Pipeline de Données AWS avec Airflow et dbt

Un guide complet pour orchestrer l'extraction et la transformation de données météorologiques et COVID depuis Amazon S3 en utilisant Apache Airflow et dbt avec DuckDB.

Prérequis du Projet

Compte AWS

Avoir un compte AWS actif pour accéder aux services S3 et IAM.

Connaissances Linux

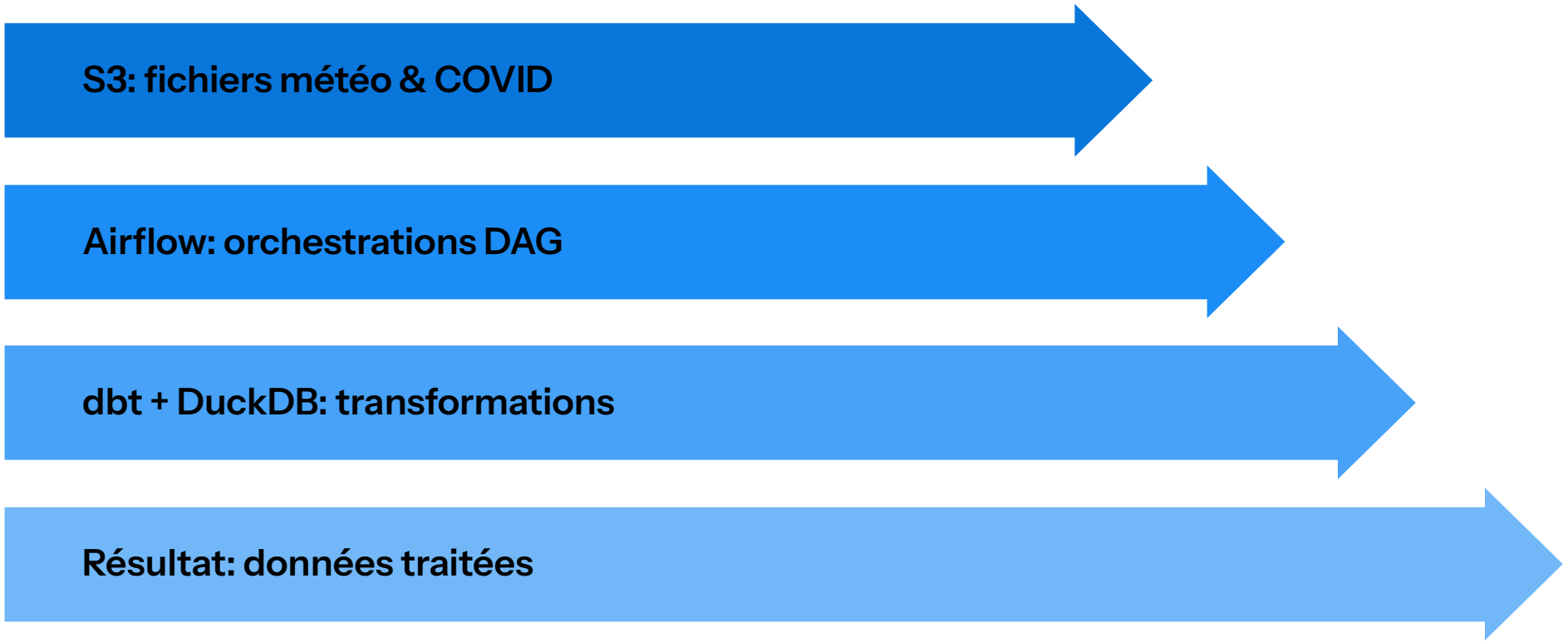
Petites notions sur Linux, nous travaillerons sur Ubuntu 24.04.

Docker et Docker Compose

Connaissance de base de Docker et docker-compose pour la conteneurisation.

Besoin d'un compte AWS ? Consultez ce guide étape par étape : Guide de création et sécurisation de compte AWS

Architecture du Pipeline

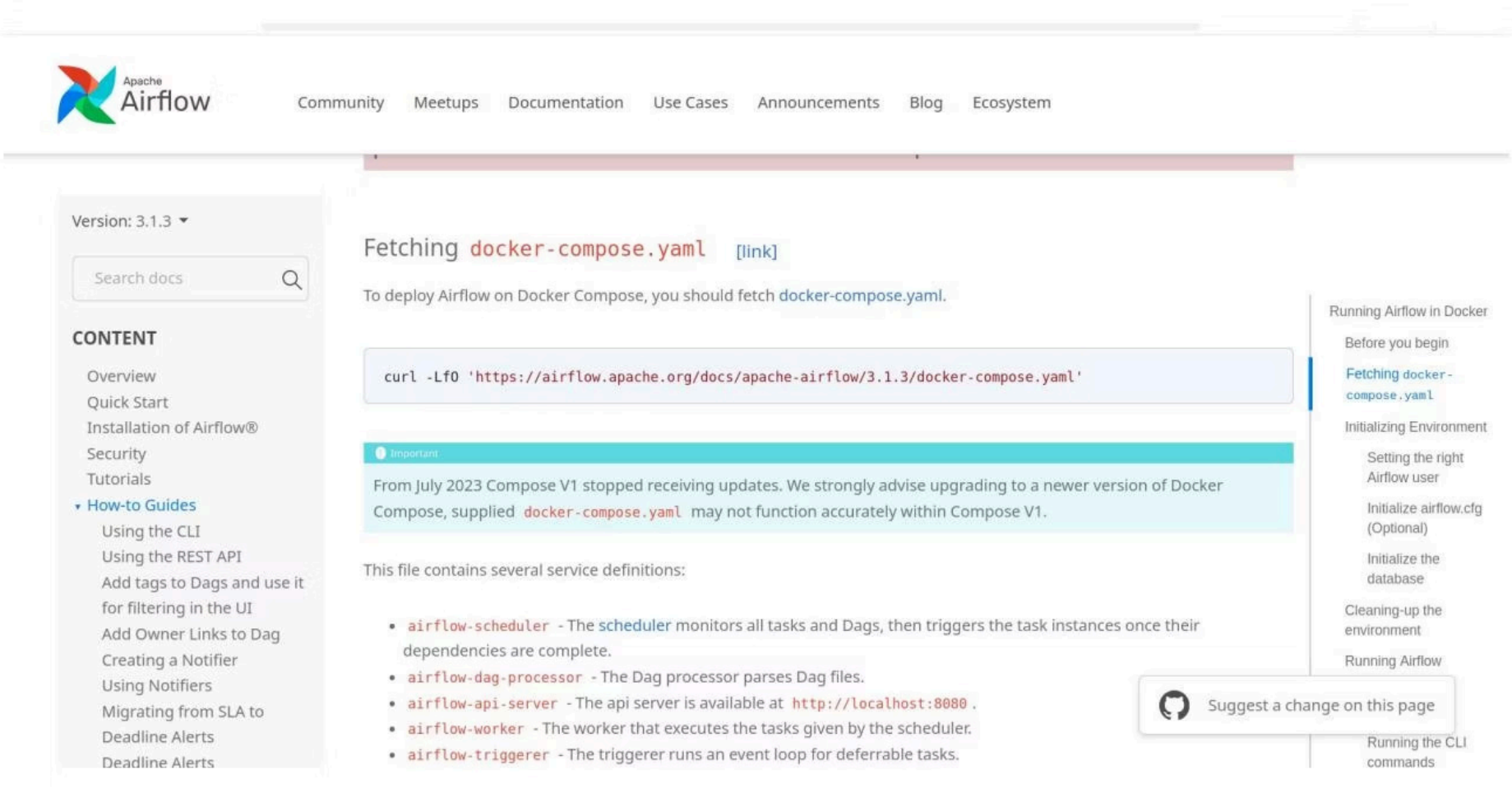


Un pipeline Airflow orchestrant des extractions de données depuis un bucket S3 de Amazon (weather/météo, COVID) et ensuite des transformations utilisant DuckDB via dbt.

Installation d'Apache Airflow

Étape 1 : Télécharger le script Docker Compose

Installation de Airflow via le script docker compose que vous pouvez trouver sur le site officiel de Airflow : [Documentation officielle Airflow]



Cliquez sur le lien en bleu dans l'image ou directement ici : **docker-compose.yml** pour télécharger le script docker compose.

Étape 2 : Créer la structure du projet

1. Créé un dossier airflow et mettre le script docker-compose.yml téléchargé, dans ce dossier
2. Créé cette structure de base à l'intérieur du dossier airflow avec cette commande :

```
mkdir -p dags plugins scripts config data/{raw,processed} tests
```

Étape 3 : Lancer les conteneurs

On se déplace dans le dossier où se trouve le script et on exécute la commande :

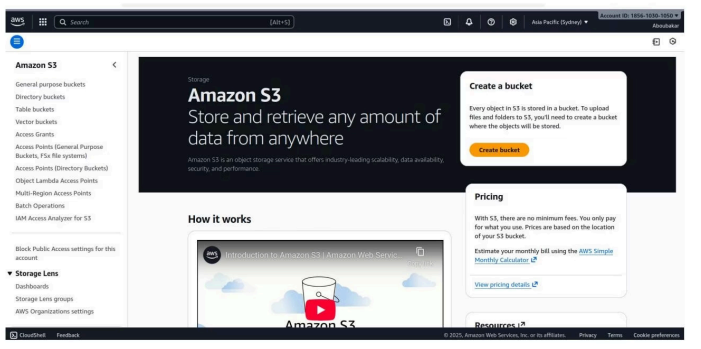
```
docker-compose up -d
```

Il installera tous les conteneurs liés à Docker et va les lancer.

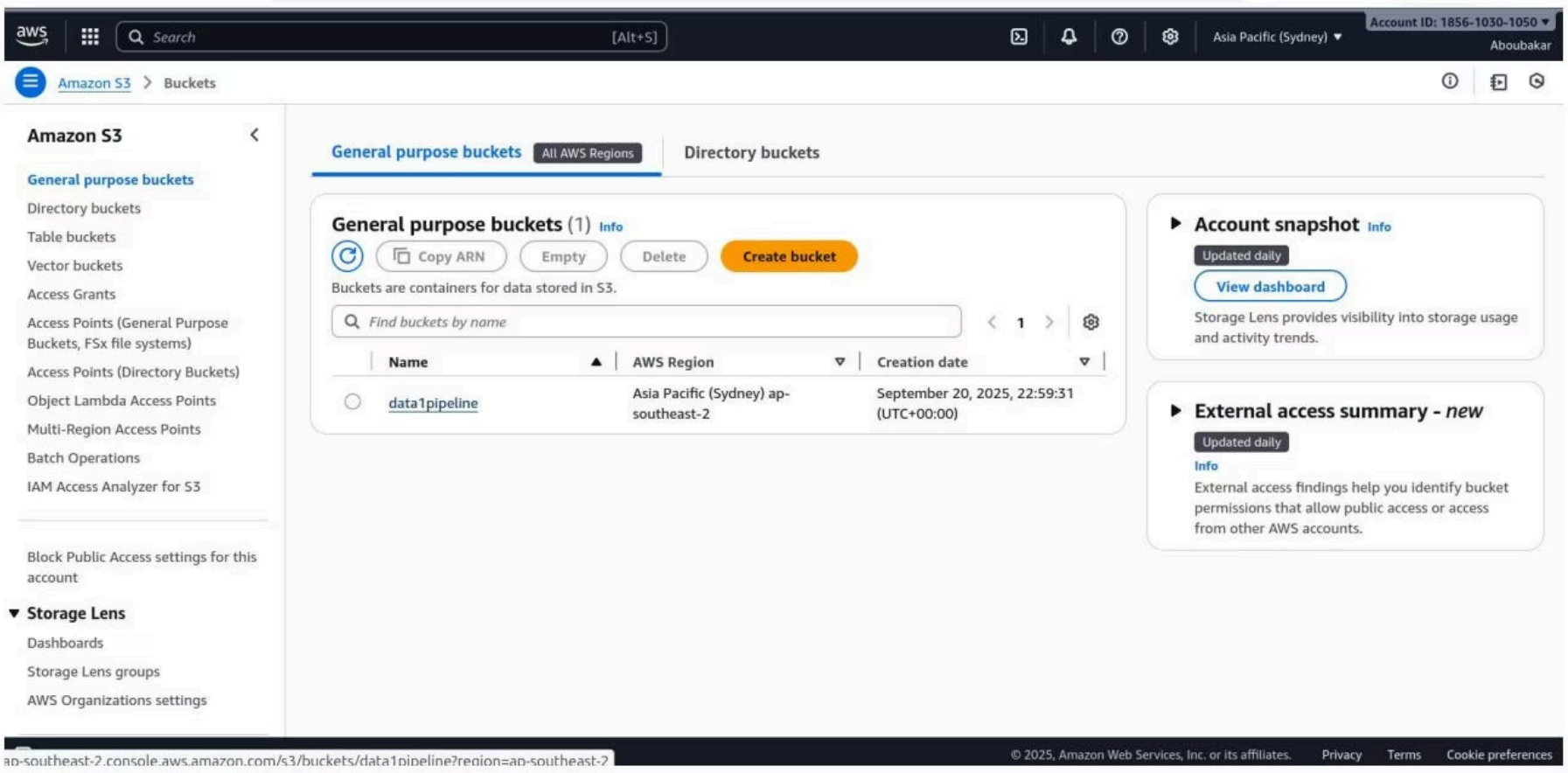
Configuration du Bucket S3

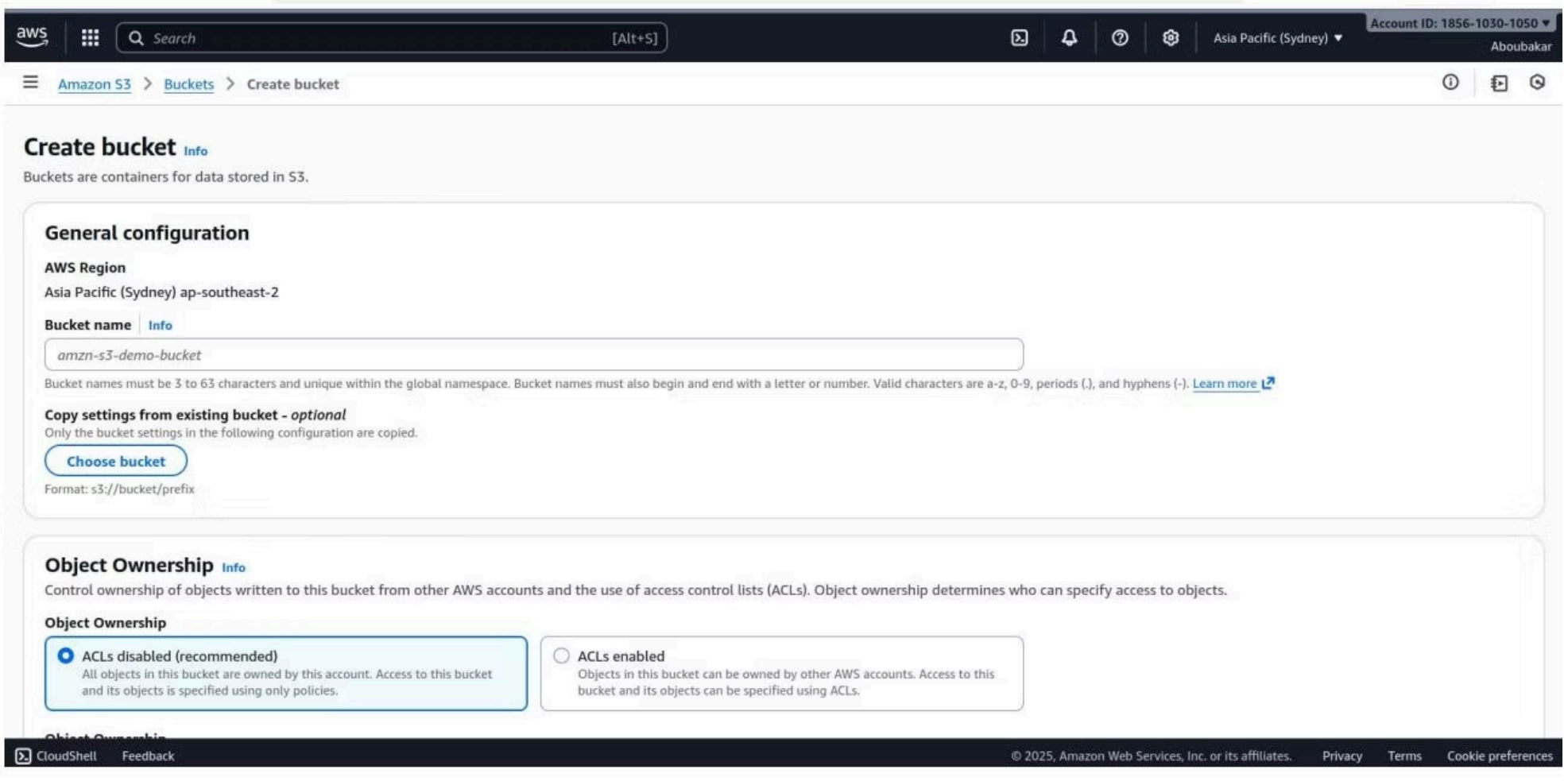
Créer un bucket S3

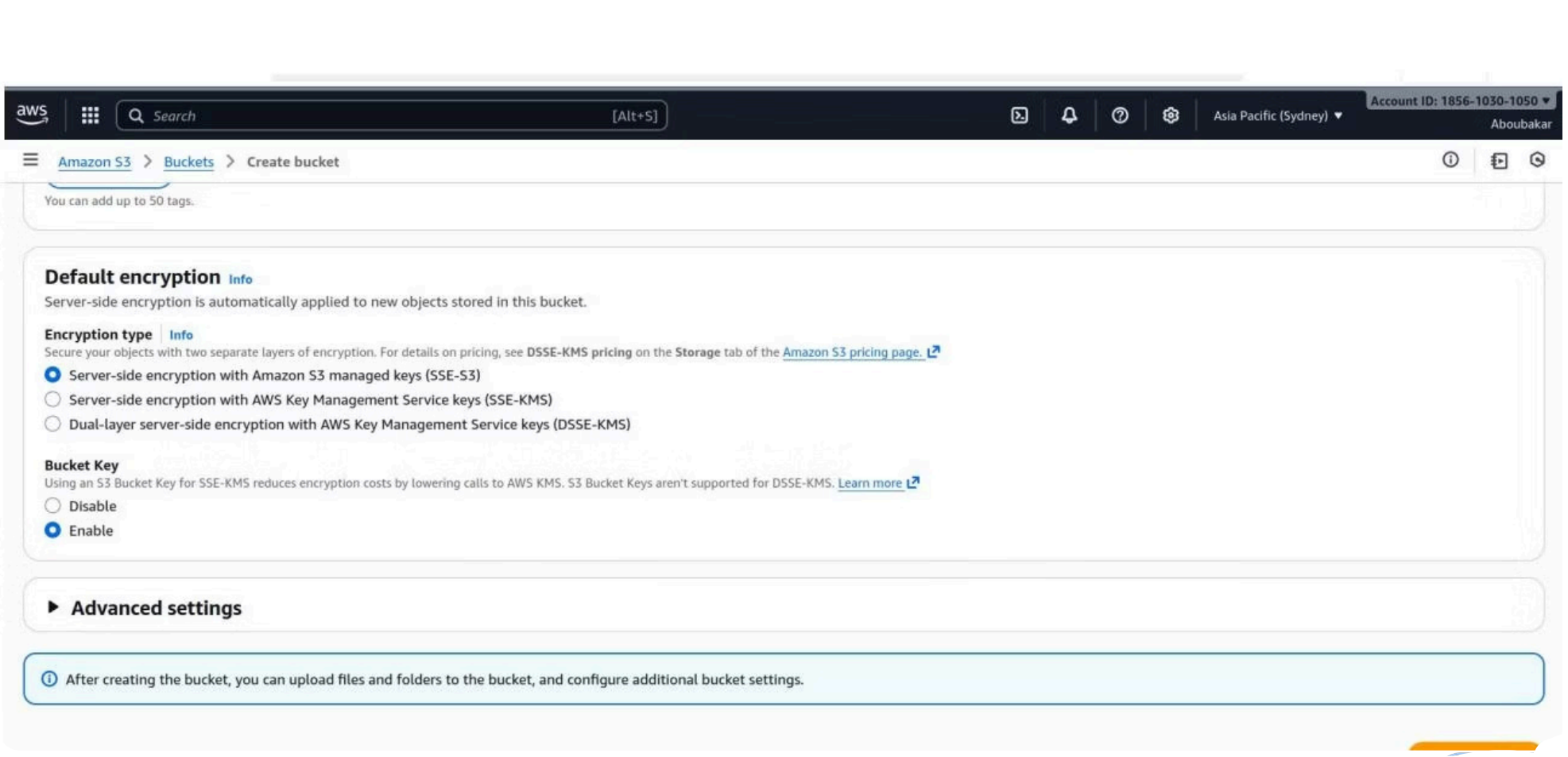
On l'a nommé dans notre cas **data1pipeline** vous pouvez le changer.



1. Allez sur votre console AWS, connectez-vous
2. Tapez dans la barre de recherche S3
3. Cliquez sur general purpose bucket
4. Cliquez sur create bucket et entrez le nom
5. Laissez les paramètres par défaut
6. Cliquez tout en bas sur create bucket







Créer les Clés d'Accès S3

Pourquoi créer des clés d'accès ?

Ces identifiants permettent à dbt (ou tout autre outil) d'accéder à votre bucket S3. Ils sont générés depuis la console AWS IAM (Identity and Access Management).

01

Créer un utilisateur IAM

Créer un utilisateur IAM avec accès programmatique (Programmatic access).

02

Attacher une politique

Attacher une politique (Policy) permettant l'accès à S3 (ex: AmazonS3FullAccess pour test ou une policy restreinte pour la production).

03

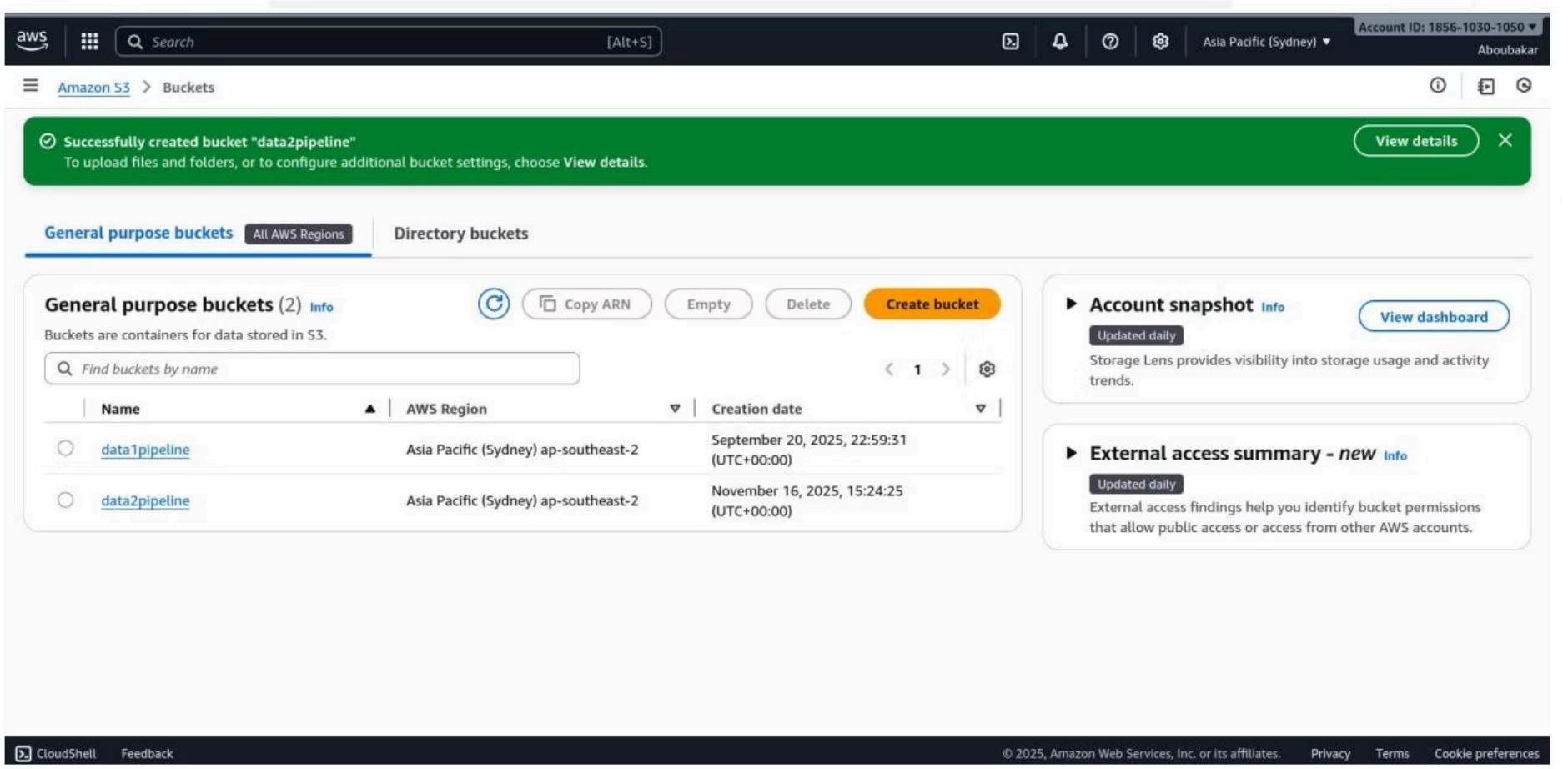
Générer les clés

Cliquez sur votre utilisateur et cherchez : create access key et généré les.

04

Récupérer les identifiants

La console génère alors une Access Key ID et une Secret Access Key.



Important : Ces clés peuvent être utilisées dans dbt, Airflow, ou tout script pour se connecter à S3.

Configuration de dbt et DuckDB

Installation de dbt-core avec l'adaptateur DuckDB

On ne va pas utiliser Postgres mais DuckDB vu que S3 est un outil de stockage objet qui permet de faire des lacs de données qui n'ont pas une structure relationnelle. Pour pouvoir effectuer des requêtes sur S3, on ne peut pas utiliser Postgres car gérant les DB relationnelles, on va utiliser DuckDB qui peut requêter sur des fichiers JSON, CSV et a des connecteurs pour S3 qu'on va utiliser.

Créer la structure du projet dbt

```
# Aller dans ton dossier airflow (si tu es sur ton Desktop par exemple)
cd ~/Desktop/airflow

# Créer le dossier dbt à l'intérieur du dossier airflow
mkdir -p dbt

# Aller dans ce dossier dbt
cd dbt

# Créer le dossier pour ton projet dbt (exemple : my_dbt_project)
mkdir -p my_dbt_project

# Initialiser un projet dbt à l'intérieur
cd my_dbt_project
dbt init my_dbt_project
```

Volumes montés dans le conteneur Airflow

Volumes montés dans le conteneur Airflow via Docker Compose :

- `/home/d-a-s/Desktop/airflow/dbt:/opt/airflow/dbt` : on le met dans la partie volume de notre fichier docker compose. Le dossier local ~/airflow/dbt est monté dans le conteneur à /opt/airflow/dbt. Sert à stocker les projets dbt (models, macros, analyses, seeds, etc.) de manière persistante. Toute modification dans le conteneur sera visible sur la machine hôte et vice-versa.
- `/home/d-a-s/.dbt:/opt/airflow/.dbt` : on le met dans la partie volume de notre fichier docker compose. Le dossier local ~/.dbt, il contient le fichier profile, vous pouvez le mettre ailleurs mais il faut spécifier le dossier et monté dans le conteneur à /opt/airflow/.dbt. Contient les fichiers de configuration dbt, notamment profiles.yml qui définit les connexions aux bases de données. Permet au conteneur d'utiliser les mêmes configurations que celles définies sur la machine hôte.

Configuration dbt (profiles.yml)

Configuration dbt pour le projet my_dbt_project (profiles.yml) :

```
target: dev

outputs:
  dev:
    type: duckdb
    path: /opt/airflow/dbt/my_dbt_project/my_duckdb.duckdb
    extensions:
      - httpfs
    threads: 1
    schema: main
    s3_region: ap-southeast-2
    s3_access_key_id: AKIASWNZ4NZ5JMMRMLOV
    s3_secret_access_key: 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
```

target: dev

Définit l'environnement de travail par défaut pour dbt.

path

Fichier DuckDB stocké dans le dossier du projet dbt monté dans le conteneur Airflow. Persistant : toutes les modifications restent sur la machine hôte via le volume Docker.

extensions: httpfs

Permet à DuckDB d'accéder à des fichiers stockés sur S3 directement depuis dbt.

threads: 1

Nombre de threads utilisés pour l'exécution des modèles dbt.

schema: main

Schéma par défaut utilisé dans DuckDB.

Identifiants S3

s3_region, s3_access_key_id, s3_secret_access_key : Informations de connexion pour accéder au bucket S3 depuis dbt/DuckDB.

Configuration des Sources et Modèles dbt

Fichier sources.yml (dans models/staging)

Rôle : Définit les sources de données externes que dbt va utiliser dans les modèles. Permet à dbt de référencer les fichiers S3 (ou d'autres sources) de manière centralisée et documentée. Facilite la traçabilité et la gestion des données avant toute transformation (staging).

```
version: 2

sources:
  - name: raw
    tables:
      - name: weather_data
        description: "Fichiers météo depuis S3"
        external:
          location: "s3://data1pipeline/raw/weather/"
          format: json
          pattern: "*.json"
      - name: covid_data
        description: "Fichiers COVID depuis S3"
        external:
          location: "s3://data1pipeline/raw/covid/"
          format: csv
          pattern: "*.csv"
```

Résumé : Ce fichier n'est pas strictement obligatoire pour faire fonctionner dbt, mais il est fortement recommandé. Il sert à documenter les sources et permettre aux modèles dbt de les référencer facilement via source('raw', 'weather_data') ou source('raw', 'covid_data'). Cela centralise la configuration S3, donc si le chemin change, il suffit de modifier ici.

Étapes pour ajouter tes modèles

1. Aller dans ton projet dbt : `cd ~/Desktop/airflow/dbt/my_dbt_project`
2. Créer les dossiers nécessaires :

```
mkdir -p models/staging
mkdir -p models/marts
```

Mettez-y les fichiers :

- -- models/staging/stg_weather.sql
- -- models/staging/stg_covid.sql
- -- models/mart/daily_weather_metrics.sql

Création et récupération de la clé API OpenWeatherMap

1. Créer un compte : Aller sur **https://home.openweathermap.org/users/sign_up**
2. Connexion : Se connecter à **https://home.openweathermap.org**
3. Générer une clé API : Aller dans le menu API keys. Cliquer sur "Generate" ou utiliser la clé par défaut fournie. Donner un nom à la clé (exemple : airflow_project_key).

Vous utiliserez cette clé dans le data_extraction_dag.py pour récupérer les données de météo (weather).

Installation Finale et Exécution

1️⃣ Accéder au conteneur Airflow

Identifie d'abord le nom du conteneur Airflow en cours d'exécution avec :

```
docker ps
```

Ensuite, connecte-toi au conteneur (par ex. le scheduler ou le webserver) avec la commande :

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

2️⃣ Installer dbt et dbt avec l'adaptateur DuckDB

Dans le conteneur avec :

```
pip install --no-cache-dir dbt-core dbt-duckdb
```

Vérifier l'installation avec :

```
dbt --version
```

Vous devez voir quelque chose comme :

```
installed version: 1.8.x
plugins:
  - duckdb: 1.8.x
```

3️⃣ Placer les fichiers DAG

Placer les fichiers data_extraction_dag.py et dbt_pipeline_dag.py dans le dossier airflow créé par docker compose.

4️⃣ Exécuter les pipelines

Ouvrir Airflow

Ouvrir la page web de Airflow **http://localhost:8080/**

Exécuter les DAGs

Exécuter les DAG : **data_extraction_dag** et **dbt_pipeline_dag**

Félicitations ! Votre pipeline de données est maintenant opérationnel. Les données météorologiques et COVID seront extraites de S3, transformées par dbt avec DuckDB, et orchestrées par Airflow.