S3

The TV Show Song Search Engine

Gestione Avanzata dei Dati 2015/2016

Ciaparrone Gioele Vitale Luca

Specifica del problema - 1

- Hai appena visto un telefilm e ti è rimasta in mente una canzone
- Come sapere quale brano hai ascoltato?
- Vuoi riprodurre e cantare subito la canzone appena scoperta?
- Quali serie hanno usato le canzoni del tuo artista preferito?

Problema: per trovare tutte queste informazioni è necessario visitare più siti -> ricerca noiosa

Specifica del problema - 2

S3 semplifica e velocizza il tutto!

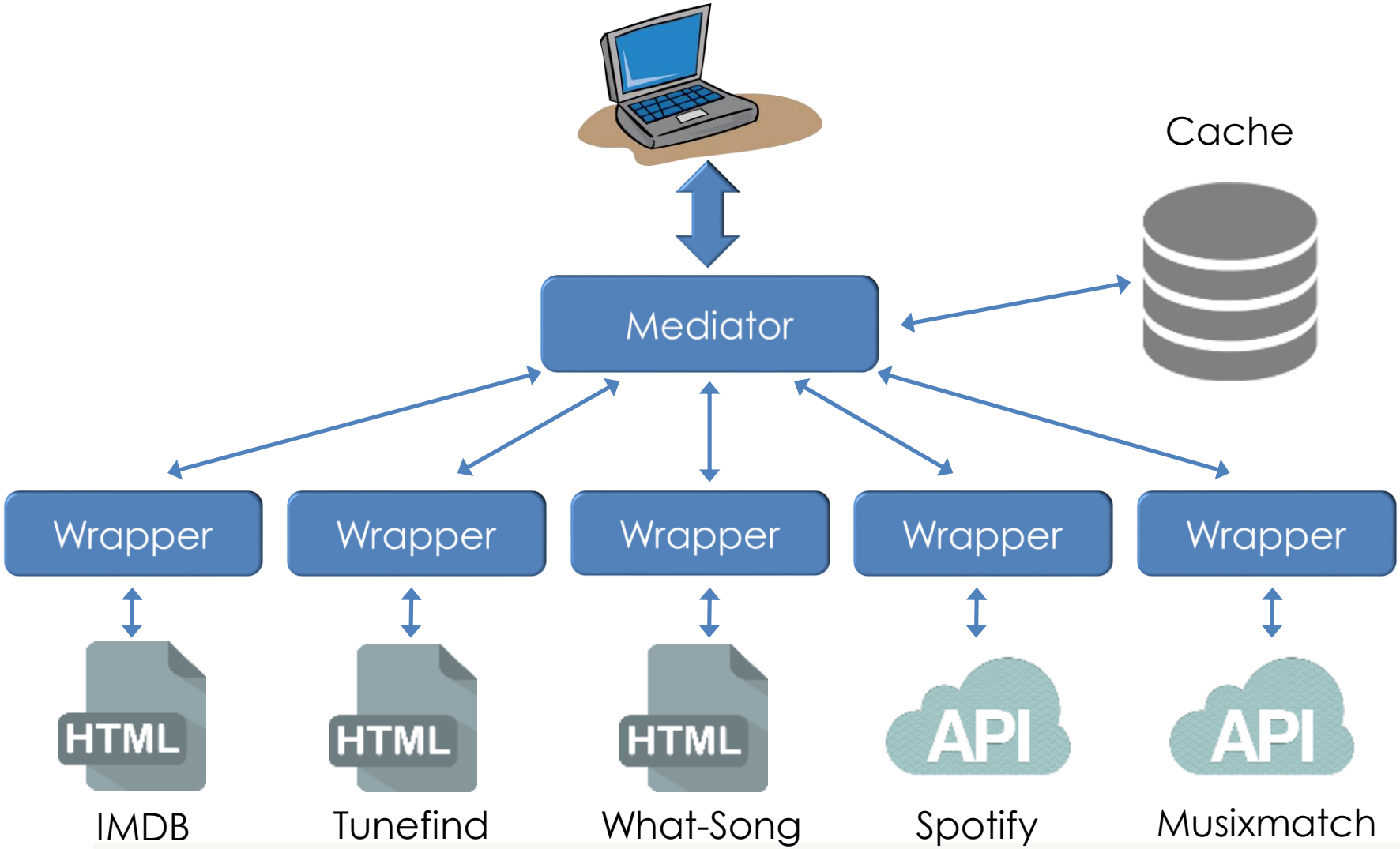
Con S3 puoi:

- Cercare le canzoni presenti in un episodio.
- Cercare gli episodi per nome o visualizzare tutti gli episodi di una serie TV.
- Cercare gli episodi con le canzoni del tuo artista preferito.
- Visualizzare testo e widget di riproduzione insieme alle informazioni del brano.

Demo

Demo

Architettura

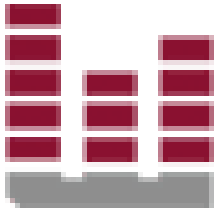


Cache

- Per ogni query globale e locale (wrapper) viene salvato il risultato in cache.
- Ad ogni query viene associato un tempo di scadenza differente:
  - Ricerca episodi per nome: 1 giorno
  - Elenco di episodi di una stagione:
    - 7 giorni se ultima stagione
    - 30 giorni altrimenti
  - Ricerca artisti per nome: 7 giorni
  - ecc.

Fonti - 1

- IMDB (http://www.imdb.com) - The Internet Movie DataBase
  - Utilizzato per ottenere i dati sulle serie TV.
  - Web scraping - cambi periodici.
- Tunefind (http://www.tunefind.com) & What-Song (http://www.what-song.com)
  - Utilizzati per ottenere le associazioni canzone-episodio.
  - Web scraping - cambi periodici.



WHAT-SONG

Fonti - 2

- Musixmatch (https://www.musixmatch.com)
  - Utilizzato per ottenere i testi delle canzoni.
  - API - informazioni quasi statiche.
- Spotify (https://www.spotify.com)
  - Utilizzato per permettere la riproduzione dei brani.
  - API – informazioni quasi statiche.

Collegamento fonti (join)

- La ricerca di episodi e serie viene effettuata su IMDB
  - Usiamo nome di serie, nome e numero episodio per cercare la stessa puntata su Tunefind e What-Song, da cui ricaviamo le informazioni sui brani
  - Usiamo poi titolo brano e artista per ottenere le informazioni da Spotify e Musixmatch
- La ricerca per artista viene effettuata su Tunefind e What-Song, da cui ricaviamo tutti i brani e gli episodi in cui appaiono
  - Usiamo nome di serie ed episodio per cercare l'episodio su IMDB

Descrizione schemi locali - 1

- IMDB
  - **series**(id, title, description, year, image, creator, actors, genres, rating, seasonsNum)
  - **episode**(id, seriesId, season, number, title, description, image)
- Tunefind (TF)
  - **episode**(seriesName, season, number, title, url)
  - **song**(episodeUrl, title, artist, moment)
- What-Song (WS)
  - **episode**(seriesName, season, number, title, url)
  - **song**(episodeUrl, title, artist, moment)

Descrizione schemi locali - 2

- Spotify (SP)
  - **song**(title, artist, uri)
- Musixmatch (MM)
  - **lyrics**(songTitle, artist, text, url)

Wrapper - 1

- IMDB - http://www.imdb.com/title/<seriesId> (series)
  - title -> '[itemprop="name"] ~ div.originalTitle'[0]
  - description -> '[itempropr="description"]'[0]
  - year -> '[itemprop="name"] + div > *:last-child'[0]
  - image -> 'div.poster [itemprop="image"]'[0]
  - creator -> '[itemprop="creator"] > a'[0]
  - actors -> 'span[itemprop="actors"]'[0]
  - genres -> 'span[itemprop="genre"]'[0]
  - rating -> 'span[itemprop="ratingValue"]'[0]
  - seasonsNum -> '#title-episode-widget > div > br'[0]
- Per la tabella IMDB.episode i path sono simili.

Wrapper - 2

- Tunefind (TF.song)
  - title -> ‘div.tf-episode-container li.list-group-item div.media-body a’[0]
  - artist -> ‘div.tf-episode-container li.list-group-item div.media-body a’[1]
  - moment -> ‘div.tf-episode-container li.list-group-item div.media-body a’[2]
- What-Song (WS.song)
  - title -> ‘tr td.song-col-2 div.song-name’[0]
  - artist -> ‘tr td.song-col-2 div.artist a’[0]
  - moment -> ‘tr td.song-col-2 span.song-description’[0]

Wrapper - 3

- Spotify & Musixmatch (API)
  - function getAdditionalSongDataAsynchronous($trackname, $artistname, $lyrics = true, $lyricsUrl = true, $spotifyUri = true)
  - Richieste eseguite in parallelo con Guzzle
- https://api.spotify.com/v1/search?q=track:<track>%20artist:<artist>&type=track
- http://api.musixmatch.com/ws/1.1/matcher.lyrics.get?q_track=<track>&q_artist=<artist>&apikey=<apiKey>
- http://api.musixmatch.com/ws/1.1/track.search?q_track=<track>&q_artist=<artist>&apikey=<apiKey>

Schema globale

- **series**(id, title, description, year, image, creator, actors, genres, rating, seasonsNum)
- **episode**(id, seriesId, season, number, title, description, image)
- **song**(episodeId, title, artist, moment, spotifyURI, lyrics, lyricsURL)
- **episodeURLs**(episodeId, urlTF, urlWS)

Mapping GAV - 1

- **series**(id, title, description, year, image, creator, actors, genres, rating, seasonsNum) :- **IMDB.series**(id, title, description, year, image, creator, actors, genres, rating, seasonsNum)
- **episode**(id, seriesId, season, number, title, description, image) :- **IMDB.episode**(id, seriesId, season, number, title, description, image)
- **episodeURLs**(episodeId, urlTF, urlWS) :- **IMDB.episode**(episodeId, seriesId, season, number, epTitle, v1, v2), **IMDB.series**(seriesId, seriesTitle, v3, v4, v5, v6, v7, v8, v9, v10), **TF.episode**(seriesTitle, season, number, epTitle, urlTF), **WS.episode**(seriesTitle, season, number, epTitle, urlWS)

Mapping GAV - 2

- **song**(episodeId, title, artist, moment, spotifyURI, lyrics, lyricsURL) :- **IMDB.episode**(episodeId, seriesId, season, number, epTitle, v1, v2), **IMDB.series**(seriesId, seriesTitle, v3, v4, v5, v6, v7, v8, v9, v10), **TF.episode**(seriesTitle, season, number, epTitle, urlTF), **TF.song**(urlTF, title, artist, moment), **SP.song**(title, artist, spotifyURI), **MM.lyrics**(title, artist, lyrics, lyricsURL)

- **song**(episodeId, title, artist, moment, spotifyURI, lyrics, lyricsURL) :- **IMDB.episode**(episodeId, seriesId, season, number, epTitle, v1, v2), **IMDB.series**(seriesId, seriesTitle, v3, v4, v5, v6, v7, v8, v9, v10), **WS.episode**(seriesTitle, season, number, epTitle, urlWS), **WS.song**(urlWS, title, artist, moment), **SP.song**(title, artist, spotifyURI), **MM.lyrics**(title, artist, lyrics, lyricsURL)

Mapping LAV - 1

- **IMDB.series**(id, title, description, year, image, creator, actors, genres, rating, seasonsNum) :- **series**(id, title, description, year, image, creator, actors, genres, rating, seasonsNum)
- **IMDB.episode**(id, seriesId, season, number, title, description, image) :- **episode**(id, seriesId, season, number, title, description, image)
- **TF.episode**(seriesName, season, number, title, url) :- **episodeURLs**(id, url, v1), **episode**(id, seriesId, season, number, title, v2, v3), **series**(seriesId, seriesName, v4, v5, v6, v7, v8, v9, v10, v11)

Mapping LAV - 2

- **TF.song**(episodeUrl, title, artist, moment) :- **song**(id, title, artist, moment, v1, v2, v3), **episodeURLs**(id, episodeUrl, v4)
- **WS.episode**(seriesName, season, number, title, url) :- **episodeURLs**(id, v1, url), **episode**(id, seriesId, season, number, title, v2, v3), **series**(seriesId, seriesName, v4, v5, v6, v7, v8, v9, v10, v11)
- **WS.song**(episodeUrl, title, artist, moment) :- **song**(id, title, artist, moment, v1, v2, v3), **episodeURLs**(id, v4, episodeUrl)
- **SP.song**(title, artist, uri) :- **song**(v1, title, artist, v2, uri, v3, v4)
- **MM.lyrics**(songTitle, artist, text, url) :- **song**(v1, songTitle, artist, v2, v3, text, url)

Query - 1

- Ricerca dei nomi delle canzoni presenti in un episodio
- **findSongsByEpisode**(episodeTitle, seriesName, song) :-
  series(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8),
  episode(epId, seriesId, v9, v10, episodeTitle, v11, v12),
  song(epId, song, v13, v14, v15, v16, v17)
- SELECT song.title as song
  FROM series, episode, song
  WHERE series.id = episode.seriesId AND episode.id = song.episodeId AND episode.title = 'Breakage' AND series.title = 'Breaking Bad';

Query - 2

- Ricerca degli episodi di una serie TV
- **findEpisodes**(seriesName, episodeName, season, number) :- **series**(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8), **episode**(v9, seriesId, season, number, episodeName, v10, v11)
- SELECT episode.title as episodeName, episode.season as season, episode.number as number
  FROM series, episode
  WHERE series.id = episode.seriesId AND series.title = ‘Breaking Bad’;

Query - 3

- Ricerca delle canzoni di un artista presenti nelle serie TV
- **findSongsByArtist**(artist, song) :- **song**(v1, song, artist, v2, v3, v4, v5)
- SELECT song.title as song
  FROM song
  WHERE artist = ‘Queen’;

Query - 4

- Ricerca di un episodio in cui viene riprodotta una canzone di un dato artista
- **findEpisodeByArtist**(artist, episodeName, seriesName) :- **song**(episodeId, v1, artist, v2, v3, v4, v5), **episode**(episodeId, seriesId, v6, v7, episodeName, v8, v9), **series**(seriesId, seriesName, v10, v11, v12, v13, v14, v15, v16, v17)
- SELECT episode.title as episodeName ,series.title as seriesName
  FROM song, episode, series
  WHERE song.episodeId = episode.id AND episode.seriesId = series.id AND song.artist = 'Queen';

GAV query unfolding - 1

- findSongsByEpisode(episodeTitle, seriesName, song) :-
  series(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8),
  episode(epId, seriesId, v9, v10, episodeTitle, v11, v12),
  song(epId, song, v13, v14, v15, v16, v17)
- findSongsByEpisode(episodeTitle, seriesName, song) :-
  IMDB.series(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8),
  episode(epId, seriesId, v9, v10, episodeTitle, v11, v12),
  song(epId, song, v13, v14, v15, v16, v17)
- findSongsByEpisode(episodeTitle, seriesName, song) :-
  IMDB.series(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8),
  IMDB.episode(epId, seriesId, v9, v10, episodeTitle, v11, v12),
  song(epId, song, v13, v14, v15, v16, v17)

GAV query unfolding - 2

- Song è unione di due query e quindi l'unfolding produce l'OR delle due seguenti query:

1. findSongsByEpisode(episodeTitle, seriesName, song) :- IMDB.series(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8), IMDB.episode(epId, seriesId, v9, v10, episodeTitle, v11, v12), **IMDB.episode(epId, seriesId', season, number, epTitle, a1, a2), IMDB.series(seriesId', seriesTitle, a3, a4, a5, a6, a7, a8, a9, a10), TF.episode(seriesTitle, season, number, epTitle, urlTF), TF.song(urlTF, song, v13, v14), SP.song(song, v13, v15), MM.lyrics(song, v13, v16, v17)**

GAV query unfolding - 3

2. findSongsByEpisode(episodeTitle, seriesName, song) :- IMDB.series(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8), IMDB.episode(epId, seriesId, v9, v10, episodeTitle, v11, v12), **IMDB.episode(epId, seriesId', season, number, epTitle, a1, a2), IMDB.series(seriesId', seriesTitle, a3, a4, a5, a6, a7, a8, a9, a10), WS.episode(seriesTitle, season, number, epTitle, urlWS), WS.song(urlWS, song, v13, v14), SP.song(song, v13, v15), MM.lyrics(song, v13, v16, v17)**

- Semplifichiamo le query:
  - epId e seriesId sono chiavi -> possiamo richiamare IMDB.episode e IMDB.series una sola volta
  - non ci servono testi di Musixmatch e URI di Spotify e non serve controllare che le canzoni trovate esistano nei loro DB

GAV query unfolding - 4

1. **findSongsByEpisode**(episodeTitle, seriesName, song) :- **IMDB.series**(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8), **IMDB.episode**(epId, seriesId, season, number, episodeTitle, v11, v12), **TF.episode**(seriesName, season, number, episodeTitle, urlTF), **TF.song**(urlTF, song, v13, v14)

2. **findSongsByEpisode**(episodeTitle, seriesName, song) :- **IMDB.series**(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8), **IMDB.episode**(epId, seriesId, season, number, episodeTitle, v11, v12), **WS.episode**(seriesName, season, number, episodeTitle, urlWS), **WS.song**(urlWS, song, v13, v14)

LAV bucket algorithm - 1

- findSongsByEpisode(episodeTitle, seriesName, song) :-
  series(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8),
  episode(epId, seriesId, v9, v10, episodeTitle, v11, v12),
  song(epId, song, v13, v14, v15, v16, v17)

- bucket(**series**(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8)):
  - **IMDB.series**(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8)
  - **TF.episode**(seriesName, a1, a2, a3)
  - **WS.episode**(seriesName, b1, b2, b3)

LAV bucket algorithm - 2

- bucket(**episode**(epId, seriesId, v9, v10, episodeTitle, v11, v12)):
  - **IMDB.episode**(epId, seriesId, v9, v10, episodeTitle, v11, v12)
  - **TF.episode**(c1, v9, v10, episodeTitle, c2)
  - **WS.episode**(d1, v9, v10, episodeTitle, d2)
- bucket(**song**(epId, song, v13, v14, v15, v16, v17)):
  - **TF.song**(e1, song, v13, v14)
  - **WS.song**(f1, song, v13, v14)
  - **SP.song**(song, v13, v15)
  - **MM.lyrics**(song, v13, v16, v17)

LAV bucket algorithm - 3

- Sono possibili 3 x 3 x 4 = 36 riformulazioni. Ne mostriamo solo una come esempio
- q(episodeTitle, seriesName, song) :- IMDB.series(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8), IMDB.episode(epId, seriesId, v9, v10, episodeTitle, v11, v12), TF.song(e1, song, v13, v14)
- q'(episodeTitle, seriesName, song) :- **series(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8)**, IMDB.episode(epId, seriesId, v9, v10, episodeTitle, v11, v12), TF.song(e1, song, v13, v14)

LAV bucket algorithm - 4

- q'(episodeTitle, seriesName, song) :- series(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8), **episode(epId, seriesId, v9, v10, episodeTitle, v11, v12)**, TF.song(e1, song, v13, v14)

- q'(episodeTitle, seriesName, song) :- series(seriesId, seriesName, v1, v2, v3, v4, v5, v6, v7, v8), episode(epId, seriesId, v9, v10, episodeTitle, v11, v12), **song(id, song, v13, v14, k1, k2, k3), episodeURLs(id, e1, k4)**

q' ⊄ findSongsByEpisode

Poiché nel bucket di song tutti i letterali hanno come url degli episodi una variabile libera (ed epId non appare mai) non è possibile trovare una riformulazione il cui unfolding sia contenuto nella query originale.

LAV bucket algorithm - 5

- findSongsByArtist(artist, song) :- **song**(v1, song, artist, v2, v3, v4, v5)
- bucket(**song**(v1, song, artist, v2, v3, v4, v5)):
  - **TF.song**(a1, song, artist, v2)
  - **WS.song**(b1, song, artist, v2)
  - **SP.song**(song, artist, v3)
  - **MM.lyrics**(song, artist, v4, v5)
- Con un unico bucket abbiamo 4 possibili riformulazioni:
  - q1(artist, song) :- **TF.song**(a1, song, artist, v2)
  - q2(artist, song) :- **WS.song**(b1, song, artist, v2)
  - q3(artist, song) :- **SP.song**(song, artist, v3)
  - q4(artist, song) :- **MM.lyrics**(song, artist, v4, v5)

LAV bucket algorithm - 6

- q1(artist, song) :- **TF.song**(a1, song, artist, v2)
  - q1(artist, song) :- **song**(id, song, artist, v2, z1, z2, z3), **episodeURLs**(id, a1, z4)
  - **q1 ⊆ findSongsByArtist**
- q2(artist, song) :- **WS.song**(b1, song, artist, v2)
  - q2(artist, song) :- **song**(id, song, artist, v2, z1, z2, z3), **episodeURLs**(id, z4, b1)
  - **q2 ⊆ findSongsByArtist**
- q3(artist, song) :- **SP.song**(song, artist, v3)
  - q3(artist, song) :- **song**(z1, song, artist, z2, v3, z3, z4)
  - **q3 ⊆ findSongsByArtist**

LAV bucket algorithm - 7

- q4(artist, song) :- **MM.lyrics**(song, artist, v4, v5)
  - q4(artist, song) :- **song**(z1, song, artist, z2, z3, v4, v5)
  - **q4 ⊆ findSongsByArtist**
- Tutti e 4 gli unfolding delle riformulazioni sono incluse nella query originale, quindi la query riformulata finale è l'unione delle 4 riformulazioni:
  - **qr**(artist, song) :- **TF.song**(a1, song, artist, v2)
  - **qr**(artist, song) :- **WS.song**(b1, song, artist, v2)
  - **qr**(artist, song) :- **SP.song**(song, artist, v3)
  - **qr**(artist, song) :- **lyrics**(song, artist, v4, v5)

Tecnologie utilizzate

- PHP 5.6
- Javascript & jQuery
- MySQL
- Simple HTML DOM Parser (CSS Path)
- Guzzle
- JSON

FINE