Pipeline Big Data pour l'analyse des données d'aéroports

BEN SALEM Eya

AMOR Yasmine

Plan

- Contexte et objectifs du projet
- Architecture globale du pipeline
- Ingestion des données
- Streaming et transport
- Traitement en temps réel
- Stockage des données
- Visualisation
- Conclusion et perspectives

Contexte et objectifs du projet

Le projet **Pipeline ETL d'analyse des données d'aéroports** vise à concevoir une chaîne d’acquisition, de traitement et de visualisation de données d'aéroports issues **d’API Openaip**.

Objectifs principaux :

- Collecter et traiter des flux de données en temps réel.
- Mettre en place un pipeline **ETL distribué** .
- Stocker les données nettoyées dans une base distribuée.
- Créer un **tableau de bord** pour visualiser les KPIs.

Contexte et objectifs du projet

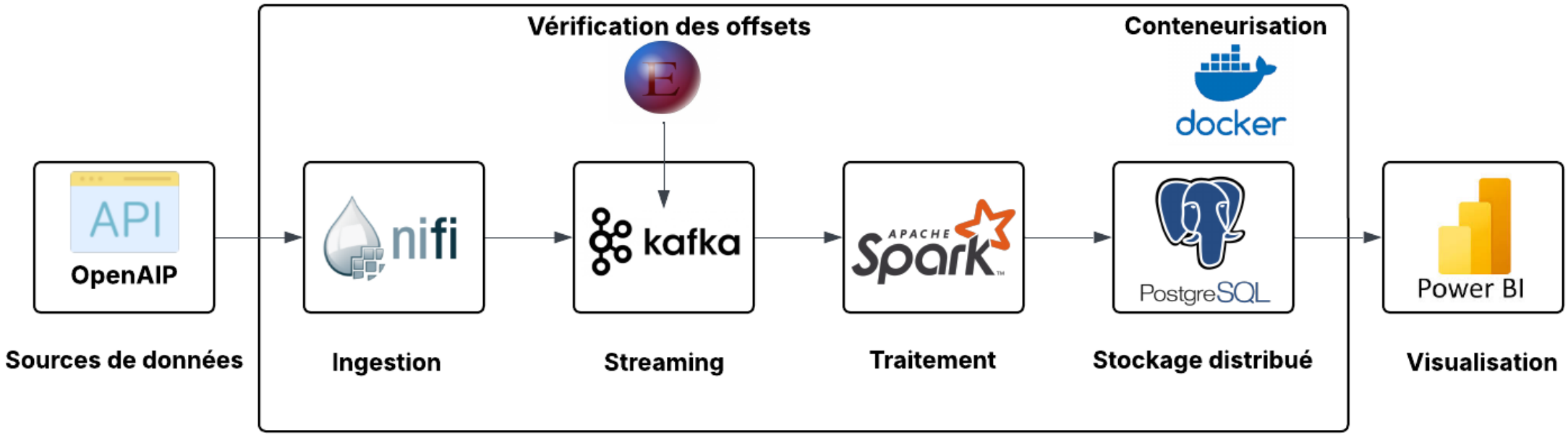
🎯 **Cibles:**

Autorités aériennes (DGAC, ICAO, IATA)

Compagnies aériennes

Gestionnaires d’aéroports

Architecture global



Ingestion des données : NIFI

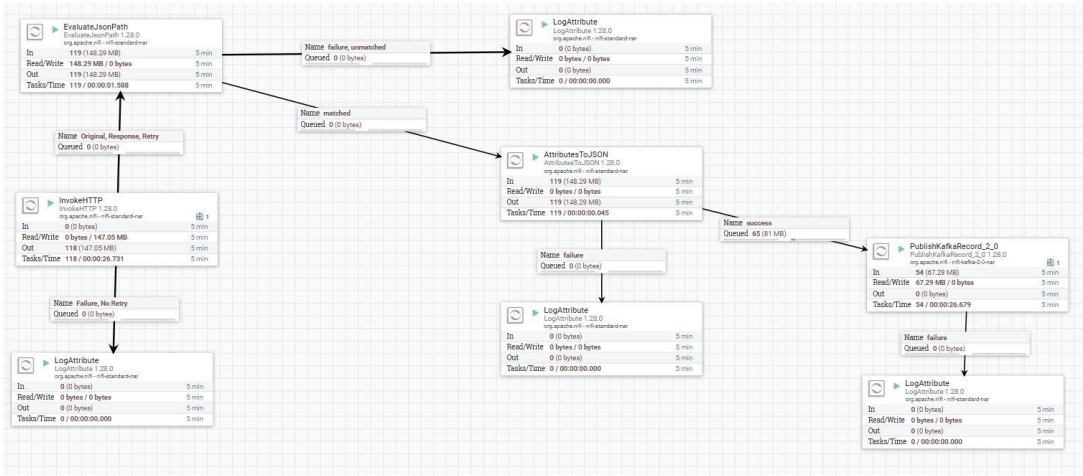
- **Les fonctions des processeurs NIFI :**

  - **InvokeHTTP:** Appele l'API OpenAIP et récupére les données

  - **EvaluateJsonPath :** Extraire les champs utiles du JSON

  - **AttributesToJSON**:Transforme les attributs en message JSON en format organisé

  - **PublishKafkaRecord_2_0:** Publie le JSON dans un topic Kafka pour Spark



Description des données

Champ JSON	Description	Utilité
`airport_id`	Identifiant unique OpenAIP	Primary key
`name`	Nom de l'aéroport	Libellé dans dashboard
`icao`	Code ICAO (international aviation)	Référence aviation
`iata`	Code IATA (3 lettres)	Reconnaissance publique (ex: CDG)
`country`	Pays	Groupement par pays
`lat`	Latitude	Carte Power BI
`lon`	Longitude	Carte Power BI
`elevation_m`	Altitude en mètres	Indicateur topographique
`runway_count`	Nombre de pistes	Analyse infrastructure
`max_runway_length_m`	Longueur piste max	Capacités aéroport
`ingested_at`	Timestamp Spark	Suivi streaming ETL

Streaming et transport : KAFKA

Propriétés du processeur Kafka dans Nifi:

Property	Value
Kafka Brokers	kafka:9092
Topic Name	flights_positions
Record Reader	JsonTreeReader
Record Writer	JsonRecordSetWriter
Use Transactions	false
Failure Strategy	Route to Failure
Transactional Id Prefix	No value set
Delivery Guarantee	Guarantee Replicated Delivery
Attributes to Send as Headers (Regex)	No value set
Message Header Encoding	UTF-8
Security Protocol	PLAINTEXT
SASL Mechanism	GSSAPI

Affichage du Topic Name crée de Kafka:

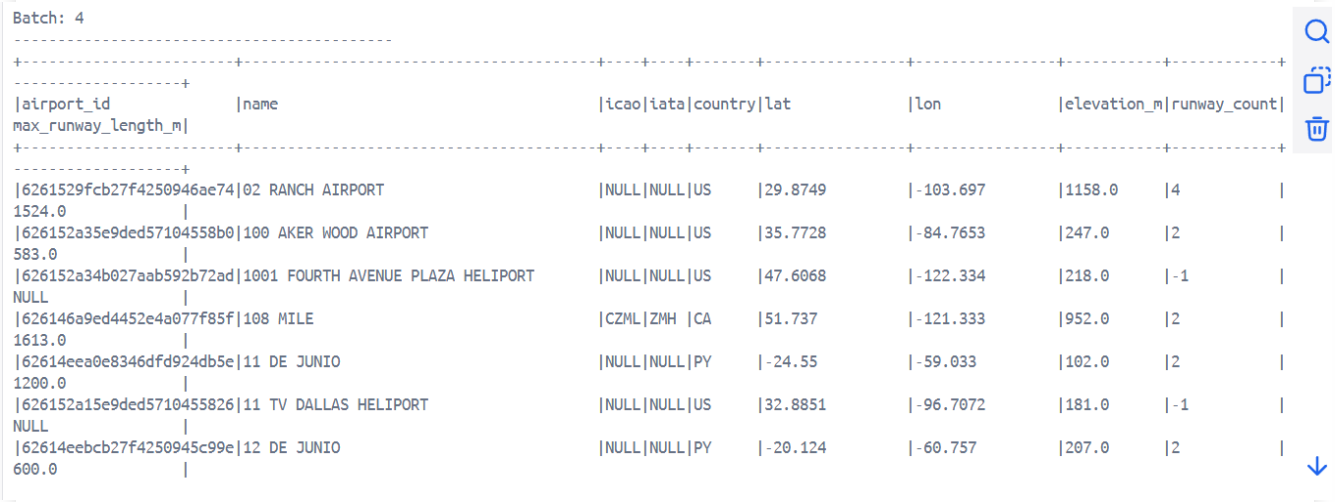
```
root@bca55b25a495:/# kafka-topics.sh --bootstrap-server localhost:9092 --list
__consumer_offsets
__transaction_state
flights_positions
```

Affichage des messages reçus par Kafka:

{"limit":1000,"totalCount":46491,"totalPages":47,"nextPage":2,"page":1,"items":[{"_id":"6261529fcb27f4250946ae74","name":"02 RANCH AIRPORT","type":2,"trafficType":[0],"magneticDeclination":6.279,"country":"US","geometry":{"type":"Point","coordinates":[-103.697,29.8749]},"elevation":{"value":1158,"unit":0,"referenceDatum":1},"ppr":false,"private":false,"skydiveActivity":false,"winchOnly":false,"runways":[{"designator":"05","trueHeading":50,"alignedTrueNorth":false,"operations":0,"mainRunway":false,"turnDirection":2,"takeOffOnly":false,"landingOnly":false,"surface":{"composition":[22],"mainComposite":22,"condition":0},"dimension":{"length":{"value":1066,"unit":0},"width":{"value":30,"unit":0}},"declaredDistance":{"tora":{"value":1066,"unit":0},"lda":{"value":1066,"unit":0}},"pilotCtrlLighting":false," id":"6261529fcb27f4250946ae75"},{"designator":"23","t

Traitement en temps réel : SPARK

Spark consomme les messages depuis Kafka, effectue un **traitement temps réel** (nettoyage et agrégation ..) et prépare les données pour le stockage.



```
Batch: 4
-------------------------------------------
+----------------------+--------------------------------------+----+----+-------+---------------+---------------+-----------+------------+
------------------+
|airport_id            |name                                  |icao|iata|country|lat            |lon            |elevation_m|runway_count|
max_runway_length_m|
+----------------------+--------------------------------------+----+----+-------+---------------+---------------+-----------+------------+
------------------+
|6261529fcb27f4250946ae74|02 RANCH AIRPORT                    |NULL|NULL|US     |29.8749        |-103.697       |1158.0     |4           |
1524.0             |
|626152a35e9ded57104558b0|100 AKER WOOD AIRPORT               |NULL|NULL|US     |35.7728        |-84.7653       |247.0      |2           |
583.0              |
|626152a34b027aab592b72ad|1001 FOURTH AVENUE PLAZA HELIPORT   |NULL|NULL|US     |47.6068        |-122.334       |218.0      |-1          |
NULL               |
|626146a9ed4452e4a077f85f|108 MILE                            |CZML|ZMH |CA     |51.737         |-121.333       |952.0      |2           |
1613.0             |
|62614eea0e8346dfd924db5e|11 DE JUNIO                         |NULL|NULL|PY     |-24.55         |-59.033        |102.0      |2           |
1200.0             |
|626152a15e9ded5710455826|11 TV DALLAS HELIPORT               |NULL|NULL|US     |32.8851        |-96.7072       |181.0      |-1          |
NULL               |
|62614eebcb27f4250945c99e|12 DE JUNIO                         |NULL|NULL|PY     |-20.124        |-60.757        |207.0      |2           |
600.0              |
```

Description du Script Spark

Spark **analyse le tableau runways**, extrait les pistes, et garde la plus longue.

Étape	Ce que ça fait	Objectif
1 Démarrer Spark	Lance l'application de streaming	Préparer l'environnement
2 Définir le schéma JSON	Décrit la structure des données d'aéroports	Comprendre les données reçues
3 Lire depuis Kafka	Récupère les données en temps réel	Ingestion streaming
4 Convertir & parser JSON	Transforme le texte Kafka en colonnes	Rendre les données exploitables
5 Aplatir & nettoyer	Extraire id, nom, pays, lat, lon, altitude, pistes	Obtenir table propre
6 Calculs	Nombre de pistes, longueur piste max	Enrichir les données
7 Écrire dans PostgreSQL	Sauvegarde les données traitées	Stockage pour le dashboard

Champ Spark	Origine / Calcul
`runway_count`	`count(runways)`
`max_runway_length_m`	`max(runways.length_m)`
`ingested_at`	`current_timestamp()`

Stockage des données : PostgreSQL

Les données traitées sont sauvegardées dans une base relationnelle PostgreSQL.Ce stockage structuré permet d'alimenter les outils de visualisation.

	airport_id [PK] text	name text	icao text	iata text	country text	lat double precision	lon double precisi
963	626147aff2228b5341bfdefd	ABECHE	FTTC	AEH	TD	13.84741467980163	20.8437047
964	626152b04b027aab592b75...	AERO-BEE RANCH AIRSTRIP	[null]	[null]	US	30.8793	
965	626152cecb27f4250946b8...	AL S AIRWAY AIRPORT	[null]	[null]	US	43.1631	
966	626142821e911989f7b2cbac	ALAGRO FUMIGACIONES	[null]	[null]	AR	-32.888	
967	626144cded4452e4a0773d...	AGROPECUÁRIA CÉU ABERTO	SNLJ	[null]	BR	-9.517	
968	62614be9cb27f4250944e4f3	AEROCLUB IL GRIFO	[null]	[null]	IT	38.63626	

Visualisation et analyse

Conclusion et perspectives

Ce projet nous a permis de **concevoir et déployer un pipeline Big Data complet** assurant la **collecte, le traitement et la visualisation en temps réel des données aériennes**.

Nous avons **automatisé l'ingestion avec Apache NiFi**, **diffusé les flux avec Kafka**, **traité les données en continu avec Spark Streaming**, **stocké les résultats dans PostgreSQL**, et enfin **créé des tableaux de bord interactifs dans Power BI**.

En **perspective**, il serait intéressant d'**étendre ce pipeline au suivi des vols en temps réel**, en intégrant des données des vols aeriens afin d'analyser **le trafic aérien dynamique** et d'enrichir encore les visualisations.

Merci pour votre attention!