# Primeiro exercício de programação

[PDMN2020-2] Exercício de programação (Lemonade)

**Aluno:** Danilo Pimentel de Carvalho Costa
**Matrícula:** 2016058077

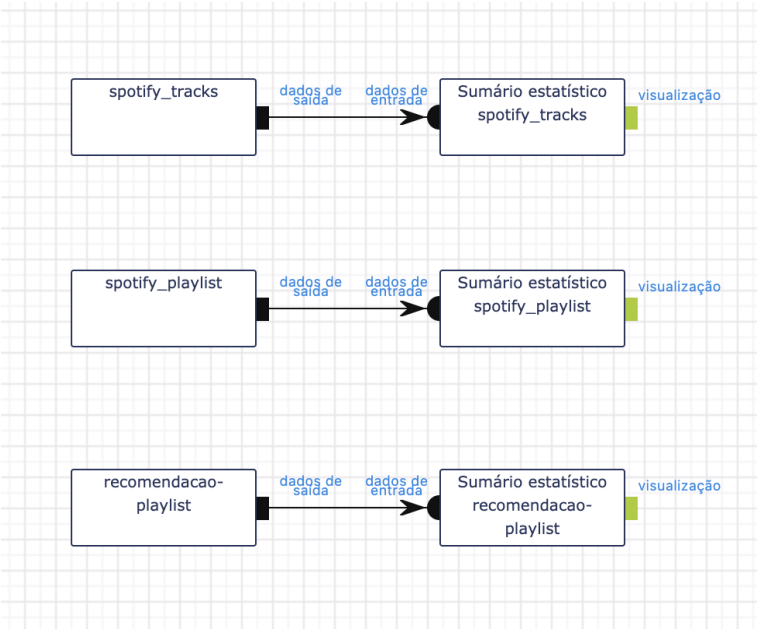
Deverá ser escrito um relatório para explicar as análises, os resultados produzidos e como cada análise foi construída no Lemonade. O aluno deverá entregar um único arquivo compactado contendo o relatório produzido com as imagens dos fluxos criados, junto com seus arquivos JSON produzidos pelo Lemonade. Será criada uma tarefa no moodle para a entrega. Caso tenha alguma sugestão para a plataforma do Lemonade, fique à vontade para incluir no relatório.

## Introdução

O primeiro exercício de programação tem como objetivo introduzir o processamento de dados massivos utilizando a plataforma Lemonade. Este sistema fornece uma ferramenta visual para construção de fluxos de trabalho para diversas tarefas de análise de dados. Para este trabalho são propostas 4 análises em uma base de dados de música extraídos do Spotify. A seguir, se encontram explicações, resultados  e como cada análise foi construída no Lemonade, bem como uma breve análise do conjunto de dados.

## Conjunto de dados

Para conhecer melhor o conjunto de dados antes de realizar as análises, foram feitos sumários estatísticos das 3 partes do conjunto de dados: recomendacao-playlist, spotify_playlist e spotify_tracks.



Sumário estatístico recomendacao-playlist

| Atributo | Máximo | Mínimo | Desvio padrão | Contagem | Média | Distintos (aproximado) | Ausentes | Assimetria (skewness) | Curtose (kurtosis) |
|---|---|---|---|---|---|---|---|---|---|
| b_artist_name | Willie Nelson | Andy Williams | - | 37 | - | 36 | 0 | - | - |

One record

A parte recomendacao-playlist é uma lista de nomes de artistas. As informações relevantes A estatística relevante aqui é a Contagem. Pela contagem, podemos ver que esta parte do conjunto de dados não é grande.

Sumário estatístico spotify_playlist

| Atributo | Máximo | Mínimo | Desvio padrão | Contagem | Média | Distintos (aproximado) | Ausentes | Assimetria (skewness) | Curtose (kurtosis) |
|---|---|---|---|---|---|---|---|---|---|
| pid | 199999 | 0 | 57776.3287 | 191000 | 102331.9607 | 200282 | 0 | -0.07 | -1.19 |
| name | WWW | ! 2017 Songs | - | 191000 | - | 34735 | 0 | - | - |
| modified_at | 1509494400 | 1280275200 | 36717052.0199 | 191000 | 1476323420.2304 | 2180 | 0 | -1.32 | 1 |
| duration_ms | 332180189 | 555696 | 12874787.3425 | 191000 | 15651272.4295 | 202361 | 0 | 1.51 | 5.34 |
| num_albums | 243 | 2 | 39.9121 | 191000 | 49.7862 | 241 | 0 | 1.38 | 1.69 |
| num_artists | 224 | 3 | 30.1484 | 191000 | 38.0994 | 214 | 0 | 1.5 | 2.56 |
| num_edits | 191 | 2 | 20.7985 | 191000 | 17.8479 | 187 | 0 | 2.5 | 7.96 |
| num_followers | 31539 | 1 | 112.5808 | 191000 | 2.6665 | 231 | 0 | 193.47 | 44618.81 |
| num_tracks | 250 | 5 | 53.8317 | 191000 | 66.7624 | 249 | 0 | 1.27 | 1.02 |

9 records

A parte spotify_playlist contém mais campos. Olhando novamente para a Contagem, podemos ver que é um conjunto um pouco maior, com 191000 registros. Para todos os registros, todas as colunas têm valores. A playlist mais antiga é de 28 de julho de 2010, e a mais recente é de 1 de novembro de 2017. Isto confirma a informação inicial sobre o período de coleta das playlists. Como curiosidade, podemos ver que a playlist mais longa possui incríveis 92 horas.

Sumário estatístico spotify_tracks

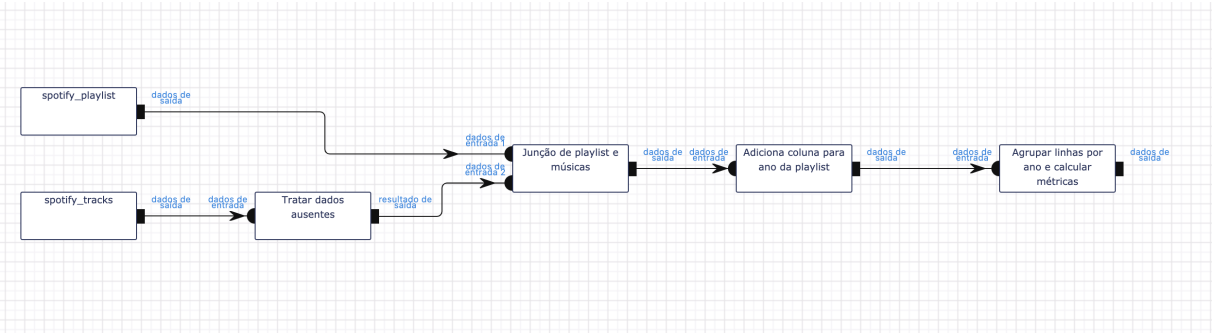
| Atributo | Máximo | Mínimo | Desvio padrão | Contagem | Média | Distintos (aproximado) | Ausentes | Assimetria (skewness) | Curtose (kurtosis) |
|---|---|---|---|---|---|---|---|---|---|
| pid_playlist | 199999 | 0 | 57724.7887 | 12733217 | 102220.9618 | 200282 | 8561 | -0.07 | -1.19 |
| album_name | 환생 Rebirth | ! | - | 12733217 | - | 303561 | 8561 | - | - |
| album_uri | spotify:track:5b88tNlNg4Q4nrRbrCXUmg | 1963 (Oh What A Night!)" | - | 12733217 | - | 388962 | 8561 | - | - |
| artist_name | 2 8 1 4 | Bump" | - | 12733203 | - | 144640 | 8575 | - | - |
| artist_uri | spotify:track:6T7xieqFqFkWemSLvV30Ch | spotify:artist:0001ZVMPt41Vwzt1zsmuzp | - | 12733133 | - | 150922 | 8645 | - | - |
| duration_ms | 10435467 | 0 | 73488.8314 | 12733110 | 234372.3999 | 196577 | 8668 | 13.34 | 735.83 |
| pos | 249 | 0 | 48.338 | 12733110 | 54.5788 | 254 | 8668 | 1.2 | 0.97 |
| track_name | ◆äkäskero | ! (Foreword) | - | 12733110 | - | 693323 | 8668 | - | - |
| track_uri | spotify:track:7zzyrYnZlfvYAGwl7lRb7X | '71 - '93" | - | 12733110 | - | 930212 | 8668 | - | - |

9 records

Na parte referente à spotify_tracks, podemos ver que há dados ausentes para todas as colunas. Com isso, nas análises feitas é preciso adicionar tratamento para dados ausentes a fim de melhorar os resultados para as perguntas.

# Análise 1

**Existe uma mudança de preferência na duração das músicas com o passar dos anos? Apresente a relação da duração mínima, média e máxima das músicas em função dos anos dessa amostra. Considere o último timestamp de alteração da lista de reprodução para responder essa pergunta.**




A resposta desta pergunta requer a junção de duas partes do conjunto de dados: spotify_playlist e spotify_tracks. Como visto, é preciso tratar valores ausentes em spotify_tracks. Os atributos utilizados no fluxo foram pid_playlist e duration_ms. Foram removidas as linhas que possuem valores ausentes para estes atributos.

A relação requerida é de ano para duração máxima, mínima e média das músicas. Para determinar qual o ano para cada música, é preciso juntar seus dados com os dados das listas de reprodução em que ela foi colocada.

Uma música pode ter sido colocada em mais de uma lista de reprodução, e portanto pode aparecer em mais de ano. Isto não é um problema, uma vez que o pedido é para analisar o tempo de duração das músicas executadas naquele ano.

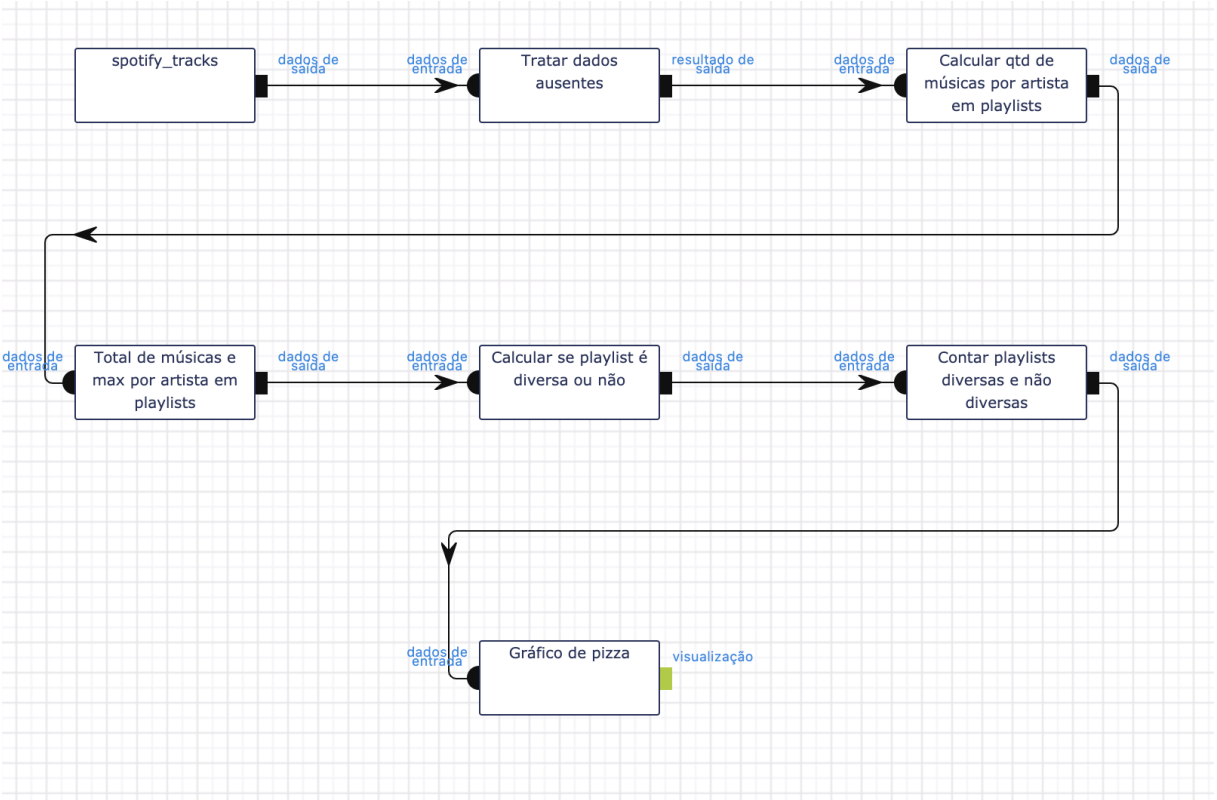
O momento da última alteração de uma lista está representado em timestamp, e o tipo de dado é inteiro. Foi preciso fazer uma conversão para o tipo Timestamp, e a partir dele formatar para o ano do qual este timestamp está. Esta nova coluna é adicionada para cada linha.

Agora podemos agrupar as músicas por ano. O agrupamento calcula o que se pede: máxima, média, e mínima duração das músicas em cada ano. O resultado é mostrado a seguir:

| # | year | max_duration_ms | min_duration_ms | avg_duration_ms |
|---|---|---|---|---|
| 1 | 2016 | 9744610 | 0 | 233630.5768679875 |
| 2 | 2012 | 4195000 | 4520 | 241151.13290607271 |
| 3 | 2017 | 10435467 | 0 | 233176.0770812087 |
| 4 | 2014 | 5425981 | 176 | 240208.5452555248 |
| 5 | 2013 | 6348017 | 192 | 241781.12652170748 |
| 6 | 2011 | 4088794 | 25364 | 244664.25819672132 |
| 7 | 2015 | 7551112 | 0 | 237816.8486539514 |
| 8 | 2010 | 578106 | 179493 | 260998.0 |

# Análise 2

**O que é mais comum, playlists onde existem mais músicas distintas de um mesmo artista (considere mais de 50%) ou playlists mais diversificadas?**



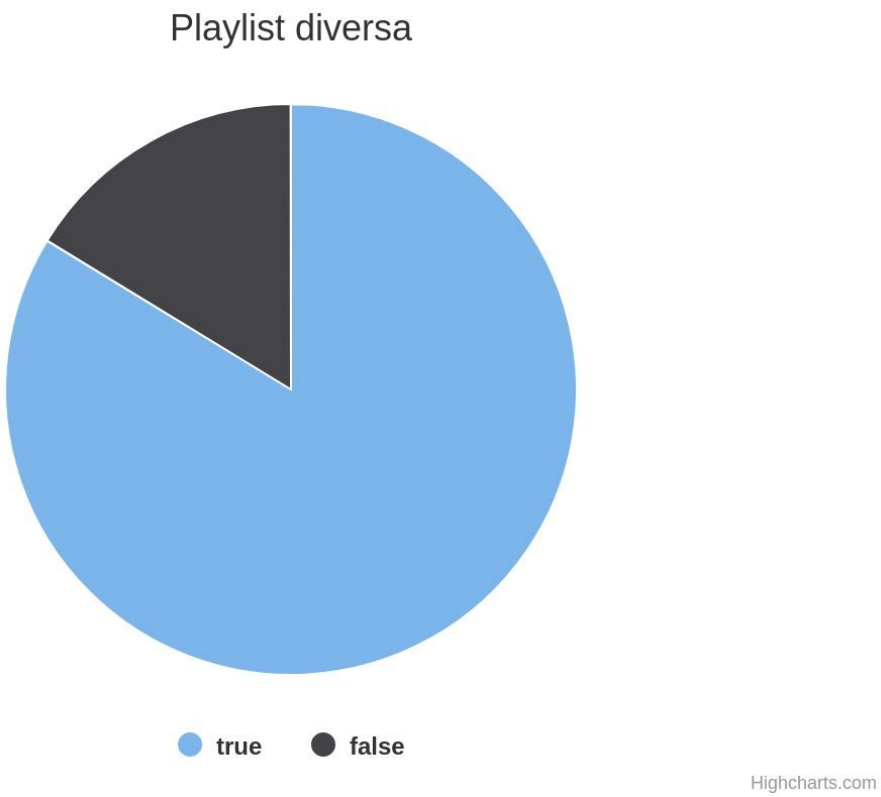

O fluxo para responder esta pergunta depende somente de spotify_tracks. Esta parte do conjunto de dados já possui informações de qual playlist (pid_playlist) e a qual artista as músicas pertencem (artist_uri). As linhas que possuem valores ausentes para estes atributos foram removidas.

Para cada playlist, precisamos determinar a quantidade de músicas por artista. Foi realizada uma agregação por pid_playlist e artist_uri, e um novo atributo foi adicionado com a contagem de ocorrências.

Com a quantidade de músicas por artista, podemos agora saber para cada playlist qual o artista com mais músicas, além de poder determinar o total de músicas em cada playlist. Para isto, fazemos uma agregação por pid_playlist somando a quantidade de músicas de todos os artistas na playlist e determinando o maior número de músicas do artista com mais músicas na playlist.

Com o total e a maior quantidade de músicas por artista da playlist, podemos dizer se a playlist é diversificada (a maior quantidade é menor que metade do total de músicas) ou não. Para fazer isso, criamos um novo atributo "diversificada" com uma transformação que realiza o cálculo descrito.

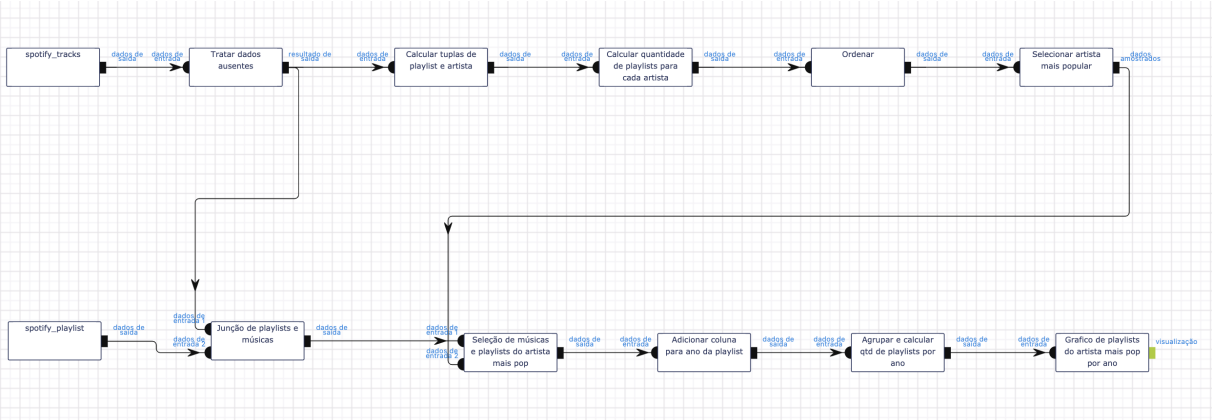
Agora temos a informação de diversificada ou não para cada playlist. Finalmente, fazemos uma agregação em "diversificada", contando quantas playlists são diversificadas e quantas não são. Com estes valores em mão, podemos responder a pergunta gerando um gráfico de pizza mostrado a seguir:




Claramente, há mais playlists diversificadas no conjunto de dados.

# Análise 3

**Para o artista mais popular (isto é, presente no maior número de listas), crie um gráfico que relacione o número de playlists distintas em que esse artista apareceu com o passar dos anos.**




Esta análise requer duas partes: encontrar o artista mais popular e encontrar o número de playlists deste ao longo dos anos. No diagrama, o fluxo nas caixas de cima são referentes à primeira parte, e o fluxo nas caixas de baixo se referem à segunda.

O critério de popularidade para um artista é calculado com base na presença em listas de reprodução. A parte spotify_tracks é suficiente para determinar o artista mais popular, uma vez que possui as informações de pid_playlist e artist_uri. Para começar o fluxo, é preciso tratar os dados ausentes nestas colunas removendo as linhas com valores ausentes.

Após a limpeza de spotify_tracks, precisamos determinar em quais listas um artista aparece, ou seja, teve uma música de sua autoria adicionada. Podemos encontrar esta informação agrupando as linhas em spotify_track por pid_playlist e artist_uri, contando quantas músicas estão no grupo somente para ser um agrupamento válido, já que este resultado não será de fato utilizado.

Tendo as tuplas de playlist e artista, precisamos determinar o número de playlists para cada artista. Agrupamos estas tuplas por artista, e contamos quantas linhas temos em cada grupo. Chamamos este resultado de artista_qtd_playlists.

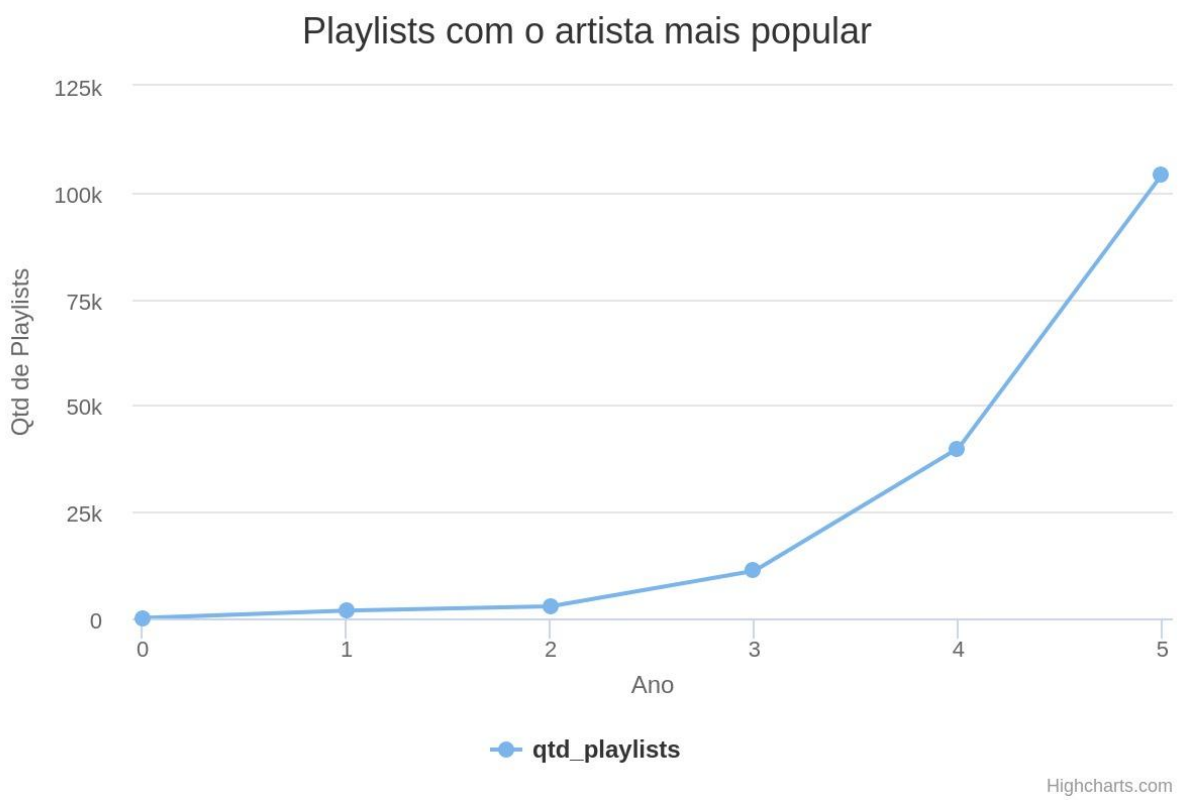
Uma vez que foi calculada a quantidade de playlists por artista, podemos determinar qual o artista mais popular. Basta ordenar em ordem descendente os registros por artista_qtd_playlists, e selecionar o primeiro artista da lista. Assim terminamos a primeira parte da análise.

Na segunda parte da análise, com o artista mais popular em mãos, temos que determinar a quantidade de playlists em que este artista esteve presente ao longo dos anos. Fazemos a seleção de playlists deste artista através de uma junção de spotify_playlists com o artista mais popular. Desta forma, filtramos pelo artista mais popular e recuperamos a informação de última modificação na mesma operação.

Para determinar a qual ano pertence uma playlist, utilizamos a mesma estratégia descrita na primeira análise, e criamos uma nova coluna year. É importante notar que a transformação foi feita depois da junção, para não transformar mais dados que precisaríamos.

Para calcular a quantidade de playlists do artista mais popular por ano, fazemos um agrupamento pelo atributo de ano criado. Contamos a quantidade de registros no grupo, e

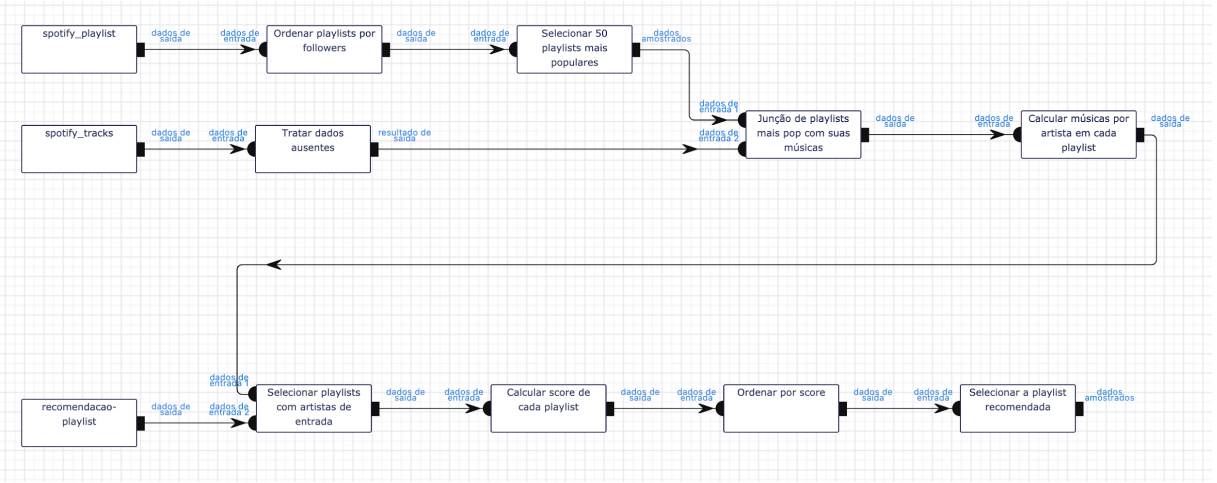
com isso estamos com todos os dados necessários para o gráfico requerido. A última parte do fluxo constrói o gráfico abaixo:




Podemos ver que o artista mais popular veio crescendo no período de análise. Nos primeiros anos este artista não era tão popular, mas veio sendo adicionado em mais playlists em uma quantidade crescente e possivelmente exponencial até o último ano de análise.

# Análise 4

**Usando as 50 playlists com o maior número de seguidores, crie um sistema simples de recomendação capaz de indicar uma única playlist com base em um conjunto de músicas passadas por parâmetro. Para isso, utilize o arquivo de entrada recomendacao-playlist, no Lemonade, contendo um conjunto de músicas para o qual você deverá indicar uma lista.**




Desambiguação: o enunciado especifica que o sistema de recomendação recebe um conjunto de músicas. Os dados disponíveis em recomendacao-playlist são somente dos autores das músicas, e não de todos os dados disponíveis para cada música em spotify_tracks.

O sistema de recomendação construído utiliza a quantidade de músicas por artista em uma playlist. Para determinar a melhor lista de reprodução, determinamos a playlist com a maior quantidade de músicas dos autores recebidos na entrada.

Primeiramente, a análise pede para considerar somente as 50 playlists mais populares. Tal resultado pode ser obtido na parte spotitfy_playlists, simplesmente ordenando em ordem decrescente por num_followers e selecionando os 50 primeiros registros.

Para determinar os autores por playlist, precisamos de pid_playlist e artist_name em spotify_tracks. Tratamos os dados ausentes removendo linhas que não possuem valores para estes atributos, e fazemos a junção das 50 playlists mais populares com os dados de suas músicas.

Para as listas de reprodução selecionadas, precisamos da quantidade de músicas por artista. Podemos calcular esta informação agrupando por pid_playlist e artist_name, e contamos a quantidade de registros em cada grupo. Assim, temos o suficiente para recomendar uma playlist para uma lista de artistas.

A lista de artistas das músicas está em recomendacao-playlist. Nosso resultado até agora nos permite realizar uma junção com estes dados por nome do autor. Desta forma, mantemos somente tuplas de playlists/autor que têm os autores em recomendacao-playlist.

Agora, determinamos a lista recomendada. Agrupamos os resultados por pid_playlist, e somamos os valores da quantidade total de músicas dos artistas do grupo criando uma espécie de "score". Basta ordenar em ordem decrescente por "score", e selecionar o primeiro resultado. O resultado para o conjunto fornecido se encontra abaixo:

| # | ds0_ds1_pid_playlist | score |
|---|---|---|
| 1 | 139482 | 16 |

# Conclusão

Pude aprender mais sobre o processo de descoberta de conhecimento (KDD) nas análises de dados propostas. Tive também uma introdução suave a ambientes de computação distribuída, e com isso posso afirmar que o Lemonade cumpre o seu papel. O tema escolhido para o primeiro exercício de programação foi complexo o suficiente para a aprendizagem dos conceitos, e simples o suficiente para implementação no prazo descrito.