

ANGELE DUTILLEUL - JULIEN GODFROY

Documentation projet **BIG-DATA**

Sommaire

| | |
|---|----------|
| I - Introduction | 3 |
| Sujet choisi : | 3 |
| Technologies utilisées : | 3 |
| II - Ingestion des données | 4 |
| III - Transformation avec dbt | 6 |
| IV - Indexation et visualisation | 8 |
| V - Automatisation Airflow | 9 |

I - Introduction

Ce rapport est réalisé dans le cadre du projet de BigData.

Il présentera le traitement entier de la donnée : acquisition, traitements et analyse.

Sujet choisi :

Nous avons décidé d'étudier **l'algorithme de recommandation de Youtube**. Plus précisément, nous souhaitons essayer de **déterminer quels sont les facteurs caractéristiques d'une chaîne Youtube populaire**. Pour ce faire, nous disposerons de plusieurs tables de données récupérées dynamiquement, à savoir :

- Le TOP 100 des chaînes françaises selon le nombre d'abonnés ([Web Scrapping](#)) ;
- Des statistiques sur les 25 dernières vidéos de toutes les chaînes du TOP 100 (API Youtube).

Technologies utilisées :

Les principaux outils et technologies utilisés dans ce projet sont les suivants :

- Airflow (local) ;
- Elasticsearch (online) ;
- Kibana (online) ;
- PostgreSQL (online) ;
- Dbt (local, repo : https://github.com/jugodfroy/dbt_BigDataProject) ;
- Web Scrapping Python : BeautifulSoup4.







II - Ingestion des données

Pour notre projet, nous avons besoin des deux sources suivantes :

1. Le TOP 100 des chaînes françaises selon le nombre d'abonnés ([Web Scrapping](#)) ;
2. Des statistiques sur les 25 dernières vidéos de toutes les chaînes du TOP 200 (API Youtube).

1. RECUPERATION DU TOP 100 DES CHAINES YOUTUBE FRANÇAISES :

L'API youtube ne permet pas de réaliser un quelconque classement entre les chaînes Youtube. Après quelques recherches, il est apparu plus simple de scraper le classement depuis la source suivante : socialblade.com/youtube/top/country/fr/mostsubscribed

| Rank | Grade | Username | Uploads | Subs | Video Views |
|-----------------------|-------|--|----------------------|---------------------------|-------------------------------------|
| 1st | B+ |  SQUEEZIE 🐾 | 1,515 | 18M | 9,893,750,177 |
| 2nd Ranking | B |  Cyprien 📺 category Channel name | 217 videos | 14.4M followers | 3,075,291,521 total views |
| 3rd | B+ |  Lofi Girl 🎵 | 396 | 12.8M | 1,640,749,784 |
| 4th | B |  Norman 😊 | 210 | 11.7M | 2,710,140,414 |
| 5th | B+ |  OGGY 🎬 | 1,050 | 11.6M | 5,955,959,930 |
| | |  https://socialblade.com/youtube/channel/UCyWqModMQblo8274Wh_ZsQ Channel id | 158 | 11.1M | 6,223,056,043 |

Depuis ce site, nous pouvons récupérer le TOP 100 des chaînes Youtube françaises avec les données suivantes pour chaque chaîne :

- Ranking ;
- Channel_name ;
- Channel_Id ;
- Channel_link ;

- Category ;
- Number of followers ;
- Total views.

Les données sont ensuite stockées dans une base de données PostgreSQL en ligne, ce qui donne le résultat suivant :

| landing_channels_ranking | | | | | | | | | |
|--------------------------|------|------------------------------|--|---------------------------|---------|----------|------------|---------------|--|
| | rank | channel_name | channel_link | channel_id | uploads | subs | views | category | |
| 1 | 1 | SQUEEZIE | /youtube/channel/UCWeg2Pkate69NfDBeuRFTAw | UCWeg2Pkate69NfDBeuRFTAw | 1514 | 18000000 | 9884927323 | entertainment | |
| 2 | 2 | Cyprien | /youtube/channel/UCyWqModMQIblo8274Wh_ZsQ | UCyWqModMQIblo8274Wh_ZsQ | 217 | 14400000 | 3074762148 | comedy | |
| 3 | 3 | Lofi Girl | /youtube/channel/UCSJ4gkVC6Nrvll8umztf0Ow | UCSJ4gkVC6Nrvll8umztf0Ow | 395 | 12800000 | 1634753601 | music | |
| 4 | 4 | Norman | /youtube/channel/UCww2zZWg4Cf5xcRKG-ThmXQ | UCww2zZWg4Cf5xcRKG-ThmXQ | 210 | 11700000 | 2709954513 | comedy | |
| 5 | 5 | OGGY | /youtube/channel/UCNEKMkg_DG8eAyR1BNWsSvw | UCNEKMkg_DG8eAyR1BNWsSvw | 1048 | 11600000 | 5950498800 | film | |
| 6 | 6 | GIMS | /youtube/channel/UCCB1Byx5yTbLpQaV-rifmtA | UCCB1Byx5yTbLpQaV-rifmtA | 158 | 11100000 | 6218110178 | music | |
| 7 | 7 | Soolking Officiel | /youtube/channel/UCz6JjQtnK9XjMwKuqLEkRw | UCz6JjQtnK9XjMwKuqLEkRw | 63 | 9820000 | 2715553924 | music | |
| 8 | 8 | Lama Faché | /youtube/channel/UCH0XvUpYcxn4V0iZGnZXMnQ | UCH0XvUpYcxn4V0iZGnZXMnQ | 2171 | 9450000 | 2363738606 | entertainment | |
| 9 | 9 | Tibo InShape | /youtube/channel/UCpWaR3gNAQGsX48cllQC0qw | UCpWaR3gNAQGsX48cllQC0qw | 1307 | 9390000 | 3943050785 | entertainment | |
| 10 | 10 | Michou | /youtube/channel/UCo3i0nUzZjLuM7VjAVz4zA | UCo3i0nUzZjLuM7VjAVz4zA | 697 | 8470000 | 2183755846 | entertainment | |
| 11 | 11 | L'Algerino | /youtube/channel/UCp8LmJx1i93YVokA4fNAs1g | UCp8LmJx1i93YVokA4fNAs1g | 116 | 8250000 | 2813136823 | music | |
| 12 | 12 | VLOG | /youtube/channel/UCDnZvnYAnTPywqPlj1jwm_g | UCDnZvnYAnTPywqPlj1jwm_g | 9 | 8160000 | 320735 | entertainment | |
| 13 | 13 | Amixem | /youtube/channel/UCgqvqBoSHB1ctlyhoHrGwQ | UCgqvqBoSHB1ctlyhoHrGwQ | 760 | 8090000 | 2762637507 | comedy | |
| 14 | 14 | Osratouna tv - قناة أسرتن... | /youtube/channel/UCWldqSQeKeGmUWISFeCiEnA | UCWldqSQeKeGmUWISFeCiEnA | 431 | 7720000 | 5101420827 | music | |
| 15 | 15 | قناة الشيخ كشك | /youtube/channel/UCr73LItzbXNVVl4c2a0vLgXO | UCr73LItzbXNVVl4c2a0vLgXO | 4340 | 7590000 | 1611008775 | entertainment | |

Notons qu'en réalité, pour avoir des données le plus à jour possible, les colonnes *uploads*, *subs* and *views* sont récupérées directement grâce à l'api Youtube donc la méthode est identique à la partie ci-dessous.

2. RECUPERATION DES STATISTIQUES SUR LES 25 DERNIERES VIDEOS DE CHAQUE CHAINE DU TOP100

L'API Youtube nous permet de récupérer de nombreuses données concernant la plateforme. Voici les deux requêtes Youtube qui nous sont utiles :

```

• request = youtube.search().list(
    part="snippet",
    channelId=channel_id,
    maxResults=25,
    order="date",
    type="video"
)
response = request.execute()

```

Avec cette première requête, nous pouvons récupérer une liste contenant l'id des 25 dernières vidéos d'une chaîne Youtube donnée.

```

• request = youtube_token.videos().list(

    part="statistics",
    id=video_id
)
response = request.execute()

```

Cette deuxième requête nous permet d'avoir de nombreuses statiques pour une vidéo donnée par son id unique : à savoir :

- Son titre ;
- Sa description ;
- Sa date de publication ;
- Son nombre de views ;
- Son nombre de commentaires ;
- Son nombre de like.

Les données sont à chaque fois renvoyées en json, et sont traitées pour être converties en *pandas.dataframe* puis être envoyé sur la base de données POSTGRESQL.

| | asc channel_id | asc video_id | asc title | asc description | asc publishedAt | asc viewCount | asc likeCount |
|----|-------------------------|--------------|---|--|----------------------|---------------|---------------|
| 1 | UCWeg2Pkate69NfdeuRFTAw | WAeqwwWxq_k | QUI EST L'IMPOSTEUR ? (ft Jérôme Niel & Panayotis | Merci à Vinted d'avoir sponsorisé cette vidéo, vous pouvez su | 2023-05-15T17:33:30Z | 6519621 | 390259 |
| 2 | UCWeg2Pkate69NfdeuRFTAw | i2obqgmjvY | Les pires idées des scientifiques... | Merci à Honkai: Star Rail d'avoir sponsorisé la vidéo ! Téléchar | 2023-05-07T14:47:26Z | 4789802 | 331366 |
| 3 | UCWeg2Pkate69NfdeuRFTAw | ypT5BGms-Z4 | Le meilleur duo évite la prison #2 (ft mon frère, Bigflo & | Merci à Rhinoshield d'avoir sponsorisé cette vidéo, vous pouve | 2023-04-30T16:48:52Z | 7300496 | 407472 |
| 4 | UCWeg2Pkate69NfdeuRFTAw | okKBQ1Gp5pQ | Ce prêteur mène une double vie incroyable | Retrouve toutes les vidéos histoires dans la playlist officielle ... | 2023-04-10T17:13:36Z | 4017129 | 366192 |
| 5 | UCWeg2Pkate69NfdeuRFTAw | dWbPdqs5Xk | QUI EST L'IMPOSTEUR ? (ft Pierre Niney & François | Merci à Fruitz d'avoir sponsorisé cette vidéo ! Vous pouvez télé | 2023-04-05T17:48:48Z | 10326466 | 550130 |
| 6 | UCWeg2Pkate69NfdeuRFTAw | hAKdXopG31M | Ces marques sont allées trop loin... | Retrouve toutes les vidéos histoires dans la playlist attitrée ... | 2023-03-29T17:02:11Z | 6054706 | 431887 |
| 7 | UCWeg2Pkate69NfdeuRFTAw | 0o4_-8LQOjk | T'es pas drôle, tu sors (ft Joyca, Maxenss) | Merci à Prime Video d'avoir sponsorisé cette vidéo ! Tous les é | 2023-03-21T17:50:22Z | 7315853 | 446072 |
| 8 | UCWeg2Pkate69NfdeuRFTAw | 6XGj35f-row | QUI EST L'IMPOSTEUR ? (ft SCH & Soso Maness) | Retrouvez SCH le 24 Mai à l'Accorhotels Arena et le 22 Juillet à | 2023-03-12T11:43:33Z | 10135743 | 531579 |
| 9 | UCWeg2Pkate69NfdeuRFTAw | N-TClquxFk | Le meilleur duo évite la prison (ft Mister V, Léna, Theo Juice) | Merci à NordVPN d'avoir sponsorisé la vidéo ! Télécharge Non | 2023-03-05T14:38:01Z | 12826683 | 648306 |
| 10 | UCWeg2Pkate69NfdeuRFTAw | r67zVQK7zE0 | Elle a fait une sombre découverte... (4 vraies histoires d' | Merci à @bagherajones, Nelly et @lilcursedlays (Laissy) pour | 2023-02-22T17:49:02Z | 6389452 | 370922 |
| 11 | UCWeg2Pkate69NfdeuRFTAw | 3kRyQsFRj3E | Il a fait exploser le casino qui l'a ruiné | Laissez-moi vous raconter l'histoire d'une vengeance sans préc | 2023-02-15T18:12:15Z | 5639746 | 393072 |

III - Transformation avec dbt

DBT, acronyme de "Data Build Tool", est une plateforme open-source conçue pour faciliter le développement et la gestion de flux de données dans le domaine de l'analyse de données. Il s'agit d'un outil populaire utilisé par de nombreuses entreprises et équipes d'ingénierie des données. DBT permet de transformer et de modéliser les données, en utilisant du code SQL, afin de les préparer pour des analyses ultérieures. Il offre des fonctionnalités telles que la gestion des dépendances, la documentation automatique et la vérification des erreurs de syntaxe SQL.

Après avoir installé dbt core et configuré l'accès à notre base de données POSTGRES, nous avons défini nos deux tables précédentes comme sources de transformation (cf. photo de gauche), puis nous avons écrit notre transformation en sql (cf. photo de droite) :

landing_sources.yml X

models > sources > ! landing_sources.yml

```

1 version: 2
2
3 sources:
4   - name: landing_sources
5     database: postgres
6     schema: godfroy_dutilleul
7     description: This is a replica of the Postgres database used by our app
8     tables:
9       Generate model
10      - name: landing_video_stats
11        description: >
12          Data on the 25 last published videos for each channel of the TOP 100
13        columns:
14          - name: channel_id
15            description: unique id of the youtube channel which published the video
16          - name: title
17            description: video title
18          - name: description
19            description: description of the video made by the publisher
20          - name: publishedAt
21            description: datetime the video has been published
22          - name: viewCount
23            description: number of views of the video
24          - name: likeCount
25            description: number of likes of the video
26          - name: commentCount
27            description: number of comments of the video
28          - name: video_id
29            description: unique id of the video
30
31      Generate model
32      - name: landing_channels_ranking
33        description: >
34          Data on the TOP 100 French YouTube channels ranked by the subscribers (DESC)
35        columns:
36          - name: rank
37            description: youtube channel ranking in the TOP 100

```

marts_youtube_data.sql X

models > marts_youtube_data.sql

```

1 select
2   t2.title,
3   t2.description,
4   cast(t2."publishedAt" as timestamp) as "publishedAt",
5   cast(t2."viewCount" as bigint) as "viewCount",
6   cast(t2."likeCount" as bigint) as "likeCount",
7   cast(t2."commentCount" as integer) as "commentCount",
8   t2.video_id,
9
10  cast(t1."rank" as integer) as channel_rank,
11  t1.channel_name,
12  t1.channel_link,
13  t1.channel_id,
14  cast(t1.uploads as bigint) as channel_uploads,
15  cast(t1.subs as bigint) as channel_subs,
16  cast(t1."views" as bigint) as channel_totalviews,
17  t1.category as channel_categ
18
19 from {{ source('landing_sources', 'landing_channels_ranking') }} t1
20
21 inner join {{ source('landing_sources', 'landing_video_stats') }} t2
22   on t1.channel_id = t2.channel_id
23

```

La nouvelle table transformée viendra se sauvegarder dans PostgreSQL sous le nom marts_youtube_data, suivant les conventions de nommage de dbt.

Avec dbt nous pouvons utiliser la commande *dbt docs generate* & *dbt docs serve --port 8888* pour générer de façon automatique une documentation directement sur un serveur local :

The screenshot shows the dbt documentation interface at localhost:8888. The main content area displays the details for the **marts_youtube_data** table. The description states: "This table is the made from channels_ranking and videos_stats tables combination." The columns table lists the following columns and their descriptions:

| COLUMN | TYPE | DESCRIPTION |
|--------------|-----------------------------|--|
| title | text | Video title |
| description | text | Description of the video made by the publisher |
| publishedAt | timestamp without time zone | Datetime the video has been published |
| viewCount | bigint | Number of views of the video |
| likeCount | bigint | Number of likes of the video |
| commentCount | integer | Number of comments of the video |

On the right side, the Lineage Graph shows the data flow from the sources **landing_sources.landing_channels_ranking** and **landing_sources.landing_video_stats** to the target **marts_youtube_data**.

Cette documentation permet à un analyste de comprendre les différentes tables, leurs dépendances et leurs données.

IV - Indexation et visualisation

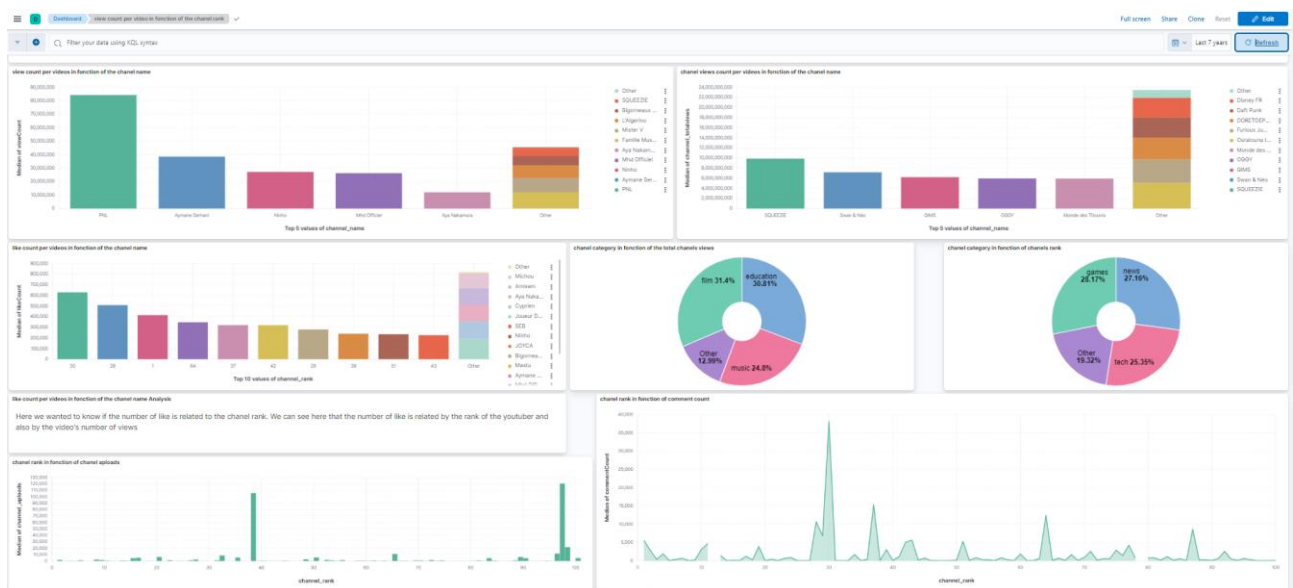
A l'aide d'un script python, nous pouvons indexer nos données vers Elasticsearch.

Ce script se déroule en plusieurs étapes, à savoir :

- Connexion à la base de données PostgreSQL ;
- Récupération de la table sous forme de *pandas.dataframe* ;
- Connexion à elasticsearch et création de l'index *video_data* ;
- Envoie des données.

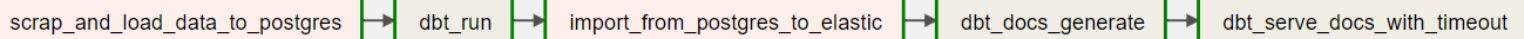
Un fois les données envoyées sur Elasticsearch, le dashboard se met à jour automatiquement.

Pour ce qui est de la visualisation et de l'analyse, nous l'avons entièrement réalisé avec Kibana. Voici notre dashboard :



V - Automatisation Airflow

Pour automatiser toute la chaîne de traitement de la donnée, nous avons mis en place un DAG Airflow. Ce DAG codé en python est composé des 5 tâches suivantes :



- **scrap_and_load_data_to_postgre** : pour exécuter les fonctions de scrapping du TOP 100 et de récupération des données vidéos via l'API Youtube et créer les 2 tables sources dans PostgreSQL ;
- **dbt_run** : pour run la transformation en se sourcant depuis les 2 tables précédentes pour créer de la table finale, stockée sur PostgreSQL ;
- **Import_from_postgre_to_elastic** : envoie de la table finale de PostgreSQL vers Elasticsearch ;
- **dbt_docs_generate** : permet la création de la documentation automatique dbt ;
- **dbt_serve_docs_with_timeout** : permet de mettre en ligne la documentation pendant 5 minutes.

```
with DAG(
    'youtube_dag',
    default_args=default_args,
    description='A simple dbt pipeline',
    schedule_interval='@daily',
) as dag:

    # Define your tasks
    t0 = PythonOperator(
        task_id='scrap_and_load_data_to_postgres',
        python_callable=scrape_and_load_data
    )

    t1 = execute_dbt_command('dbt run')

    t2 = PythonOperator(
        task_id='import_from_postgres_to_elastic',
        python_callable=load_to_elastic
    )

    t3 = execute_dbt_command('dbt docs generate')

    t4 = BashOperator(
        task_id='dbt_serve_docs_with_timeout',
        bash_command='cd /home/julien/dbt_BigDataProject/YoutubeBigData && timeout 200 dbt docs serve --port 8888 || true'
    )

    # Define your pipeline
    t0 >> t1 >> t2 >> t3 >> t4
```