with DAG('sample_dag',
        default_args=default_args,
        schedule_interval='@daily',
        catchup=False) as dag:
```

En los `default_args` configurar en `email` la casilla de correo donde queremos que lleguen las alertas y configurar `email_on_failure: True`, de esta manera al fallar un job nos enviará un correo avisando que ese job ha fallado.

Airflow - envío de alertas por slack



Name app & choose workspace ×

App Name
 22
Don't worry - you'll be able to change this later.

Pick a workspace to develop your app in:
 ▼
Keep in mind that you can't change this app's workspace later. If you leave the workspace, you won't be able to manage any apps you've built for it.
[Sign into a different workspace](#)

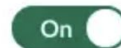
By creating a **Web API Application**, you agree to the [Slack API Terms of Service](#).

Crear una nueva APP desde Slack

Airflow - envío de alertas por slack



Activate Incoming Webhooks



[Incoming webhooks](#) are a simple way to post messages from external sources into Slack. They make use of normal HTTP requests with a JSON payload, which includes the message and a few other optional details. You can include [message attachments](#) to display richly-formatted messages.

Adding incoming webhooks requires a bot user. If your app doesn't have a [bot user](#), we'll add one for you.

Each time your app is installed, a new Webhook URL will be generated.

If you deactivate incoming webhooks, new Webhook URLs will not be generated when your app is installed to your team. If you'd like to remove access to existing Webhook URLs, you will need to [Revoke All OAuth Tokens](#).

Activar entradas desde Webhooks

Airflow - envío de alertas por slack



https://hooks.slack.com/services/T****/B****/*****

Copiar la URL del webhook

Airflow - envío de alertas por slack



Add Connection

Conn Id *	slack_webhook
Conn Type *	HTTP <small>Conn Type missing? Make sure you've installed the corresponding Airflow Provider Package.</small>
Description	
Host	https://hooks.slack.com/services/
Schema	
Login	
Password	*****
Port	
Extra	

Save

Agregar una conexión en Airflow, conn type = HTTP, ingresar el host <https://hooks.slack.com/services/> y la password la parte con el formato T00000000/B00000000/XXXXXXXXXXXXXXXXXXXXXXXXXXXX

Airflow - envío de alertas por slack



```
from airflow.providers.slack.operators.slack_webhook import SlackWebhookOperator

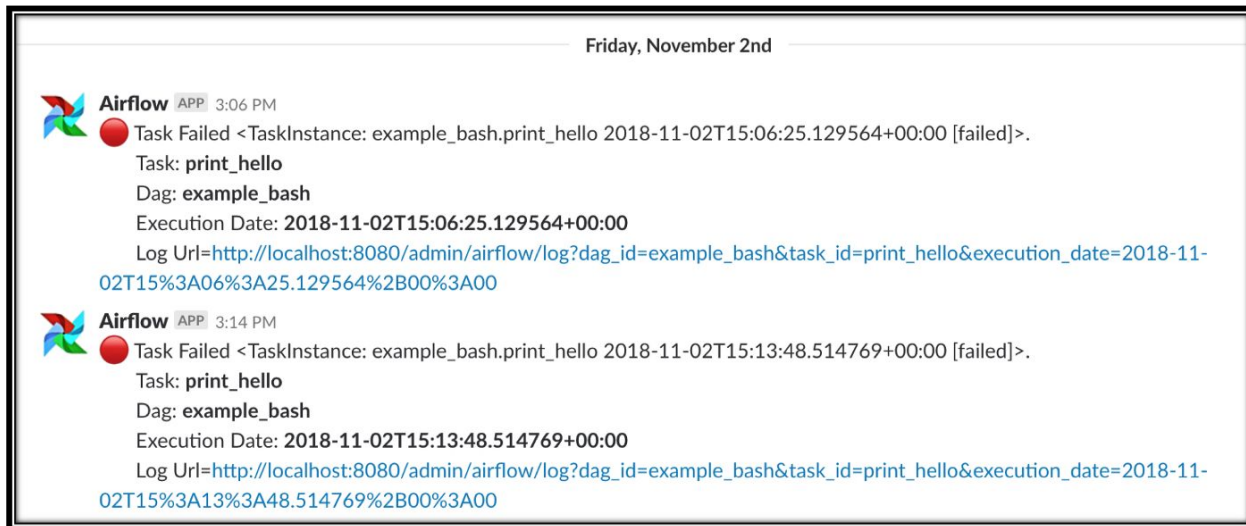
def slack_notification(context):
    slack_msg = """
        :red_circle: Task Failed.
        *Task*: {task}
        *Dag*: {dag}
        *Execution Time*: {exec_date}
        *Log Url*: {log_url}
    """.format(
        task=context.get('task_instance').task_id,
        dag=context.get('task_instance').dag_id,
        ti=context.get('task_instance'),
        exec_date=context.get('execution_date'),
        log_url=context.get('task_instance').log_url,
    )

    failed_alert = SlackWebhookOperator(
        task_id='slack_notification',
        http_conn_id='slack_webhook',
        message=slack_msg)

    return failed_alert.execute(context=context)
```

Luego en el DAG configurar los parámetros que queremos enviar a Slack

Airflow - envío de alertas por slack





Cloud Datawarehouse

BIG QUERY



Google Cloud

data-toc-fede

Search (/) for resources, docs, products, and more

Search

14

?

:

F

Explorer

+ ADD

IK

Type to search

?

Viewing workspace resources.

SHOW STARRED ONLY

data-toc-fede

☆

:

Untitled

×

+

Home


Info

Help

Fullscreen

Welcome to your SQL Workspace!

Get started



Try the Google Trends Demo Query

This simple query generates the top search terms in the US from the Google Trends public dataset.


[OPEN THIS QUERY](#)

[VIEW DATASET](#)

[COMPOSE A NEW QUERY](#)

[ADD DATA](#)


Add your own data



Local file

Upload a local file


[LAUNCH THIS GUIDE](#)



Google Drive

Google storage service

[LAUNCH THIS GUIDE](#)



Google Cloud Storage

Google object storage service

[LAUNCH THIS GUIDE](#)

☒ Show welcome page on startup

PERSONAL HISTORY

PROJECT HISTORY

REFRESH

Show debug panel



Preguntas ?



Gracias