

# LinguaLearnX

Tomasoni Francesco - 1080980

Homework 2

# Job PySpark - Watch Next

Il Job seguente raggruppa i dati per ogni id, creando una lista di video correlati per ciascun video. Infine, unisce queste informazioni aggregate con il dataset esistente, mantenendo tutti i record del dataset TEDx e aggiungendo le liste di video correlati dove ci sono corrispondenze.

```
# READ WATCH_NEXT DATASET
watch_next_data_path = "s3://tomasonidata/related_videos.csv"
watch_next_data = spark.read.option("header", "true").csv(watch_next_data_path)

# ADD WATCH_NEXT TO AGGREGATE MODEL
watch_next_data_agg = watch_next_data.groupBy(col("id").alias("id_reference_watch_next")) \
    .agg(collect_list("related_id").alias("related_id"))
tedx_dataset_agg = tedx_dataset_agg.join(watch_next_data_agg,
    tedx_dataset_agg._id == watch_next_data_agg.id_reference_watch_next,
    "left").drop("id_reference_watch_next")
```

# Watch Next-Dataset

Dopo l'esecuzione del Job, il dataset avrà un nuovo campo chiamato "related\_id".

```
_id: "527254"  
slug: "bonnie_hancock_my_epic_journey_becoming_the_fastest_person_to_paddle_a..."  
speakers: "Bonnie Hancock"  
title: "My epic journey becoming the fastest person to paddle around Australia"  
url: "https://www.ted.com/talks/bonnie_hancock_my_epic_journey_becoming_the_..."  
description: "What challenges lie ahead of a staggering 12,700-kilometer paddle arou..."  
duration: "600"  
publishedAt: "2024-04-30T14:47:22Z"  
▼ tags: Array (4)  
  0: "sports"  
  1: "motivation"  
  2: "personal growth"  
  3: "humanity"  
▼ related_id: Array (6)  
  0: "86532"  
  1: "117221"  
  2: "123149"  
  3: "1981"  
  4: "1360"  
  5: "23981"
```

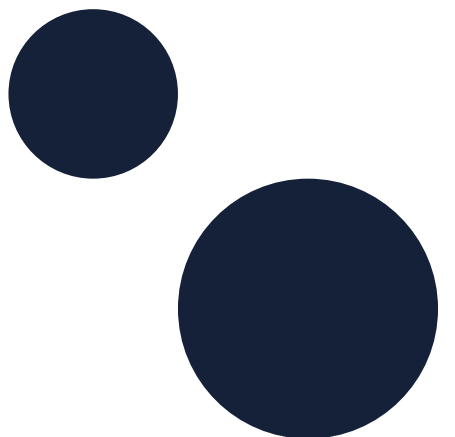
# Possibile utilizzo dei watch next in LinguaLearnX

Il campo "related\_id" contiene una lista di video correlati.

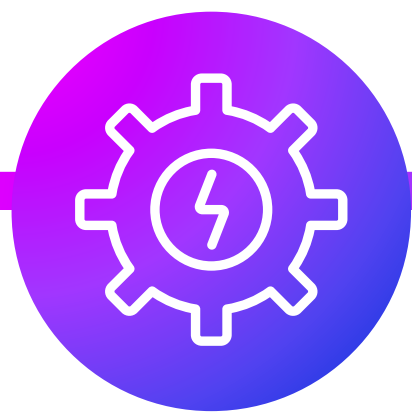
Pertanto, una possibile funzionalità aggiuntiva dell'applicazione potrebbe essere la creazione di collezioni basate sulle categorie dei video correlati.

In base al numero di video visualizzati e al numero di quiz superati, verrebbe assegnato un punteggio alla categoria.

Questo approccio favorirebbe un ulteriore apprendimento, sia della lingua che dei contenuti specifici.



# Implementazione del job dedicato:



## Scraper

Recupera le trascrizioni per ogni video TED presente nel Dataset



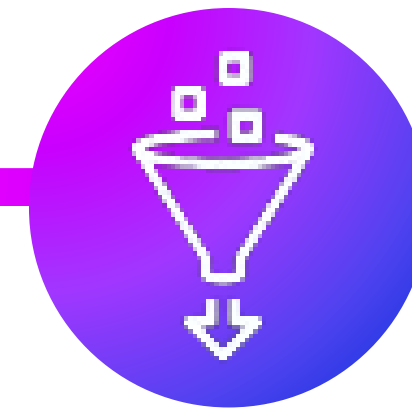
## Nuovo Dataset

il file "transcript\_dataset", conterrà l'id del TED con accanto la trascrizione



## Bucket

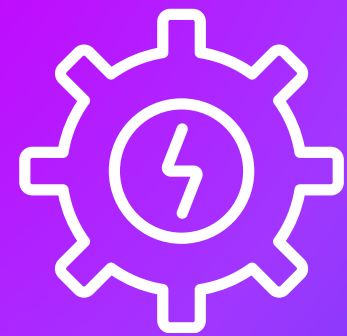
Il nuovo file contenente le trascrizioni viene inserito nel bucket S3



## Glue

Attraverso Job Glue viene aggiunta la colonna transcript al database presente su MongoDB e vengono etichettati i TED che non hanno una trascrizione





# Scraper

01

Per ogni id presente nel file  
“final\_list” viene aperta la  
pagina: [url] + '/transcript',

02

cerca uno script contenente un oggetto  
JSON che include vari dettagli del video,  
tra cui la trascrizione completa.

03

se viene trovato lo salva all'interno di un  
file “transcript\_dataset”, altrimenti l'id  
viene salvato in “missing\_transcripts”

04

Codice

```
# Configurare la connessione a MongoDB
```

```
mongo_options = {  
    "connectionName": "TEDX2024",  
    "database": "unibg_tedx_2024",  
    "collection": "tedx_data"  
}
```

```
# Leggere i dati da MongoDB
```

```
mongo_data = glueContext.create_dynamic_frame.from_options(  
    connection_type="mongodb",  
    connection_options=mongo_options  
)  
mongo_data.show()
```

```
# Percorso del bucket S3 contenente il dataset delle trascrizioni
```

```
transcript_data_path = 's3://tomasonidata/transcript_dataset.csv'
```

```
# Caricare il dataset delle trascrizioni da S3
```

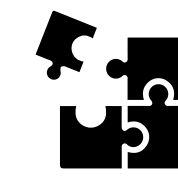
```
transcript_data = spark.read.option("header", "true").csv(transcript_data_path)  
transcript_data.show()
```

```
# Eseguire la join sui due dataset utilizzando l'ID come chiave
```

```
mongo_data = mongo_data.join(transcript_data, mongo_data['_id'] == transcript_data['id'], 'left')  
mongo_data.show()
```

```
# Selezionare le colonne necessarie e aggiungere il campo 'transcript'
```

```
final_data = mongo_data.withColumn("transcript", col("transcript"))  
final_data.show()
```



# Job PySpark

## Add Transcription

Il Job seguente legge il dataset presente su MongoDB, esegue una join tra il dataset il file contenente le trascrizioni

# Dataset Mongo DB

Dopo lo scraping e l'esecuzione del job il dataset su Mongo si presenta in questo modo:

```
_id: "526880"  
description: "You might think of gas masks as clunky military-looking devices. But i..."  
duration: "254"  
publishedAt: "2024-04-30T15:14:51Z"  
▶ related_id: Array (3)  
slug: "george_zaidan_how_do_gas_masks_actually_work"  
speakers: "George Zaidan"  
▶ tags: Array (8)  
title: "How do gas masks actually work?"  
url: "https://www.ted.com/talks/george_zaidan_how_do_gas_masks_actually_work"  
id: "526880"  
transcript: "You might think of gas masks as clunky, spooky, military-looking devic..."
```



# Criticità

## Parsing del JSON:

Il codice assume che il JSON contenuto nel tag `<script>` sia sempre ben formato e contenga la chiave `transcript`. In caso di cambiamenti nel formato del JSON o nella struttura della pagina, il codice potrebbe fallire.

## Validazione e Pulizia dei Dati:

Le trascrizioni potrebbero contenere rumore (come testi non rilevanti). Potrebbe essere utile implementare funzioni di pulizia e validazione del testo estratto.

## Prestazioni

Il Dataset è formato da una grossa quantità di url, è necessario molto tempo affinché venga completato il tutto

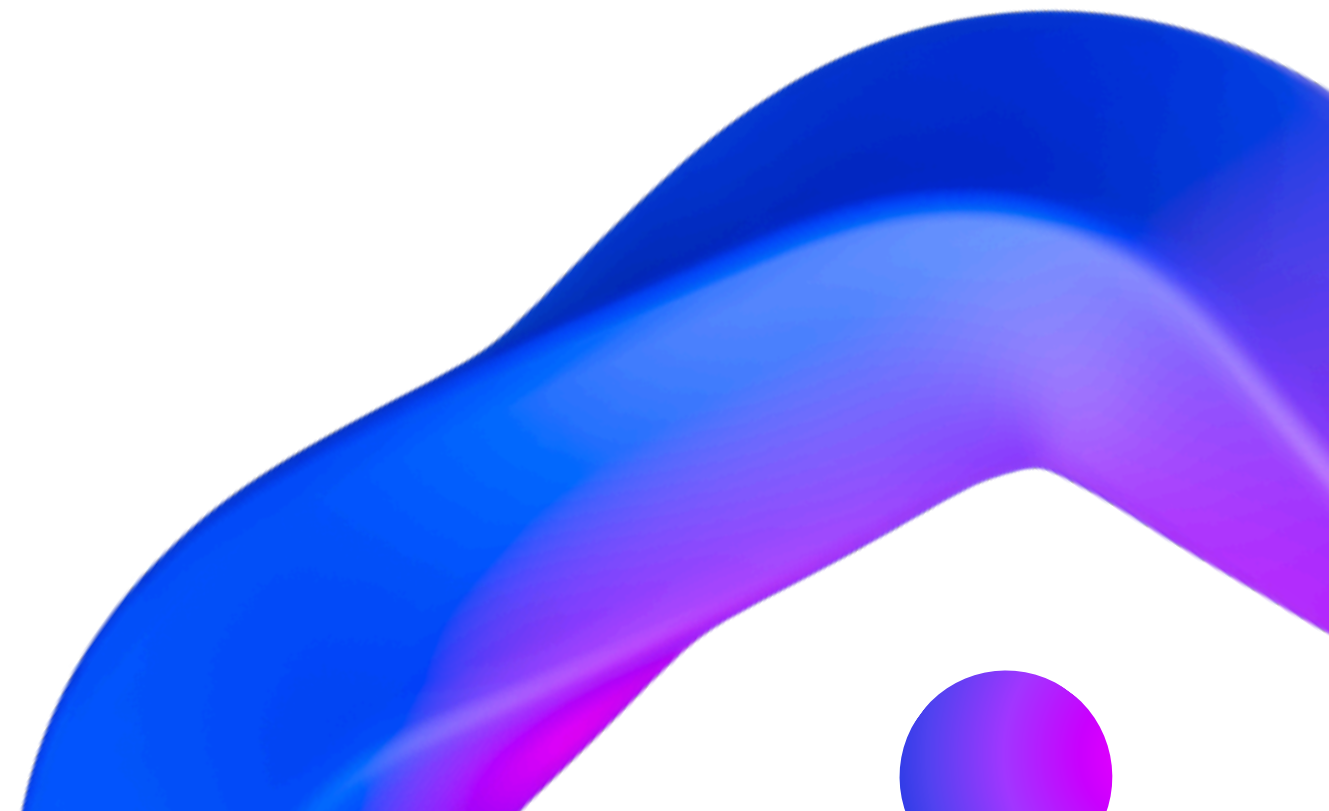
# Possibili evoluzioni



## Scraper

Modificare lo scraper in modo tale che recuperi anche il timestamp del testo

## Scraping

Integrare lo scraper all'interno di AWS per automatizzare il processo di lettura.





 Esplora, Impara, con  
TEDx e le lingue!

# LinguaLearnX

