The new way to ingest data with Airflow [PUBLIC]

Prerequisites

→ AWS account

→ Bigquery account

→ Astro CLI

→ Docker

Dataset

retail_transaction_dataset.csv.zip

The Dockerfile

```
FROM quay.io/astronomer/astro-runtime:11.3.0

RUN python -m venv .venv-source-s3 && source .venv-source-s3/
    pip install --no-cache-dir airbyte-source-s3 && deactivat
```

```
RUN python -m venv .pyairbyte-venv && source .pyairbyte-venv/
    pip install --no-cache-dir airbyte==0.10.5 && deactivate
```

The .env file

```
AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AIRFLOW__CORE__TEST_CONNECTION=Enabled
```

The packages

```
build-essential
libsnappy-dev
```

The DAG

```
# Description: This file contains the DAG definition for the
from airflow.decorators import dag, task
from datetime import datetime
from airflow.providers.google.cloud.hooks.bigquery import Big
from airflow.models.baseoperator import chain

@dag(
    start_date=datetime(2024, 1, 1),
    schedule='@daily',
    catchup=False,
    tags=['el', 's3', 'bigquery'],
    max_consecutive_failed_dag_runs=5,
    doc_md=__doc__,
)
def el():

    @task.external_python(python='/usr/local/airflow/.pyairby
    def extract():
```

```
        import airbyte as ab
        from airbyte.caches import BigQueryCache

        source = ab.get_source(
            "source-s3",
            config={
                "bucket": "m-yt-el",
                "region_name": "eu-north-1",
                "streams": [
                  {
                      "name": "transaction",
                      "format": {
                          "filetype": "csv"
                      },
                  }
                ],
                "credentials": {
                    "aws_access_key_id": ab.get_secret("AWS_A
                    "aws_secret_access_key": ab.get_secret("A
                }
            },
            install_if_missing=False,
        )
        source.select_all_streams()
        read_result = source.read(cache=BigQueryCache(
            project_name="yt-el-423508",
            dataset_name="retail",
            credentials_path="/usr/local/airflow/include/el-b
        ))

        first_record = next((record for record in read_result
        print(f"First record: {first_record}")

    @task
    def check():
        bigquery = BigQueryHook(gcp_conn_id='bigquery',
                                use_legacy_sql=False,
                                location='US')
```

```
        df = bigquery.get_pandas_df("SELECT COUNT(*) FROM ret
        print(f"Number of rows in the table: {df.iloc[0, 0]}"

    chain(extract(), check())

el()
```