

CS2032 - Cloud Computing (Ciclo 2024-2)

Orquestar pipelines de datos

Semana 15 - Taller 1: Apache Airflow

ELABORADO POR: GERALDO COLCHADO

Con apoyo de Asistente de Cátedra y Laboratorio:

- Sofía García (sofia.garcia@utec.edu.pe)

Contenido

Orquestar pipelines de datos

1. **Objetivo del taller 1**
2. Concepto: Pipelines de Datos
3. Apache Airflow
4. Ejercicio 1: Instalar Apache Airflow
5. Ejercicio 2: ETL con data ficticia
6. Ejercicio 3: ETL de MySQL a S3
7. Ejercicio propuesto
8. Cierre

Orquestar pipelines de datos

Objetivo del Taller 1

- Aprender qué es un Pipeline de Datos
- Aprender a Orquestar un Pipeline de Datos ETL

Contenido

Orquestar pipelines de datos

1. Objetivo del taller 1
2. **Concepto: Pipelines de Datos**
3. Apache Airflow
4. Ejercicio 1: Instalar Apache Airflow
5. Ejercicio 2: ETL con data ficticia
6. Ejercicio 3: ETL de MySQL a S3
7. Ejercicio propuesto
8. Cierre

Concepto

Pipelines de Datos y Orquestación

*“Una canalización de datos (o **pipeline de datos**) es el medio que permite que los datos viajen desde una ubicación a otra”*

*“La **orquestación de pipelines** es el proceso de definir y automatizar una secuencia de tareas y sus dependencias para completar un proceso específico”*

Contenido

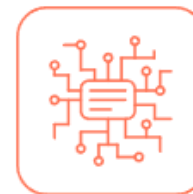
Orquestar pipelines de datos

1. Objetivo del taller 1
2. Concepto: Pipelines de Datos
3. **Apache Airflow**
4. Ejercicio 1: Instalar Apache Airflow
5. Ejercicio 2: ETL con data ficticia
6. Ejercicio 3: ETL de MySQL a S3
7. Ejercicio propuesto
8. Cierre

Apache Airflow Plataforma

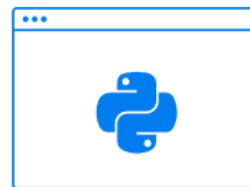
Apache Airflow®

Apache Airflow® is a platform created by the community to **programmatically author, schedule and monitor workflows**.



Dynamic

Apache Airflow® **pipelines are defined in Python**, allowing for dynamic pipeline generation. This allows for writing code that instantiates pipelines dynamically.



Pure Python

No more command-line or XML black-magic! **Use standard Python features to create your workflows**, including date time formats for scheduling and loops to dynamically generate tasks. This allows you to maintain full flexibility when building your workflows.

Contenido

Orquestar pipelines de datos

1. Objetivo del taller 1
2. Concepto: Pipelines de Datos
3. Apache Airflow
4. **Ejercicio 1: Instalar Apache Airflow**
5. Ejercicio 2: ETL con data ficticia
6. Ejercicio 3: ETL de MySQL a S3
7. Ejercicio propuesto
8. Cierre

Ejercicio 1

Instalar Apache Airflow

Servicios



Apache Airflow administrado

Ejecute Apache Airflow sin aprovisionar ni administrar servidores.

- **Paso 1:** Crear “MV airflow” con plantilla de CloudFormation e ingresar.
- **Paso 2:** Ejecutar los siguientes comandos:
mkdir airflow
cd airflow
curl -LfO 'https://airflow.apache.org/docs/apache-airflow/2.10.3/docker-compose.yaml'
mkdir -p ./dags ./logs ./plugins ./config
echo -e "AIRFLOW_UID=\$(id -u)" > .env
docker compose up airflow-init
docker compose up -d
- **Paso 3:** Esperar unos 2 minutos e ingresar a la interfaz web reemplazando la IP y usando usuario airflow y password airflow: http://IP:8080

Contenido

Orquestar pipelines de datos

1. Objetivo del taller 1
2. Concepto: Pipelines de Datos
3. Apache Airflow
4. Ejercicio 1: Instalar Apache Airflow
5. **Ejercicio 2: ETL con data ficticia**
6. Ejercicio 3: ETL de MySQL a S3
7. Ejercicio propuesto
8. Cierre

Ejercicio 2

ETL con data ficticia

- **Paso 1:** Ingrese a directorio `/home/ubuntu/airflow/dags/` y cree el archivo `"etl_data_ficticia.py"` y copie el código fuente entregado por el docente:

```
pico etl_data_ficticia.py
```

- **Paso 2:** Espere aprox. unos 5 minutos para que la interfaz web muestre el DAG `etl_data_ficticia` o en su defecto reinicie los contenedores para poder visualizar el DAG creado para lo cual debe realizar:

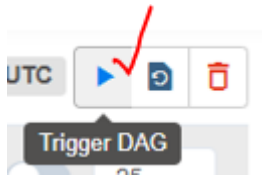
```
cd ..  
docker compose down  
docker compose up airflow-init  
docker compose up -d
```

Espere unos 2 minutos para ingresar a la interfaz web

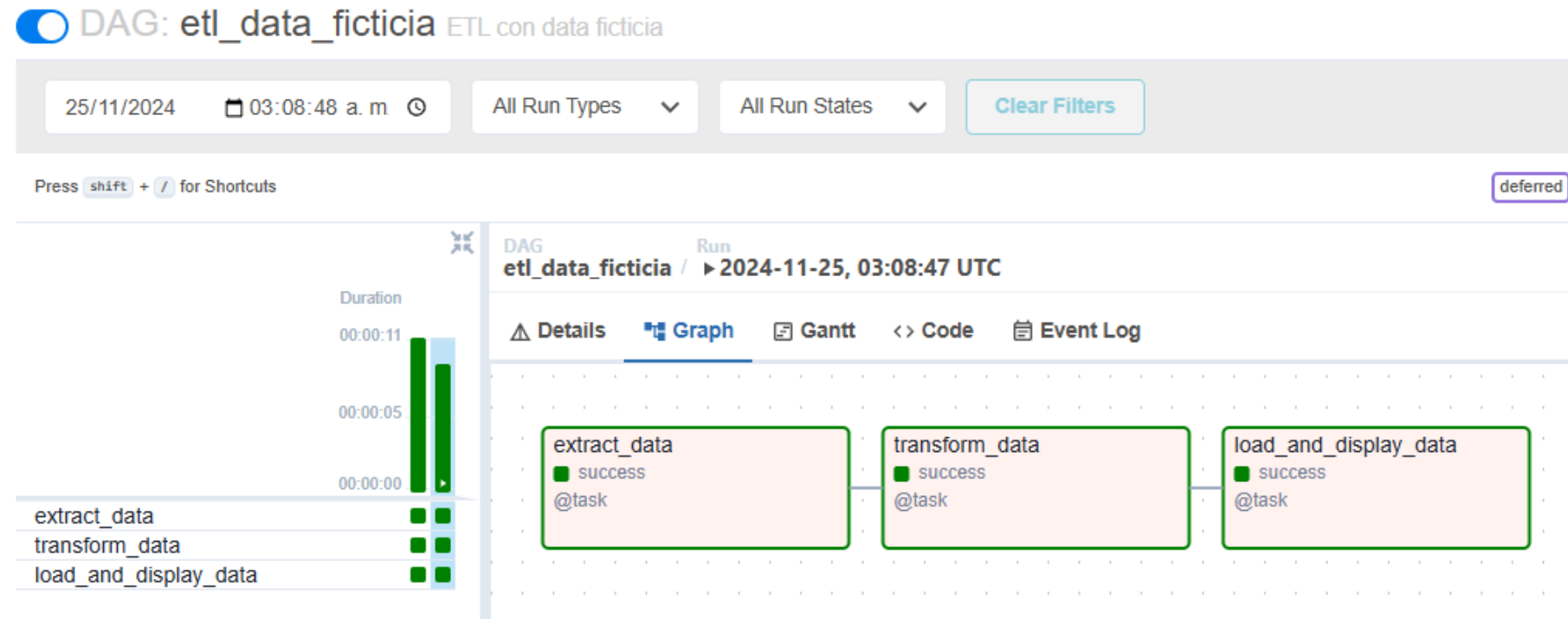
Ejercicio 2

ETL con data ficticia

- Paso 3:** Ingrese al DAG etl_data_ficticia y ejecútelo

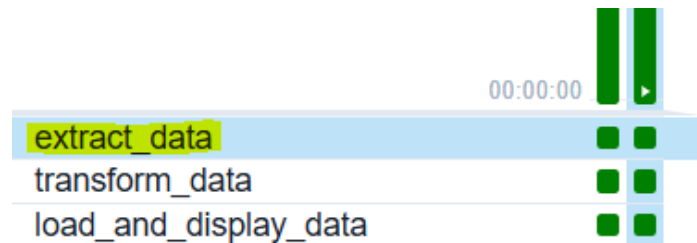


- Paso 4:** Analice

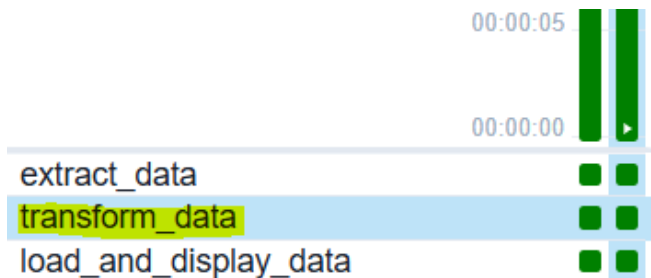


Ejercicio 2

ETL con data ficticia



```
a90a33eef29b
*** Found local files:
*** * /opt/airflow/logs/dag_id=etl_data_ficticia/run_id=manual__2024-11-25T03:08:47.8
[2024-11-25, 03:08:50 UTC] {local_task_job_runner.py:123} ► Pre task execution logs
[2024-11-25, 03:08:50 UTC] {logging_mixin.py:190} INFO - NAME AGE CITY
0 Alice 25 New York
1 Bob 30 Los Angeles
2 Charlie 35 Chicago
3 David 40 Houston
[2024-11-25, 03:08:50 UTC] {logging_mixin.py:190} INFO - Tarea 1: OK
[2024-11-25, 03:08:50 UTC] {python.py:240} INFO - Done. Returned value was: None
[2024-11-25, 03:08:50 UTC] {taskinstance.py:340} ► Post task execution logs
```



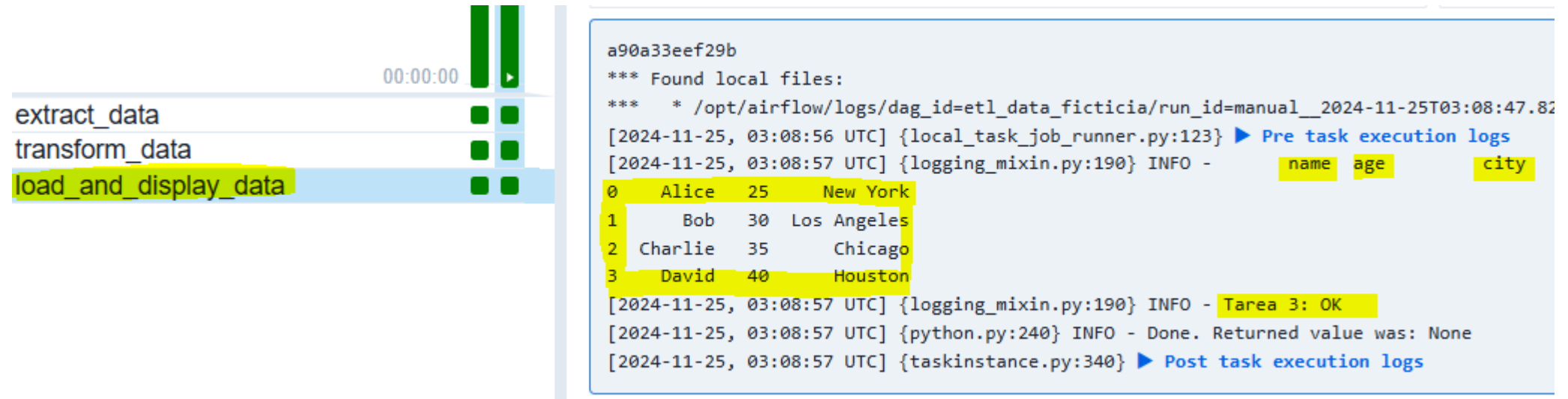
All Levels

All File :

```
a90a33eef29b
*** Found local files:
*** * /opt/airflow/logs/dag_id=etl_data_ficticia/run_id=manual__2024-11-25T03:08:47
[2024-11-25, 03:08:53 UTC] {local_task_job_runner.py:123} ► Pre task execution logs
[2024-11-25, 03:08:54 UTC] {logging_mixin.py:190} INFO - Tarea 2: OK
[2024-11-25, 03:08:54 UTC] {python.py:240} INFO - Done. Returned value was: None
[2024-11-25, 03:08:54 UTC] {taskinstance.py:340} ► Post task execution logs
```

Ejercicio 2

ETL con data ficticia



The screenshot displays an Apache Airflow DAG with three tasks: `extract_data`, `transform_data`, and `load_and_display_data`. The `load_and_display_data` task is currently running, indicated by a green bar and a play button icon. A timer shows 00:00:00. To the right, the task's logs are visible, showing the execution of a script that finds local files and displays a table of fictitious data.

```
a90a33eef29b
*** Found local files:
*** * /opt/airflow/logs/dag_id=etl_data_ficticia/run_id=manual__2024-11-25T03:08:47.82
[2024-11-25, 03:08:56 UTC] {local_task_job_runner.py:123} ► Pre task execution logs
[2024-11-25, 03:08:57 UTC] {logging_mixin.py:190} INFO -      name  age  city
0    Alice   25   New York
1     Bob    30  Los Angeles
2  Charlie   35   Chicago
3    David   40   Houston
[2024-11-25, 03:08:57 UTC] {logging_mixin.py:190} INFO - Tarea 3: OK
[2024-11-25, 03:08:57 UTC] {python.py:240} INFO - Done. Returned value was: None
[2024-11-25, 03:08:57 UTC] {taskinstance.py:340} ► Post task execution logs
```

Contenido

Orquestar pipelines de datos

1. Objetivo del taller 1
2. Concepto: Pipelines de Datos
3. Apache Airflow
4. Ejercicio 1: Instalar Apache Airflow
5. Ejercicio 2: ETL con data ficticia
6. **Ejercicio 3: ETL de MySQL a S3**
7. Ejercicio propuesto
8. Cierre

Ejercicio 3

ETL de MySQL a S3

- **Paso 1:** Ejecute los contenedores de MySQL

```
docker run -d --rm --name mysql_c -e MYSQL_ROOT_PASSWORD=utec -p 8005:3306 -v mysql_data:/var/lib/mysql mysql:8.0
docker run -d --rm --name adminer_c -p 8081:8080 adminer
```

- **Paso 2:** Ingrese a adminer y cree una tabla con datos

```
DROP DATABASE IF EXISTS tienda;
CREATE DATABASE tienda CHARSET utf8mb4;
USE tienda;

CREATE TABLE fabricantes (
    ID INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    NOMBRE VARCHAR(100) NOT NULL
);

INSERT INTO fabricantes(NOMBRE) VALUES('Asus');
INSERT INTO fabricantes(NOMBRE) VALUES('Lenovo');
INSERT INTO fabricantes(NOMBRE) VALUES('Hewlett-Packard');
INSERT INTO fabricantes(NOMBRE) VALUES('Samsung');
```


Ejercicio 3

ETL de MySQL a S3

- Paso 3:** Cree una conexión a MySQL en Airflow (Menú Admin/Connections)

The screenshot shows the 'Add Connection' form in the Airflow Admin interface. The form is divided into two main sections. The left section contains fields for 'Connection Id *' (filled with 'mysql_credentials'), 'Connection Type *' (filled with 'MySQL'), 'Description' (empty), and 'Host' (filled with '54.172.176.41'). The right section contains fields for 'Schema' (empty), 'Login' (filled with 'root'), 'Password' (filled with '****'), and 'Port' (filled with '8005'). At the bottom right, there are three buttons: 'Save' (highlighted with a red arrow), 'Test', and a back arrow.

Add Connection	
Connection Id *	mysql_credentials
Connection Type *	MySQL
Description	
Host	54.172.176.41
Schema	
Login	root
Password	****
Port	8005
<div>Save Test ←</div>	

Ejercicio 3

ETL de MySQL a S3

- **Paso 4:** Cree una conexión a AWS en Airflow (Menú Admin/Connections)

The screenshot shows the 'Add Connection' form in the Airflow web interface. The form is divided into two main sections: a left sidebar for general connection details and a right panel for specific AWS credentials and extra parameters.

Left Sidebar (Add Connection):

- Connection Id ***: aws_credentials
- Connection Type ***: Amazon Web Services
- Description**: (Empty text area)
- AWS Access Key ID**: ASIAY2PXOJQPBHK7H7UC

Right Panel:

- AWS Secret Access Key**: (Redacted with dots)
- Extra**: A JSON object containing an AWS session token:

```
{  "aws_session_token":  "IQoJb3JpZ21uX2VjEGMaCXVzLXd1c3QtMiJIMEYCI"}
```
- Buttons**: 'Save' (with a red arrow pointing to it), 'Test', and a back arrow.

Ejercicio 3

ETL de MySQL a S3

- **Paso 5:** Instale la librería PyMySQL en dos contenedores en ejecución de Airflow

```
docker exec -it airflow-airflow-worker-1 /bin/bash
```

```
pip install PyMySQL
```

```
docker exec -it airflow-airflow-scheduler-1 /bin/bash
```

```
pip install PyMySQL
```

Ejercicio 3

ETL de MySQL a S3

- **Paso 6:** Ingrese a directorio `/home/ubuntu/airflow/dags/` y cree el archivo `“etl_mysql_a_s3.py”` y copie el código fuente entregado por el docente previo **reemplazo del bucket_name:**

```
pico etl_mysql_a_s3.py
```

- **Paso 7:** Espere aprox. unos 5 minutos para que la interfaz web muestre el DAG `etl_mysql_a_s3` o en su defecto reinicie los contenedores para poder visualizar el DAG creado para lo cual debe realizar:

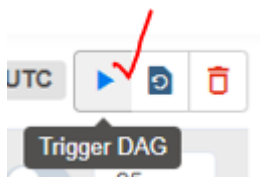
```
cd ..  
docker compose down  
docker compose up airflow-init  
docker compose up -d  
docker exec -it airflow-airflow-worker-1 /bin/bash  
pip install PyMySQL  
docker exec -it airflow-airflow-scheduler-1 /bin/bash  
pip install PyMySQL
```

Espere unos 2 minutos para ingresar a la interfaz web

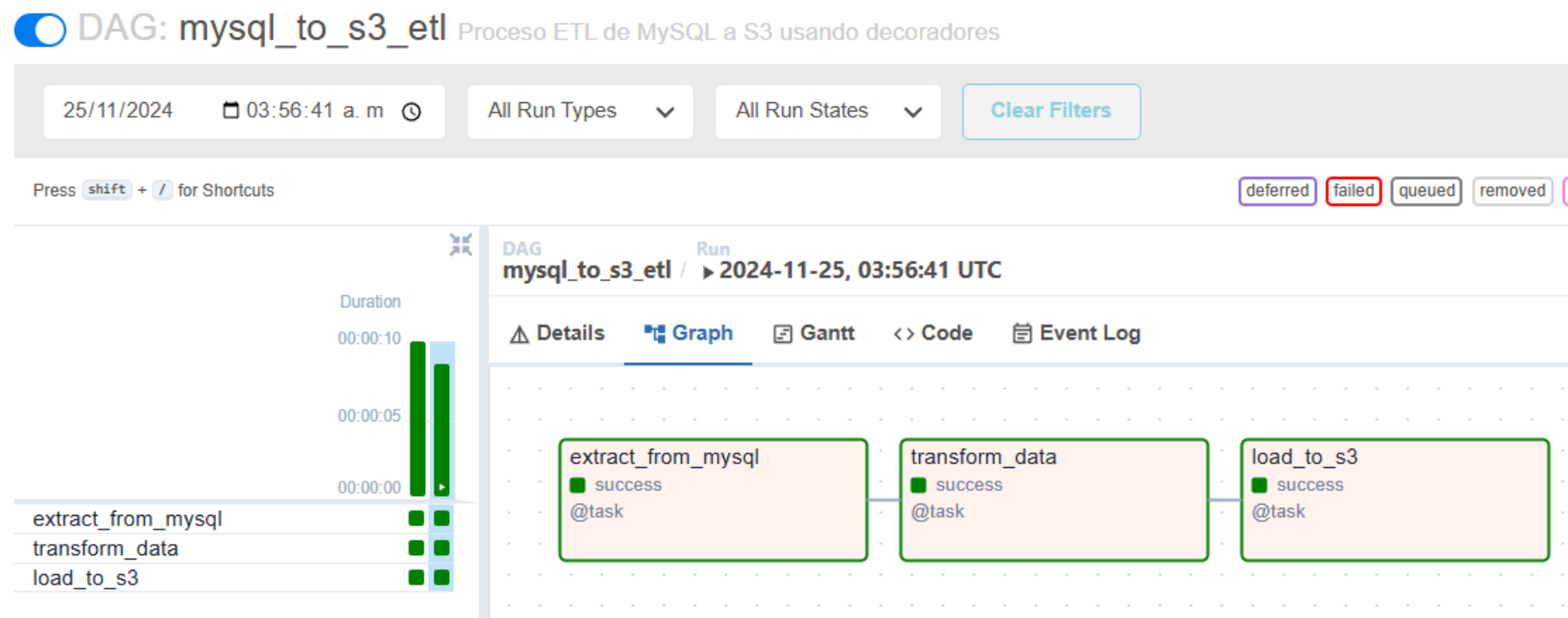
Ejercicio 3

ETL de MySQL a S3

- Paso 8:** Ingrese al DAG `etl_mysql_a_s3` y ejecútelo



- Paso 9:** Analice



Ejercicio 3

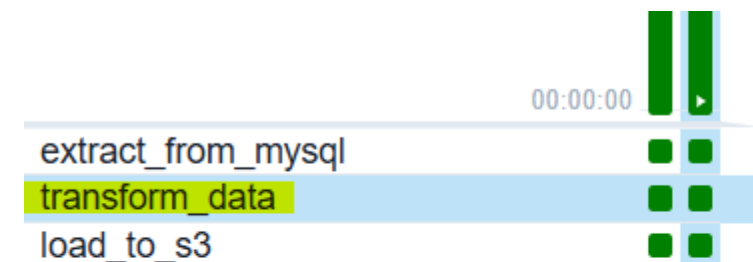
ETL de MySQL a S3

00:00:00	▶
extract_from_mysql	■ ■
transform_data	■ ■
load_to_s3	■ ■

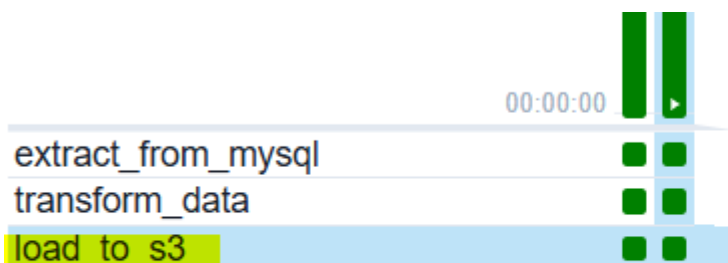
```
a90a33eef29b
*** Found local files:
***   * /opt/airflow/logs/dag_id=mysql_to_s3_etl/run_id=manual__2024-11-25T03:56:41.43871
[2024-11-25, 03:56:43 UTC] {local_task_job_runner.py:123} ▶ Pre task execution logs
[2024-11-25, 03:56:44 UTC] {base.py:84} INFO - Retrieving connection 'mysql_credentials'
[2024-11-25, 03:56:44 UTC] {warnings.py:112} WARNING - /opt/***/dags/etl_mysql_a_s3.py:56
    df = pd.read_sql(query, connection)
[2024-11-25, 03:56:44 UTC] {logging_mixin.py:190} INFO - ID NOMBRE
0 1 Asus
1 2 Lenovo
2 3 Hewlett-Packard
3 4 Samsung
[2024-11-25, 03:56:44 UTC] {logging_mixin.py:190} INFO - Tarea 1 - Extraer
[2024-11-25, 03:56:44 UTC] {python.py:240} INFO - Done. Returned value was: None
[2024-11-25, 03:56:44 UTC] {taskinstance.py:340} ▶ Post task execution logs
```

Ejercicio 3

ETL de MySQL a S3



```
a90a33eef29b
*** Found local files:
*** * /opt/airflow/logs/dag_id=mysql_to_s3_etl/run_id=manual__2024-11-25T03:56:41.43
[2024-11-25, 03:56:47 UTC] {local_task_job_runner.py:123} ▶ Pre task execution logs
[2024-11-25, 03:56:47 UTC] {logging_mixin.py:190} INFO - id nombre
0 1 Asus
1 2 Lenovo
2 3 Hewlett-Packard
3 4 Samsung
[2024-11-25, 03:56:47 UTC] {logging_mixin.py:190} INFO - Tarea 2 - Transformar a minús
[2024-11-25, 03:56:47 UTC] {python.py:240} INFO - Done. Returned value was: None
[2024-11-25, 03:56:47 UTC] {taskinstance.py:340} ▶ Post task execution logs
```



```
a90a33eef29b
*** Found local files:
*** * /opt/airflow/logs/dag_id=mysql_to_s3_etl/run_id=manual__2024-11-25T03:56:41.4387
[2024-11-25, 03:56:50 UTC] {local_task_job_runner.py:123} ▶ Pre task execution logs
[2024-11-25, 03:56:50 UTC] {base.py:84} INFO - Retrieving connection 'aws_credentials'
[2024-11-25, 03:56:51 UTC] {logging_mixin.py:190} INFO - Tarea 3 - Cargar csv a S3
[2024-11-25, 03:56:51 UTC] {python.py:240} INFO - Done. Returned value was: None
[2024-11-25, 03:56:51 UTC] {taskinstance.py:340} ▶ Post task execution logs
```

Ejercicio 3

ETL de MySQL a S3

gcr-documentos-3 Información

Objetos

Propiedades

Permisos

Métricas

Administración

Puntos de acceso

Objetos (2) Información

Copiar URI de S3

Copiar URL

Descargar

Abrir



Eliminar

Acciones

Crear carpeta

Los objetos son las entidades fundamentales que se almacenan en Amazon S3. Puede utilizar el [inventario de Amazon S3](#) para obtener una lista de objetos. Para que otras personas obtengan acceso a sus objetos, tendrá que concederles permisos de forma explícita. [Más información](#)

Buscar objetos por prefijo

	Nombre	Tipo	Última modificación	Tamaño
<input type="checkbox"/>	 transformed_data.csv	csv	24 Nov 2024 10:56:52 PM -05	54.0 B
<input type="checkbox"/>	 universidades/	Carpeta	-	-

	A	B
1	id	nombre
2	1	Asus
3	2	Lenovo
4	3	Hewlett-Packard
5	4	Samsung

Contenido

Orquestar pipelines de datos

1. Objetivo del taller 1
2. Concepto: Pipelines de Datos
3. Apache Airflow
4. Ejercicio 1: Instalar Apache Airflow
5. Ejercicio 2: ETL con data ficticia
6. Ejercicio 3: ETL de MySQL a S3
7. **Ejercicio propuesto**
8. Cierre

Ejercicio Propuesto

ETL de DynamoDB a S3

- Modifique el programa “etl_mysql_a_s3.py” a “etl_dynamodb_a_s3.py” y cambia el origen de datos de una tabla MySQL a una tabla en DynamoDB.
- Coloque todas las evidencias en un pdf y suba al padlet indicado por el docente.

Contenido

Orquestar pipelines de datos

1. Objetivo del taller 1
2. Concepto: Pipelines de Datos
3. Apache Airflow
4. Ejercicio 1: Instalar Apache Airflow
5. Ejercicio 2: ETL con data ficticia
6. Ejercicio 3: ETL de MySQL a S3
7. Ejercicio propuesto
8. **Cierre**

Cierre:

Orquestar pipelines de datos - Qué aprendimos?

- Qué es un Pipeline de Datos
- Orquestar un Pipeline de Datos ETL con Airflow

Gracias

Elaborado por docente: Geraldo Colchado