Autre

Extraction et Analyse des Produits avec PySpark

1️⃣ Télécharger les images des produits

📌 Script PySpark pour télécharger les images

import os
import requests
from pyspark.sql import SparkSession
from pyspark.sql.functions import col

Initialisation de Spark

spark = SparkSession.builder.appName("DownloadImages").getOrCreate()

Chargement des données

df = spark.read.csv("path_to_data.csv", header=True, inferSchema=True)

Filtrer les produits sélectionnés (ex: top 100 par unique_scans_n)

df_filtered = df.orderBy(col("unique_scans_n").desc()).limit(100)

Créer le dossier si besoin

output_dir = "off_raw/images"
os.makedirs(output_dir, exist_ok=True)

Fonction pour télécharger une image

def download_image(image_url, code_produit):
if not image_url:
return
filename = f"{output_dir}/{code_produit}.jpg"
try:
response = requests.get(image_url, timeout=10)

```
if response.status_code == 200:
with open(filename, "wb") as f:
f.write(response.content)
except Exception as e:
print(f"Erreur téléchargement {image_url} : {e}")
```

Télécharger les images

```
df_filtered.select("image_url", "code_produit").toPandas().apply(
lambda row: download_image(row["image_url"], row["code_produit"]), axis=1
)
```

2️⃣ Job Spark qui donne un output par pays

📌 Job PySpark

```
output_path = "off_raw/output_by_country"

df_by_country = df.groupBy("pays").count()

df_by_country.write.mode("overwrite").csv(output_path, header=True)
```

3️⃣ Job pour calculer des statistiques (distribution des scores par pays)

📌 Calcul des statistiques par pays

```
from pyspark.sql.functions import avg, count, stddev

stats_df = df.groupBy("pays").agg(
avg("score").alias("moyenne_score"),
stddev("score").alias("ecart_type_score"),
count("code_produit").alias("nombre_produits")
)

stats_df.write.mode("overwrite").csv("off_raw/stats_by_country", header=True)
```

4️⃣ Sélectionner les top 10 produits par pays (unique_scans_n)

📌 Sélection des Top 10 par pays

```
from pyspark.sql.window import Window
from pyspark.sql.functions import rank

window_spec = Window.partitionBy("pays").orderBy(col("unique_scans_n").desc())

df_top10 = df.withColumn("rank", rank().over(window_spec)).filter(col("rank") <= 10)

df_top10.write.mode("overwrite").csv("off_raw/top10_products_by_country", header=True)
```

5️⃣ Splitting des données

📌 Splitting une colonne

```
from pyspark.sql.functions import split, explode

df_split = df.withColumn("categorie", explode(split(df["categories"], ", "))).select("produit", "categorie")
df_split.show()
```

📌 Splitting un DataFrame

```
train_df, test_df = df.randomSplit([0.7, 0.3], seed=42)
```

6️⃣ Conversion des dates en UTC

```
from pyspark.sql.functions import to_utc_timestamp

df_utc = df.withColumn("utc_timestamp", to_utc_timestamp(df["timestamp"], df["timezone"]))
```

7️⃣ Gestion des valeurs NULL

📌 Supprimer les lignes avec NULL

```
df_cleaned = df.dropna()
```

📌 Remplacer les valeurs NULL

```
df_filled = df.fillna({"nom": "Inconnu", "age": 0})
```

📌 Filtrer les NULLs

```
df_nulls = df.filter(df["colonne"].isNull())
```

8️⃣ Regrouper les produits similaires

```
from pyspark.sql.functions import regexp_replace

df_cleaned = df.withColumn("produit_normalise", regexp_replace(df.produit, "Nutella.*", "Nutella"))
```

9️⃣ Supprimer les données incohérentes

📌 Supprimer poids <= 0

```
df_cleaned = df.filter(df.poids > 0)
```

📌 Supprimer les valeurs aberrantes

```
q1, q3 = df.approxQuantile("poids", [0.25, 0.75], 0.05)
iqr = q3 - q1
lower_bound = q1 - 1.5 *iqr*
*upper_bound* = *q3* + *1.5* iqr

df_cleaned = df.filter((df.poids > lower_bound) & (df.poids < upper_bound))
```

🚀 Résumé des jobs Spark :

1️⃣ Télécharger les images dans off_raw/images/[code_produit].jpg.
2️⃣ Créer un output par pays (off_raw/output_by_country).
3️⃣ Calculer des statistiques sur les produits par pays (off_raw/stats_by_country).
4️⃣ Extraire les 10 produits les plus populaires par pays (off_raw/top10_products_by_country).
5️⃣ Gérer les valeurs NULL et nettoyer les données incohérentes.
6️⃣ Grouper les produits similaires.
7️⃣ Convertir les dates en UTC.
8️⃣ Splitting des données et colonnes.