**Guida allo Sviluppo della Pipeline ETL con Apache Airflow**

**1. Introduzione e Setup dell'Ambiente**

Questo documento descrive passo per passo il processo di sviluppo ed esecuzione di una pipeline di dati ETL utilizzando Apache Airflow. L'obiettivo della pipeline, denominata globalstay\_full\_etl\_pipeline, è orchestrare le fasi di ingestione (Bronze), pulizia (Silver) e aggregazione (Gold) dei dati di prenotazione alberghiera.

L'intera infrastruttura Airflow è stata configurata ed eseguita all'interno di una **macchina virtuale (VM) con sistema operativo Debian**, creando un ambiente di sviluppo isolato e controllato.

**1.1 Avvio dei Servizi Airflow**

Per poter operare, Airflow necessita di due componenti principali in esecuzione: il Webserver e lo Scheduler.

1. **Attivazione dell'Ambiente Virtuale:** Prima di avviare i servizi, è stato attivato l'ambiente virtuale Python dedicato (airflow-venv) per garantire che tutte le dipendenze siano correttamente caricate.

Bash

source ~/airflow-venv/bin/activate

1. **Avvio del Webserver:** Il Webserver è stato avviato sulla porta 8080. Questo servizio fornisce l'interfaccia utente (UI) web, fondamentale per monitorare e interagire con i DAG.

Bash

A screenshot of a computer

AI-generated content may be incorrect.airflow webserver -p 8080

1. **Avvio dello Scheduler:** In un terminale separato, è stato avviato lo Scheduler. Questo componente è il cuore di Airflow: si occupa di monitorare i DAG, schedulare le esecuzioni e inviare i task agli executor. Per questo ambiente di sviluppo è stato utilizzato il SequentialExecutor.

Bash

A screenshot of a computer

AI-generated content may be incorrect.airflow scheduler

**2. Struttura del Progetto**

Una corretta organizzazione dei file è essenziale per un progetto. La logica è stata suddivisa tra il file del DAG e gli script di trasformazione.

**2.1 Cartella dei DAG**

**A screenshot of a computer

AI-generated content may be incorrect.**Tutti i file relativi ai DAG sono contenuti nella cartella airflow/dags/. Il file principale che definisce la nostra pipeline è globalstay\_full\_etl\_pipeline.py.

**2.2 Cartella degli Script**

Per mantenere il file del DAG pulito e leggibile, la logica ETL vera e propria è stata separata in script Python dedicati, contenuti nella sottocartella scripts/. Questa cartella è un package Python (grazie alla presenza del file \_\_init\_\_.py), permettendo di importare le funzioni in modo pulito all'interno del DAG.

I file principali sono:

* ingest\_to\_bronze.py: Logica per la fase Bronze.
* bronze\_to\_silver.py: Logica per la fase Silver.
* A screenshot of a computer

  AI-generated content may be incorrect.silver\_to\_gold.py: Logica per la fase Gold.

**3.Analisi del Codice del DAG**

Il cuore della pipeline è definito nel file globalstay\_full\_etl\_pipeline.py. Questo file non contiene la logica ETL, ma si occupa di definire la struttura del workflow e di orchestrare l'esecuzione degli script Python esterni.

**3.1 Configurazione e Importazioni**

Per rendere il progetto modulare, il percorso della cartella scripts/ viene aggiunto al sys.path di Python, permettendo al DAG di importare le funzioni di trasformazione. La definizione del DAG include parametri chiave:

* schedule=None: Il DAG è progettato per essere eseguito solo su richiesta (manualmente).
* catchup=False: Previene esecuzioni non desiderate di run passate.
* tags: Etichettano il DAG per una facile ricerca e organizzazione nell'interfaccia utente.

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**

**3.2 Definizione dei Task e Dipendenze**

Ogni fase della pipeline (Bronze, Silver, Gold) è un task di tipo PythonOperator, che esegue una specifica funzione Python (python\_callable). La sequenza di esecuzione è garantita definendo le dipendenze con l'operatore >>, che assicura che un task parta solo dopo il successo del precedente.

**A screenshot of a computer

AI-generated content may be incorrect.**

**4. Esecuzione e Monitoraggio del DAG dall'Interfaccia Utente**

Una volta avviati i servizi, tutta l'interazione con la pipeline avviene tramite l'interfaccia web di Airflow, accessibile all'indirizzo http://localhost:8080.

**4.1 Attivazione ed Esecuzione**

1. **Visualizzazione del DAG:** Nella schermata principale, il DAG globalstay\_full\_etl\_pipeline è visibile nell'elenco. Inizialmente è in stato "inattivo".

A screenshot of a computer

AI-generated content may be incorrect.

1. **A screenshot of a computer

   AI-generated content may be incorrect.Attivazione del DAG:** Per renderlo operativo, il DAG è stato attivato tramite l'interruttore a levetta.
2. A screenshot of a computer

   AI-generated content may be incorrect.**Avvio Manuale:** Una volta attivo, il DAG è stato avviato manualmente cliccando sul pulsante "Play". L'interfaccia mostra immediatamente il DAG in stato di "running" (cerchio verde chiaro).

**4.2 Monitoraggio dell'Esecuzione**

A screenshot of a computer

AI-generated content may be incorrect.Cliccando sul nome del DAG, si accede alla vista di dettaglio, dove è possibile monitorare lo stato di avanzamento di ogni singolo task. In questa vista, si può osservare il flusso di esecuzione e lo stato di ogni task (successo, in esecuzione, fallito, etc.).

**5. Analisi Dettagliata dell'Esecuzione del DAG**

Dopo aver avviato il DAG, l'interfaccia di Airflow offre molteplici viste per analizzare e monitorare l'esecuzione in dettaglio.

**5.1 Vista "Graph"**

**A screenshot of a computer

AI-generated content may be incorrect.**La vista **"Graph"** è una delle più utili in quanto mostra la struttura delle dipendenze tra i task. Al termine dell'esecuzione, tutti i task sono bordati di verde scuro, a indicare che l'intera pipeline è stata completata con successo.

**5.2 Vista "Gantt"**

**A screenshot of a computer

AI-generated content may be incorrect.**La vista **"Gantt"** offre una prospettiva temporale dell'esecuzione. Mostra la durata di ogni task e come si posizionano lungo la linea del tempo, rendendo facile identificare i task più lunghi e i possibili colli di bottiglia della pipeline.

**5.3 Viste di Dettaglio del Task**

Selezionando un singolo task, è possibile accedere a informazioni ancora più granulari.

* **A screenshot of a computer

  AI-generated content may be incorrect.Event Log:** Questa vista mostra un log di basso livello degli eventi del ciclo di vita del task (es. running, success), utile per il debugging.
* A screenshot of a computer

  AI-generated content may be incorrect.**Task Duration:** Questo grafico visualizza la durata delle esecuzioni di un task nel tempo, permettendo di monitorare le sue performance e identificare eventuali degradazioni.

**6. Configurazione della Connessione ad Azure**

Per permettere ad Airflow di comunicare con servizi esterni in modo sicuro, senza esporre credenziali nel codice, si utilizza il sistema delle **"Connections"**.

1. **Accesso alle Connessioni:** La gestione delle connessioni è accessibile dal menu **Admin -> Connections**.**A screenshot of a computer

   AI-generated content may be incorrect.**
2. **A screenshot of a computer

   AI-generated content may be incorrect.Lista delle Connessioni:** L'interfaccia mostra tutte le connessioni predefinite e quelle create dall'utente.
3. **Connessione ad Azure Storage:** Per questo progetto, è stata configurata una connessione di tipo wasb (Windows Azure Storage Blob) con l'ID **azure\_storage\_connection**. Questa connessione contiene in modo sicuro le chiavi di accesso necessarie per leggere e scrivere dati sull'account di Azure Blob Storage. L'ID della connessione viene poi richiamato all'interno degli script Python per autenticarsi al servizio cloud.

**7. Conclusioni**

Al termine dell'esecuzione del DAG, la pipeline ha correttamente processato i dati e salvato i

5 file CSV contenenti i KPI di business nella cartella gold/ del container datalake su Azure

Blob Storage. Questo conferma il successo dell'intero processo ETL.

**A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.**

Questo documento ha illustrato l'intero ciclo di vita di una pipeline ETL orchestrata con Apache Airflow su un ambiente di sviluppo virtualizzato. Partendo dalla configurazione dei servizi, passando per la strutturazione del codice e il monitoraggio tramite l'interfaccia web, la pipeline ha eseguito con successo tutte le fasi di trasformazione dei dati.

La corretta configurazione della connessione ad Azure ha permesso di interagire in modo sicuro con il cloud, dove sono stati salvati i 5 file CSV finali contenenti i KPI di business. Questo conferma il successo dell'intero processo e la validità dell'architettura scelta."