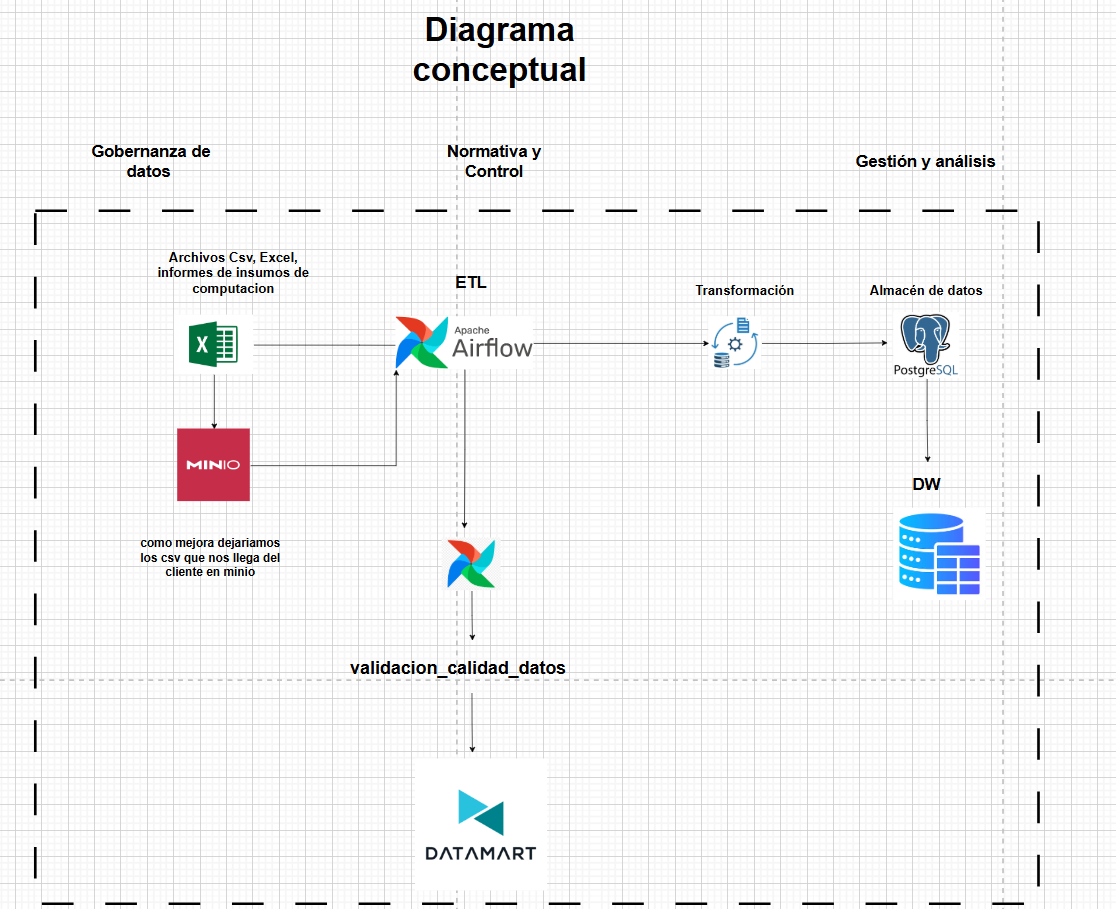
PROYECTO ON-PRIMECE :



1. COMO INSTALAR DOCKER:
2. BUSCAR COMO HACERLO CON ESTE LINK DE YOUTUBE:

<https://www.youtube.com/watch?v=d2pYQmY90tI&ab_channel=TheCoderCave%7CProgramaci%C3%B3nyTecnolog%C3%ADa>

# 

# 

# 

# **Instalación y Configuración de Apache Airflow con Docker y PostgreSQL**

## **Requisitos Previos**

Antes de iniciar, asegúrate de tener instalado:

* **Docker**: Para instalarlo, sigue la guía oficial según tu sistema operativo: Docker Installation
* **Docker Compose**: Para instalarlo, sigue la guía oficial: Docker Compose Installation

## **Paso 1: Crear el archivo docker-compose.yml**

El siguiente archivo docker-compose.yml define los servicios necesarios para ejecutar Airflow junto con PostgreSQL como backend:

version: '3.8'

services:

postgres:

image: postgres:13

container\_name: postgres\_dw

restart: always

environment:

POSTGRES\_USER: airflow

POSTGRES\_PASSWORD: airflow

POSTGRES\_DB: airflow

ports:

- "5432:5432"

volumes:

- pg\_data:/var/lib/postgresql/data

- ./init-db.sql:/docker-entrypoint-initdb.d/init-db.sql

healthcheck:

test: ["CMD-SHELL", "pg\_isready -U airflow"]

interval: 10s

retries: 5

airflow:

image: apache/airflow:2.5.1

container\_name: airflow\_dw

restart: always

depends\_on:

postgres:

condition: service\_healthy

environment:

AIRFLOW\_\_CORE\_\_EXECUTOR: LocalExecutor

AIRFLOW\_\_DATABASE\_\_SQL\_ALCHEMY\_CONN: postgresql+psycopg2://airflow:airflow@postgres/airflow

AIRFLOW\_\_CORE\_\_SQL\_ALCHEMY\_CONN: postgresql+psycopg2://airflow:airflow@postgres/airflow # Compatibilidad

AIRFLOW\_\_CORE\_\_LOAD\_EXAMPLES: 'False'

AIRFLOW\_\_API\_\_AUTH\_BACKENDS: airflow.api.auth.backend.basic\_auth

\_AIRFLOW\_WWW\_USER\_USERNAME: admin

\_AIRFLOW\_WWW\_USER\_PASSWORD: admin

ports:

- "8080:8080"

volumes:

- ./dags:/opt/airflow/dags

- ./logs:/opt/airflow/logs

- ./plugins:/opt/airflow/plugins

- ./data\_csv:/opt/airflow/data\_csv

command: >

bash -c "

airflow db init &&

airflow users create --username admin --password admin --role Admin --firstname Felix --lastname Cardenas --email admin@example.com &&

airflow webserver & airflow scheduler

"

volumes:

pg\_data:

## **Paso 2: Explicación del docker-compose.yml**

### **Servicio PostgreSQL (postgres)**

* Utiliza la imagen oficial de PostgreSQL versión 13.
* Se le asigna el nombre de contenedor postgres\_dw.
* Se establecen las credenciales de usuario airflow y contraseña airflow.
* Se mapean los puertos 5432:5432.
* Se montan volúmenes para persistir los datos de PostgreSQL y ejecutar un script de inicialización (init-db.sql).
* Se configura un healthcheck para validar que la base de datos esté lista antes de iniciar Airflow.

### **Servicio Airflow (airflow)**

* Usa la imagen oficial apache/airflow:2.5.1.
* Se le asigna el nombre de contenedor airflow\_dw.
* Se establece LocalExecutor como ejecutor.
* Se conecta a PostgreSQL mediante la variable AIRFLOW\_\_DATABASE\_\_SQL\_ALCHEMY\_CONN.
* Deshabilita la carga de ejemplos (AIRFLOW\_\_CORE\_\_LOAD\_EXAMPLES: 'False').
* Se configuran credenciales de usuario admin.
* Se mapean los puertos 8080:8080.
* Se montan volúmenes para dags, logs, plugins y data\_csv.
* Se ejecutan los siguientes comandos al iniciar:
  1. airflow db init → Inicializa la base de datos.
  2. airflow users create → Crea un usuario administrador.
  3. Inicia los servicios de Airflow: airflow webserver y airflow scheduler.

## **Paso 3: Levantar los Contenedores**

Ejecuta el siguiente comando para iniciar los contenedores:

docker-compose up -d

Esto iniciará los servicios en segundo plano.

## **Paso 4: Verificar el Estado de los Contenedores**

Para ver los contenedores en ejecución, usa:

docker ps

Si deseas ver los logs en tiempo real:

docker-compose logs -f

## **Paso 5: Acceder a Airflow Web UI**

Una vez que los servicios estén en ejecución, abre un navegador y accede a:

http://localhost:8080

Usa las credenciales configuradas:

* **Usuario**: admin
* **Contraseña**: admin

## **Paso 6: Detener y Eliminar los Contenedores**

Para detener los contenedores sin eliminarlos:

**docker-compose stop**

Para detener y eliminar todos los contenedores y volúmenes:

**docker-compose down -v**

no logueamos en este caso usamos la contraseña que definimos

airflow para usuario y contraseña.

Conexiones en Airflow

Airflow permite definir conexiones a servicios externos como bases de datos, APIs y otros recursos para que los DAGs puedan interactuar con ellos. Las conexiones son configuradas desde la interfaz de usuario de Airflow.

1. Connection Id

⦁ Descripción: Este es el identificador único para la conexión. Este Connection Id es usado para referirse a la conexión dentro de los DAGs de Airflow.

⦁ Valor: En este caso, es postgres\_dwh. Este nombre debe ser único para cada conexión.

2. Connection Type

⦁ Descripción: Define el tipo de conexión. Airflow soporta múltiples tipos de conexión como Postgres, MySQL, Http, etc.

⦁ Valor: En este caso, se selecciona Postgres de la lista desplegable, ya que la conexión es a una base de datos PostgreSQL.

3. Description

⦁ Descripción: Este campo es opcional y se utiliza para proporcionar una descripción adicional sobre la conexión.

⦁ Valor: Puedes dejarlo en blanco o escribir una breve descripción que explique el propósito de esta conexión. Ejemplo: "Conexión a la base de datos de Data Warehouse".

4. Host

⦁ Descripción: Aquí se coloca el nombre del host o la dirección IP del servidor donde se encuentra la base de datos PostgreSQL.

⦁ Valor: En este caso, se ha establecido como postgres\_dwh, que parece ser el nombre de un servicio o un alias de un servidor que Airflow puede resolver. Si tu servidor tiene una dirección IP específica o un nombre de dominio, colócalo aquí.

5. Schema

⦁ Descripción: Este campo es opcional y se utiliza para especificar el esquema de la base de datos donde se almacenarán los datos. Un esquema es una estructura que contiene tablas y objetos relacionados dentro de la base de datos.

⦁ Valor: En este caso, se ha especificado dwh, que puede ser el nombre del esquema que contiene las tablas de tu Data Warehouse.

6. Login

⦁ Descripción: Aquí debes colocar el nombre de usuario con el que Airflow se conectará a la base de datos PostgreSQL.

⦁ Valor: En este caso, se usa dwh, lo cual es el nombre de usuario configurado para acceder a la base de datos.

7. Password

⦁ Descripción: Este campo es donde se debe ingresar la contraseña asociada al nombre de usuario (Login) para autenticar la conexión a PostgreSQL.

⦁ Valor: Debes ingresar la contraseña correspondiente al usuario dwh. Asegúrate de que esté segura y de que sea correcta.

8. Port

⦁ Descripción: Este es el puerto en el que la base de datos PostgreSQL escucha las conexiones. El puerto predeterminado para PostgreSQL es 5432, que es el valor que aparece en este formulario.

⦁ Valor: Si tu base de datos usa el puerto predeterminado, puedes dejar este valor como está (5432). Si utiliza un puerto personalizado, asegúrate de cambiarlo a la configuración correcta.

9. Test Connection

⦁ Descripción: Al finalizar la configuración, es recomendable hacer clic en el botón Test Connection (que no aparece en la imagen, pero está disponible en la interfaz) para verificar si los parámetros de conexión son correctos y si Airflow puede conectarse con éxito a la base de datos.

una vez que probamos todas las configuraciones probamos el test de conexion

y luego procedemos a guardar.

1. Lista de DAGs

En la interfaz se muestran dos DAGs activos: data\_quality\_check y etl\_postgres. Los DAGs son representaciones de los flujos de trabajo que ejecutan tareas en Airflow.

2. Estado de los DAGs

⦁ data\_quality\_check: Tiene un total de 5 ejecuciones pasadas (Runs), y está configurado para ejecutarse de manera diaria (@daily). La última ejecución fue el 15 de febrero de 2025 a las 18:34:02.

⦁ etl\_postgres: Tiene 3 ejecuciones pasadas (Runs), y también está configurado para ejecutarse de manera diaria (@daily). La última ejecución fue el 15 de febrero de 2025 a las 18:33:40.

3. Acciones disponibles

⦁ Cada DAG tiene un botón de acción:

⦁ Play Button : Permite ejecutar el DAG manualmente. Al hacer clic en este botón, Airflow iniciará la ejecución del DAG de acuerdo con su programación (o inmediatamente si se ejecuta manualmente).

⦁ Delete Button : Elimina el DAG, aunque esto es poco frecuente, ya que generalmente los DAGs se mantienen en el sistema.

4. Detalles adicionales

⦁ Schedule: Los DAGs están configurados para ejecutarse todos los días (@daily), lo que significa que Airflow ejecutará cada uno de estos DAGs automáticamente una vez al día.

⦁ Last Run: Muestra la fecha y hora de la última ejecución exitosa del DAG.

⦁ Next Run: Muestra cuándo está programado para ejecutarse el DAG nuevamente, en este caso, ambos están programados para ejecutarse a las 00:00:00 del 15 de febrero de 2025.

5. Botones y filtros

⦁ Auto-refresh: Esta opción permite que la vista se actualice automáticamente para mostrar el estado más reciente de los DAGs sin necesidad de recargar la página manualmente.

⦁ Search DAGs: Un campo de búsqueda que permite filtrar los DAGs por su nombre o etiqueta.

1. Barra de navegación superior

⦁ Auto-refresh: El toggle que permite la actualización automática de la interfaz para reflejar los cambios de estado sin necesidad de recargar manualmente la página.

⦁ Schedule: El DAG está configurado para ejecutarse diariamente (@daily).

⦁ Next Run: El próximo run del DAG está programado para 2025-02-15, 00:00:00 UTC.

2. Resumen de los DAG Runs

⦁ Total Runs Displayed: Se han mostrado 3 ejecuciones del DAG.

⦁ Total success: Las 3 ejecuciones fueron exitosas.

⦁ First Run Start: La primera ejecución del DAG comenzó el 2025-02-15 a las 16:50:59 UTC.

⦁ Last Run Start: La última ejecución comenzó el 2025-02-15 a las 18:33:41 UTC.

⦁ Max Run Duration: La duración máxima de ejecución fue de 5 minutos y 14 segundos.

⦁ Mean Run Duration: La duración promedio de ejecución fue de 1 minuto y 53 segundos.

⦁ Min Run Duration: La duración mínima fue de 12 segundos.

3. Vista de los tasks del DAG (Grid)

La tabla muestra una lista de todas las tareas que componen el DAG etl\_postgres. Cada fila corresponde a una tarea y muestra las siguientes columnas:

⦁ Task Names: Los nombres de las tareas dentro del DAG, como create\_table, extract\_data\_pacientes, transform\_data\_pacientes, etc.

⦁ Duración de las tareas: Al lado de cada nombre de tarea, se visualiza la duración en un gráfico de barras. Las barras verdes indican que las tareas se ejecutaron exitosamente. En este caso, la tarea create\_table duró 5 minutos y 14 segundos, mientras que otras tareas, como extract\_data\_pacientes, transform\_data\_pacientes, etc., tienen una duración considerablemente menor.

⦁ Color de la barra: Las barras verdes indican que las tareas fueron exitosas en todas las ejecuciones. No hay tareas fallidas o en estado de error (lo cual sería indicado con colores diferentes como failed o up\_for\_retry).

4. Detalles adicionales

⦁ DAG Summary: Se resumen estadísticas sobre las tareas de este DAG:

⦁ Total Tasks: El DAG tiene un total de 10 tareas.

⦁ Las demás métricas de rendimiento son calculadas automáticamente, proporcionando una visión rápida sobre el rendimiento y la eficiencia del DAG.

explicación:

parte 1:

<https://www.youtube.com/watch?v=CB0Lcc2c7Ds&ab_channel=FelixCardenas>

parte 2:

https://www.youtube.com/watch?v=Eu46YZRw3K4&ab\_channel=FelixCardenas

luego que vemos que corrió bien el dag, revisamos una consulta si creo la tabla en sql.

git:

<https://github.com/Felo30/proyecto_airflow>

documentacion mas detallada:

https://github.com/Felo30/Proyecto-de-taligent