Guida Operativa all'utilizzo dello Scraper

Questa guida ha lo scopo di spiegare all'utente come utilizzare lo scraper in modo corretto, fornendo una breve overview del programma. Si passerà in rassegna la modifica delle impostazioni di base, la modifica delle variabili fondamentali ai fini della ricerca di artisti, il modo in cui il programma agisce in caso di disconnessione e infine verrà illustrata come dovrebbe presentarsi la struttura completa di una collezione del database.

1. INSTALLAZIONE LIBRERIE NECESSARIE

Per prima cosa è necessario installare tutte le librerie che verranno utilizzate dal programma. All'interno della directory è presente il file requirements.txt, il quale contiene tutti i pacchetti utilizzati. Per poter installare le librerie sarà necessario eseguire il seguente comando dal terminale Windows dopo aver selezionato la cartella in cui il file è presente.

pip install -r requirements.txt

Questo comando permette di leggere dal foglio di testo i pacchetti necessari e successivamente di installarli rapidamente.

2. MODIFICA DEL FILE SETTINGS

Nella cartella ***Setup*** è presente il file ***Settings.py*** che contiene tutte le impostazioni inerenti al database di MongoDB e alle credenziali delle API di GeniusLyrics e Spotify. Nell'immagine sottostante si possono osservare le impostazioni standard: di default il programma si connetterà a MongoDB sul server locale (con porta d'accesso 27017). Nel caso in cui si desiderasse salvare i dati utilizzando Mongo ATLAS basterà sostituire <localhost> con il proprio codice **SVR**. Sempre riguardo alle impostazioni di Mongo, è possibile modificare il nome del database e delle rispettive collezioni. Nel medesimo file possono essere cambiate anche le chiavi d'accesso di Spotify e Genius Lyrics, indispensabili per il funzionamento del programma. L'ultimo paramentro riguarda il PATH: questo è utile nell'ultima parte del programma, ovvero quando i dati verranno aggregati tra di loro, questa impostazione serve appunto per stabilire, se richiesto, dove creare la cartella che conterrà i file json da esportare.

```
# Inserire path dove è presente la directory del progetto

PATH = 'C:/Users/giaco/Desktop/Spotify Project'

# MongoDB settings.
connection = 'localhost'
port = '27017'

cluster_name = 'DataMan'
artists_collection_name = 'Artists'
albums_collection_name = 'albums'
main_collection_name = 'lyrics'
natio_collection_name = 'nationality'
SexAgeCity_collection_name = 'sexagecity'
related_collection_name = 'related'
streams_collection_name = 'Streams11'

# Credenziali Spotify.
spotify_id = "183c3f4d12544afbbbc4406ae45c4fb9"
spotify_secret = "2722aea9ff01479ca4a4c5fbae49b6de"
client_credentials_manager = SpotifyClientCredentials(spotify_id, spotify_secret)
sp = spotipy.Spotify(client_credentials_manager=client_credentials_manager)

# Credenziali GeniusLyrics
genius = lyricsgenius.Genius("pAQ_YE4KjoW9yQn1jw0dBrhAmwaHBYDK8EfFIjg8A9ykfaBrlzBVmwR0JmfFFROw")
```

3. MODIFICA DEL FILE VARIABLES

```
# Combinazioni di ricerca.
search1 = list(string.ascii_lowercase)
search2 = [letter + let for letter in search1 for let in search1]
search3 = [letter + let for letter in search2 for let in search1]
total_search = search1 + search2 + search3

# Livello minimo popolarità.

popularity = 60
```

L'immagine soprastante presenta le due variabili principali che l'utente può modificare per cambiare i criteri di ricerca degli artisti. Il file ***Variables.py*** è collocato anch'esso nella cartella ***Setup***. La variabile ***total_search*** indica il criterio con il quale vengono ricercati gli artisti. Questa variabile indica la somma di disposizioni con ripetizione delle lettere dell'alfabeto prese prima singolarmente poi a gruppi di due ed infine a gruppi formati da tre elementi. Sono state scelte le disposizioni con ripetizione perchè, ad esempio, i risultati prodotti dalle ricerche effettuate con la combinazione ***'aba'*** saranno diverse dai risultati derivanti dalla ricerca effettuata con la combinazione ***'baa'*** o ***'aab'***.

$$\sum D'_{i,26} = D'_{1,26} + D'_{2,26} + D'_{3,26} = 18278$$

Le ricerche totali saranno quindi 18278 nel caso in cui si decidesse di far partire il programma con questo tipo di ricerca (verra richiesto esplicitamente nel terminale).
La seconda variabile riguarda la soglia minima di popolarità da tenere in considerazione: infatti se lasciata immutata il programma scarterà tutti gli artisti con un valore di popolarità inferiore a 60. Entrambe queste variabili possono essere modificate a piacimento dell'utilizzatore e sono in grado di modificare sostanzialmente la ricerca degli artisti. Nella sezione successiva verrà mostrata un immagine dello scraper in funzione che mostra il funzionamento del programma utilizzando la ricerca totale.

4. RACCOLTA DEI DATI

Per avviare il programma basterà eseguire il file ***start.py*** da terminale: una volta eseguito il comando verranno visualizate sullo schermo una serie di operazioni eseguibili antecedute da un numero che se digitato eseguirà l'operazione selezionata. L'obbiettivo è stato quello di rendere il programma *user-friendly* cercando di renderlo interattivo e di facile utilizzo. Come si può osservare dall'immagine seguente, all'avvio del programma si aprirà un menù dove è richiesto che venga digitato un numero corrispondente all'operazione che si intende eseguire.

```
Benvenuto, con questo script è possibile raccogliere tutte le informazioni inerenti agli artisti.
Si prega di seguire l'ordine prestabilito nel menu.

1. Scraping Artisti.
2. Scraping Albums.
3. Scraping Lyrics e variabili canzoni.
4. Scraping Nazionalità
5. Scraping Sesso, Provenienza, Età.
6. Scraping Artisti correlati.
7. Scraping Streams giornalieri
8. Aggregazione
Selezionare il numero corrispondente:
```

1. La prima operazione permette di scaricare le informazioni di base degli artisti: nome, numero di followers, genere musicale, url spotify e popolarità

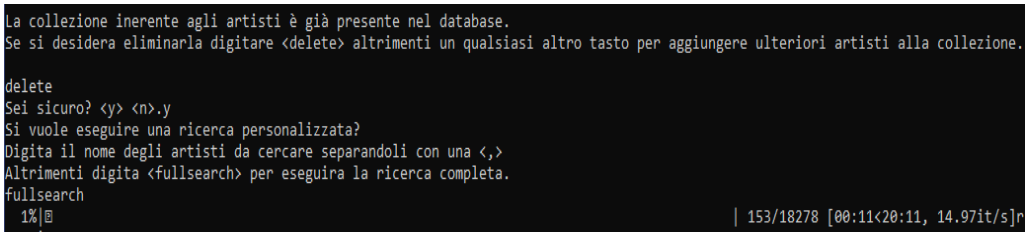
2. La seconda iterazione permette di scaricare tutte le url degli album e dei singoli degli artisti presenti nella collezione creata al punto precedente.

3. la terza operazione permette di scaricare le lyrics e le variabili relative a tutte le canzoni presenti negli album scaricati al punto precedente.

4. Dalla quarta alla settima operazione sono facilmente comprensibili e si occupano di raccogliere informazioni soggettive riguardo agli artisti.

5. l'Ottava operazione invece permette di aggeregare tutti i dati collezionati e verrà tattata piu in dettaglio in segito.

Ogni volta che si esegue un'operazione, come prima cosa verrà chiesto se si intende eliminare la collezione qualora fosse preesistente. Dopo aver digitato <delete>, il programma richiede una ulteriore conferma di sicurezza volta alla cancellazione definitiva della collezione. È consigliato seguire l'ordine proposto, anche se non si preclude l'esecuzione delle operazioni in ordine diverso (ma in questo caso si potranno eseguire solo alcune di esse ad esempio artisti, streams e nazionalità). Resta fondamentale eseguire la prima operazione (quella inerente agli artisti), dato che tutte le altre operazioni sono dipendenti da essa: il nome e l'id dell'artista vengono utilizzati per iterare e aggiungere le variabili nel corso delle varie operazioni di scraping. Nell'immagine seguente viene illustrato l'avvio della prima operazione qualora si scegliesse di eseguire la ricerca totale come specificato all'interno del file ***Variables.py***.



5. DIPENDENZE TRA LE OPERAZIONI

Come è stato detto in precedenza, tutte le operazioni sono dipendenti dalla prima. Per quanto riguarda invece la terza operazione, quella relativa alle lyrics e alle variabili delle canzoni, abbiamo una dipendenza dalla seconda operazione, la quale scarica le url degli album per ciascun artista. É possibile aggiungere artisti in un secondo momento, senza che questo comprometta i criteri di matching tra le collezioni: tuttavia, ciò deve avvenire ricordando che alcune collezioni possono essere dipendenti da altre.

6. PROBLEMI DI DISCONNESSIONE DALLE API

I problemi legati alla disconnessione dal server di GeniusLyrics e di Spotify sono stati gestiti quasi interamente all'interno del codice, tuttavia può comunque capitare che compaiano rari errori di connessione. Nel caso di un errore di connessione basta far ripartire lo scraper: questo ricomincerà a scaricare i dati a partire dall'ultima collezione sulla quale si era interrotto.

7. AGGREGAZIONE DEI DATI

Dopo aver estratto tutte le variabili d'interesse, in ultima istanza dobbiamo aggregare i dati: questa facoltà ci viene data scegliendo l'ottava operazione. Una volta aggregati i dati, ci verrà chiesto se intendiamo cancellare le collezioni usate per creare il dataset finale; verrà chiesto anche se si desidera creare un file json con destinazione il ***PATH*** specificato all'interno del file ***Settings.py***

8. STRUTTURA DEI DATI

Dato che MongoDB permette di sfruttare uno schema flessibile, può capitare che durante l'attività si siano creati dei documenti (poi inseriti nella collezione finale) che presentano strutture diverse: è il caso di valori mancanti causati dalla difficoltà nel reperire determinate informazioni.

Di conseguenza, si rende utile presentare la struttura predefinita che ciascuna collezione dovrebbe assumere.

```
|_ _id                    # id dell'artista
|_ artist                 # nome dell artista
|_ albums                 # Contiene una lista di dizionari, uno per ogni album
|____ name                # Nome dell'album
|____ tracks              # Lista di dizionari inerenti alle canzoni
|_______ songname         # Nome della canzone
|_______ language         # Lingua del testo della canzone
|_______ lyrics           # Lyrics
|_______ variables        # lista di dizionari dove ogni variabile è chiave del valore effettivo.
|__________ danceability
            ...
            ...
            ...
|__________ loudness
|_ related                # Lista contenente i 4 principali artisti correlati.
|_ age                    # Età dell'artista.
|_ città                  # Città di nascita dell'artista.
|_ sex                    # Sesso dell'artista.
|_ nationality            # Nazionalità di provenienza.
|_ followers              # Numero di followers su Spotify.
|_ genres                 # Lista di generi dell'artista
|_ popularity             # Livello di popolarita su Spotify
|_ url                    # Url della pagina Spotify dell'artista
|_ streams                # Lista di dizionari dove il timestamp è chiave.
|____ 01/01/2020
      ...
      ...
      ...
|____ 31/12/2020
|_______ città 1          # Dizionario per ogni citta dove la città e chiave e
         ...              # il numero di streams è il valore.
         ...
         ...
|_______ città n
```