

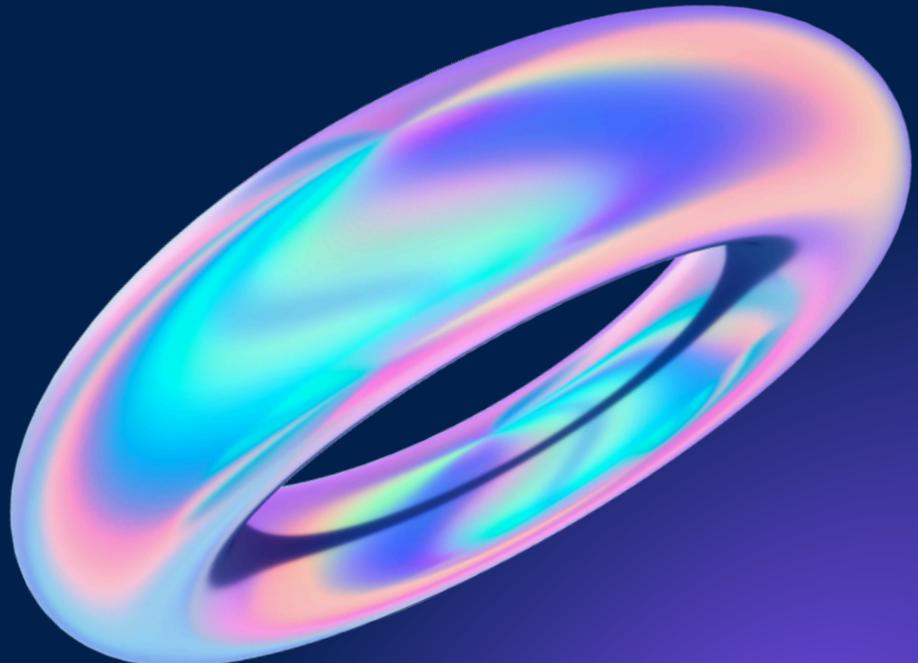
Tecnologie cloud & mobile

[Trello](#) [GitHub](#)



HealthyTEDx

Migliora la tua salute



Luisi Giuseppe 1073970
Zanchi Michael 1080756

WATCH NEXT

```
related_video_path = "s3://tedx-prova-2024/related_videos.csv"
related_video = spark.read \
    .option("header", "true") \
    .option("quote", "") \
    .option("escape", "\") \
    .csv(related_video_path)

## DELETE DUPLICATES
related_video = related_video.dropDuplicates()
related_video_agg = related_video.groupBy(col("id").alias("id_rel")) \
    .agg(collect_list("related_id").alias("id_related_video"),
         collect_list("title").alias("title_related_video"))

## JOIN WITH DATASET
tedx_dataset_agg_with_related_video = tedx_dataset_main.join(
    related_video_agg, tedx_dataset_main.id == related_video_agg.id_rel, "left") \
    .drop("id_rel") \
    .select(col("id").alias("id_related"), col("*"))

tedx_dataset_agg_with_related_video.printSchema()
```

In questo primo job, abbiamo integrato i dati dei related video (cioè i "watch next") con il dataset di partenza. Dopo aver eseguito la lettura del file, abbiamo eseguito un primo filtraggio dei dati, eliminando i possibili duplicati. Successivamente, abbiamo aggregato i video raccogliendo gli ID e i titoli dei video correlati in liste. Infine, abbiamo eseguito una join tra i dati dei video aggregati e il dataset principale, e abbiamo stampato il risultato, aggiornando i dati su MongoDB Atlas

Esempio MongoDB

```
_id: "526880"
id_related : "526880"
slug : "george_zaidan_how_do_gas_masks_actually_work"
speakers : "George Zaidan"
title : "How do gas masks actually work?"
url : "https://www.ted.com/talks/george_zaidan_how_do_gas_masks_actually_work"
description : "You might think of gas masks as clunky military-looking devices. But i..."
duration : "254"
publishedAt : "2024-04-30T15:14:51Z"
image_url : "https://talkstar-assets.s3.amazonaws.com/production/talks/talk_128547/..."
► id_related_video : Array (3)
    ▶ id_related_video : Array (3)
        0: "100294"
        1: "109914"
        2: "76541"
    ▶ title_related_video : Array (3)
        0: "Why plague doctors wore beaked masks"
        1: "Whatever happened to the hole in the ozone layer?"
        2: "What's in the air you breathe?"
► tags : Array (8)
    0: "environment"
    1: "technology"
    2: "design"
    3: "education"
    4: "natural disaster"
    5: "chemistry"
    6: "TED-Ed"
    7: "animation"
```

Questo è un esempio dei risultati che abbiamo ottenuto dopo aver implementato la funzionalità dei “watch next” all’interno dei dataset principale

JOB 2

Script completo

Per quanto riguarda il job 2 abbiamo deciso di implementare la funzionalità riguardante i tag. Partendo dal file csv, riguardante i tag, visto a lezione, abbiamo pensato ad una serie di tag che potevano essere utili alla nostra applicazione. Quindi dopo aver letto i file, presente in S3, abbiamo definito una lista di nuovi tag di interesse. Utilizzando questa lista, il dataset viene filtrato in modo tale che includa solo i tag pertinenti.

Dopo aver filtrato questi tag, abbiamo creato un file di output in S3. Questo file csv è stato poi riutilizzato in seguito

```
# Definisci i tag di interesse
new_tags = ['nutrition', 'food', 'diet', 'healthy', 'vegetarian', 'veganism', 'protein', 'carbohydrates', 'vitamins', 'minerals', 'calories',
    'cuisine', 'recipes', 'meals', 'eat', 'sports', 'exercise', 'fitness', 'physical activity', 'motivation']

# Filtra i tag relativi alla nutrizione e allo sport
filtered_tags_dataset = tags_dataset.filter(
    col("tag").isin(new_tags))
|
# output con i nuovi tag
output_path = "s3://tedx-prova-2024/filtered_tags.csv"

# Scrivi il dataset filtrato in output
filtered_tags_dataset.write.mode("overwrite").csv(output_path, header=True)
```

ESEMPI JOB 2

Ecco qualche esempio

```
_id: ObjectId('66675cf37b77992d7e76fc9f')
id : "466416"
slug : "malachy_mchugh_how_stretching_actually_changes_your_muscles"
internalId : "104174"
tag : "exercise"
```

```
_id: ObjectId('66675cf37b77992d7e76fc9d')
id : "527254"
slug : "bonnie_hancock_my_epic_journey_becoming_the_fastest_person_to_paddle_a..."
internalId : "128758"
tag : "sports"
```

```
_id: ObjectId('66675cf37b77992d7e76fc9f')
id : "525470"
slug : "louise_mabulo_a_climate_solution_the_wisdom_passed_down_through_genera..."
internalId : "127861"
tag : "food"
```

JOB 2.1

Script completo

```
## READ TAGS DATASET
tags_dataset_path = "s3://tedx-prova-2024/filtered_tags.csv/part-00000-9d31db9c-25da-49ca-8d6f-63336ebbf33d-c000.csv"
tags_dataset = spark.read.option("header","true").csv(tags_dataset_path)

# CREATE THE AGGREGATE MODEL, ADD TAGS TO TEDX_DATASET
tags_dataset_agg = tags_dataset.groupBy(col("id").alias("id_ref")).agg(collect_list("tag").alias("tags"))
tags_dataset_agg.printSchema()
tedx_dataset_agg = tedx_dataset_agg_with_related_video.join(
    tags_dataset_agg, tedx_dataset.id == tags_dataset_agg.id_ref, "left") \
    .drop("id_ref") \
    .select(col("id").alias("_id"), col("*")) \
    .drop("id")
tedx_dataset_agg.printSchema()

tedx_dataset_filtered = tedx_dataset_agg.filter(col("tags").isNotNull() & (size(col("tags")) > 0))
tedx_dataset_filtered.printSchema()
```

Dopo aver creato il nuovo file csv, l'abbiamo inserito all'interno dello script creato in precedenza (quello dei "watch next") in modo tale da poter collegare i tag ai video.

Usando questo nuovo file alcuni video non avevano nessun tag, quindi abbiamo deciso di eseguire un ulteriore filtraggio. Nello specifico abbiamo eliminato tutti i video che non possedevano nessun tag. Così facendo abbiamo ottenuto una lista di video con associati i tag filtrati.

ESEMPIO JOB 2.1

```
_id: "519458"
id_related : "519458"
slug : "dan_kwartzler_one_of_the_world_s_oldest_condiments"
speakers : "Dan Kwartler"
title : "One of the world's oldest condiments"
url : "https://www.ted.com/talks/dan_kwartzler_one_of_the_world_s_oldest_condi..."
description : "In the mid-18th century, England was crazy for ketchup. The sauce was ..."
duration : "300"
publishedAt : "2024-03-12T15:03:49Z"
image_url : "https://talkstar-assets.s3.amazonaws.com/production/talks/talk_125407/..."
> id_related_video : Array (3)
```

```
  ▾ id_related_video : Array (3)
    0: "104173"
    1: "112485"
    2: "68197"
  ▾ title_related_video : Array (3)
    0: "Why do we eat popcorn at the movies?"
    1: "Food expiration dates don't mean what you think"
    2: "The dark history of bananas"
  ▾ tags : Array (1)
    0: "food"
```

CRITICITÀ

- Una prima criticità riguarda il fatto che alcuni tag, pur essendo pertinenti all'argomento della nostra applicazione, potrebbero comunque riportare a video non rilevanti per l'applicazione stessa.
- un'altra criticità di questa implementazione potrebbe essere la poca presenza di video inerenti all'argomento dell'applicazione. Per risolvere questa criticità si potrebbe pensare di scaricare ulteriori video e integrarli al database.

FINE