BI Engineer technical test

Pré-requis

Avant de commencer le test technique assurez vous que les vérifications suivantes sont **OK** pour vous :

- ☐ Vous avez bien le dossier `bi-hiring-test/` dans l'archive fourni avec cet énoncé
- ☐ Vous avez lu le fichier `README.md` présent dans ce dossier
- ☐ Vous avez suivi la partie `Setup requirements` et les outils évoqués sont bien installés

Contexte

L'équipe commerciale de Meilleurs Agents aimerait avoir un tableau de bord leur donnant une vue d'ensemble de leur parc d'agences clientes ainsi que son évolution dans le temps. Ce dashboard devrait comprendre plusieurs indicateurs leurs permettant de suivre la cycle de vie commercial des agences.

Pour cela, l'équipe s'est rapprochée de vous pour que vous mettiez en place un datamart qui permettrait à leur analyste de construire un dashboard de suivi facilement.

Pour vous aider dans le traitement de cette demande, l'équipe Web vous a mis à disposition des exports de données journaliers sur le mois de janvier 2021.
Il y a deux types d'exports :

- `FULL` : Image complète d'une table au moment de l'extrait
- `INCREMENTALE` : Image des nouvelles données (données du jour)

Les données à disposition sont le suivantes :

▼ Realtor

Type de l'extrait : `FULL`

Chemin jusqu'à l'extrait : `realtor/realtor_yyyymmdd.json`

Description : Extrait de vingt agences clientes

Schema de l'extrait :

Nom du champ	Type	Description
realtor_id	STRING	Identifiant unique de l'agence
ds	DATE	Date de l'image des données
realtor_name	STRING	Nom de l'agence
city_name	STRING	Nom de la ville de l'agence

▼ Realtor_agent

Type de l'extrait : `FULL`

Chemin jusqu'à l'extrait : `realtor_agent/realtor_agent_yyyymmdd.json`

Description : Extrait des collaborateurs d'agences

Schema de l'extrait :

Nom du champ	Type	Description
ds	DATE	Date de l'image des données
realtor_agent_id	STRING	Identifiant unique du collaborateur d'une agence
realtor_agent_name	STRING	Nom du collaborateur
realtor_id	STRING	Identifiant unique de l'agence
user_id	STRING	Identifiant unique de l'utilisateur MeilleursAgents
user_name	STRING	Nom de l'utilisateur MeilleursAgents
is_enabled	BOOLEAN	Booléen indiquant si l'utilisateur est actif
role	STRING	Role du collaborateur
role_label	STRING	Label du rôle du collaborateur

▼ Event

Type de l'extrait : `INCREMENTALE`

Chemin jusqu'à l'extrait : `realtor_agent_event/realtor_agent_eventt_yyyymmdd.json`

Description : Extrait de l'activité des collaborateurs par agence sur l'interface B2B

Schema de l'extrait :

Nom du champ	Type	Description
event_id	INT64	Identifiant unique de l'évènement
ds	DATE	Date de l'image des données
event_created_date	DATE	Date de l'évènement
event_page_main_category	STRING	Catégorie de la page sur laquelle a eu lieu l'évènement
realtor_id	INT64	Identifiant unique de l'agence

▼ Listing

Type de l'extrait : `FULL`

Chemin jusqu'à l'extrait : `realtor_listing/realtor_listing_yyyymmdd.json`

Description : Annonces des agences

Schema de l'extrait :

Nom du champ	Type	Description
listing_id	STRING	Identifiant unique d'une annonce
ds	DATE	Date de l'image des données
realtor_id	STRING	Identifiant unique de l'agence
created_ts	TIMESTAMP	Timestamp à laquelle l'annonce a été crée
last_updated_ts	TIMESTAMP	Timestamp à laquelle l'annonce a été modifiée la dernière fois
listing_name	STRING	Nom de l'annonce
transaction_type	STRING	Type de transaction de l'annonce
start_ts	TIMESTAMP	Timestamp à laquelle l'annonce a été publié sur le site
end_ts	TIMESTAMP	Timestamp à laquelle l'annonce a été retiré sur le site
item_type	STRING	Type du bien de l'annonce

▼ Past_sale

Type de l'extrait : `FULL`

Chemin jusqu'à l'extrait : `realtor_past_sale/realtor_past_sale_yyyymmdd.json`

Description : biens vendus par les agences

Schema de l'extrait :

Nom du champ	Type	Description
past_sale_id	STRING	Identifiant unique du bien vendu
ds	DATE	Date de l'image des données
realtor_id	STRING	Identifiant unique de l'agence
created_ts	TIMESTAMP	Timestamp à laquelle le bien vendu a été crée
last_updated_ts	TIMESTAMP	Timestamp à laquelle le bien vendu a été modifiée la dernière fois
past_sale_name	STRING	Nom du bien vendu
sale_ts	TIMESTAMP	Timestamp a laquelle le bien vendu a été vendu
item_type	STRING	Type de bien du bien vendu

▼ Review

Type de l'extrait : `INCREMENTALE`

Chemin jusqu'à l'extrait : `realtor_review/realtor_review_yyyymmdd.json`

Description : Avis sur les agences

Schema de l'extrait :

Nom du champ	Type	Description
review_id	STRING	Identifiant unique de l'avis
ds	DATE	Date de l'image des données
review_name	STRING	Nom de l'avis
review_ts	TIMESTAMP	Timestamp à laquelle l'avis a été crée
realtor_id	STRING	Identifiant unique de l'agence
type	STRING	Type d'avis
moderation_status	STRING	Status de modération de l'avis

realtor_recommendation	INTEGER	Note sur l’agence sur cinq

Ces fichiers sont disponibles dans le dossier `ma_bi_eng_technical_test/bi-hiring-test/dags/data/`

1. Conception

L’équipe commerciale aimerait que vous lui expliquiez comment vous comptez récupérer les différentes sources de données nécessaires, quels traitements vous allez faire dessus pour simplifier le travail à leur analyste et quels seraient les indicateurs pertinents pour le suivi du parc d’agences que vous imaginez.

Globalement, elle serait intéressée par une présentation des différents flux de traitement de donnés que vous comptez créer ainsi que de la/les table(s) finale(s).

Ce qui est attendu:

- À partir des fichiers, réfléchir aux indicateurs qui seraient pertinents dans le suivi du parc d’agences
- Un document présentant le modèle de données (type MCD)
- Un ou plusieurs schéma décrivant le/les workflow(s) de traitement des données pour répondre au besoin

> 💡 Outre les indicateurs calculés dans la/les table(s) finale(s), nous attendons de voir comment vous allez traiter chaque flux de données et sous quelle forme vous allez restituer l’information

2. Développement

2.1. Prise en main de Airflow et implémentation de fonction

Le premier exercice consistera à faire fonctionner un DAG Airflow incomplet, vous trouverez ce DAG dans le fichier `db_analytics_example.py`.

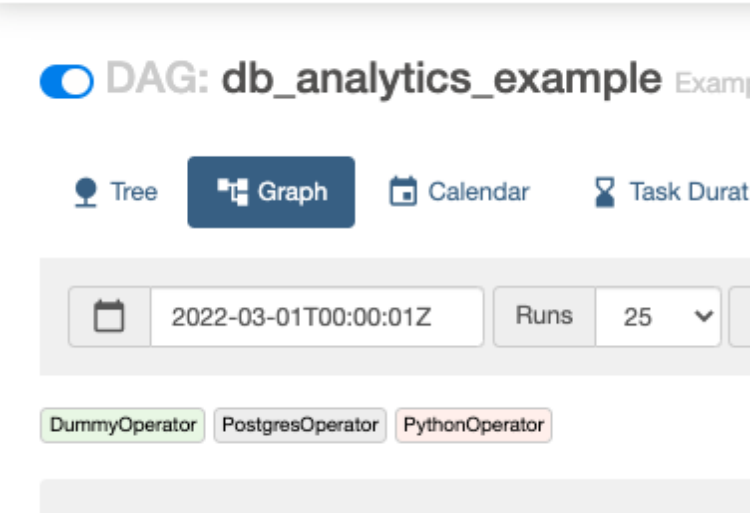
Ce DAG à pour but de faire des statistiques sur les propriétaires d'animaux du fichier `pets.json`.

Les principales étapes de ce DAG sont écrites, sauf la fonction `insert_values` que vous devez implementer vous même pour faire fonctionner le dag.

La fonction doit charger des données depuis un fichier JSON (NEWLINE DELIMITED JSON) dans une table PostgreSQL.

Quelques tips :

- N'hésitez pas à aller voir la documentation de Apache Airflow
- Vous êtes libre de changer la signature de la fonction si vous le voulez, tant que la fonction donne le résultat décrit plus haut
- Pour visualiser le DAG pensez à lancer la commande `docker-compose up` (lire le `README` de l'archive fourni). Le DAG devrait être accessible dans votre navigateur web après un délai d'initialisation (http://localhost:8080/graph?dag_id=db_analytics_example).
- **Activez le DAG** en cliquant sur le toggle switch à gauche du nom du DAG (sinon il ne sera pas exécuté par le scheduler, il ne se passera rien)



2.2. Mise en application

Maintenant que vous avez un premier DAG fonctionnel, Il est temps de vous en inspirer pour développer ce que vous avez prévu dans la **partie 1**.

À partir des fichiers mis à disposition :

- **Implémenter le modèle de données que vous avez conçu précédemment sur la base PostgreSQL mise à disposition (dans un schéma spécifique)**
- **Implémenter le/les workflow(s) de traitement de données conçu précédemment à l'aide du Airflow mise à disposition**

Ce qui est attendu:

- une archive contenant le/les DAG (fichier `.py`) et les scripts SQL (fichiers `.sql`) nécessaires au calcul de/des table(s) pour l'équipe commerciale.