Zuzanna Mróz, Kacper Staroń, Jakub Szypuła

BD3-AO3 Raport końcowy

1. Opis celu biznesowego projektu

Ważnym elementem budowania marki jest nie tylko szeroka popularność w ogóle społeczeństwa ale również rozmiar oddanej grupy fanów, którzy będą głównymi osobami które kupują gadżety i inne produkty, a w przypadku muzyków płyty i uczęszczają na koncerty. Pojedynczy zakup albumu dla artysty może oznaczać tyle samo co nawet 1000 odsłuchań na Spotify. Dlatego ważnym elementem monitorowania popularności zespołu w celu dobrania odpowiedniej strategii marketingowej musi być monitoring zarówno jak i tej “ogólnej” popularności jak i rozwoju tej bardziej zaangażowanej publiczności.

Celem naszego projektu jest przygotowanie architektury która pozwoli pobierać dane z serwisów pisanej twórczości fanowskiej (tzw. fanfiction), która w połączeniu z adekwatnymi danymi dostępnymi przez spotify pozwoli użytkownikom na dokonanie bardziej adekwatnej analizy biznesowej w celu zbudowania marki.

Wszystkie skrypty wykorzystane w tym projekcie znajdują się na poniższym repozytorium GitHub:
https://github.com/HeroOfElFacil/BD3-AO3

2. Opis wykorzystywanych zbiorów danych

W poniższej sekcji opisane zostaną zestawy danych wykorzystane w projekcie. Na potrzeby analizy danych pobraliśmy informacje o 10 najpopularniejszych zespołach pod względem liczby tekstów w serwisie AO3.

2.1 Spotify

Spotify jest najpopularniejszym muzycznym serwisem streamingowym, który umożliwia streaming muzyki na żądanie.

Dane ze spotify pobierane są za pomocą skryptu napisanego w języku Python poprzez udostępnione API (biblioteka spotipy). Dotyczą one podstawowych informacji dostępnych w serwisie streamingu muzycznego. Częstotliwość pobierania zależy od potrzeb użytkownika, gdyż przede wszystkim opiera się na liczbie analizowanych zespołów. Dane używane tutaj można rozumieć jako odpowiednik wymiaru w modelu hurtowni danych.

W poniższych tabelkach znajdują się opisy tabeli pobieranych za pomocą API spotify:

bands (liczba rekordów: taka jak liczba zespołów których interesuje użytkownik) - tabela zawierająca podstawowe informacje o zespołach informacje

Nazwa kolumny	**Opis**	**Typ**	**Format**
BandName	Nazwa zespołu	String	tekst
Followers	Liczba osób która podąża za danym artystą na Spotify	Int	liczba całkowita
Popularity	Wskaźnik popularności wg spotify (od 1 do 100)	Int	liczba całkowita

genres (liczba rekordów: kilka razy więcej niż w tabeli **bands**) - tabela słownikowa zawierająca pary nazwa zespołu-gatunek muzyczny na podstawie informacji ze Spotify

Nazwa kolumny	**Opis**	**Typ**	**Format**
BandName	Nazwa zespołu	String	tekst
Genre	Nazwa gatunku muzycznego	String	tekst

albums (liczba rekordów: kilka razy więcej niż w tabeli **bands**) - tabela zawierająca informacje o albumach wydanych przez każdy zespół. Ze względu na różne wersje regionalne, zdarza się, że w spotify pojawiają się więcej niż raz pary zespół-nazwa albumu-data wydania, w takim wypadku duplikaty były usuwane.

Nazwa kolumny	**Opis**	**Typ**	**Format**
BandName	Nazwa zespołu	String	tekst
AlbumName	Tytuł albumu	String	tekst
ReleaseDate	Data wydania albumu	Date	YYYY-MM-DD

2.2 AO3

AO3 jest internetowym serwisem zbierającym fanowską twórczość pisaną (tzw. fanfiction), który w marcu 2021 odnotowywał 60 milionów wejść dziennie. Dane są pobierane przez Python za pomocą biblioteki AO3. Ponownie, jak w wypadku spotify, dane mogą być pobierane wedle uznania użytkownika.

W poniższych tabelkach znajdują się opisy tabeli pobieranych za pomocą pakietu AO3:

ao3_band_fic_metadata_clean (liczba rekordów: jeden rekord na jeden tekst, w naszym wypadku 424665) - tabela zbierające informacje na temat tekstów zamieszczonych w serwisie AO3

Nazwa kolumny	**Opis**	**Typ**	**Format**
id	ID tekstu	int	liczba całkowita
date_updated	Data zamieszczenia tekstu w serwisie	datetime	YYYY-MM-DD hh:mm:ss
bookmarks	Liczba “zakładek” odnoszących się do tekstu	int	liczba całkowita
nchapters	Liczba rozdziałów	int	liczba całkowita
complete	Informacja czy tekst został zakończony, czy nie	Bool	True/False
comments	Liczba komentarzy	int	liczba całkowita
hits	Liczba wyświetleń	int	liczba całkowita
kudos	Liczba “polubień”	int	liczba całkowita
language	Język w jakim napisano tekst	String	tekst
rating	Klasyfikacja wiekowa	String	tekst
status	Status tekstu	String	tekst
words	Liczba słów	int	Liczba całkowita

ao3_band_tag_metadata_clean (liczba rekordów: n razy więcej niż w przypadku **ao3_band_fic_metadata_clean**, zależne to od ilości metadanych) - tabela słownikowa zawierająca informacje na temat metadanych powiązanych z tekstem.

Nazwa kolumny	**Opis**	**Typ**	**Format**
id	ID tekstu	int	liczba całkowita
meta_name	Typ metadanych zawartych w wierszu	String	tekst ze zbioru: authors, categories, characters,

			fandoms, relationships, series, tags, warnings
meta_val	Wartość metadanej	String	tekst

3. Diagram oraz opis architektury stworzonego systemu i wykorzystanych narzędzi

Dane pochodzą, jak wspomniano, z dwóch źródeł:

- AO3
- Spotify

Dane te są pobierane za pomocą skryptów Pythonowych, które obsługują połączenia przez API oraz dokonują podstawowej inżynierii danych. Podstawowy przepływ został przedstawiony na diagramie poniżej:

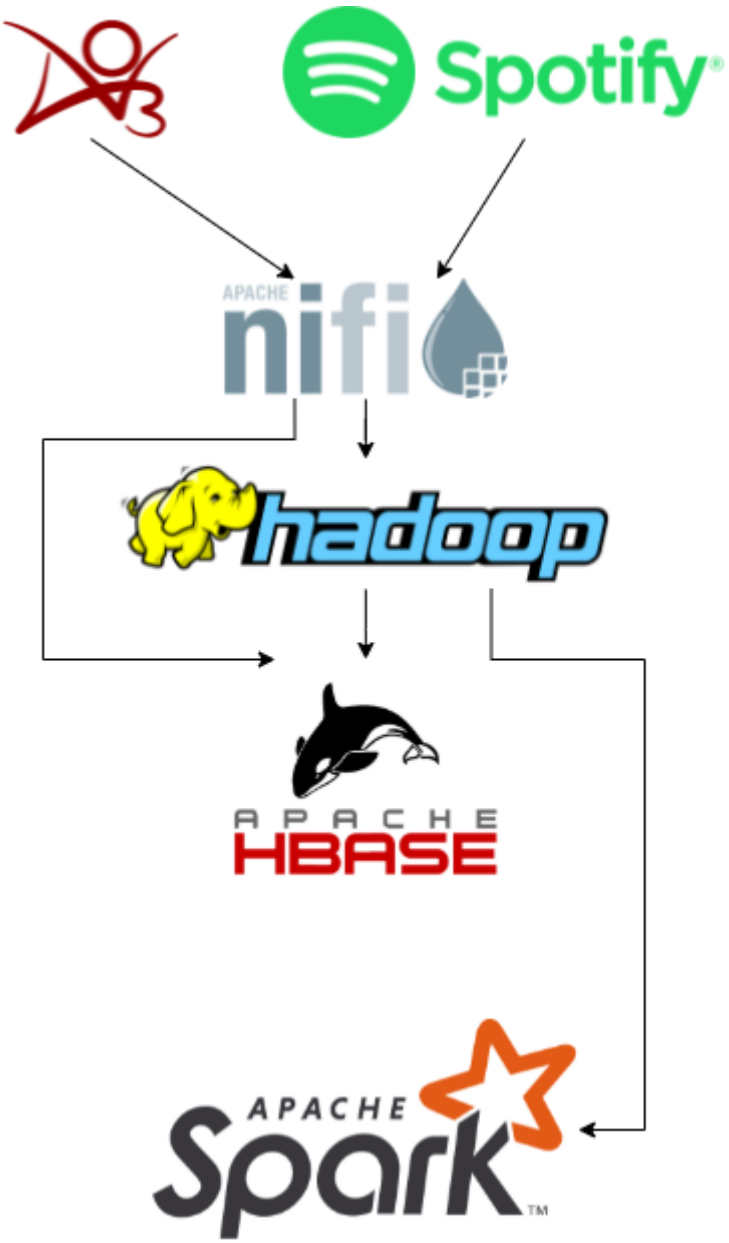
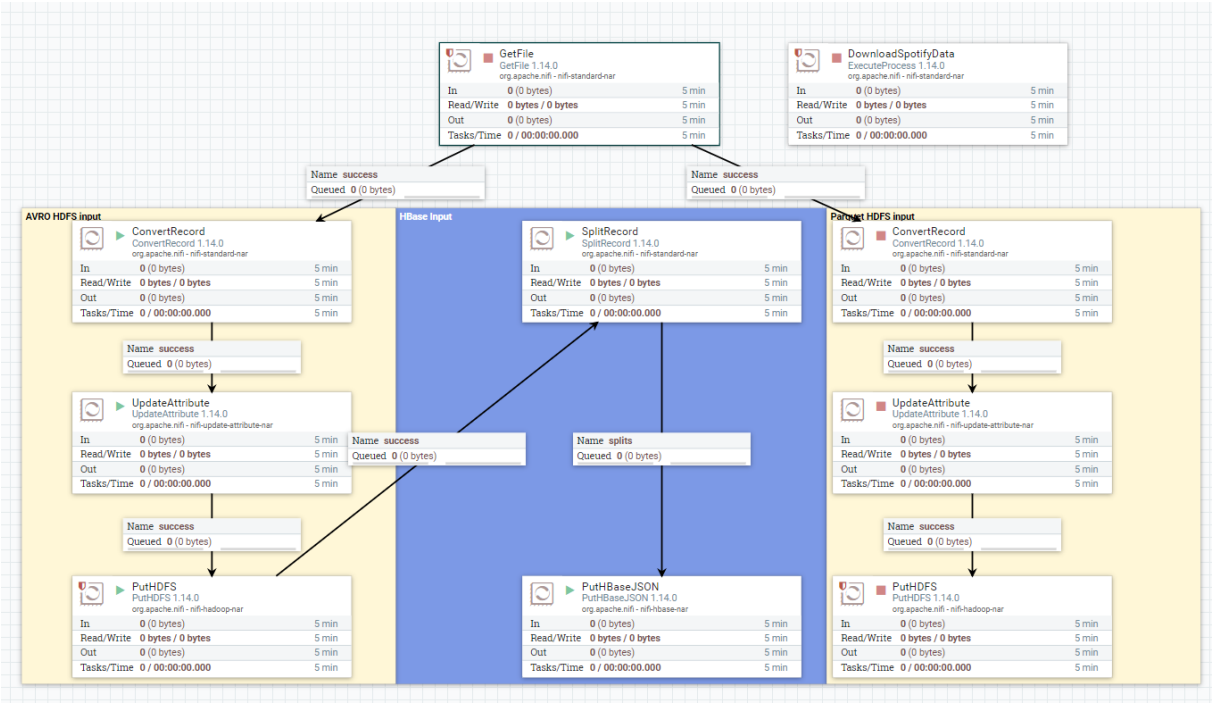


Diagram faktycznych i potencjalnych przepływów danych

Role poszczególnych komponentów:
-NiFi: koordynacja przepływów i ładowania danych, przetwarzanie danych wejściowych
-Hadoop: przechowywanie przetworzonych plików źródłowych
-HBase: przechowywanie danych ze wsparciem random read w celu potencjalnego szybszego przetwarzania go później
-Spark: narzędzie do inżynierii i analizy danych które może czerpać dane z HDFS oraz HBase

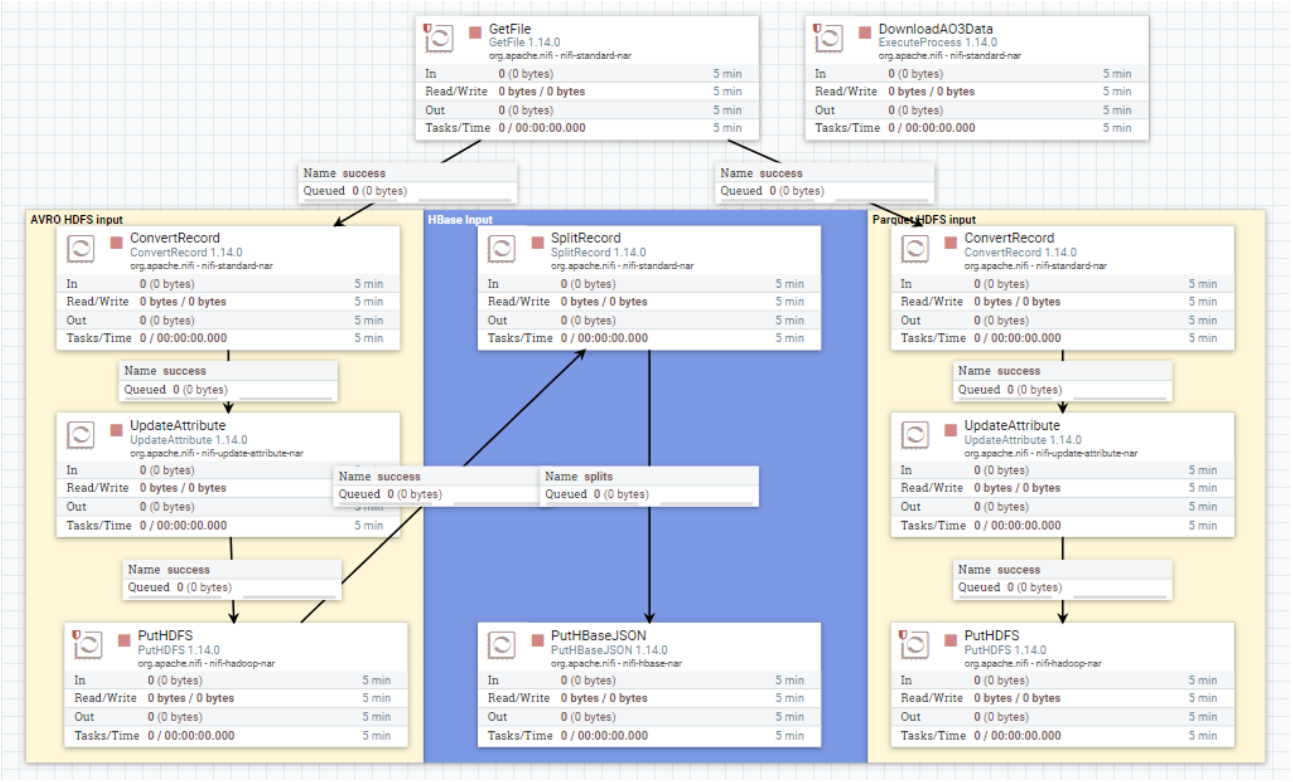
4. Opis sposobu pozyskiwania, przetwarzania i składowania danych źródłowych

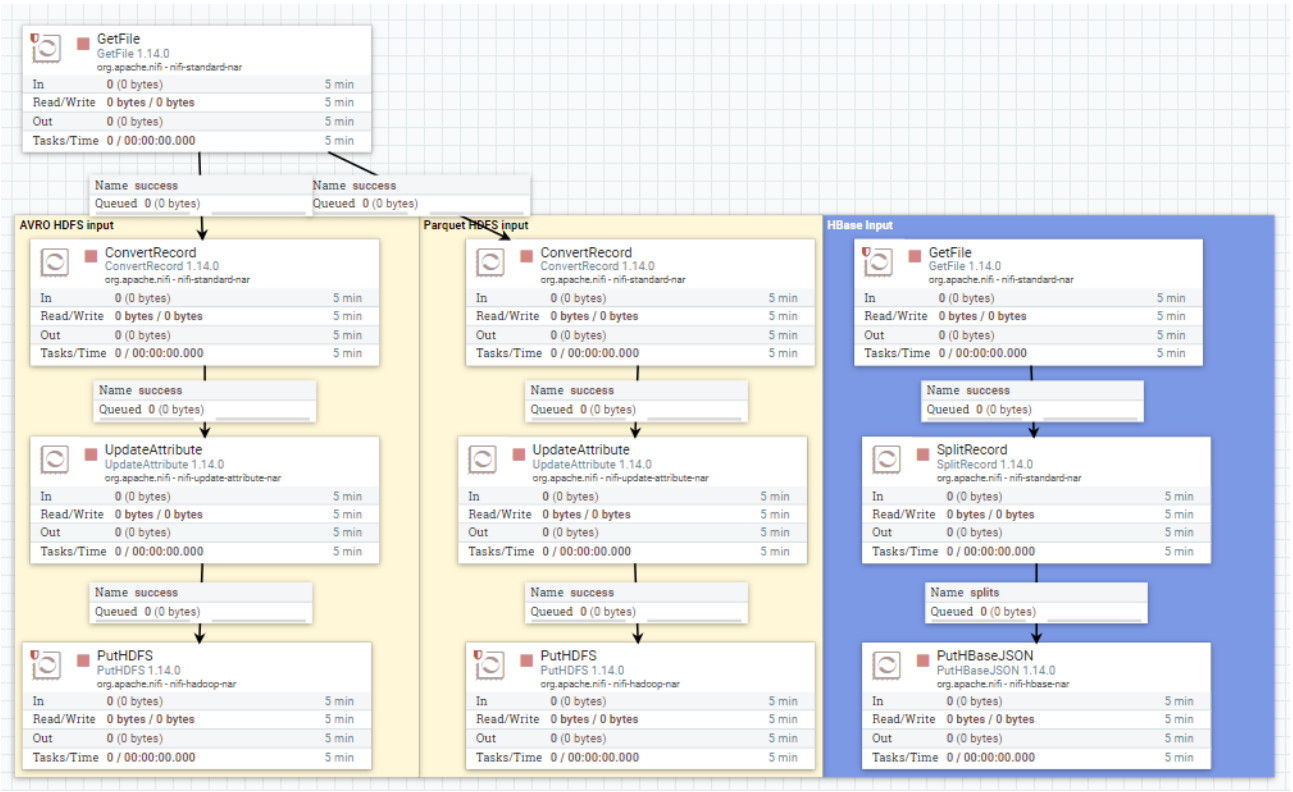
W pierwszej kolejności dane są ładowane poprzez skrypty do plików .csv, które są ładowane do Apache NiFi. Poniżej przedstawione są DataFlow w NiFi:



Ładowanie danych Spotify w NiFi

Pierwszym krokiem jest uruchomienie skryptu (tutaj procesor DownloadSpotifyData), który tworzy odpowiednie pliki .csv: albums, bands, genres (w naszym wypadku w folderze /home/vagrant/nifi/). Format danych zawartych w tych plikach jest taki sam jak zawarty w sekcji 2. Następnie dane te są transformowane w dwa różne formaty: AVRO i Parquet. Pierwszy jest domyślnym, ze względu na fakt, że przechowuje on dane wierszowo, a naszą główną motywacją jest to, żeby móc łatwo dostać się do poszczególnych wykonawców, albumów, czy tekstów. Drugi jest opcją pomocniczą, wybraną ze względu na dobre wsparcie przy ładowaniu danych do analiz w Apache Spark. Następnie zmieniana jest nazwa pliku, żeby nadać mu poprawne rozszerzenie, a ostatecznie dane są ładowane do HDFS.
Dalej w tym samym flow dane są ładowane do odpowiednich tabel w HBase, a uprzednio konwertowane w tym celu do zbioru wierszy w formacie JSON.





Ładowanie danych AO3 w NiFi

Flow dla AO3 jest analogiczny, jest odrębny ze względu na fakt, że są to osobne źródła i dzięki temu możliwa jest równoległa praca nad ładowaniem danych dla Spotify i AO3. W tym wypadku ładowane .csv to ao3_band_fic_metadata_clean.csv i

ao3_band_tag_metadata_clean.csv, ponownie, z takim samym formatem danych jak opisany został w 2. sekcji.
Główną zmianą w stosunku do flow dla danych Spotify jest ładowanie danych do HBase'a z plików pomocniczych, trzymających jedynie podzbiory kolumn pliku wejściowego odpowiadające column families w tabel ao3_band_tag_metadata_clean.

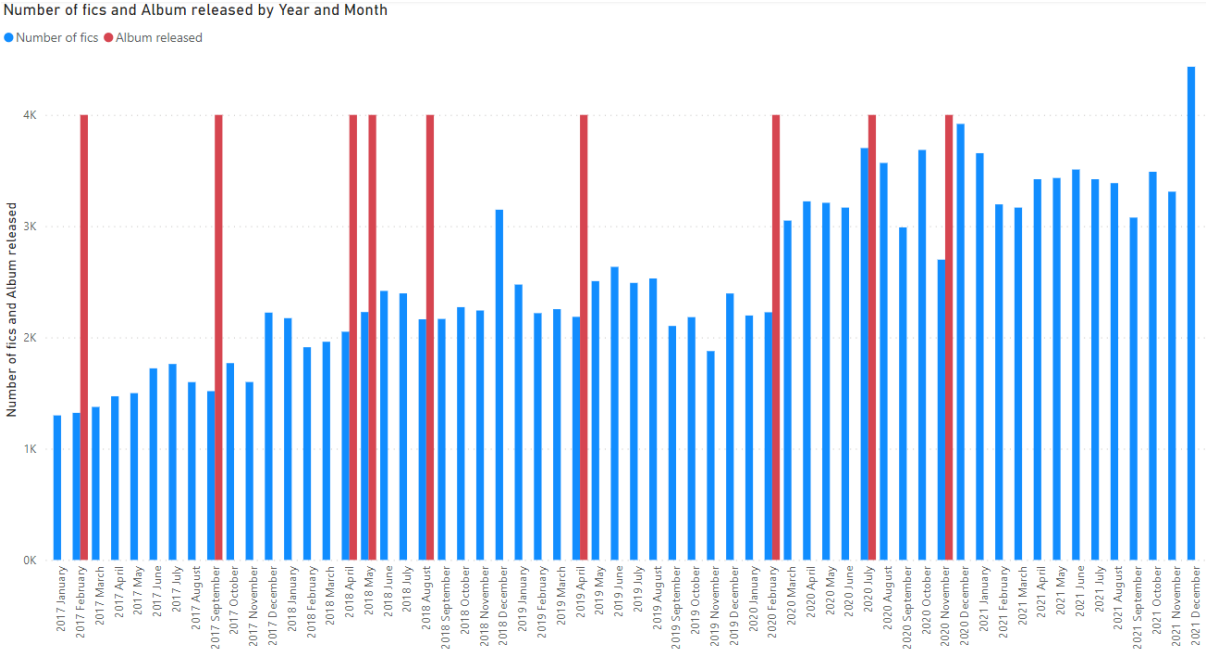
5. Opis sposobu analizy danych i rodzaju generowanych widoków wsadowych

Dane są analizowane poprzez skrypty Pythonowe wykonywane przez Apache Spark. W naszym rozwiązaniu dostępne są następujące rodzaje widoków wsadowych:

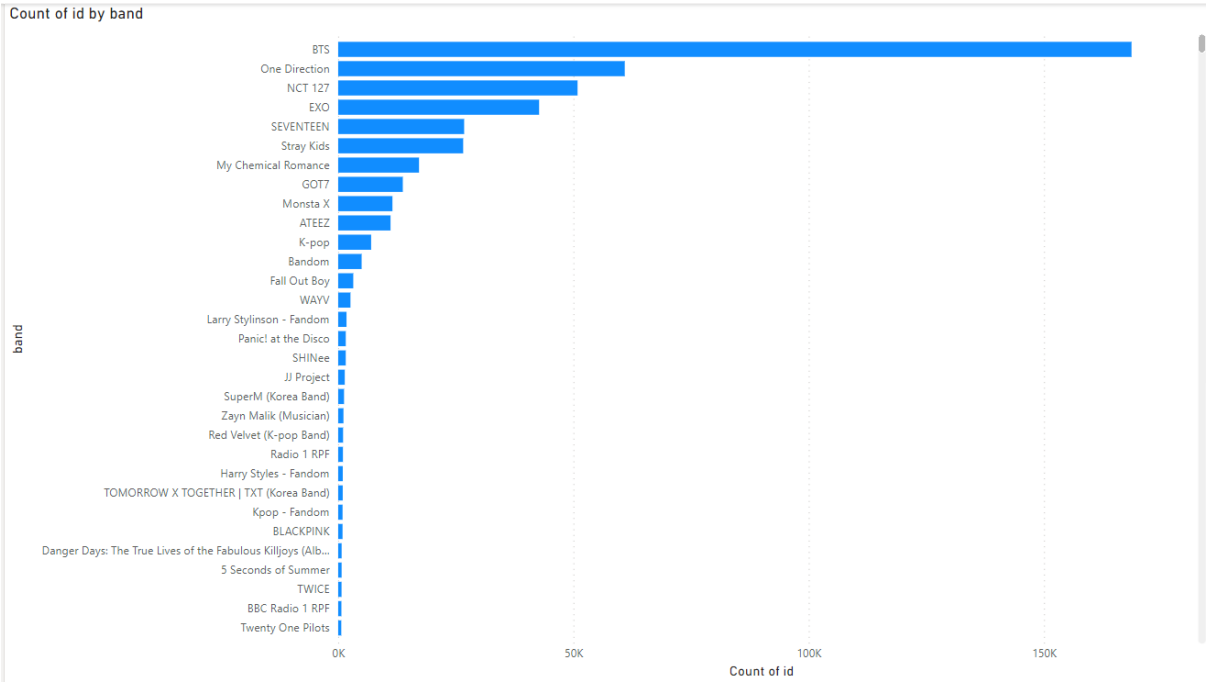
- bezpośredni dostęp do pobranych tabel
- zliczenie liczby wierszy w poszczególnych tabelach
- lista unikalnych tagów
- top 10 najpopularniejszych zespołów po ilości tekstów
- najczęstsze relacje między członkami zespołu w tekstach na podstawie liczby tekstów
- liczba tekstów dla zespołu BTS dla każdego miesiąca wg daty wydania
- kiedy wychodziły poszczególne albumy dla danych zespołów wg miesięcy

6. Przykłady danych dostępnych dla warstwy prezentacyjnej

Dla warstwy prezentacyjnej dostępne są wszystkie widoki opisane w poprzedniej sekcji. Poniżej znajdują się przykładowe wizualizacje wykonane w Power BI Desktop:



Liczba tekstów w poszczególnych miesiącach z naniesionymi datami wydania albumów.



Liczba tekstów dla każdego zespołu w pobranych danych (ponieważ pobierane teksty mogą dotyczyć więcej niż jednego utworu, jest więcej zespołów niż nasze oryginalne 10).

7. Podsumowanie finalnej wersji rozwiązania

Nasze rozwiązanie obejmuje cały proces pobierania, przetwarzania i analizy danych z serwisów Spotify oraz AO3. Dzięki temu rozwiązaniu można w prosty i zautomatyzowany sposób analizować interesujące nas dane. W obecnym rozwiązaniu, żeby użytkownik mógł pobrać interesujące go dane na temat danego zespołu, musi przeprowadzić następujące kroki:

1. Edytować plik .csv z listą nazw zespołów
2. Zmodyfikować Pythonowy skrypt pobierania danych z AO3
3. Uruchomić flow NiFi
4. Uruchomić analizy Apache Spark

Więc jest to bardzo prosty proces, który pozwala na oszczędzenie pracy przy próbie analizy danych w ten sposób.

8. Opis podziału pracy w zespole

Zuzanna Mróz - pobieranie i wstępna obróbka danych z AO3, analizy Spark, testy
Kacper Staroń - ładowanie danych do HBase, NiFi, analizy Spark, testy
Jakub Szypuła - pobieranie danych ze spotify, NiFI, wizualizacje pomocnicze, dokumentacja końcowa