U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Tunix - Music Search System

Information Processing and Retrieval – Milestone 1

13-10-2025

Nuno Rios (up202206272@up.pt)
Rodrigo Resende (up202108750@up.pt)
Rodrigo Ribeiro (up202206396@up.pt)
António Pereira (up202205501@up.pt)

Tunix - Music Search System

Main points:

•Traditional systems (e.g., **Shazam**) rely on **audio fingerprinting**.

•However, users often **remember lyrics**, not melodies.

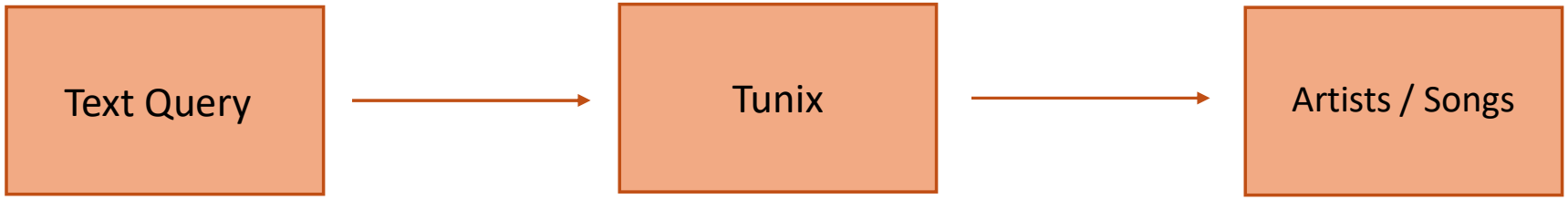
•**Goal:** Retrieve songs, artists, and metadata **from text queries**.

•**Tunix** uses a single dataset for songs and retrieves all additional information (artist details, album, etc.) from the Last.fm API.

•Provides contextual info such as:

- **Artist biography**, **album name, lyrics.**

•Contributes to **Information Retrieval** by building a **structured pipeline** for text-based music search.



Data Sources

Dataset	Attributes	Pros	Cons
Audio features and lyrics of Spotify songs	track_id, track_name, track_artist, lyrics, track_popularity, track_album_id, track_album_name, track_album_release_date, playlist_name, playlist_id, playlist_genre, playlist_subgenre, danceability, energy, key, loudness, mode, speechiness, instrumentalness, liveness, valence, tempo, duration_ms, language	A lot of information about the songs and many different tracks	Some songs don't have lyrics and the number of songs per artist is a bit limited. Too many non-essential attributes
Spotify Million Song Dataset	song_name, artist_name, link, lyrics	Good balance of attributes, consistent number of songs per artist, and overall a well-sized dataset	Some inconsistencies in how values are obtained — description states data comes from the Spotify API, but song links are from a lyrics website
Song Lyrics Dataset	Artist, Title, Album, Date, Lyric, Year	Well-structured by artist; good complementary dataset to enrich the main dataset with lyrics information	Some songs do not have lyrics (marked consistently with the string "lyrics for this song have yet to be released please check back once the song has been released"); easy to detect and filter
Lyrics and Metadata from 1950 to 2019	artist_name, track_name, release_date, genre, lyrics, len, dating, violence, world/life, night/time, shake the audience, family/gospel, romantic, communication, obscene, music, movement/places, light/visual perceptions, family/spiritual, like/girls, sadness, feelings, danceability, loudness, acousticness, instrumentalness, valence, energy, topic, age	Very rich dataset with a large number of songs and detailed metadata	May have too many artists with only a few songs each and includes many non-essential attributes
Artists	artist_name,artist_mbid,biography,top_10_similar_mbids,dbpedia_uri	Very rich dataset with a large number of artists and detailed metadata	Its not a raw dataSet, its a set of files that need to be conected can be harder to work

Exploratory Data Analysis

Attributes:

- artist → Artist name
- song → Song title
- link → Unique song identifier
- text → Lyrics

Link Structure:

- Each link points to a **unique song page**
- URL encodes **artist name** and **song title**
- End number = unique song ID

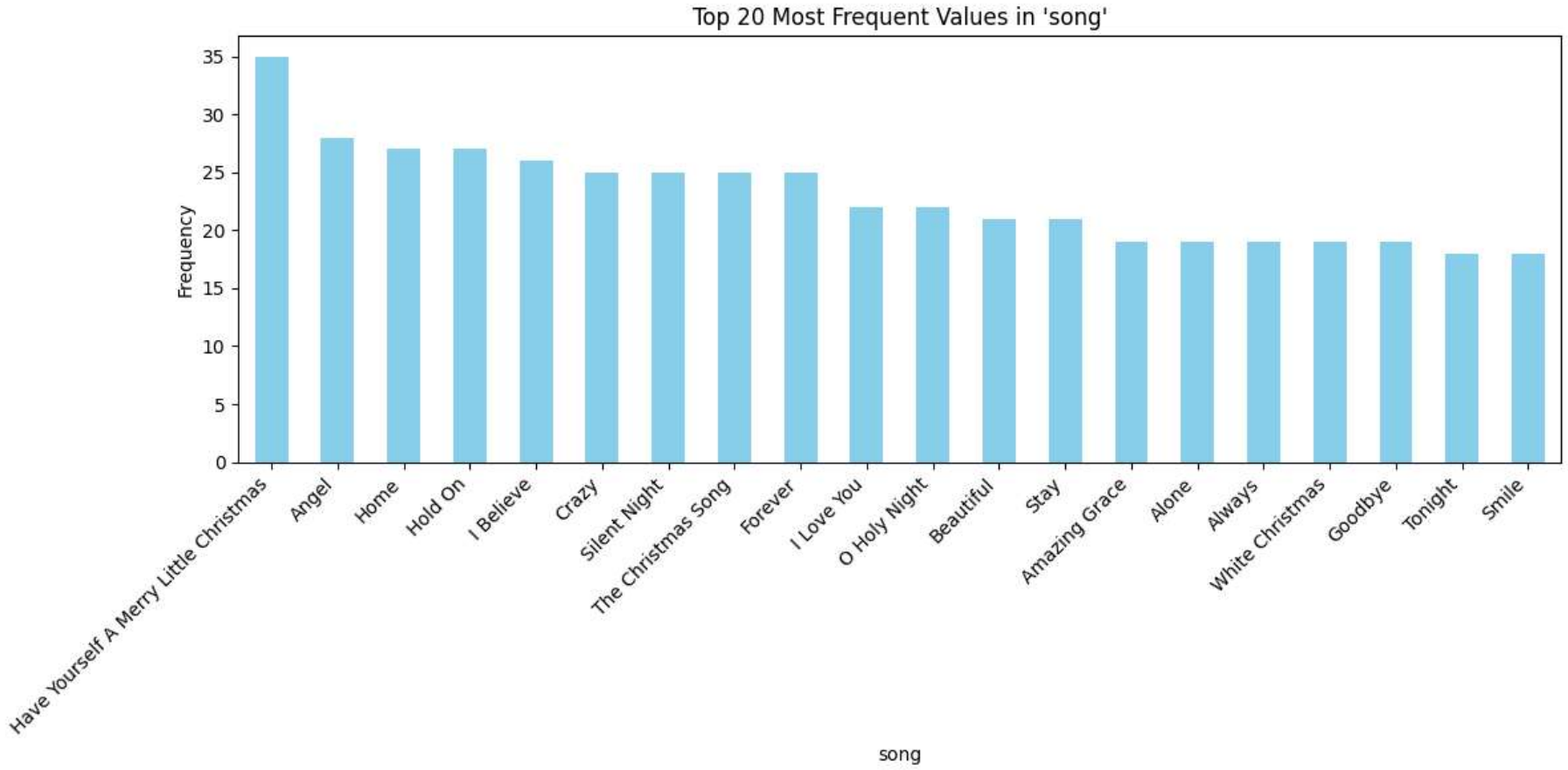
Lyrics Structure:

- Lyrics split by verses/paragraphs (\r\n)
- Repeated lines = choruses/refrains
- Variable length; may include [Chorus] or musical instructions
- Keywords indicate themes (e.g., “love”, “Christmas”)
- Formatting may vary (spaces, capitalization)
- Same lyrics can appear for different artists (covers)

Unique Values Insights:

- 89 songs per artist on average
- Many songs share the same title → different versions/artists
- Some lyrics are repeated across artists → same song, different performer

Exploratory Data Analysis



Data Processing Pipeline

Goal

Enhance the dataset with **musical metadata** retrieved from the **Last.fm API**:

- Artist description
- Album title
- Release date
- Track description

General Setup

API Root: http://ws.audioscrobbler.com/2.0/
Common Parameters: method, api_key, format=json

Main Steps

1. Normalize Input Rows:
   - Extract clean (song, artist) pairs
   - Standardize capitalization and spacing
2. Track SearchAPI Method: track.search
   - Retrieves→ track_mbid, artist_mbid, track_name, artist_name
   - Used to identify the most accurate match for each song-artist pair.
3. Artist MetadataAPI Method: artist.getInfo
   - Extracts→ bio.content / bio.summary, name
   - Cleans HTML and removes boilerplate text
   - Provides background and descriptive details about the artist.
4. Track MetadataAPI Method: track.getInfo
   - Extracts→ wiki.content / summary (track description)→ wiki.published (release date)→ album.title, album.mbid
   - Adds contextual metadata such as album title and release info.

Enriched Dataset

The dataset produced by the pipeline is organized into two main tables, Musics and Artists, each containing enriched metadata collected from the Last.fm API.

1. Musics Table

- **Lyrics** – Song lyrics
- **Album** – lastfm_album_name
- **Link** – Song identifier
- **Description** – lastfm_track_description
- **Release Date** – lastfm_release_date

2. Artists Table

- **Artist** – Artist name
- **Last.fm Name** – lastfm_artist_name
- **Bio** – lastfm_artist_bio

Data Analysis

General Checks

- Null values → Identify missing data
- Duplicate rows → Detect repeated records
- Unique values per column → Check data variety

Individual Checks

- Short strings → Validate artist_bio and lyrics
- Wrong artist bios → Remove nulls and detect invalid entries containing:
  - "There are"
  - "Read more on Last.fm"

Columns Removed

- Songs: link, lastfm_track_description, lastfm_release_date
- Artists: lastfm_artist_name

Handling Nulls

- album_name → Replace nulls with "Single"

Column Renaming

- lastfm_artist_bio → artist_bio
- lastfm_album_name → album_name

Columns Added

- Songs: artist_id (connect to Artists), id (unique song identifier)

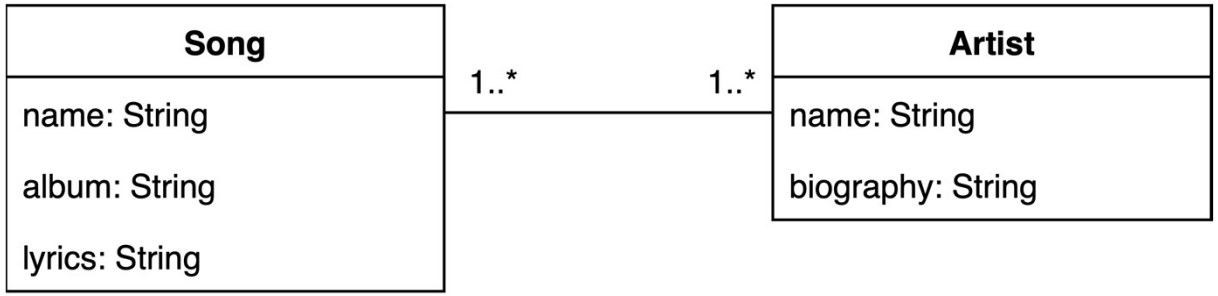
System Conceptual Model

Entities:

- **Artist** (id, name, description)
  - Has → **Songs**
- **Song** (id, name, lyrics, album_name, artist_id)
  - Performed by → **Artist**

Relationships:

One Artist → One to many songs
One Song → One artist



Definition and Characterization of the Final Collection

After the **data cleaning** and **preprocessing** steps, two final datasets were obtained:

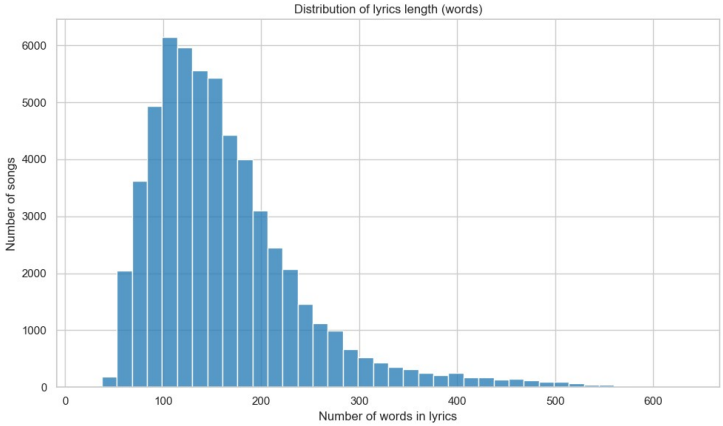
•Song → Contains information about each song, including: id, artist_id, title, album_name, and lyrics.
  •Each row in the **Song** dataset represents a unique song, including relevant metadata and a reference to its artist. In total, the dataset contains **55,185 entries**.

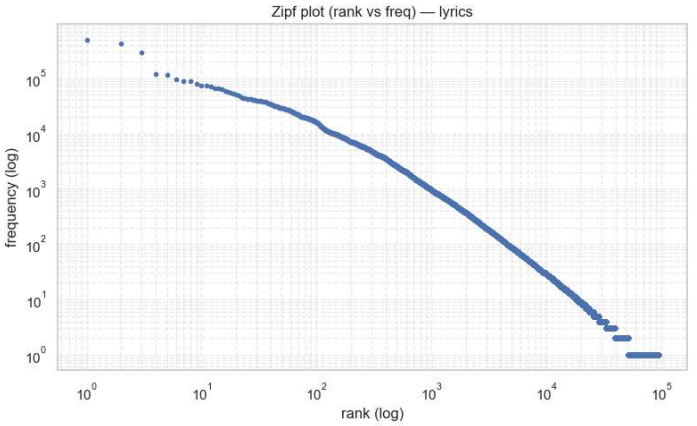
•Artist → Contains information about each artist, including: id, artist_name, and artist_bio.
  •Each row in the **Artist** dataset represents a unique musical artist with cleaned and validated information. The dataset contains **643 entries** in total.

These datasets are **consistent**, **well-structured**, and **ready for analysis or integration** into the Solr.
We chose to **store them in CSV format** because it is:

•**Lightweight** and **easy to handle**
•**Widely supported** across different tools and programming languages
•**Easy to visualize** in tools like Excel or pandas
•Ideal for **data exchange** and **further processing**

Definition and Characterization of the Final Collection





Information Needs

- **Search songs by sentiment** – Ability to find songs based on emotions or mood expressed in the lyrics.
- **Similarity-based search** – Find songs that are similar in style, theme, or lyrical content.
- **Filter songs by violence** – Identify and exclude or focus on songs containing violent themes.
- **Relate artist background to lyrics** – Connect the personal or professional background of an artist with the content of their songs.