UNIVERSITÀ DEGLI STUDI DI BERGAMO

MOBILE AND CLOUD TECHNOLOGIES

EMANUELE PERICO 1046601

CHRISTIAN CELESTINO 1046204

DESCRIZIONE JOB PYSPARK REALIZZATO

Il job scritto è composto da tre parti principali:

1. Lettura del dataset *watch_next_dataset.csv;*
2. Creazione del modello aggregato *watch_next_dataset_agg;*
3. Join di *watch_next_dataset_agg* a *tedx_dataset_agg* (contenente i dati del singolo TEDx talk con i relativi tag)

```
66
67  ### READ WATCH_NEXT DATASET
68  watch_next_dataset_path = "s3://unibg-tedx-data-emanuele/watch_next_dataset.csv"
69  watch_next_dataset = spark.read.option("header","true").csv(watch_next_dataset_path)
70
71  ### CREATE THE AGGREGATE MODEL, ADD WATCH_NEXT TO TEDX_DATASET
72  watch_next_dataset_agg = watch_next_dataset.groupBy(col("idx")).agg(collect_list("url").alias("wn_url"), collect_list("watch_next_idx").alias("wn_idx"))
73  watch_next_dataset_agg.printSchema()
74
75  # join fra il dataset iniziale (contenente anche i tags) e quello dei watch_next
76  tedx_dataset_watch_next_agg = tedx_dataset_agg.join(watch_next_dataset_agg, tedx_dataset_agg._id == watch_next_dataset_agg.idx, "left").drop("idx")
77  tedx_dataset_watch_next_agg.printSchema()
78
```

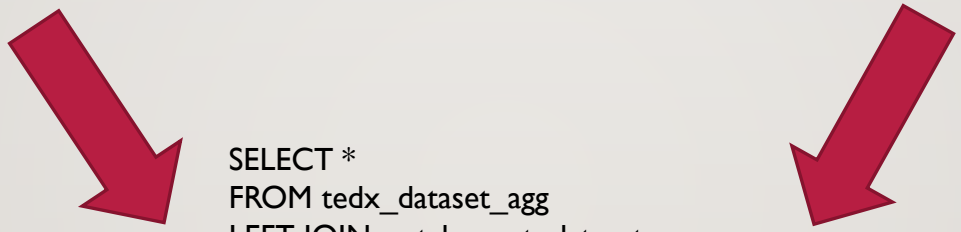
DATI TRATTATI

tedx_dataset_agg

idx	main_speaker	title	details	posted	url	tag
A	Pippo	...	...	apr-20	https://...	#ted
B	Pluto	...	...	mag-20	https://...	#science
C	Paperino	...	...	mar-20	https://...	#tech

watch_next_dataset_agg

idx	wn_url	wn_idx
A	https://...	D
A	https://...	E
B	https://...	F
B	https://...	G
C	https://...	H



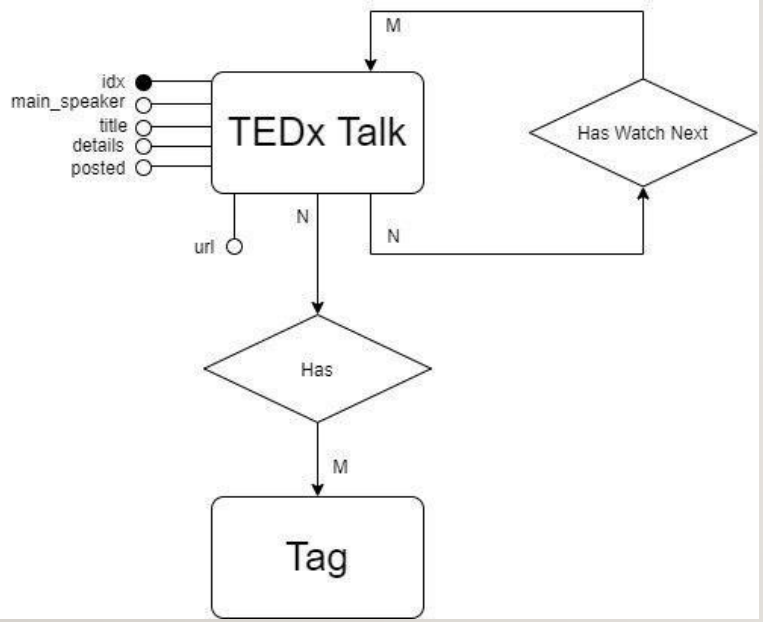
```
SELECT *
FROM tedx_dataset_agg
LEFT JOIN watch_next_dataset_agg
ON tedx_dataset_agg.idx=watch_next_dataset_agg.idx
```

tedx_dataset_watch_next_agg

idx	main_speaker	title	details	posted	url	tag	wn_url	wn_idx
A	Pippo	...	...	apr-20	https://...	#ted	https://...; https://...	D; E
B	Pluto	...	...	mag-20	https://...	#science	https://...; https://...	F; G
C	Paperino	...	...	mar-20	https://...	#tech	https://...	H

SCHEMA FINALE

Schema ER



MongoDB Visual


```
_id: "8c1fad5ce0dab8908dee527f88697ce2"
main_speaker: "Todd Dufresne"
title: "History vs. Sigmund Freud"
details: "Working in Vienna at the turn of the 20th century, he began his career..."
posted: "Posted Mar 2020"
url: "https://www.ted.com/talks/todd_dufresne_history_vs_sigmund_freud"
> tags: Array
> wn_url: Array
> wn_idx: Array
```


Query result di MongoDB

EVOLUZIONI

Per comodità e semplicità sarebbe preferibile raggruppare gli URL e gli ID dei watch_next sotto un'unica matrice, in modo da avere un unico oggetto contenente entrambi i campi di interesse.
Come fare?
Utilizzare la funzione *struct* di pyspark:

```
67  ### READ WATCH_NEXT DATASET
68  watch_next_dataset_path = "s3://unibg-tedx-data-emanuele/watch_next_dataset.csv"
69  watch_next_dataset = spark.read.option("header","true").csv(watch_next_dataset_path)
70
71  ### CREATE THE AGGREGATE MODEL, ADD WATCH_NEXT TO TEDX_DATASET
72  watch_next_dataset_agg = watch_next_dataset.select(col("idx"),struct(col("watch_next_idx").alias("wn_id"), col("url")).alias("watch_next"))
73                                             .groupBy(col("idx")).agg(collect_list("watch_next").alias("watch_next"))
74  watch_next_dataset_agg.printSchema()
75
76  # join fra il dataset iniziale (contenente anche i tags) e quello dei watch_next
77  tedx_dataset_full_agg = tedx_dataset_agg.join(watch_next_dataset_agg, tedx_dataset_agg._id == watch_next_dataset_agg.idx, "left").drop("idx")
78  tedx_dataset_full_agg.printSchema()
79
```

Ottenendo:

idx	main_speaker	title	details	posted	url	tag	watch_next	
A	Pippo	...	...	apr-20	https://...	#ted	wn_url	wn_idx
B	Pluto	...	...	mag-20	https://...	#science	wn_url	wn_idx
C	Paperino	...	...	mar-20	https://...	#tech	wn_url	wn_idx

Visual MongoDB:

CRITICITÀ

Eventuali criticità riscontrate potrebbero essere ad esempio:

- Errori nei Dataset CSV, come errori di scrittura e di separazione dei campi

- Possibili duplicati che vanno ad appesantire il DataLake, risolvibili attraverso l'uso della funzione *collect_set* al posto della classica *collect_list* ;
  Nel nostro caso l'uso di questa funzione riduce lo spazio del nostro DataLake di circa la metà (da 12 MB a 7 MB)