



## **Relatório TP I - Integração de Sistemas de Informação**

No. 27960 – Pedro Ribeiro

No.27961 – Ricardo Fernandes

**Licenciatura em Engenharia Sistemas Informáticos**

**3ºano**

**Docente: Óscar Ribeiro**

Barcelos | outubro, 2025

## Índice de Figuras

Figura 1 - Arquitetura Geral do Sistema .....	13
Figura 2 - Componente de obter endpoint para álbum .....	15
Figura 3 - Componente de obter endpoint para playlist .....	16
Figura 4 - Componente de obter token [Bearer <token>].....	18
Figura 5 - Componente de obtenção de dados acerca de álbum (Parte 1) .....	18
Figura 6 - Componente de obtenção de dados acerca de álbum (Parte 2) .....	19
Figura 7 - Componente de obtenção de dados acerca da playlist (Parte 3) .....	20
Figura 8 - Workflow completo da obtenção de dados acerca da playlist .....	21
Figura 9 - Componente de obtenção de dados acerca da playlist (Parte 1) .....	22
Figura 10 - Componente de obtenção de dados acerca da playlist (Parte 2) .....	23
Figura 11 - Workflow da obtenção de dados acerca da playlist.....	24
Figura 12 - Componente de obtenção de dados acerca de artistas (Parte 1).....	25
Figura 13 - Componente de obtenção de dados acerca de artistas (Parte 2).....	26
Figura 14 - Workflow completo da obtenção de dados acerca de artistas.....	26
Figura 15 - Componente de obtenção de dados acerca da playlist/artista .....	27
Figura 16 - Componente de obtenção de dados acerca da exportação para base de dados.....	28
Figura 17 - Organização de todos os componentes .....	28
Figura 18 - Transformação Obter Token.....	29
Figura 19 - Transformação Get Request Playlist (Parte 1) .....	30
Figura 20 - Transformação Get Request Playlist (Parte 2) .....	31
Figura 21 - Transformação Get Request Playlist (Parte 3) .....	32
Figura 22 - Transformação Get Request Playlist (Parte 4) .....	33
Figura 23 - Transformação Get Request Playlist completa.....	33
Figura 24 - Transformação Get Request Artistas (Parte 1).....	34
Figura 25 - Transformação Get Request Artistas (Parte 2).....	35
Figura 26 - Transformação Get Request Artistas (Parte 3).....	35
Figura 27 - Transformação Get Request Artistas (Parte 4).....	36
Figura 28 - Transformação Get Request Artistas completa .....	36
Figura 29 - Transformação Join (Parte 1) .....	37
Figura 30 - Transformação Join (Parte 2) .....	38
Figura 31 - Transformação Join completa .....	38
Figura 32 - Transformação Get Album (Parte 1).....	39
Figura 33 - Transformação Get Album (Parte 2).....	40
Figura 34 - Transformação Get Album (Parte 3).....	40
Figura 35 - Transformação Get Album (Parte 4).....	41
Figura 36 - Diagrama representativo do Job.....	44
Figura 37 - Job principal do sistema .....	46
Figura 38 - Dashboard com resultado final (Pentaho Kettle) .....	47

Figura 39 - Dashboard com resultado final (Knime).....	48
Figura 40 - Tabela Dados Playlist.....	48
Figura 41 - Tabela Artistas.....	49
Figura 42 - Tabela Join .....	49
Figura 43 - Tabela Album .....	50

## Índice

1. Enquadramento .....	5
2. Problema / Objetivo .....	7
3. Estratégia utilizada.....	9
3.1 Estratégia utilizada na Plataforma KNIME.....	9
3.2 Estratégia utilizada na plataforma Pentaho Kettle .....	10
4. Abordagem geral utilizada .....	13
5. Transformações de Dados.....	15
5.1 Transformações de Dados na plataforma Knime .....	15
5.1.1. Componente de obter endpoint para álbum .....	15
5.1.2. Componente de obter endpoint para playlist.....	16
5.1.3. Componente de obter token [Bearer <token>] .....	16
5.1.3. Componente de obtenção de dados acerca de album .....	18
5.1.3. Componente de obtenção de dados acerca da playlist.....	21
5.1.4. Componente de obtenção de dados acerca de artistas .....	24
5.1.5. Componente de obtenção de dados acerca da playlist/artista .....	26
5.1.5. Componente de obtenção de dados acerca da exportação para base de dados .....	27
5.1.6. Organização de todos os componentes .....	28
5.2 Transformações de Dados na plataforma Pentaho Kettle .....	29
5.2.1 Transformação Obter Token:.....	29
5.2.2 Transformação Get Request Playlist.....	29
5.2.3 Transformação Get Request Artistas .....	33
5.2.4 Transformação Join.....	36
5.2.5 Transformação Get Album .....	38
6. Utilização de Jobs.....	43
6.1 Utilização de um Job para controlo do fluxo – Knime.....	43
6.2 Utilização de um Job para controlo do fluxo – Pentaho Kettle.....	44
7. Visualização e Resultados .....	47
7.1 Dashboard .....	47
7.2 Tabelas da base de dados .....	48
7.2.1 Tabela Dados Playlist:.....	48
7.2.2 Tabela Artistas: .....	49
7.2.3 Tabela Join:.....	49
7.2.4 Tabela Album: .....	50
8. Demonstração.....	51
9. Funcionalidades Não Implementadas.....	53
10. Conclusão e Trabalhos Futuros.....	55
11. Referências Bibliográficas.....	57

## **1. Enquadramento**

O presente trabalho insere-se na disciplina de Integração de Sistemas de Informação (ISI) da Licenciatura em Engenharia de Sistemas Informáticos, cujo objetivo é consolidar conhecimentos sobre processos de ETL (Extract, Transform, Load) e integração de dados provenientes de múltiplas fontes. A motivação para o tema escolhido relaciona-se com a análise de tendências musicais, explorando dados de playlists, álbuns e artistas para entender a popularidade e relações entre conteúdos na plataforma Spotify. O trabalho foi desenvolvido em grupo, nomeadamente pelos alunos Ricardo Fernandes, aluno número 27961, e Pedro Ribeiro, aluno número 27960, permitindo o aprofundamento prático das metodologias ETL e do uso de APIs. Para a execução do projeto, foram utilizadas tecnologias como Pentaho Kettle/PDI, Knime, API do Spotify, manipulação de ficheiros CSV e XML e integração com base de dados SQL Server, proporcionando um pipeline ETL completo e estruturado. O código e os recursos desenvolvidos encontram-se disponíveis nos repositórios Git: [FernandeeSxS/Spotify-ETL](#) e [Pedroxyz/Spotify-ETL](#).



## **2. Problema / Objetivo**

O problema abordado neste trabalho consiste em extrair dados de playlists, álbuns e artistas a partir da API do Spotify, transformá-los através de processos de limpeza, normalização, desnormalização e agregação, e carregá-los numa base de dados SQL Server e em ficheiros XML para posterior análise e visualização em dashboards. Este cenário permite demonstrar a integração de múltiplas fontes de dados e a aplicação prática de um pipeline ETL completo, refletindo situações reais de tratamento e organização de dados em ambientes corporativos ou de análise musical.

A relevância deste projeto reside no facto de que a informação musical nas plataformas digitais é vasta, dinâmica e heterogénea, tornando essencial a utilização de processos estruturados de ETL para garantir consistência, fiabilidade e acessibilidade dos dados. Além disso, permite explorar metodologias e ferramentas de integração de dados amplamente utilizadas no mercado, preparando para desafios reais de análise e gestão de dados.

Os objetivos principais do projeto são construir um pipeline ETL funcional e automatizado, capaz de extrair, transformar e carregar dados de múltiplas fontes de forma estruturada, garantindo a sua reutilização em análises e dashboards. Os objetivos secundários incluem experimentar a integração com APIs externas, explorar diferentes formatos de dados (CSV, XML), aplicar técnicas de limpeza e normalização, realizar agregações e joins entre diferentes tabelas e consolidar a experiência prática no uso do Pentaho Kettle/PDI e do Knime como ferramentas ETL.





## 3. Estratégia utilizada

### 3.1 Estratégia utilizada na Plataforma KNIME

O projeto desenvolvido tem como objetivo implementar um pipeline ETL (Extract, Transform, Load) sobre dados musicais obtidos a partir da Spotify Developer API, utilizando o KNIME Analytics Platform como ferramenta de orquestração. A abordagem foi concebida para permitir a integração dinâmica de dados de álbuns, playlists e artistas, garantindo que a informação extraída seja normalizada, transformada e posteriormente carregada em múltiplos destinos para análise e reutilização.

#### Extract:

Os dados utilizados neste pipeline têm origem em dois componentes principais:

##### 1 - Spotify Developer API:

Para obter as credenciais necessárias para o uso do projeto, o utilizador deve primeiro criar uma conta na plataforma Spotify for Developers e, de seguida, criar uma aplicação dentro do seu painel pessoal. Ao fazer isto, terá acesso ao *Client ID* e ao *Client Secret*, que são essenciais para gerar o token de autenticação necessário para consumir a API. Inicialmente, os links de playlists e de álbuns presentes nos ficheiros são convertidos para os respetivos endpoints da API.

##### 2 - Ficheiros CSV de entrada:

Os dados utilizados no pipeline têm origem em ficheiros CSV de entrada bem como na API do Spotify. Os ficheiros estão organizados dentro da pasta do projeto, no diretório `data/input`. Esta estrutura inclui três ficheiros principais: `album_link.txt`, `playlist_link.txt` e `credentials.txt`. O primeiro contém os links dos álbuns cujos dados serão extraídos, o segundo guarda os links das playlists a consultar e o terceiro armazena a informação necessária para a autenticação, podendo conter diretamente o token em Base64 ou os dados que permitem a sua geração através de um script auxiliar em Python.

#### Transform:

Após a extração, os dados em formato JSON passam por um conjunto de transformações no KNIME. Estas transformações incluem:

- Conversão e limpeza de campos,
- Normalização de estruturas JSON em tabelas,
- Separação de arrays em múltiplas linhas,

- Aplicação de expressões regulares e manipulações textuais,
- Conversões de unidades e filtragem de atributos irrelevantes,
- Combinação de dados de diferentes origens (por exemplo, músicas e artistas).

Estas operações têm como objetivo uniformizar e estruturar a informação, preparando-a para a fase de carregamento.

### **Load:**

Na fase final, os dados transformados são exportados e disponibilizados em diferentes formatos e destinos:

- **Ficheiros CSV e JSON** todas as tabelas finais são exportadas para ficheiros em formato csv. Os dados intermédios gerados ao longo do pipeline são armazenados em ficheiros de texto no formato JSON.
- **Ficheiros XML** Todas as tabelas, exceto a tabela resultante do join entre músicas e artistas, são exportadas para ficheiros XML, garantindo compatibilidade e portabilidade para outros sistemas ou processos que consumam estes formatos.
- **Base de dados:** Todas as tabelas transformadas são carregadas para uma base de dados SQL Server, permitindo armazenamento centralizado, execução de consultas SQL e reutilização em análises futuras.
- **Dashboard,** A tabela resultante do join entre playlists e artistas é utilizada diretamente em dashboards, permitindo a visualização de relações entre os dados, como a popularidade de artistas por álbum ou as interações entre artistas e géneros musicais.

### **Ferramentas e Bibliotecas Utilizadas**

- **KNIME Analytics Platform**: plataforma central para orquestração do ETL
- **Spotify Developer API**: fonte principal de dados
- Python (script auxiliar para geração do token de autenticação)
- Principais nós KNIME utilizados: CSV Reader, GET Request, JSON Path, Ungroup, String Manipulation, Math Formula, Rule Engine, Joiner, DB Writer, entre outros.

### **3.2 Estratégia utilizada na plataforma Pentaho Kettle**

O projeto desenvolvido tem como objetivo implementar um pipeline ETL (Extract, Transform, Load) sobre dados musicais obtidos a partir da Spotify Developer API, utilizando o Pentaho Data Integration (Kettle) como ferramenta de orquestração.

A abordagem foi concebida para permitir a integração dinâmica de dados de playlists, álbuns e artistas, garantindo que a informação extraída seja limpa, normalizada e transformada,

antes de ser carregada em bases de dados e ficheiros CSV, permitindo análises consistentes e reutilização em dashboards ou relatórios.

#### **Extract:**

Os dados são extraídos das seguintes fontes, utilizando passos específicos do Pentaho Kettle/PDI:

##### **1 - Spotify API (Playlist e Album):**

Na transformação Get Request Playlist, é feita uma requisição à API do Spotify para obter informações sobre as músicas e o nome dos artistas correspondentes contidos numa playlist específica. Os nomes dos artistas obtidos através da requisição são exportado para CSV. Na transformação Get Album, é novamente realizada uma requisição à API do Spotify através do para obter informações detalhadas sobre um determinado album.

##### **2 – Ficheiro CSV (nomes\_artistas.csv)**

Na transformação Get Request Artistas, os CSV gerado anteriormente (nomes\_artistas.csv) é usado como input. Cada artista do CSV é passado como parâmetro para o passo REST Client, que realiza chamadas à API do Spotify para obter informações detalhadas (como popularidade, género, número de seguidores).

#### **Transform:**

Após a extração, os dados passam por um conjunto de transformações no Pentaho Kettle/PDI. Estas transformações incluem:

- Limpeza e substituição de valores inconsistentes ou nulos;
- Normalização e padronização de campos textuais;
- Filtragem de registos irrelevantes ou duplicados;
- Separação de campos compostos em múltiplas linhas para granularidade da informação;
- Reorganização e classificação dos dados por critérios como artista, álbum ou popularidade;
- Combinação e agregação de dados para melhorar a lógica dos dados;
- Renomeação e reestruturação de campos para compatibilidade com o destino final.

Estas operações têm como objetivo uniformizar, estruturar e preparar os dados para a fase de carregamento.

**Load:**

Após a fase de transformação, os dados são carregados para diferentes destinos, de acordo com o objetivo de cada conjunto de informação:

- **Base de dados:** Todas as tabelas transformadas são carregadas para uma base de dados SQL server, permitindo armazenamento centralizado, consultas SQL e reutilização em análises futuras.
- **Ficheiros XML:** Todas as tabelas, exceto a tabela resultante do join, são exportadas para ficheiros XML, garantindo compatibilidade e portabilidade para outros sistemas ou processos que consumam estes formatos.
- **Dashboard:** A tabela que resulta do join entre playlists e artistas é utilizada diretamente em dashboards, permitindo a visualização de relações entre os dados, como popularidade de artistas por álbum ou interações entre músicas e playlists.

Este carregamento assegura que os dados estão disponíveis para análise, visualização e integração com outros sistemas, completando o ciclo ETL.

**Ferramentas e Bibliotecas Utilizadas**

- **Pentaho Kettle** - plataforma central para orquestração do ETL
- **Spotify Developer API** - fonte principal de dados
- Python (script auxiliar para geração do token de autenticação)
- **Principais passo Pentaho utilizados:** CSV File Input, REST Client, JSON Input, Modified JavaScript value, Select Values, Sort Rows, Merge Join, String Operations, entre outros.

## 4. Abordagem geral utilizada

Para representar de forma clara a estratégia adotada em ambos os projetos, foi criado um diagrama de fluxo utilizando a ferramenta Mermaid no site [mermaid.live](https://mermaid.live). Este diagrama ilustra visualmente a sequência de operações do pipeline ETL, desde a extração de dados, passando pelas transformações e normalizações realizadas, até ao carregamento final.

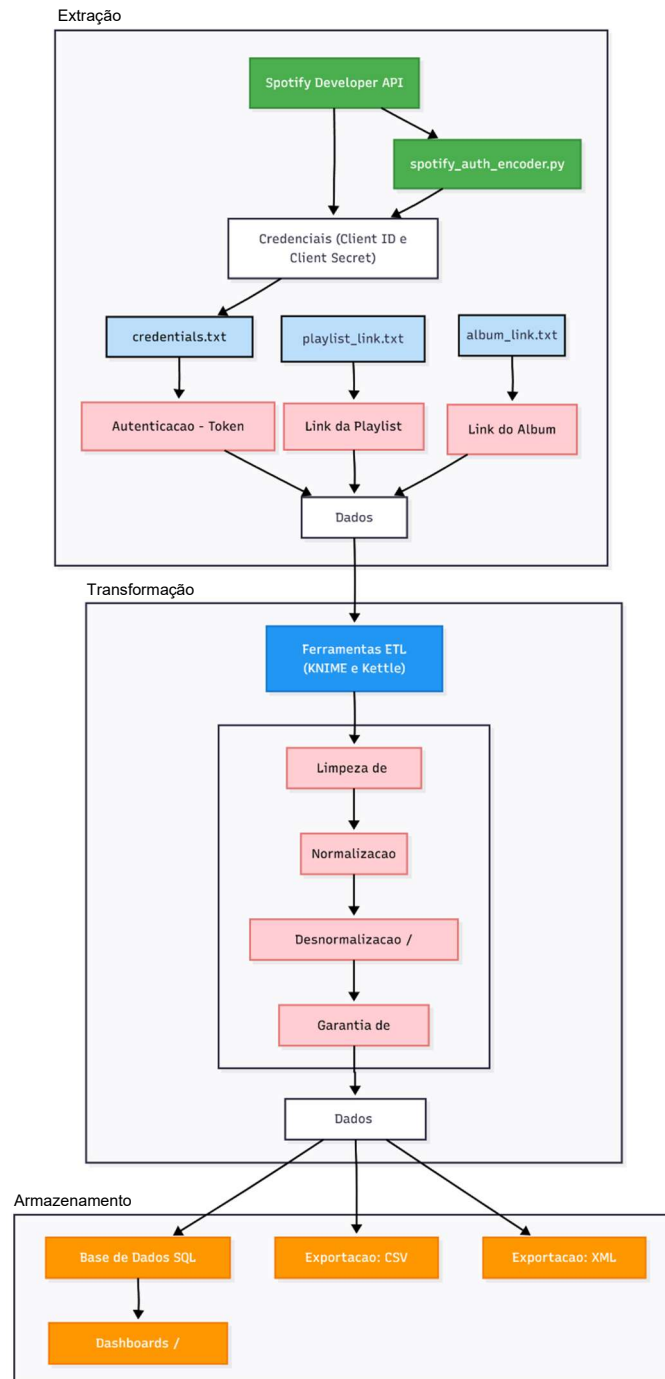


Figura 1 - Arquitetura Geral do Sistema



## 5. Transformações de Dados

### 5.1 Transformações de Dados na plataforma Knime

#### 5.1.1. Componente de obter endpoint para álbum

O workflow abaixo detalha a preparação do endpoint da API utilizando o link de um álbum inserido no ficheiro `album_link.txt`. O workflow inicia com o nó Component Input, seguido do CSV Reader (Read `album_link.txt`) que carrega o link como tabela. O nó String Manipulation (API Endpoint Construction) extrai o ID do álbum a partir do link do mesmo e constrói o endpoint final da API. De seguida, o Table Row to Variable (FlowVariable Creation) transforma o endpoint numa variável de fluxo, enviada pelo Component Output para o workflow principal, onde será usada na chamada HTTP à API. Este design modular garante uma gestão clara e isolada dos parâmetros de entrada.

Transformação / Nó KNIME	Objetivo / Função
<b>CSV Reader</b>	Carrega link de álbum do ficheiro <code>album_link.txt</code> e guarda na coluna <code>linkalbum</code>
<b>String Manipulation</b>	<p>Limpeza de strings, construção do endpoint da API, manipulação de texto.</p> <p>Script utilizada:</p> <pre>replace(\$linkalbum\$, "https://open.spotify.com/intl-pt/album/", "https://api.spotify.com/v1/albums/")</pre>
<b>Table Row to Variable</b>	Converter valor da <code>linkalbum</code> em variáveis de fluxo (Flow Variables)

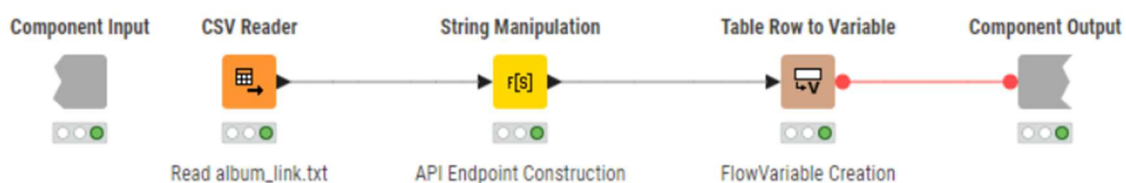


Figura 2 - Componente de obter endpoint para álbum

### 5.1.2. Componente de obter endpoint para playlist

Este módulo de *workflow* tem como objetivo preparar o *endpoint* da API para uma *playlist*. O processo inicia com o nó CSV Reader que lê o *link* da *playlist* (playlist\_link.txt). O nó String Manipulation (API Endpoint Construction) usa este *link* para extrair o ID da *playlist* e construir a URL completa (o *endpoint*) da API. Por fim, o Table Row to Variable (Flow Variable Creation) transforma este *endpoint* numa variável de fluxo, que é enviada pelo Component Output ao *workflow* principal para ser utilizada na chamada à API. Em suma, o componente realiza a extração do ID e a construção do *endpoint* da *playlist* de forma modular e eficiente.

Transformação / Nó KNIME	Objetivo / Função
CSV Reader	Carrega link de playlist do ficheiro playlist_link.txt e guarda na coluna linkplaylist
String Manipulation	Limpeza de strings, construção de endpoints da API, manipulação de texto.  Script utilizada:  replace(\$linkplaylist\$, "https://open.spotify.com/playlist/", "https://api.spotify.com/v1/playlists/")
Table Row to Variable	Converter valor da linkplaylist em variáveis de fluxo (Flow Variables)

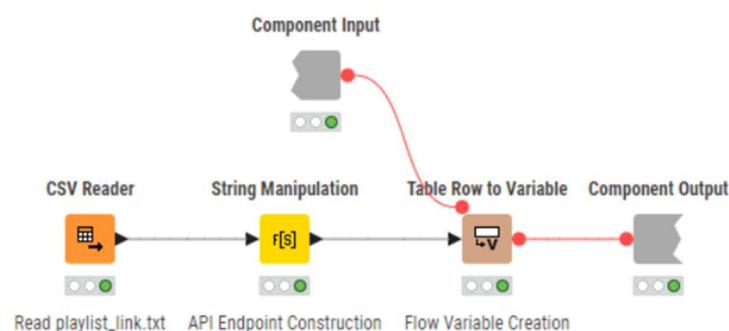


Figura 3 - Componente de obter endpoint para playlist

### 5.1.3. Componente de obter token [Bearer <token>]

Este *workflow* é responsável pela geração de um *Access Token* para a API. O processo começa com o CSV Reader lendo as credenciais (Client Id e Client Secret em BASE64) de



autenticação (credentials.txt). Em seguida, o String Manipulation (Add Basic) adiciona o prefixo "Basic ", e a coluna é convertida numa variável de fluxo pelo Table Row to Variable. Esta string é usada pelo nó POST Request (Generate Token) para solicitar o *Access Token* à API. A resposta JSON é então processada pelo JSON Path (Extract Token), isolando o valor do *token* de acesso. Por fim, o String Manipulation (Add Bearer) adiciona o prefixo "Bearer " ao *token* extraído, e o Table Row to Variable final o converte numa variável de fluxo. O Component Output disponibiliza essa variável (Bearer Token) para o *workflow* principal, permitindo chamadas autorizadas subsequentes à API.

Transformação / Nó KNIME	Objetivo / Função
CSV Reader	Carrega as credenciais de autenticação (Client Id e Client Secret em BASE64) do arquivo credentials.txt.
String Manipulation	Adiciona o prefixo "Basic " à <i>string</i> codificada das credenciais.  string(join("Basic " + \$BASE64\$))
Table Row to Variable	Transforma a <i>string</i> "Basic [token BASE64]" em uma variável de fluxo.
POST Request	Utiliza a variável de fluxo " Basic [token BASE64]" para fazer a chamada POST ao <i>endpoint</i> da API e solicitar o <i>Access Token</i> .
JSON Path	Analisa a resposta JSON da requisição POST e isola o valor puro do <i>Access Token</i> .
String Manipulation	Adiciona o prefixo "Bearer " ao <i>Access Token</i> extraído, formatando-o para uso em chamadas autorizadas.
Table Row to Variable	Transforma o <i>token</i> formatado ("Bearer [token]") em uma variável de fluxo (flow_accesstoken).

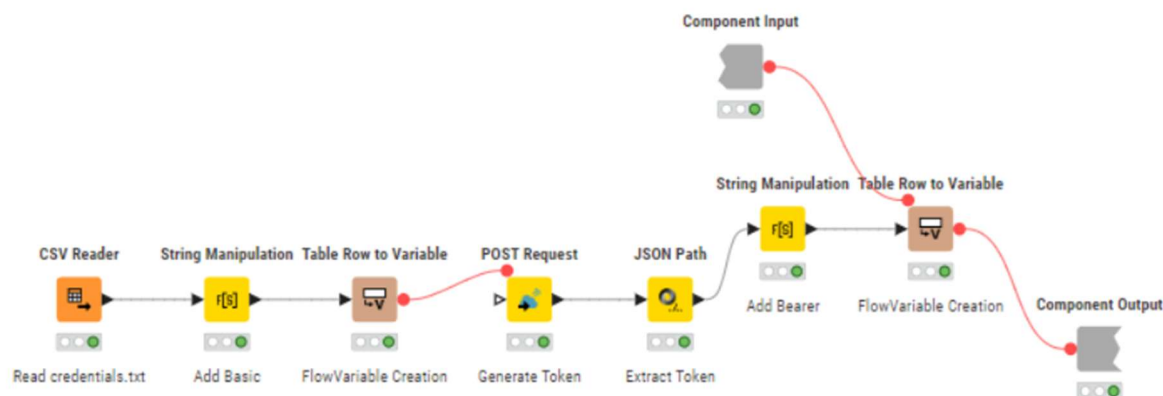


Figura 4 - Componente de obter token [Bearer <token>]

### 5.1.3. Componente de obtenção de dados acerca de álbum

#### 5.1.3.1. Primeira Parte (Workflow principal – Spotify-ETL)

Este workflow é responsável por requisitar e guardar os dados brutos de um álbum da API. O processo inicia-se com o Component Input que recebe o endpoint da API do álbum e o Bearer Token de autorização. Em seguida, o nó GET Request (Request Album Endpoint) executa a chamada autorizada à API, utilizando o endpoint e o token, e retorna a resposta com os dados. O nó CSV Writer (Write Intermediate File) recebe esta resposta bruta (em JSON) e guarda em um ficheiro intermediário para registo e backup.

Transformação / Nó KNIME	Objetivo / Função
<b>GET Request</b>	Executa uma requisição HTTP GET autorizada, usando o <i>endpoint</i> e o <i>Bearer Token</i> , para obter os dados do álbum/playlist da API.
<b>CSV Writer</b>	Guarda a resposta bruta (dados JSON) obtida da API em um ficheiro intermediário <code>json_album_intermediate.txt</code>



Figura 5 - Componente de obtenção de dados acerca de álbum (Parte 1)

### 5.1.3.1. Segunda Parte (Workflow secundário – Component \_Album)

Este *workflow* localizado em outro separador no Knime é responsável pelo tratamento, extração e transformação dos dados brutos em formato JSON obtidos da API. O processo começa com o CSV Reader a ler o ficheiro intermédio com a resposta JSON. De seguida, o nó String to JSON (String into JSON) converte o conteúdo de texto para um objeto JSON estruturado, permitindo a sua manipulação. O nó JSON Path (Extract JSON fields) extrai os campos de dados relevantes (como os detalhes das músicas) da estrutura JSON. Estes dados são seguidos pelo nó Ungroup (Ungroup Array), que expande a lista (o *array*) de músicas, criando uma linha na tabela para cada uma delas (normalização). Finalmente, o nó Math Formula (Milliseconds to Minutes) aplica uma fórmula para converter a duração das músicas de milissegundos para minutos, tornando os dados prontos para a análise subsequente.

Transformação / Nó KNIME	Objetivo / Função
CSV Reader	Carrega a resposta JSON bruta da API, que foi previamente guardada num ficheiro intermédio json_album_intermediate.txt.
String to JSON	Converte o conteúdo do ficheiro (que é uma <i>string</i> de texto) para um objeto de dados JSON estruturado.
JSON Path	Navega na estrutura JSON e <b>extrai campos de dados específicos</b> (p. ex., título da música, artista, duração).
Ungroup	Expandes o <i>array</i> de itens (como as músicas do álbum/playlist), criando uma linha individual na tabela para cada item.
Math Formula	Aplica uma fórmula matemática para converter a duração da música de milissegundos para minutos, para facilitar a análise.  round(\$Duracao\$ / 60000, 2)

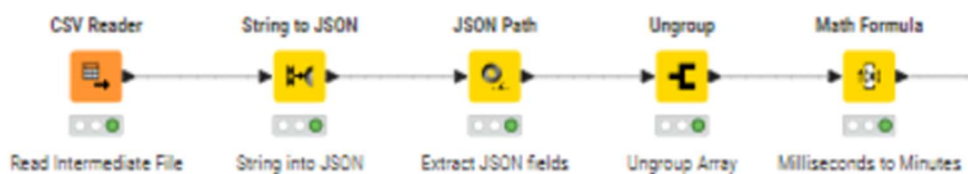


Figura 6 - Componente de obtenção de dados acerca de album (Parte 2)

O segmento final deste *workflow*, após a etapa anterior de transformação, foi utilizado o nó Column Filter (Remove Columns) que limpa a tabela final, removendo colunas desnecessárias. A partir deste ponto, o *workflow* divide-se em duas rotas de exportação. A rota inferior utiliza o CSV Writer (Write album\_output.txt) para gravar os dados limpos num ficheiro de texto, que é a saída primária. A rota superior, para formatos alternativos, inicialmente utiliza o Table to JSON para converter a tabela para JSON e, de seguida, o JSON to XML para a transformar em XML. Por último, o CSV Writer (Write album\_output\_xml.txt) guarda este *output* em XML num ficheiro, garantindo que os dados finais estão disponíveis nos formatos CSV/TXT e XML.

Transformação / Nó KNIME	Objetivo / Função
Column Filter	Recebe a tabela processada e remove colunas que não são necessárias(Status e content-type) para o <i>output</i> final, simplificando o conjunto de dados.
CSV Writer (Write album_output.txt)	Grava a tabela de dados limpa e finalizada num ficheiro de texto ou CSV para uso primário.
Table to JSON	Converte a tabela filtrada numa estrutura de dados JSON como passo intermédio para a conversão para XML.
JSON to XML	Transforma o objeto JSON numa estrutura de dados XML, satisfazendo requisitos de <i>output</i> alternativos.
CSV Writer (Write album_output_xml.txt)	Guarda a <i>string</i> XML resultante num ficheiro de texto, finalizando a exportação no formato XML.

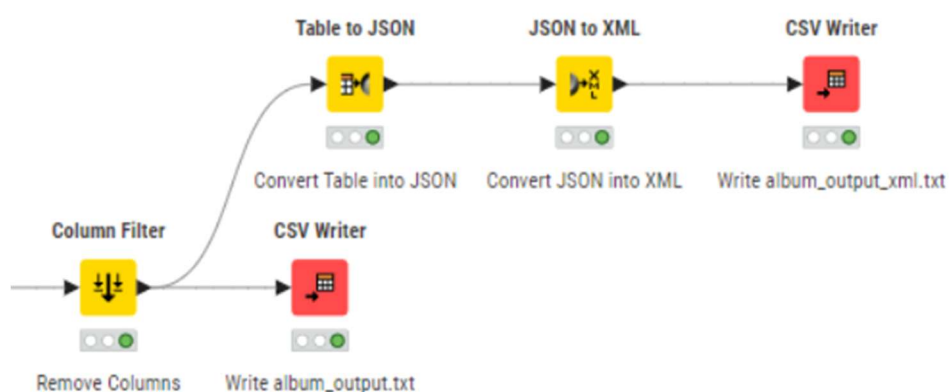


Figura 7 - Componente de obtenção de dados acerca da playlist (Parte 3)

O workflow completo tem a seguinte aparência:

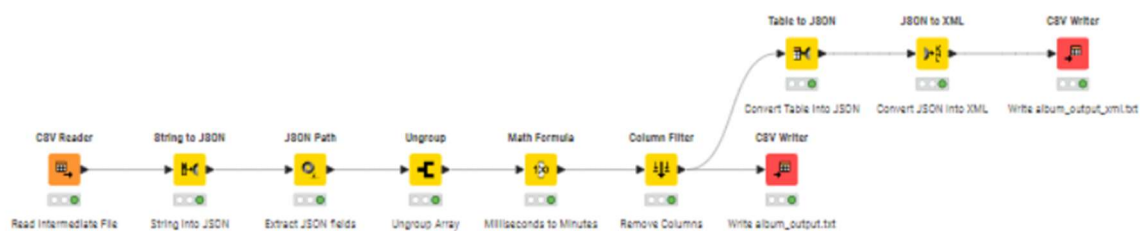


Figura 8 - Workflow completo da obtenção de dados acerca da playlist

### 5.1.3. Componente de obtenção de dados acerca da playlist

Este segmento de *workflow* é dedicado a solicitar dados da API e a normalizar campos aninhados contidos na resposta JSON. O fluxo começa com o Component Input a fornecer o *endpoint* e o *Bearer Token* necessários para a autenticação. Seguidamente, o nó GET Request (Request Endpoint) executa a chamada HTTP autorizada à API, obtendo os dados em formato JSON. O processamento do JSON é feito em duas fases de normalização: primeiro, o JSON Path (Extract JSON Fields) extrai os campos de nível superior e o Ungroup (Ungroup Array) expande o *array* principal de itens numa tabela. Depois, para lidar com subestruturas complexas, o segundo nó JSON Path (Extract JSON name Field) isola um campo específico aninhado (o campo "name", neste caso), e o último Ungroup (Ungroup name Field) expande esse *array* aninhado, de modo a garantir que todos os dados aninhados sejam normalizados em linhas separadas da tabela para análise posterior.

Transformação / Nó KNIME	Objetivo / Função
GET Request	Executa a requisição HTTP GET autorizada ao <i>endpoint</i> para obter a resposta JSON.
JSON Path	Extrai os campos de dados do primeiro nível da resposta JSON.
Ungroup	Expandi o <i>array</i> principal de itens extraído, criando uma linha para cada item de nível superior.
JSON Path	Extrai um campo aninhado específico (neste caso, "name") que é, ele próprio, um <i>array</i> dentro dos itens já extraídos.
Ungroup	Expandi o <i>array</i> do campo aninhado ("name"), finalizando a normalização dos dados aninhados para análise tabular.

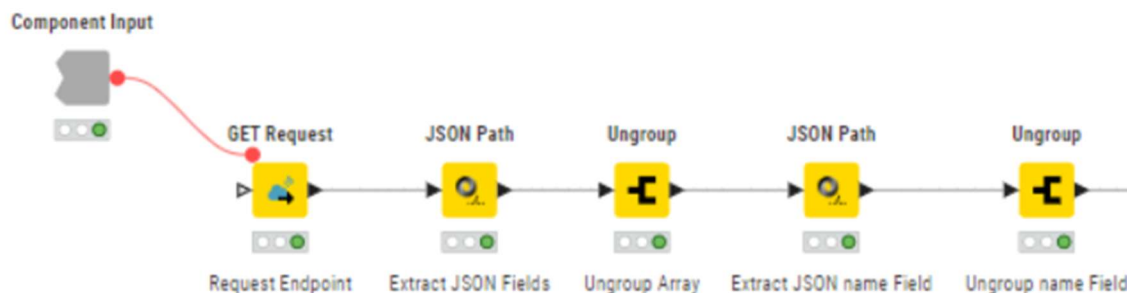


Figura 9 - Componente de obtenção de dados acerca da playlist (Parte 1)

A continuação deste workflow é a etapa final de agregação, limpeza e exportação dos dados de uma playlist. O fluxo começa com o nó String Manipulation (Remove With Regex) a limpar strings através da remoção de caracteres indesejados, enviando os dados da tabela para o Component output. Seguidamente, o nó crucial GroupBy (Group by Cantores) agrega as informações por cantor/artista, consolidando os dados (p. ex., somando durações ou contando músicas). Após a agregação, a tabela é limpa: o Column Filter (Remove Columns) elimina colunas desnecessárias, e o Column Renamer (Rename Columns) padroniza os nomes das colunas.

A partir daqui o output divide-se em três vias: uma para o Component Output (que envia a tabela limpa para o resto do workflow) e duas para exportação. A primeira exportação é feita pelo CSV Writer (Write playlist\_output.txt) para criar o ficheiro de saída final em CSV/TXT. A segunda via, para o formato alternativo XML, converte a tabela para JSON com o Table to JSON, transforma-o em XML com o JSON to XML, e, por fim, guarda o resultado XML num ficheiro através do CSV Writer (Write playlist\_output\_xml.txt).

Transformação / Nó KNIME	Objetivo / Função
<b>String Manipulation</b>	Remove caracteres indesejados ou ruído nas colunas de texto (strings) usando expressões regulares.  Script utilizada:  <code>regexReplace(\$Album\$, "[\\"]", "")</code>
<b>GroupBy</b>	Consolida as linhas da tabela agrupando as informações por cantor/artista e calcula métricas agregadas.

<b>Column Filter</b>	Elimina colunas que se tornaram desnecessárias ou redundantes após o processo de agregação.
<b>Column Renamer</b>	Renomeia as colunas da tabela final, tornando os nomes claros e consistentes para o <i>output</i> .
<b>CSV Writer (Write playlist_output.txt)</b>	Cria o ficheiro de saída final dos dados processados no formato de texto/CSV.
<b>Table to JSON</b>	Converte a tabela finalizada para uma estrutura JSON, como passo intermédio para a exportação XML.
<b>JSON to XML</b>	Transforma os dados em JSON para o formato XML (eXtensible Markup Language).
<b>CSV Writer</b>	Grava o resultado em XML num ficheiro de texto, finalizando a saída no formato alternativo.

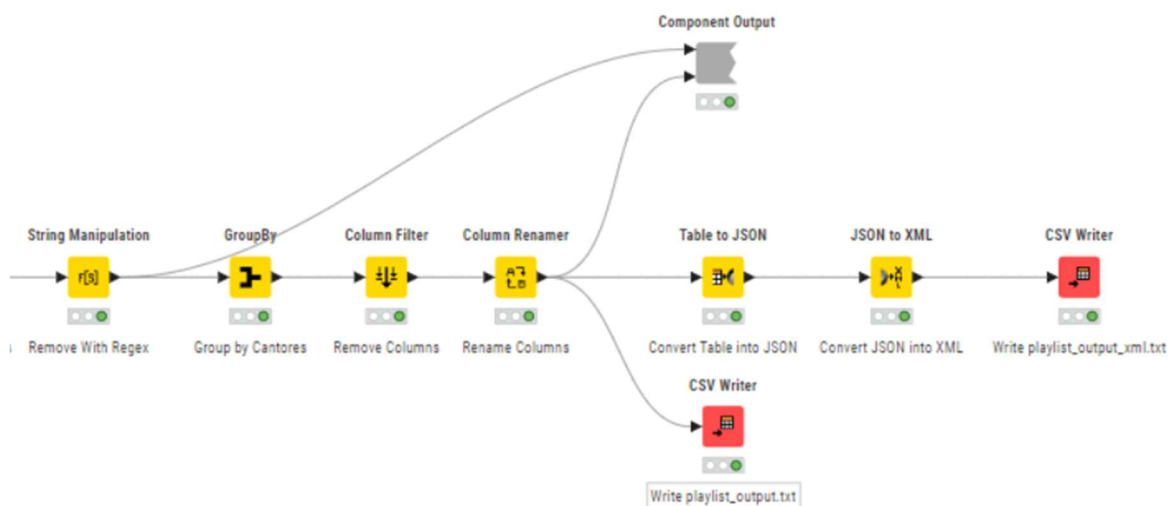


Figura 10 - Componente de obtenção de dados acerca da playlist (Parte 2)

O workflow completo tem a seguinte aparência:

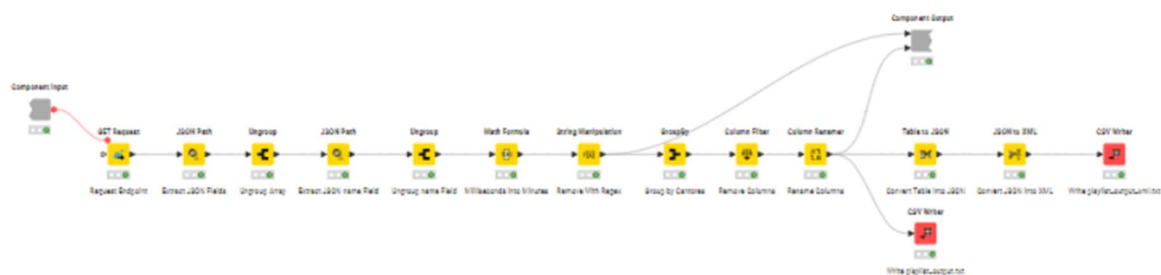


Figura 11 - Workflow da obtenção de dados acerca da playlist

#### 5.1.4. Componente de obtenção de dados acerca de artistas

Este componente de *workflow* está concebido para processar múltiplos itens de uma lista, iterando sobre cada um para fazer uma chamada à API. O fluxo começa com o Component Input a fornecer a tabela de dados das playlists que contém os itens a processar. O nó Table Row to Variable Loop Start (Initiate Loop) inicia o ciclo, convertendo cada linha dessa tabela numa variável de fluxo individual a cada iteração. Seguidamente, o String Manipulation (Variable) (Endpoint Construction) utiliza esta variável de fluxo para construir o *endpoint* (URL) do artista específico para o item em curso. De seguida, o nó GET Request (Request Endpoint) executa a requisição à API para obter os dados desse artista. Por fim, o Loop End (Ending Loop) encerra o ciclo, agregando os resultados de todas as chamadas GET Request numa única tabela de saída, permitindo o processamento sequencial e eficiente de uma lista de dados.

Transformação / Nó KNIME	Objetivo / Função
<b>Table Row to Variable Loop Start</b>	Inicia o ciclo, convertendo sequencialmente cada linha de entrada numa variável de fluxo para uso na iteração.
<b>String Manipulation (Variable)</b>	Utiliza a variável de fluxo (o ID do item atual) para construir o <i>endpoint</i> (URL) completo e específico para a requisição (um endpoint para cada artista).  <code>strip(join("https://api.spotify.com/v1/search?q=", replace(\$\${SCantores}\$\$, " ", "%20"), "&amp;type=artist&amp;limit=1"))</code>
<b>GET Request</b>	Executa a requisição HTTP GET ao <i>endpoint</i> que foi dinamicamente construído, obtendo os dados para o artista em curso.
<b>Loop End</b>	Encerra o ciclo, compilando e agregando todas as respostas da API (os resultados de cada iteração) numa única tabela de saída.



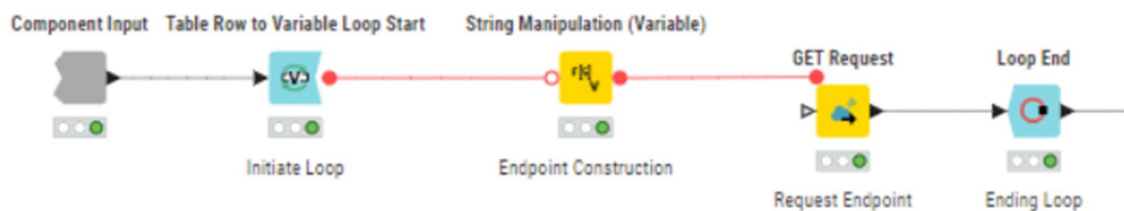


Figura 12 - Componente de obtenção de dados acerca de artistas (Parte 1)

O processo continua com o nó JSON Path (Extract JSON fields) a isolar e extrair os campos de dados da estrutura JSON, que tipicamente se apresenta como uma lista (*array*). De seguida, o Rule Engine (JSON to Array) aplica regras, que podem incluir a conversão ou verificação desse campo para o formato de lista. O nó String Manipulation (Array to String) converte então essa lista de valores numa única *string* de texto, utilizando um separador (delimitador) para concatenar os itens. Por fim, o Component Output disponibiliza a tabela resultante, com o campo formatado como uma *string* pronta para ser gravada ou processada nos passos seguintes.

Transformação / Nó KNIME	Objetivo / Função
<b>JSON Path</b>	Isola e extrai campos específicos da resposta JSON
<b>Rule Engine</b>	<p>Aplica regras condicionais e é usado para garantir/forçar a conversão do campo extraído para o tipo de dado <i>array</i> (lista).</p> <p>TRUE =&gt; \$Genero Cantor\$</p>
<b>String Manipulation</b>	<p>Converte a lista (<i>array</i>) de valores numa única <i>string</i> de texto, usando um delimitador (como uma vírgula) para separação.</p> <p>replace(replace(replace(\$Genero Cantor\$, "[", ""), "]", ""), "\", "")</p>

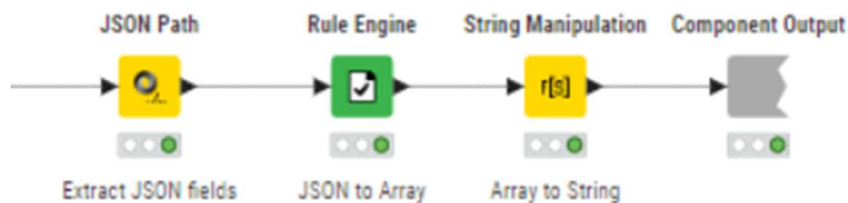


Figura 13 - Componente de obtenção de dados acerca de artistas (Parte 2)

O workflow completo tem a seguinte aparência:



Figura 14 - Workflow completo da obtenção de dados acerca de artistas

### 5.1.5. Componente de obtenção de dados acerca da playlist/artista

Este componente de *workflow* é o passo final na consolidação e exportação dos dados, relacionando a *playlist* com os seus artistas. O processo começa com o Component Input a receber as tabelas de dados da *playlist* e dos artistas processadas anteriormente. O nó Joiner (Join Playlist Artist) é o passo crucial, pois combina estas duas tabelas usando um ID comum (ID da *música*), criando uma tabela única. De seguida, o Column Filter (Remove Column) limpa a tabela, removendo colunas insignificantes, e o Duplicate Row Filter (Remove Duplicate Rows) assegura que não haja linhas duplicadas. A tabela final limpa é então dividida em três saídas: o Component Output envia a tabela para o *workflow* seguinte, um CSV Writer (Write playlist\_artists\_output.txt) exporta os dados para um ficheiro CSV/TXT, um XML Writer (Write playlist\_artists\_output.xml) envia para XML, os dados são convertidos sucessivamente por Table to JSON e JSON to XML, sendo finalmente gravados num ficheiro XML pelo outro CSV Writer (playlist\_artists\_output\_xml.txt).

Transformação / Nó KNIME	Objetivo / Função
<b>Joiner</b>	Combina as duas tabelas de entrada usando um ID comum (e.g., ID da música), relacionando as <i>playlists</i> com os seus artistas.
<b>Column Filter</b>	Remove colunas insignificantes ou redundantes após a operação de junção.(Status content-type)

<b>Duplicate Row Filter</b>	Identifica e elimina linhas duplicadas para garantir a unicidade dos dados finais.
<b>CSV Writer (Write playlist_artists_output.txt)</b>	Cria o ficheiro de saída principal dos dados processados no formato CSV/TXT.
<b>Table to JSON</b>	Converte a tabela final para uma estrutura JSON, como passo intermédio para a exportação XML.
<b>JSON to XML</b>	Transforma a estrutura JSON na estrutura de dados XML.
<b>CSV Writer (playlist_artists_output_xml.txt)</b>	Grava o resultado em XML num ficheiro de texto, finalizando a saída no formato alternativo.

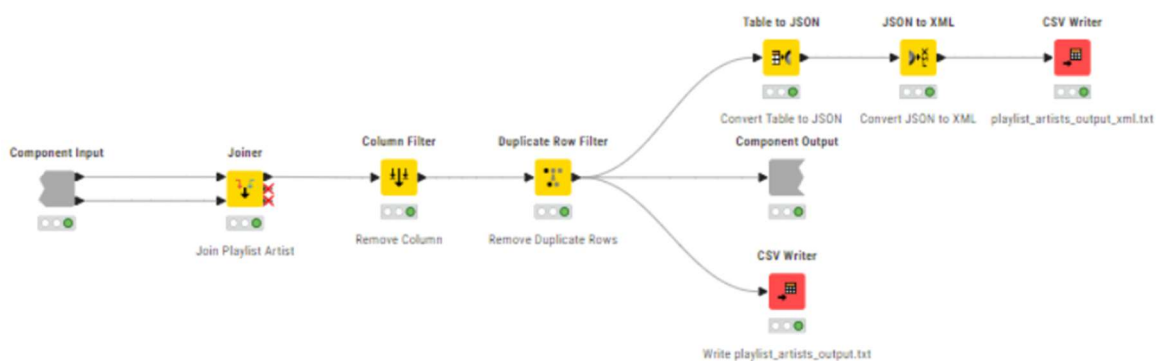


Figura 15 - Componente de obtenção de dados acerca da playlist/artista

#### 5.1.5. Componente de obtenção de dados acerca da exportação para base de dados

Este componente de workflow é o módulo de Escrita para a Base de Dados, projetado para persistir a tabela de dados final num servidor. O processo começa com o nó DB Connector (Connect Database) a estabelecer a ligação ao servidor (SQL Server). O nó DB Table Creator (Create Database Table) usa essa conexão para criar a tabela de destino na base de dados, espelhando a estrutura dos dados de entrada. Finalmente, o nó DB Writer (Write Database) recebe a tabela de dados final (do Component Input) e a grava efetivamente na base de dados. O Component Output encerra o módulo.

Nota sobre a Execução: Como não foi possível resolver o problema de conexão com o SQL Server, apesar de o workflow demonstrar a intenção correta de persistência de dados, será explicitado no ficheiro README os passos e configurações exatas para que qualquer utilizador possa replicar a conexão e a escrita na base de dados com sucesso.

Transformação / Nó KNIME	Objetivo / Função
DB Connector	Estabelece a ligação (ligação) ao servidor de base de dados (e.g., SQL Server), fornecendo as credenciais e o URL.
DB Table Creator	Cria a tabela de destino na base de dados com a estrutura correta (colunas e tipos de dados) para receber os dados.
DB Writer	Executa a inserção (gravação) dos dados da tabela de entrada para a tabela na base de dados.

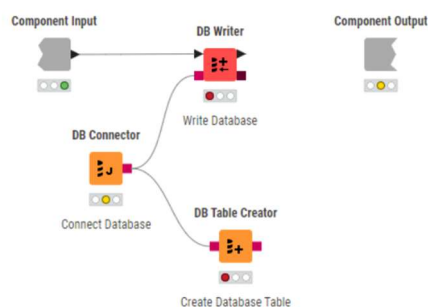


Figura 16 - Componente de obtenção de dados acerca da exportação para base de dados

### 5.1.6. Organização de todos os componentes

A organização de todos os componentes do workflow principal é a seguinte:

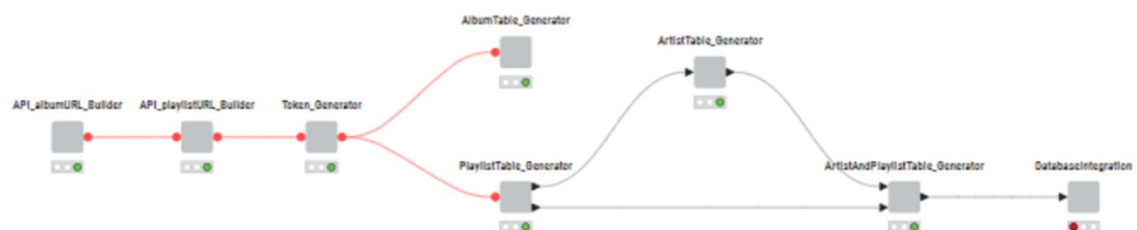


Figura 17 - Organização de todos os componentes

## 5.2 Transformações de Dados na plataforma Pentaho Kettle

### 5.2.1 Transformação Obter Token:

O fluxo começa por ler as credenciais do Spotify a partir de um ficheiro de texto, garantindo que a informação necessária para autenticação está disponível. De seguida, um passo de JavaScript constrói os campos essenciais para a requisição à API: o cabeçalho de autorização, o corpo da mensagem e o tipo de conteúdo. Depois, há uma seleção dos campos relevantes para garantir que apenas os dados necessários seguem adiante. Com estas informações, é feito o pedido HTTP POST à API de tokens do Spotify, solicitando um token de acesso. A resposta, em formato JSON, é lida e processada para extrair o `access_token`, o tipo de token e o tempo de expiração. Por fim, o token é armazenado numa variável global, disponível para todos os passos seguintes do pipeline, permitindo que outras operações que necessitem de autenticação utilizem este token de forma automática e segura.

Nome do Passo	Tipo de Passo	Função
Ler Credenciais	CSV File Input	Lê do ficheiro as credenciais codificadas em Base64 necessárias para autenticar na API do Spotify.
Construção Campos Body e Header	Modified Javascript Value	Cria os campos necessários (authorization, body e contentType) para enviar o pedido de autenticação à API.
Seleção Campos	Select Values	Seleciona apenas os campos relevantes que serão usados no pedido REST.
Solicitar Token Spotify	REST Client	Envia o pedido POST para o endpoint do Spotify e obtém a resposta com o token de acesso.
Extrair Token Spotify	Json Input	Extrai o valor do <code>access_token</code> (e outros dados) do JSON de resposta.
Armazenar Token Spotify	Set Variables	Guarda o token extraído como variável global para ser usado noutros processos ETL.



Figura 18 - Transformação Obter Token

### 5.2.2 Transformação Get Request Playlist

O processo inicia com a leitura do URL da playlist, de modo a obter o link necessário para aceder aos dados da playlist no Spotify. Paralelamente, é recuperado o token previamente gerado, Spotify ETL

que será usado para autenticação. Com este token, o passo Construção Autenticação monta o cabeçalho Authorization com o formato correto de Bearer token, enquanto o passo Construção Endpoint extrai o ID da playlist do URL e constrói o endpoint final da API do Spotify. Seguindo o fluxo, ocorre um Stream Lookup para associar o token de autenticação ao endpoint, garantindo que a requisição tenha credenciais válidas.

Nome do Passo	Tipo de Passo	Função
Ler URL da Playlist	CSV File Input	Lê o link da playlist guardado no ficheiro <code>playlist_link.txt</code> .
Construção Endpoint	Modified Javascript Value	Extrai o ID da playlist e constrói o URL final para a API do Spotify.
Obter Token Spotify	Get Variables	Vai buscar o token de autenticação guardado como variável global.
Construção Autenticação	Modified Javascript Value	Cria o header <code>Authorization: Bearer &lt;token&gt;</code> .
Seleção Campo Autenticação	Select Values	Seleciona apenas o header e a chave de junção (JOIN_KEY).
Stream Lookup	Stream Lookup	Junta o endpoint da playlist com o header de autenticação.

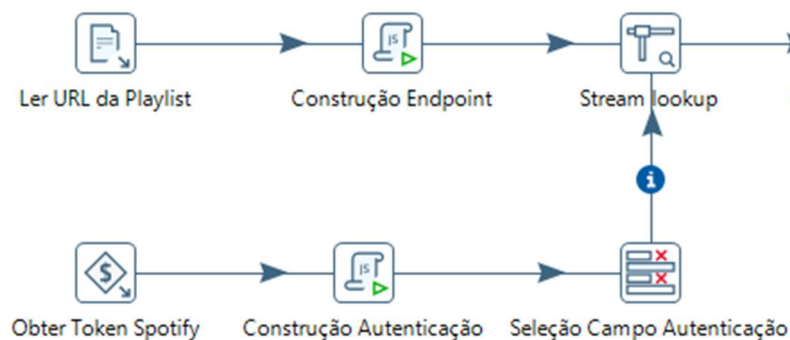


Figura 19 - Transformação Get Request Playlist (Parte 1)

O passo Requisição Playlist então realiza a chamada GET à API do Spotify e retorna os dados da playlist em formato JSON. Estes dados são processados através de um Json Input, que extrai informações detalhadas de cada música, incluindo nome da música, duração, popularidade, artistas, ID da música e nome do álbum. Em paralelo, o passo Extrair Nomes Artistas coleta especificamente os nomes dos artistas envolvidos em cada música, pois uma música pode conter

Spotify ETL 30

mais do que um artista. Ainda em paralelo com os anteriores, o nome dos artistas envolvidos é extraído para um ficheiro csv para futura manipulação

Nome do Passo	Tipo de Passo	Função
Requisição Playlist	REST Client	Faz o pedido REST à API do Spotify e obtém os dados da playlist em formato JSON.
Extrair Dados Playlist	Json Input	Extrai do JSON os campos principais (nome da música, popularidade, duração, artistas, etc.).
Extrair Nomes Artistas	Json Input	Percorre o array de artistas de cada música e extrai os nomes individuais.
Exportar Nomes Artistas	Text file output	Exporta apenas os nomes dos artistas para um ficheiro .csv.

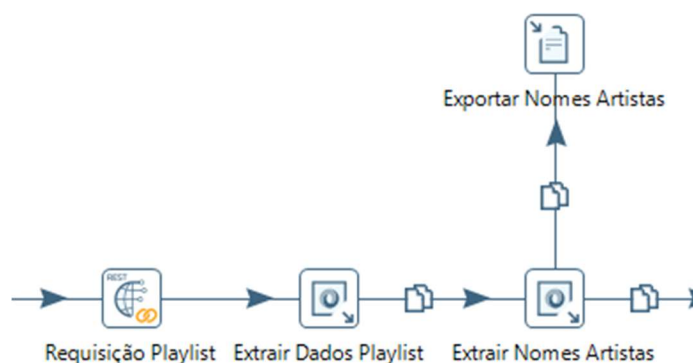


Figura 20 - Transformação Get Request Playlist (Parte 2)

Os nomes dos artistas são depois agrupados através de um Group By, concatenando múltiplos artistas de uma mesma música numa única string. Para limpar os títulos, o passo Retirar Feat remove menções a “feat.” que apareçam entre parênteses através da expressão Regex “([<sup>^</sup>])\*feat.[<sup>^</sup>]\*)([<sup>^</sup>])\*feat.[<sup>^</sup>]\*” . Seguidamente são aplicadas normalizações adicionais, como formatar a duração de milissegundos para minutos e segundos, limpar nomes de músicas e álbuns, e garantir que a popularidade seja um valor numérico consistente. Após a limpeza, o passo Filtrar Nulos remove qualquer registro incompleto, garantindo que apenas músicas com nome e artistas definidos sigam adiante.

Nome do Passo	Tipo de Passo	Função
Agrupar Artistas	Group by	Agrupa os artistas de cada música num só campo, separados por vírgulas.
Retirar Feat	Replace in String	Remove menções "feat." dos nomes das músicas.
Formatar Dados	Modified Javascript Value	Limpa e formata os campos (duração, popularidade, nomes, etc.).
Filtrar Nulos	Filter Rows	Elimina registos sem nome de música ou artista.



Figura 21 - Transformação Get Request Playlist (Parte 3)

Posteriormente, os campos são padronizados para nomes simples e consistentes, preparando os dados para saída. De seguida, as músicas são organizadas alfabeticamente pelo nome da música. Por fim, os dados processados são exportados em dois formatos: criação de um ficheiro XML com todas as informações e exportação dos dados da playlist para a tabela *Tabela\_Dados\_Playlist* numa base de dados local.

Nome do Passo	Tipo de Passo	Função
Renomear Campos	Select Values	Ajusta os nomes dos campos para o formato final.
Ordenar Nome Musica	Sort Rows	Ordena as músicas alfabeticamente.
Exportar Playlist	XML output	Exporta os dados limpos para um ficheiro XML.
Exportar Tabela Playlist	Table Output	Carrega os dados na tabela da base de dados <i>Tabela_Dados_Playlist</i> .





Figura 22 - Transformação Get Request Playlist (Parte 4)

Desta forma, na figura abaixo podemos ver o fluxo completo descrito anteriormente:

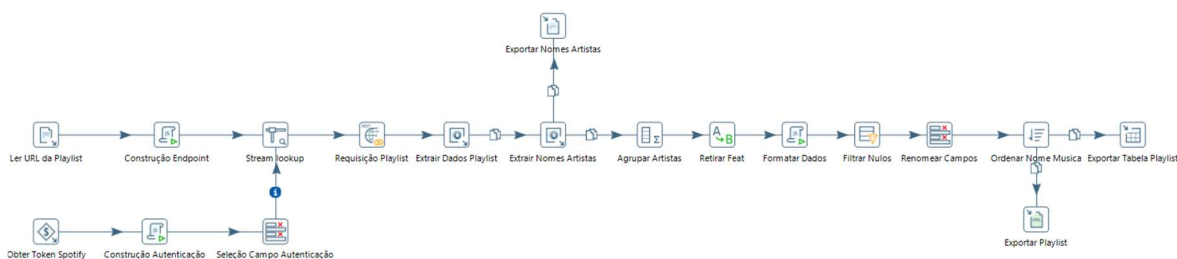


Figura 23 - Transformação Get Request Playlist completa

### 5.2.3 Transformação Get Request Artistas

O processo inicia com a leitura do ficheiro CSV de artistas, de modo a obter a lista de nomes que serão consultados na API do Spotify. Em paralelo, é recuperado o token previamente definido, que será usado para autenticação. Com este token, o passo Construção Autenticação monta o cabeçalho Authorization no formato correto de Bearer token, enquanto o passo Construção Endpoint gera o URL final de busca do Spotify para cada artista. Seguindo o fluxo, ocorre um Stream Lookup para associar o token de autenticação a cada endpoint, garantindo que cada requisição possua credenciais válidas.

Nome do Passo	Tipo de Passo	Função
Ler Artistas	CSV File Input	Lê o ficheiro CSV com os nomes dos artistas.
Ordenar Nome	Sort Rows	Ordena os artistas alfabeticamente.
Eliminar Duplicados	Unique Rows	Remove artistas repetidos para evitar chamadas redundantes à API.

Construção Endpoint	Modified Javascript Value	Cria dinamicamente o URL de pesquisa da API do Spotify para cada artista.
Obter Token Spotify	Get Variables	Recupera o token de autenticação guardado como variável global.
Construção Autenticação	Modified Javascript Value	Cria o header Authorization: Bearer <token>.
Seleção Campo Autenticação	Select Values	Seleciona apenas o header e a chave de junção (JOIN_KEY).
Stream Lookup 2	Stream Lookup	Junta o endpoint com o header de autenticação para cada requisição.

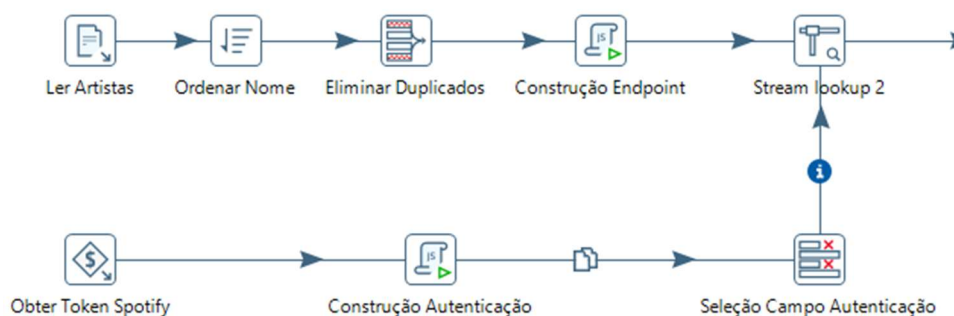


Figura 24 - Transformação Get Request Artistas (Parte 1)

O passo Requisição Artistas realiza a chamada GET à API do Spotify, retornando os dados dos artistas em formato JSON. Estes dados são processados através de um Json Input, que extrai informações detalhadas de cada artista, incluindo nome, ID, seguidores, popularidade e géneros musicais. Em paralelo, o passo Extrair Géneros coleta especificamente os géneros de cada artista. Os géneros são depois tratados pelo passo Formatação Géneros, que substitui valores nulos ou vazios por "unknown".

Nome do Passo	Tipo de Passo	Função
Requisição Artistas	REST Client	Faz o pedido REST à API do Spotify para obter os dados dos artistas.
Extrair Artistas	Json Input	Extrai do JSON os principais campos: nome, popularidade, seguidores, géneros e ID.

Extrair Géneros	Json Input	Percorre o array de géneros de cada artista e extrai os valores individuais.
-----------------	------------	--



Figura 25 - Transformação Get Request Artistas (Parte 2)

De seguida esses mesmos géneros são concatenados de modo a obter todos os géneros de um artista numa única string, garantindo uma visão consolidada das categorias musicais. Posteriormente são removidos quaisquer artistas repetidos e eliminados aqueles cujos registos se encontrem incompletos, garantindo que apenas artistas com ID definido sigam adiante.

Nome do Passo	Tipo de Passo	Função
Formatação Géneros	Modified Javascript Value	Substitui géneros nulos por "unknown" e limpa o texto.
Agrupar Géneros	Group By	Agrupa todos os géneros de um artista numa só linha, separados por vírgulas.



Figura 26 - Transformação Get Request Artistas (Parte 3)

O passo Renomear Campos padroniza os nomes dos campos para formatos simples e consistentes, preparando os dados para exportação. Por fim, os dados são enviados para duas saídas finais: criação de um ficheiro XML contendo todas as informações dos artistas e exportação das informações sobre os artistas para a tabela *Tabela\_Artistas* numa base de dados local.

Nome do Passo	Tipo de Passo	Função
Renomear Campos	Select Values	Ajusta os nomes dos campos para o formato final (ex.: <i>Nome_Artista</i> , <i>Popularidade_Artista</i> ).
Filtrar Nulos	Filter Rows	Remove artistas sem ID (registos incompletos).
Extrair Tabela Artistas	Table Output	Carrega os dados finais na tabela <i>Tabela_Artistas</i> da base de dados.
Extrair Artistas XML	XML output	Exporta os mesmos dados para um ficheiro XML com estrutura organizada.

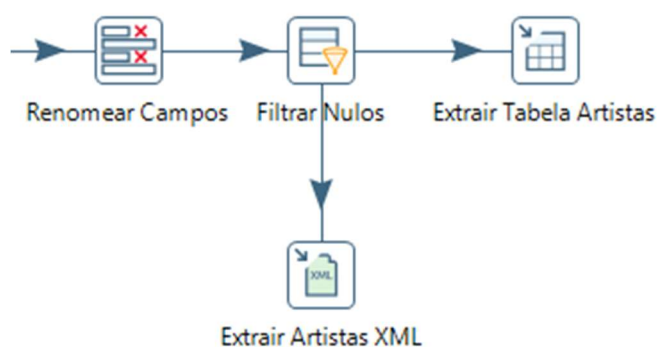


Figura 27 - Transformação Get Request Artistas (Parte 4)

Desta forma, na figura abaixo, podemos ver o fluxo completo descrito anteriormente:



Figura 28 - Transformação Get Request Artistas completa

### 5.2.4 Transformação Join

O processo inicia com a importação das tabelas da base de dados local SPOTIFY\_DB\_LOCAL. A tabela *Tabela\_Artistas* é lida, de modo a obter informações de cada artista, incluindo nome, popularidade, géneros musicais e número de seguidores. Paralelamente, lê-se a tabela *Tabela\_Dados\_Playlist*, extraindo dados de cada música, como ID da música, nome do artista, nome da música, duração em minutos, popularidade da música e nome do álbum. De seguida, o

passo Separar Artistas divide múltiplos artistas que apareçam em um mesmo campo Nome\_Artista, criando uma linha para cada artista individual (Artista\_Unico). Para garantir consistência, os nomes dos artistas são organizados tanto da tabela de artistas quanto da playlist. Posteriormente, são aplicadas operações de limpeza textual, como remover espaços extras, padronizar letras e eliminar caracteres indesejados.

Nome do Passo	Tipo de Passo	Função
Importar Tabela Playlist	Table Input	Carrega os dados das músicas da playlist na transformação.
Importar Tabela Artistas	Table Input	Carrega os dados dos artistas na transformação.
Separar Artistas	Split Fields	Separa múltiplos artistas em linhas individuais.
Ordenar Nomes Artistas / Ordenar Nomes Artistas 2	Sort Rows	Exporta os mesmos dados para um ficheiro XML com estrutura organizada.
Limpar Nomes / Limpar Nomes 2	String Operations	Limpa espaços e caracteres especiais nos nomes dos artistas.

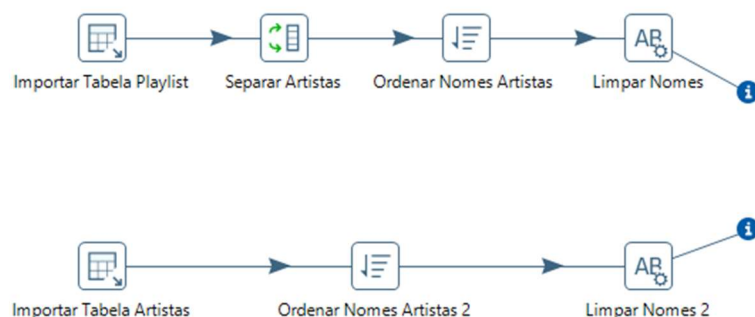


Figura 29 - Transformação Join (Parte 1)

O passo Merge Join realiza a junção interna entre os artistas da playlist e os artistas da tabela de artistas, usando como chave os campos Artista\_Unico e Nome\_Artista, permitindo associar informações complementares da tabela de artistas (popularidade, seguidores, gêneros) a cada música da playlist. Após a junção, o passo Seleção Campo Nome Artista remove colunas duplicadas geradas durante o merge, mantendo apenas os campos necessários. Por fim, o resultado consolidado é exportado para a tabela Tabela\_Analise\_Completa na base de dados local, armazenando de forma estruturada todas as informações de músicas e artistas.

Nome do Passo	Tipo de Passo	Função
Merge Join	Merge Join	Junta os dados da playlist com os dados dos artistas.
Seleção Campo Nome Artista	Select Values	Remove colunas desnecessárias após o join.
Exportar Tabela Join	Table Output	Carrega os dados finais na tabela Tabela_Analise_Completa da base de dados.



Figura 30 - Transformação Join (Parte 2)

Desta forma, na figura abaixo, podemos ver o fluxo completo descrito anteriormente:

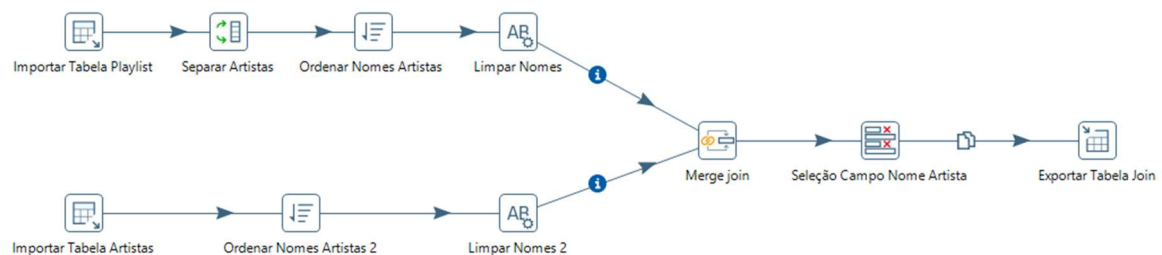


Figura 31 - Transformação Join completa

### 5.2.5 Transformação Get Album

O processo inicia com a leitura do link do álbum a partir de um ficheiro CSV e a obtenção do token de autenticação do Spotify. Com essas informações, os passos Construção Autenticação e Construção Endpoint criam, respetivamente, o header de autenticação (auth\_header) e o endpoint final da API do Spotify, extraíndo o ID do álbum a partir do URL fornecido. O passo Stream lookup associa o header de autenticação ao endpoint para permitir que a requisição à API seja autenticada corretamente.

Nome do Passo	Tipo de Passo	Função
Ler URL Album	CSV File Input	Lê o link da playlist guardado no ficheiro album_link.txt.
Construção Endpoint	Modified Javascript Value	Extrai o ID do album e constrói o URL final para a API do Spotify.
Obter Token Spotify	Get Variables	Recupera o token de autenticação guardado como variável global.
Construção Autenticação	Modified Javascript Value	Cria o header Authorization: Bearer <token>.
Seleção Campo Autenticação	Select Values	Seleciona apenas o header e a chave de junção (JOIN_KEY).
Stream Lookup	Stream Lookup	Junta o endpoint com o header de autenticação.

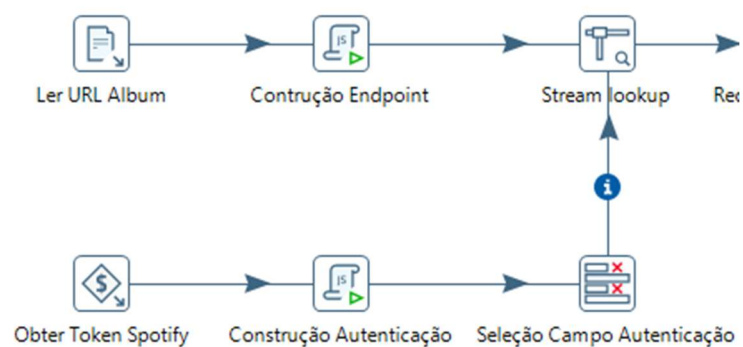


Figura 32 - Transformação Get Album (Parte 1)

De seguida, é realizada a chamada à API do Spotify para obter os dados do álbum. Os dados resultantes são processados através de um Json Input, que extrai informações de cada música, como número da música, nome da música, duração em milissegundos e ID da música.

Nome do Passo	Tipo de Passo	Função
Requisição Album	REST Client	Faz a chamada à API do Spotify para obter os dados do álbum.
Extrair Musicas Album	Json Input	Extrai os dados das músicas do álbum do JSON retornado.



Figura 33 - Transformação Get Album (Parte 2)

Posteriormente, o passo Formatar Duração converte a duração das músicas de milissegundos para o formato MM:SS, tornando os dados mais legíveis. Após essa formatação, são ajustados os nomes das colunas para corresponder ao padrão do projeto. Por fim, os dados são então ordenados pelo número da música e finalmente exportados para a tabela Tabela\_Musicas\_Album na base de dados local e, paralelamente, gerados num ficheiro XML.

Nome do Passo	Tipo de Passo	Função
Formatar Duração	Modified Script Value	Converte a duração das músicas de milissegundos para MM:SS.
Renomear Campos	Select Values	Ajusta os nomes dos campos para padronização.
Ordenar Nome Musica	Sort Rows	Ordena as músicas por número de música.
Extrair Album XML	XML output	Recupera o token de autenticação guardado como variável global.
Exportar Tabela Album	Table Output	Carrega os dados finais na tabela Tabela_Musicas_Album.

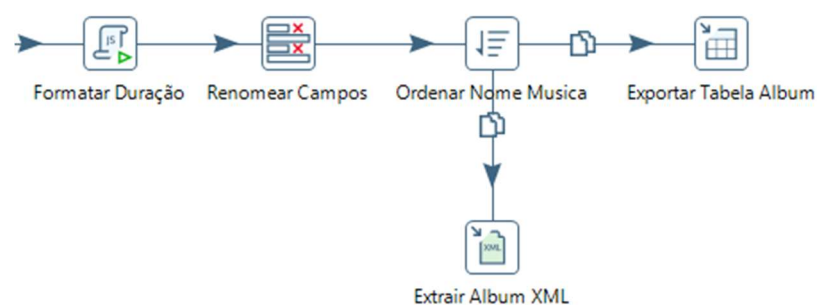


Figura 34 - Transformação Get Album (Parte 3)



Desta forma, na figura abaixo, podemos ver o fluxo completo descrito anteriormente:

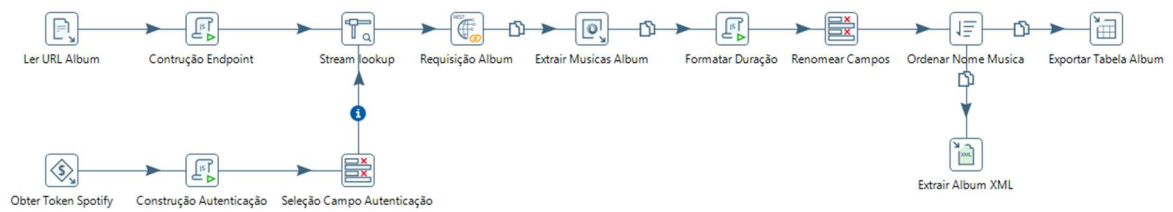


Figura 35 - Transformação Get Album (Parte 4)



## 6. Utilização de Jobs

### 6.1 Utilização de um Job para controlo do fluxo – Knime

Apesar de o KNIME oferecer mecanismos de organização de *workflows* mais formais, como as chamadas entre *workflows* utilizando nós dedicados (Container Input/Output, Workflow Executor, etc.), foi necessário adotar uma abordagem alternativa e pragmática para garantir a execução sequencial e modular do projeto, conforme a estrutura definida. Não tendo sido possível implementar a gestão do fluxo de forma tradicional através dos nós de comunicação avançada (como o Workflow Input e Workflow Output para comunicação direta entre *workflows* em diferentes separadores), a solução adotada foi baseada na persistência de ficheiros intermédios para ligar logicamente as diferentes etapas.

O projeto foi estruturado em dois *workflows* distintos (*workflow* principal e um componente secundário num separador diferente), comunicando-se implicitamente através do sistema de ficheiros:

#### **Workflow Principal (Spotify-ETL - Primeira Parte): Requisição e Registo**

Este *workflow* (que recebe o *endpoint* e o *Bearer Token*) executa a requisição principal (GET Request). O passo crucial que liga os dois *workflows* é a utilização do nó CSV Writer (Write Intermediate File), que guarda a resposta JSON bruta no ficheiro `json_album_intermediate.txt`. Este ato de escrita cumpre a função de ponto de entrega de dados para a fase seguinte, atuando como um "output" informal.

#### **Workflow Secundário (Component\_Album - Segunda Parte): Tratamento, Transformação e Exportação**

Este *workflow* é inicializado pelo nó CSV Reader, que lê explicitamente o ficheiro `json_album_intermediate.txt`. Esta leitura cumpre a função de ponto de receção de dados, atuando como um "input" informal. Após isto os dados são transformados e exportados.

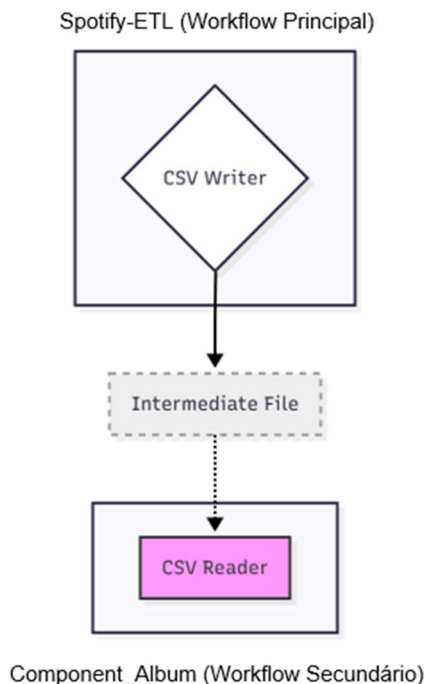


Figura 36 - Diagrama representativo do Job

#### Justificação da metodologia usada:

O método de controlo de fluxo baseado em ficheiros intermédios foi adotado como uma alternativa à utilização da funcionalidade formal de componentes nativos do KNIME, dadas as restrições de ambiente ou a complexidade na integração. As responsabilidades de cada etapa permanecem claramente separadas e isoladas em diferentes separadores lógicos, o controlo de fluxo assegurado, uma vez que a execução correta é garantida pela sequência de operações, sendo a segunda parte dependente da escrita do ficheiro intermédio pela primeira e a transparência, tornando a dependência entre os *workflows* explícita, com o ficheiro `json_album_intermediate.txt` a funcionar como a "ponte" de comunicação dos dados, conforme detalhado nas secções de explicação das transformações do projeto.

## 6.2 Utilização de um Job para controlo do fluxo – Pentaho Kettle

Este processo de orquestração foi estruturado para ser executado de forma sequencial, garantindo que cada passo se complete com sucesso antes de avançar para o próximo. O fluxo principal é o seguinte:

O processo inicia com o passo Start, que serve como o ponto de partida do fluxo. O primeiro passo funcional e crucial é a transformação ObterToken, responsável por autenticar-se na API do Spotify e obter o token de acesso necessário para todas as requisições subsequentes.

Uma vez autenticado, o Job adota uma abordagem condicional. Ele executa a etapa Verificar playlist\_link para confirmar a presença do ficheiro de entrada playlist\_link.txt, que contém a URLs da playlist a ser processada. Se o ficheiro existir, a transformação GetRequestPlaylist é executada para extrair os dados brutos das músicas e as suas informações básicas.

De seguida, o fluxo repete este padrão de verificação com o passo Verificar nomes\_artistas, que verifica se o ficheiro de saída nomes\_artistas.csv já foi gerado. Se o ficheiro existir, a etapa Get Request Artistas é acionada para obter dados detalhados sobre os artistas associados às músicas.

Finalmente, os dados obtidos são consolidados na transformação Join, onde os datasets de músicas e artistas são unidos. O ciclo de extração termina com outra verificação, Verificar album\_link, para garantir a existência de um ficheiro com o link do album desejado. Se disponível, a última transformação, Get Album, é executada para extrair informações detalhadas sobre os álbuns que completam o dataset.

Nome do Passo	Tipo de Passo	Função
Start	Start	Inicia o job de ETL do Spotify.
ObterToken	Transformation	Executa a transformação para obter o token do Spotify.
Verificar playlist_link	Checks if files exist	Verifica se o ficheiro playlist_link.txt existe..
GetRequestPlaylist	Transformation	Executa a transformação para obter dados da playlist.
Verificar nomes_artistas	Checks if files exist	Confirma se o ficheiro nomes_artistas.csv existe.
Get Request Artistas	Transformation	Executa a transformação para obter dados dos artistas.
Join	Transformation	Junta os dados de playlists e artistas.
Verificar album_link	Checks if files exist	Verifica se o ficheiro album_link.txt existe.
Get Album	Transformation	Executa a transformação para obter dados dos álbuns.

Desta forma, na figura abaixo, podemos ver o fluxo completo descrito anteriormente:



Figura 37 - Job principal do sistema

## 7. Visualização e Resultados

Nesta seção apresentam-se os dashboards principais desenvolvidos no Power BI e as tabelas da base de dados resultantes do processo ETL. O objetivo é demonstrar como os dados extraídos e transformados permitem visualizar de forma clara as tendências de popularidade das músicas na plataformas Spotify.

### 7.1 Dashboard

Os dashboards, contém quatro componentes principais cada: a média de popularidade de artista por gênero, que mostra a média de popularidade dos artistas agrupados por gênero musical, permitindo identificar quais estilos têm maior aceitação; a soma de seguidores por nome de artista, que apresenta o total de seguidores de cada artista, destacando os mais populares na plataforma; a média de popularidade de música por nome de música, que permite comparar a popularidade das músicas individualmente; e um cartão informativo, que resume dados de cada artista, incluindo nome, número de seguidores e média de popularidade, oferecendo uma visão rápida e consolidada.

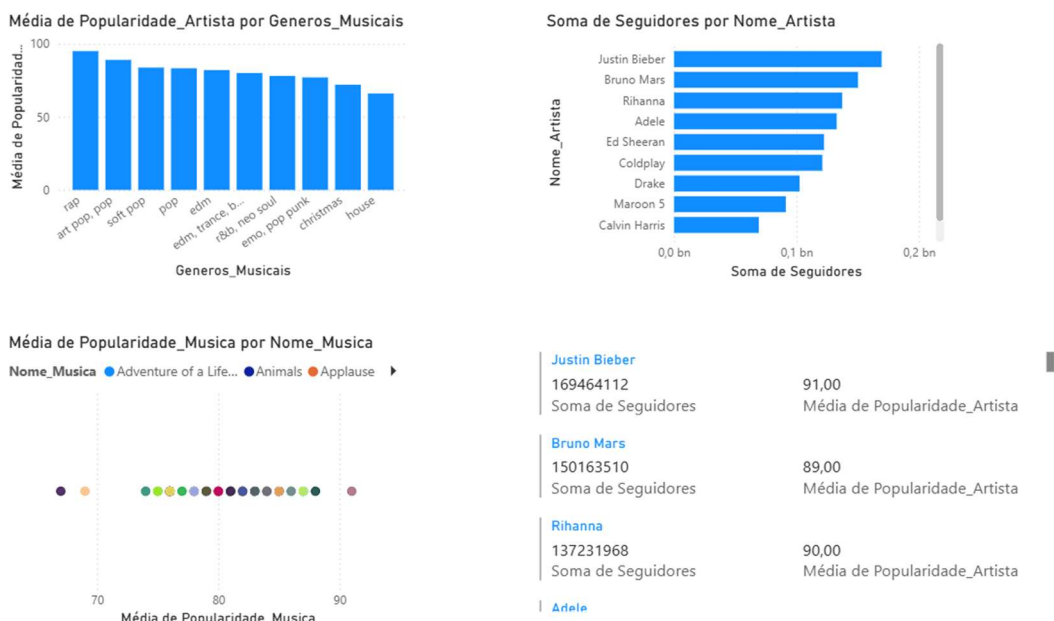


Figura 38 - Dashboard com resultado final (Pentaho Kettle)

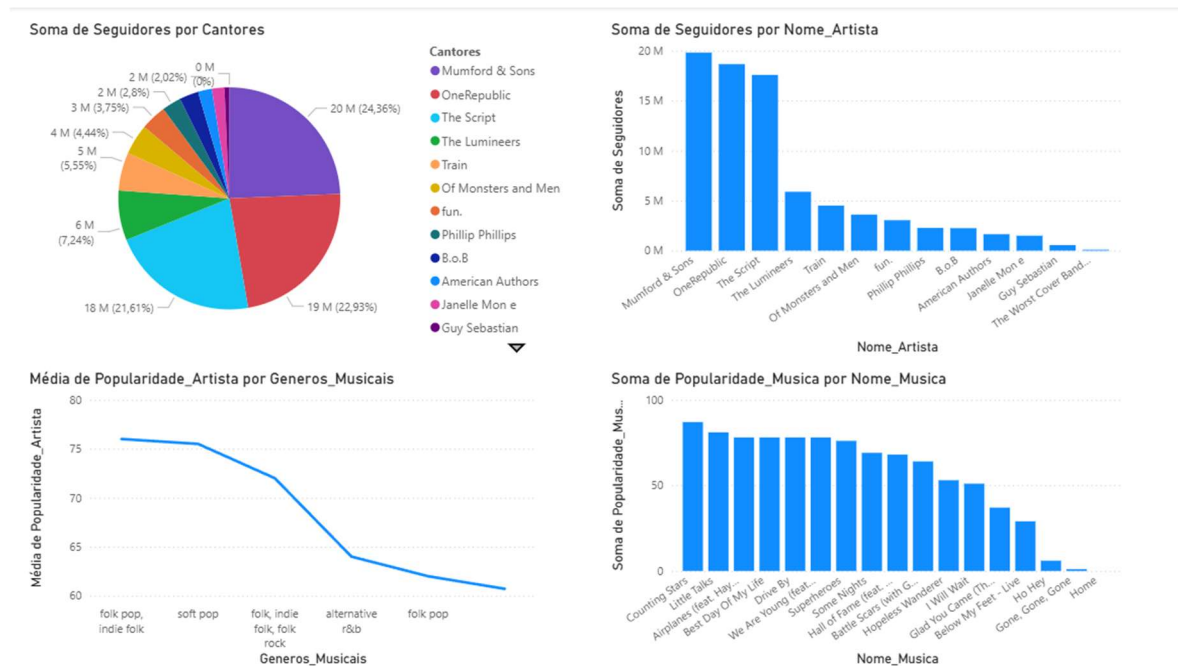


Figura 39 - Dashboard com resultado final (Knome)

## 7.2 Tabelas da base de dados

As quatro tabelas da base de dados exibem os dados processados e organizados durante o ETL. Cada tabela contém informações específicas, como dados de músicas, artistas, gêneros e estatísticas de reproduções, permitindo consultas detalhadas e suporte à análise apresentada nos dashboards.

### 7.2.1 Tabela Dados Playlist:

	ID_Musica	Nome_Musica	Nome_Artista	Duracao_Min	Popularidade_Musica	Nome_Album
1	04aAxqtGpSpv12UXAg4pkq	Centuries	Fall Out Boy	03:48	81	American Beauty/American Psycho
2	05SBRd4fXgn8FX7b8BCAE	I Need Your Love	Calvin Harris, Ellie Goulding	03:54	76	18 Months
3	07nH4fBxUB4IZcsf44Bm	Blame	Calvin Harris, John Newman	03:32	80	Motion
4	09CtPGIpYB4BrO8qb1RGsF	Sorry	Justin Bieber	03:20	85	Purpose (Deluxe)
5	0ct6r3EGToMLPrXHDvVjc	The Nights	Avicii	02:56	84	The Days / Nights
6	0DuWDLjRPyDRoPgaCsiY	Love In The Dark	Adele	04:45	80	25
7	0GNI8K3VATWBABQFAzBAYe	Stay	Rihanna, Mikky Ekko	04:00	75	Unapologetic (Deluxe)
8	0Hf4aIjpsN4Os2f0y0VqWl	Feel This Moment	Pitbull, Christina Aguilera	03:49	80	Global Warming: Meltdown (Deluxe Version)
9	0HPD5WQqrq7wPWR7P7Dw1i	Tik Tok	Kesha	03:19	84	Animal (Expanded Edition)
10	0IYBSQXN6CTvUZvg9S0IU	Let Me Love You	DJ Snake, Justin Bieber	03:25	87	Encore
11	0NIC4unbeSKZOpld9T7OaF	Secrets	Tiësto, KSHMR, VASSY	04:10	65	Club Life, Vol. 4 - New York City
12	0nrRP2bk19rLc0orkWPQk2	Wake Me Up	Avicii	04:07	85	True
13	0OPyDgTRuldCJ9B4bYSts	Hotel Room Service	Pitbull	03:57	76	Pitbull Starring In Rebelution
14	0qOnSQQF0yzuPWsXQ9paz	Stereo Hearts	Gym Class Heroes, Adam Levine	03:30	82	The Papercut Chronicles II
15	0vbtURX4qv117besfwmnD8	I Took A Pill In Ibiza - Seeb Remix	Mike Posner, Seeb	03:17	80	At Night, Alone.
16	12KUFSHFgT0XCoISlvdQ4	Break Free	Ariana Grande, Zedd	03:34	80	My Everything (Deluxe)
17	1dzQoRqT5uoxXVaAhTt0J	Just Dance	Lady Gaga, Colby O'Donis	04:01	83	The Fame
18	1giHuPhrLraKYrJMAEOIyc	Feel So Close - Radio Edit	Calvin Harris	03:26	82	18 Months
19	1mea3bSkSGXuIRvnydIB5b	Viva La Vida	Coldplay	04:02	86	Viva La Vida or Death and All His Friends
20	1nInOsHbtotAmEQhtvznP	Stronger (What Doesn't Kill You)	Kelly Clarkson	03:41	75	Stronger (Deluxe Version)

Figura 40 - Tabela Dados Playlist



## 7.2.2 Tabela Artistas:

	ID_Artista	Nome_Artista	Generos_Musicais	Popularidade_Artista	Seguidores
1	00FQb4JtyendYWaN8pK0wa	Lana Del Rey	unknown	88	51412934
2	0du5cEVh5yTK9QJze8zA0C	Bruno Mars	unknown	89	75179199
3	0EmeFodog0BfCgMzAlvKQp	Shakira	latin pop	84	39611690
4	0hCNtLu0JehylgoiP8L4Gh	Nicki Minaj	unknown	83	34259156
5	0jnsk9HBra6NMjO2oANoPY	Flo Rida	unknown	78	8952919
6	0TnOYISbd1XYRBk9myaseg	Pitbull	unknown	83	11881634
7	0X2BH1fck6amBloJhDVmmJ	Ellie Goulding	unknown	77	13180581
8	137W8MRPWKqSmrBGDBFSop	Wiz Khalifa	rap	80	12048769
9	1Cs0zKBU1kc0i8ypK3B9ai	David Guetta	edm	86	27093354
10	1I7ZsJRRS8wW3WJfPNS	Christina Aguilera	unknown	76	9218765
11	1ruutHJeECI7cos2n5TapO	Nayer	unknown	67	34656
12	1vyhD5VmyZ7KMfW5gqLgo5	J Balvin	reggaeton, latin	85	39028204
13	1yxSLGMDHIW21z4YXzZDS	Black Eyed Peas	unknown	81	9498299
14	1zNqDE7qDGCsyzJwvohVaoX	Anne-Marie	unknown	74	12808770
15	2DIgxzQSjYe5N6G9nkYghR	Jennifer Lopez	unknown	74	13690790
16	2gsggkzMSR49q6jpPvazou	Jessie J	unknown	70	12180042
17	2KsP6tYlJITBvSUxnvIVVw	Mike Posner	unknown	67	1519881
18	2o5JDhtHVPPhrJdv3cEQ99Z	Tiësto	edm, trance, big room, house	79	7962637
19	2q3GG88dVwuQPF4FmySr9I	The Score	unknown	64	1548794
20	2wX6xSig4Rig5kZU6ePWWe	KSHMR	big room, edm, slap house, electronica	60	1778366

Figura 41 - Tabela Artistas

## 7.2.3 Tabela Join:

	ID_Musica	Nome_Musica	Nome_Album	Duracao_Min	Popularidade_Musica	Nome_Artista	Generos_Musicais	Popularidade_Artista	Seguidores
1	05SBR44Fgn8FX7b8BCAE	I Need Your Love	18 Months	03:54	76	Ellie Goulding	unknown	77	13180581
2	07nH4B4UB4IZsf44Bm	Blame	Motion	03:32	80	John Newman	unknown	67	817398
3	0GNI8K3VATWBABQFAzBAYe	Stay	Unapologetic (Deluxe)	04:00	75	Rihanna	unknown	89	68718940
4	0H64aUpa4N4Oz2Ry0VqIM	Feel This Moment	Global Warming: Meltdown (Deluxe Version)	03:49	80	Christina Aguilera	unknown	76	9218765
5	0H64aUpa4N4Oz2Ry0VqIM	Feel This Moment	Global Warming: Meltdown (Deluxe Version)	03:49	80	Pitbull	unknown	83	11881634
6	0HPDSWQqrg7wPIWR7P7Dw1i	TK ToK	Animal (Expanded Edition)	03:19	84	Kesha	unknown	78	9124132
7	0IYBSQXN6CTvUzvg9S0IU	Let Me Love You	Encore	03:25	87	DJ Snake	unknown	77	9260355
8	0NIC4unbe5K2Op1d9T7OaF	Secrets	Club Life, Vol. 4 - New York City	04:10	65	KSHMR	big room, edm, slap house, electronica	60	1778366
9	0NIC4unbe5K2Op1d9T7OaF	Secrets	Club Life, Vol. 4 - New York City	04:10	65	Tiësto	edm, trance, big room, house	79	7962637
10	0NIC4unbe5K2Op1d9T7OaF	Secrets	Club Life, Vol. 4 - New York City	04:10	65	VASSY	electronica	57	41563
11	0OPyDgTRuldCj8B4bY5ths	Hotel Room Service	Pitbull Starring In Rebelution	03:57	76	Pitbull	unknown	83	11881634
12	0qOnSQQF0yzuPWsYq9paz	Stereo Hearts	The Papercut Chronicles II	03:30	82	Adam Levine	unknown	70	1062594
13	0qOnSQQF0yzuPWsYq9paz	Stereo Hearts	The Papercut Chronicles II	03:30	82	Gym Class Heroes	unknown	67	1663869
14	0vbtURX4qv1I7besfmmD8	I Took A Pill In Ibiza - Seeb Remix	At Night, Alone.	03:17	80	Mike Posner	unknown	67	1519881
15	1dsQoRqTSuozXVaAhTct0J	Just Dance	The Fame	04:01	83	Colby O'Donis	unknown	68	177150
16	1fsofagEpaACiVEHIZBje6W	Havana	Camila	03:37	80	Camila Cabello	unknown	78	35522817
17	1fsofagEpaACiVEHIZBje6W	Havana	Camila	03:37	80	Young Thug	rap, melodic rap, trap	83	12374016
18	1rC8L8YpawU5S3ymy2cC	Calling (Lose My Mind) - Radio Edit	Calling (Lose My Mind)	03:25	65	Sebastian Ingrosso	edm	67	1007823
19	1XGmz0PvufgQYYnV2i0A	Payphone	Overexposed Track By Track	03:51	85	Wiz Khalifa	rap	80	12048769
20	20I6sl0MTCk86w7yavx0	Call Me Maybe	Kiss	03:13	81	Carly Rae Jepsen	unknown	69	5133354

Figura 42 - Tabela Join



#### 7.2.4 Tabela Album:

	ID_MUSICA	NOME_MUSICA	NUMERO_MUSICA	DURACAO_MIN
1	0d2lk5rp9lcl1oqn4178x2	Robot Voices	7	03:57
2	0DGf6s54MQ3Oht41mj6s6K	Tally	12	03:32
3	0Ji9UonfwC90rbZ4laQhOb	City Walls	1	05:22
4	11QNo3ld81pncy71q6Vlic	Downstairs	6	05:26
5	1aVDGnB6ydtXwudYLgUB6h	Cottonwood	9	03:08
6	20BFo9REyGdCxrlPmD3xph	Drum Show	3	03:23
7	22BA9h34P0Q2PGZudsEBoo	The Contract	5	03:45
8	2BHSRIGgJwzTPfYvAax28m	Center Mass	8	03:48
9	3FbMtvXMojmuAzlAhgwK0l	Intentions	13	02:15
10	3yKIL7niCEJajKIVmMNBZA	Days Lie Dormant	11	03:26
11	43ee3gqWBIPKe2MeGJ2S6l	RAWFEAR	2	03:22
12	5QzlDIRuT9D5LdpybJoNYi	One Way	10	02:43
13	6YJYhPKP7MRzJSHqe3QI9K	Garbage	4	03:16

Figura 43 - Tabela Album

## 8. Demonstração

Esta secção apresenta um vídeo curto de demonstração que ilustra o funcionamento completo do projeto, abrangendo todas as fases do processo ETL, tanto para o Knime quanto para o Pentaho Kettle - Extração, Transformação, Carregamento e Visualização. O vídeo mostra de forma prática como as credenciais devem ser introduzidas no ficheiro credentials.txt e como os dados são obtidos a partir do Spotify, tratados e integrados através do Knime e do Pentaho Kettle, culminando numa visualização interativa das tendências musicais analisadas. Através dos links ou QR Codes disponibilizados, é possível acompanhar o fluxo de trabalho em tempo real e compreender a aplicação prática das etapas desenvolvidas.

Demonstração Knime	Demonstração Pentaho Kettle
 <a href="https://youtu.be/4znXSTrOFXE">https://youtu.be/4znXSTrOFXE</a>	 <a href="https://youtu.be/ujATOGgxuns">https://youtu.be/ujATOGgxuns</a>



## **9. Funcionalidades Não Implementadas**

Durante a fase de desenvolvimento, uma funcionalidade desejada para o controlo do workflow, foi o envio automático de e-mail. Esta não foi implementada com sucesso em nenhuma das ferramentas utilizadas, o KNIME e o Kettle (Pentaho Data Integration). Para tentar contornar as restrições de segurança do Gmail, foi feita uma tentativa de configurar o acesso utilizando o mecanismo de Credencial de APP (App Password) através das definições da conta Google, mas os erros de protocolo persistiram.

No KNIME, o erro foi de segurança protocolar, com a mensagem "530-5.7.0 Must issue a STARTTLS command first", indicando que o servidor SMTP do Gmail rejeitou a ligação por o nó não ter iniciado o protocolo de encriptação segura (TLS/SSL) exigido pelo RFC 3207.

Paralelamente, no Kettle, o problema era de segurança de certificação, manifestado pelo erro `SSLHandshakeException: PKIX path building failed: unable to find valid certification path to requested target`. Este erro significa que o ambiente Java do Kettle falhou na validação da cadeia de certificados do servidor SMTP, impedindo o handshake SSL.

Ambos os problemas decorreram de configurações avançadas de segurança e encriptação de rede que não foram resolvidas no ambiente de desenvolvimento, mesmo após a tentativa de utilizar a Credencial de APP.



## **10. Conclusão e Trabalhos Futuros**

O projeto permitiu desenvolver com sucesso um processo ETL completo, desde a extração de dados do Spotify até à sua transformação, integração e visualização final através de um dashboard criado no Power BI. Foram utilizadas duas ferramentas distintas de integração — KNIME e Pentaho — o que permitiu comparar abordagens e metodologias diferentes na implementação dos processos ETL. O trabalho possibilitou analisar de forma clara a popularidade de músicas e artistas, bem como identificar tendências por gênero musical e padrões de comportamento dentro da plataforma.

Durante o desenvolvimento, surgiram alguns desafios, nomeadamente a limpeza, uniformização e integração dos dados, uma vez que o conjunto de informações apresentava campos heterogêneos e formatos variados. Estes obstáculos foram superados através da aplicação de transformações específicas em ambas as ferramentas, garantindo a consistência e qualidade dos dados carregados na base de dados final. Outro desafio relevante foi a definição das métricas de análise, em particular a escolha das variáveis de popularidade e seguidores, de modo a refletir corretamente o desempenho dos artistas e músicas.

No futuro, o projeto poderá ser expandido com a implementação de um agendamento automático do processo ETL, permitindo atualizações periódicas e automatizadas dos dados. Poderá também ser interessante integrar novas fontes de dados, como YouTube Music, Apple Music ou TikTok, para enriquecer as análises e possibilitar comparações entre plataformas. Além disso, uma próxima fase poderá incluir modelos preditivos, utilizando técnicas de machine learning para antecipar tendências de popularidade e crescimento de artistas com base em dados históricos.





## 11. Referências Bibliográficas

Spotify. (s.d.). *Spotify Web Player*. Recuperado em 16 de outubro de 2025, de <https://open.spotify.com>

Spotify. (s.d.). *Spotify for Developers*. Recuperado em 16 de outubro de 2025, de <https://developer.spotify.com>

KNIME AG. (s.d.). *KNIME Analytics Platform*. Recuperado em 16 de outubro de 2025, de <https://www.knime.com>

Hitachi Vantara. (s.d.). *Pentaho Data Integration (Kettle)*. Recuperado em 16 de outubro de 2025, de <https://community.hitachivantara.com>

Microsoft. (s.d.). *Microsoft SQL Server*. Recuperado em 16 de outubro de 2025, de <https://www.microsoft.com/sql-server>

Microsoft. (s.d.). *Power BI*. Recuperado em 16 de outubro de 2025, de <https://powerbi.microsoft.com>

KNIME. (s.d.). *Getting Started Guide*. Recuperado em 16 de outubro de 2025, de <https://www.knime.com/getting-started-guide>

Roldán, M. C. (2015). *Pentaho Data Integration (Kettle) Tutorial*. Recuperado em 16 de outubro de 2025, de <https://pentahopublic.atlassian.net/wiki/spaces/EAI/pages/371557269/Pentaho%2BData%2BIntegration%2BKettle%2BTutorial>