TEDx
NEWS

2° HOMEWORK

Chiara Toffalori 1080710
Siria Ruggeri 1078977

LOAD DATA

```
# FROM FILES
tedx_dataset_path = "s3://tedx-2024-data-ruggeri/final_list.csv"
details_dataset_path = "s3://tedx-2024-data-ruggeri/details.csv"
images_dataset_path = "s3://tedx-2024-data-ruggeri/images.csv"
tags_dataset_path = "s3://tedx-2024-data-ruggeri/tags.csv"
```

Caricamento dei dati da S3: Definiamo i percorsi dei file CSV memorizzati in un bucket S3

Lettura e pulizia del Dataset TEDX

```
# READ TEDX DATASET
tedx_dataset = spark.read \
    .option("header", "true") \
    .option("quote", "\"") \
    .option("escape", "\"") \
    .csv(tedx_dataset_path)

# FILTER NULL POSTING KEY
count_items = tedx_dataset.count()
count_items_null = tedx_dataset.filter("id is not null").count()

print(f"Number of items from RAW DATA {count_items}")
print(f"Number of items from RAW DATA with NOT NULL KEY {count_items_null}")
```

LOAD DATA

```
# READ DETAILS DATASET
details_dataset = spark.read \
    .option("header", "true") \
    .option("quote", "\"") \
    .option("escape", "\"") \
    .csv(details_dataset_path) \
    .select(F.col("id").alias("id_ref"),
            F.col("description"),
            F.col("duration"),
            F.col("publishedAt"))

# JOIN WITH TEDX DATASET
tedx_dataset_main = tedx_dataset.join(details_dataset, tedx_dataset.id == details_dataset.id_ref, "left") \
    .drop("id_ref")
# READ IMAGES DATASET
images_dataset = spark.read \
    .option("header", "true") \
    .option("quote", "\"") \
    .option("escape", "\"") \
    .csv(images_dataset_path) \
    .select(F.col("id").alias("id_ref"),
            F.col("url").alias("image_url"))

# JOIN WITH TEDX DATASET
tedx_dataset_main = tedx_dataset_main.join(images_dataset, tedx_dataset_main.id == images_dataset.id_ref, "left") \
    .drop("id_ref")
```

Lettura e Unione del Dataset Details e Images:

- Leggiamo i dataset e selezioniamo alcune colonne rinominandole.
- Eseguiamo poi un join a sinistra tra tedx_dataset e details_dataset/images_dataset sulla colonna id, mantenendo tutte le righe da tedx_dataset e aggiungendo i dettagli/le immagini.

LOAD DATA

```
# READ TAGS DATASET
tags_dataset = spark.read.option("header", "true").csv(tags_dataset_path)

# AGGREGATE MODEL, ADD TAGS TO TEDX_DATASET
tags_dataset_agg = tags_dataset.groupBy(F.col("id").alias("id_ref")).agg(F.collect_list("tag").alias("tags"))

# JOIN TAGS WITH TEDX DATASET
tedx_dataset_agg = tedx_dataset_main.join(tags_dataset_agg, tedx_dataset_main.id == tags_dataset_agg.id_ref, "left") \
    .drop("id_ref") \
    .select(F.col("id").alias("_id"),
            F.col("slug"),
            F.col("speakers"),
            F.col("title"),
            F.col("url"),
            F.col("description"),
            F.col("duration"),
            F.col("publishedAt"),
            F.col("image_url"),
            F.col("tags"))

# AGGREGATE ALL TAGS FOR EACH ITEM
all_tags_dataset = tags_dataset.groupBy(F.col("id").alias("id_ref")).agg(F.collect_list("tag").alias("all_tags"))

# JOIN ALL TAGS WITH TEDX DATASET
tedx_dataset_agg_final = tedx_dataset_agg.join(all_tags_dataset, tedx_dataset_agg._id == all_tags_dataset.id_ref, "left") \
    .drop("id_ref") \
    .select(F.col("_id"),
            F.col("slug"),
            F.col("speakers"),
            F.col("title"),
            F.col("url"),
            F.col("description"),
            F.col("duration"),
            F.col("publishedAt"),
            F.col("image_url"),
            F.col("all_tags").alias("tags"))


# PRINT SCHEMA
tedx_dataset_agg_final.printSchema()
```

Lettura e Aggregazione del Dataset Tag:

- Leggiamo il dataset dei tag
- Aggreghiamo i tag per ogni id, collezionandoli in una lista.
- Uniamo il dataset principale con i tag aggregati, mantenendo tutte le righe da tedx_dataset_main e aggiungendo i tag.
- In fine aggreghiamo tutti i tag per ogni id e unisce con il dataset finale, rinominando la colonna all_tags in tags.
- Stampiamo lo schema del DataFrame finale per verificare la struttura dei dati prima della scrittura su MongoDB.

LOAD DATA

```
# WRITE TO MONGODB
write_mongo_options = {
    "connectionName": "TEDx2024",
    "database": "unibg_tedx_2024",
    "collection": "tedx_data",
    "ssl": "true",
    "ssl.domain_match": "false"
}

from awsglue.dynamicframe import DynamicFrame

tedx_dataset_dynamic_frame = DynamicFrame.fromDF(tedx_dataset_agg_final, glueContext, "nested")

glueContext.write_dynamic_frame.from_options(tedx_dataset_dynamic_frame, connection_type="mongodb", connection_options=write_mongo_options)
```

Caricamento dati su Mongo DB: Configuriamo le opzioni di connessione a MongoDB e scriviamo il DataFrame finale come DynamicFrame nella collezione MongoDB specificata.

WATCH NEXT

```
#### READ INPUT FILES TO CREATE AN INPUT DATASET
# Read final_list.csv
final_list_path = "s3://tedx-2024-data-ruggeri/final_list.csv"
final_list_df = spark.read \
    .option("header", "true") \
    .option("quote", "\"") \
    .option("escape", "\"") \
    .csv(final_list_path) \
    .select(col("id").alias("video_id"),
            col("slug"),
            col("speakers"),
            col("title"),
            col("url"))

# Read details.csv
details_path = "s3://tedx-2024-data-ruggeri/details.csv"
details_df = spark.read \
    .option("header", "true") \
    .option("quote", "\"") \
    .option("escape", "\"") \
    .csv(details_path) \
    .select(col("id").alias("detail_id"),
            col("description"),
            col("duration"),
            col("socialDescription"),
            col("presenterDisplayName"),
            col("publishedAt"))
```

Lettura :

- Definiamo la sezione in cui i file CSV verranno letti e trasformati in DataFrame per l'elaborazione.
- Leggiamo i files CSV da S3 e seleziona le colonne desiderate, rinominandole per chiarezza
- Leggiamo le informazioni sulle immagini e rinomina le colonne per abbinare gli id delle immagini ai video.
- Leggiamo le informazioni sui video correlati e rinomina le colonne per chiarire il ruolo degli ID e dei titoli dei video correlati
- Leggiamo i tag associati ai video e rinomina le colonne per gestire i tag in modo più chiar

```
# Read images.csv
images_path = "s3://tedx-2024-data-ruggeri/images.csv"
images_df = spark.read \
    .option("header", "true") \
    .option("quote", "\"") \
    .option("escape", "\"") \
    .csv(images_path) \
    .select(col("id").alias("image_id"),
            col("url").alias("image_url"))

# Read related_videos.csv
related_videos_path = "s3://tedx-2024-data-ruggeri/related_videos.csv"
related_videos_df = spark.read \
    .option("header", "true") \
    .option("quote", "\"") \
    .option("escape", "\"") \
    .csv(related_videos_path) \
    .select(col("id").alias("related_id"),
            col("related_id").alias("related_video_id"),
            col("title").alias("related_video_title"))

# Read tags.csv
tags_path = "s3://tedx-2024-data-ruggeri/tags.csv"
tags_df = spark.read \
    .option("header", "true") \
    .csv(tags_path) \
    .select(col("id").alias("tag_id"),
            col("tag"))
```

WATCH NEXT

Aggregazione
Questa sezione esegue join tra i DataFrame per combinare le informazioni dai diversi dataset in un unico DataFrame

```
#### JOIN DATASETS

# Join final_list with details
main_df = final_list_df.join(details_df, final_list_df.video_id == details_df.detail_id, "left") \
    .drop("detail_id")

# Join with images
main_df = main_df.join(images_df, main_df.video_id == images_df.image_id, "left") \
    .drop("image_id")

# Join with tags
tags_agg = tags_df.groupBy(col("tag_id")).agg(collect_list("tag").alias("tags"))
main_df = main_df.join(tags_agg, main_df.video_id == tags_agg.tag_id, "left") \
    .drop("tag_id")

# Join with related videos
related_videos_agg = related_videos_df.groupBy(col("related_id")).agg(
    collect_list("related_video_id").alias("related_videos_id_list"),
    collect_list("related_video_title").alias("title_related_videos_list")
)
main_df = main_df.join(related_videos_agg, main_df.video_id == related_videos_agg.related_id, "left") \
    .drop("related_id")
```

WATCH NEXT - filtraggio per tag

Filtraggio per Tag Richiesti:

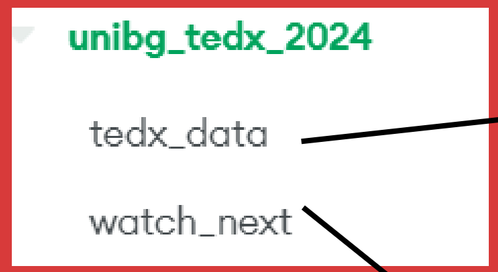
- Definiamo una lista di tag precisi e filtriamo il dataset per mantenere solo le righe che contengono almeno uno di questi tag.
- Abbiamo scelto questi tag in quanto i più inerenti a possibili video di attualità
- Eliminiamo poi eventuali duplicati
- Stampiamo lo schema del DataFrame per assicurarci che le colonne e i dati siano come previsto.

```
#### FILTER DATA BASED ON TAGS

required_tags = ["society", "politics", "economics", "health", "sustainability"]
filtered_df = main_df.filter(
    col("tags").isNotNull() & (
        array_contains(col("tags"), required_tags[0]) |
        array_contains(col("tags"), required_tags[1]) |
        array_contains(col("tags"), required_tags[2]) |
        array_contains(col("tags"), required_tags[3]) |
        array_contains(col("tags"), required_tags[4])
    )
)

# Remove duplicates if any
filtered_df = filtered_df.dropDuplicates()

# Print schema to verify
filtered_df.printSchema()
```

MONGODB

Creazione all'interno del databse unibg_tedx_2024 delle collezioni tedx_data e watch_next

Esempi di viste: abbiamo preso in considerazione un _id in cui siamo sicuri sia presente almeno un tag tra quelli selezionati (in questo caso "sustainability"). Per esempio l'_id numero 527254 non presenta nessun tag di quelli filtrati e quindi nella collezione watch_next non viene mostrato




```
_id: "528495"
slug : "ryan_panchadsaram_anjali_grover_and_david_biello_an_updated_action_pla…"
speakers : "Ryan Panchadsaram, Anjali Grover and David Biello"
title : "An updated action plan for solving the climate crisis — and a look at …"
url : "https://www.ted.com/talks/ryan_panchadsaram_anjali_grover_and_david_bi…"
description : "When it comes to climate, what are we doing right and where should we …"
duration : "722"
publishedAt : "2024-04-29T13:28:48Z"
image_url : "https://talkstar-photos.s3.amazonaws.com/uploads/585acd28-d676-449a-a0…"
▾ tags : Array (8)
    0: "climate change"
    1: "environment"
    2: "sustainability"
    3: "social change"
    4: "pollution"
    5: "electricity"
    6: "Countdown"
    7: "fossil fuels"
```



```
_id: "528495"
slug : "ryan_panchadsaram_anjali_grover_and_david_biello_an_updated_action_pla…"
speakers : "Ryan Panchadsaram, Anjali Grover and David Biello"
title : "An updated action plan for solving the climate crisis — and a look at …"
url : "https://www.ted.com/talks/ryan_panchadsaram_anjali_grover_and_david_bi…"
description : "When it comes to climate, what are we doing right and where should we …"
duration : "722"
socialDescription : "When it comes to climate, what are we doing right and where should we …"
presenterDisplayName : "Ryan Panchadsaram, Anjali Grover and David Biello"
publishedAt : "2024-04-29T13:28:48Z"
image_url : "https://talkstar-photos.s3.amazonaws.com/uploads/0476ee96-0b0e-4d00-88…"
▸ tags : Array (8)
▸ related_videos_id_list : Array (6)
▸ title_related_videos_list : Array (6)
```

MONGODB


```
▾ tags : Array (8)
    0: "climate change"
    1: "environment"
    2: "sustainability"
    3: "social change"
    4: "pollution"
    5: "electricity"
    6: "Countdown"
    7: "fossil fuels"
▾ related_videos_id_list : Array (6)
    0: "83767"
    1: "115855"
    2: "108932"
    3: "192"
    4: "243"
    5: "1380"
▾ title_related_videos_list : Array (6)
    0: "An action plan for solving the climate crisis"
    1: "What the fossil fuel industry doesn't want you to know"
    2: "How do we get the world off fossil fuels quickly and fairly?"
    3: "A critical look at geoengineering against climate change"
    4: "New thinking on the climate crisis"
    5: "Why I must speak out about climate change"
```

Implementazioni

- Il codice integra diversi dataset relativi ai video TEDx, come dettagli, video correlati e tag, arricchendo il dataset principale con informazioni utili. Questo è utile per creare un database completo e integrato che può essere utilizzato per analisi avanzate

- Analisi dei dati dettagliati sui video TEDx per identificare tendenze e confrontare con altri contenuti, identificare tendenze emergenti

Criticità

- Gestione delle prestazioni - Volume dei dati: Se i dataset sono molto grandi, potrebbero esserci problemi di memoria o prestazioni.

- Manutenzione del Codice:
  - Mantenere e aggiornare il codice per adattarsi ai cambiamenti nei dataset e nelle API
  - Documentare accuratamente il codice per facilitare la manutenzione e gli aggiornamenti futuri.

TEDxNEWS

TRELLO BOARD

GIT HUB