

✓ Lectura y librerías

```
1 # Librerías
2 import pandas as pd
```

```
1 # Leer csv del punto 1
2
3 # Especifica la ruta del archivo CSV
4 ruta_archivo_csv = '/content/dataset.csv'
5
6 # Usa la función read_csv de pandas para cargar el archivo
7 data = pd.read_csv(ruta_archivo_csv)
8
```

✓ Completeness

```
1 # Valores Nulos
2 total_null_values = data.isnull().sum()
3
4 # Total de nulos
5 total_values = data.count().sort_values(ascending=True)
6
7 # % de nulos
8 null_values_percentage = total_null_values/total_values *100
9
10 # df de los missing values con porcentaje
11 missing_values = pd.concat({'Total Values' : total_values, 'Null_values': total_null_values, 'Percentage of Missing Values': null_values_percentage}, axis=1)
12 print(missing_values)
```

	Total Values	Null_values \
album_name	477	62
track_id	531	8
track_name	532	7
audio_features.energy	537	2
audio_features.loudness	537	2
audio_features.danceability	537	2
audio_features.time_signature	538	1
audio_features.speechiness	538	1
audio_features.key	538	1
audio_features.acousticness	538	1
audio_features.liveness	538	1
audio_features.tempo	538	1
artist_name	539	0
artist_id	539	0
artist_popularity	539	0
album_id	539	0
audio_features.id	539	0
disc_number	539	0

```

audio_features.instrumentalness    539    0
album_release_date                539    0
audio_features.mode                539    0
track_popularity                  539    0
track_number                      539    0
explicit                          539    0
duration_ms                       539    0
audio_features.valence             539    0
album_total_tracks                539    0

```

Percentage of Missing Values

```

album_name                        12.997904
track_id                          1.506591
track_name                        1.315789
audio_features.energy             0.372439
audio_features.loudness           0.372439
audio_features.danceability       0.372439
audio_features.time_signature     0.185874
audio_features.speechiness        0.185874
audio_features.key                0.185874
audio_features.acousticness       0.185874
audio_features.liveness           0.185874
audio_features.tempo              0.185874
artist_name                       0.000000
artist_id                         0.000000
artist_popularity                 0.000000
album_id                          0.000000
audio_features.id                 0.000000
disc_number                       0.000000
audio_features.instrumentalness   0.000000
album_release_date                0.000000
audio_features.mode               0.000000
track_popularity                  0.000000
track_number                      0.000000
explicit                          0.000000
duration_ms                       0.000000
audio_features.valence             0.000000
album_total_tracks                0.000000

```

```

1 # Verificar el numero de columnas leidas a partir del csv
2 columnas = data.columns.tolist()
3 print(columnas)

```

```
['disc_number', 'duration_ms', 'explicit', 'track_number', 'track_popularity', 'track_id', 'track_name', 'audio_features.danceability', 'audio_features.energy', 'audio_features.key', 'au
```

Aspectos destacados

- Existe un 12.9% de valores nulos para album_name, seguido de 1.5% para track_name y audio_features.energy con un 1.3%, para los demas, se encuentra por debajo del 1% del total de registros.
- El album_name en porcentaje es mayor, al ser repetitivo en sus n-numeros de discos por cada album. Pero es de resaltar, que 8 track_name están vacios del total de 539 canciones unicas.
- Las columnas (27) se encuentran a completitud, y existe en zona raw una coherencia entre los datos del json en sus diccionarios y forma de entregarse al dataset.csv

✓ Accuracy

```
1 # duration_ms
2
3 # Filtrar valores menores a 60000
4 df_filtered = data.loc[data['duration_ms'] < 60000,['duration_ms']]
5 print(df_filtered)
```

```
      duration_ms
392      -107133
408      -223093
420         10
432        1000
472        3000
```

```
1 # track_popularity
2 minimo_track_popularity, maximo_track_popularity = data['track_popularity'].min(), data['track_popularity'].max()
3 print(minimo_track_popularity,maximo_track_popularity)
```

```
-92 152
```

```
1 # Track name
2
3 # Leer la columna y contar caracteres extraños
4 caracteres_extranos = data['track_name'].str.contains(r'^\x00-\x7F').sum()
5
6 print(f"Cantidad de caracteres extraños: {caracteres_extranos}")
```

```
Cantidad de caracteres extraños: 52
```

```
1 # audio_features.acousticness
2 minimo_audio_features_acousticness, maximo_audio_features_acousticness = data['audio_features.acousticness'].min(), data['audio_features.acousticness'].max()
3 print(minimo_audio_features_acousticness,maximo_audio_features_acousticness)
4
5
6 min_acous = data.loc[data['audio_features.acousticness'] < 0,['audio_features.acousticness']]
7 print(min_acous)
8
9 max_acous = data.loc[data['audio_features.acousticness'] > 1,['audio_features.acousticness']]
10 print(max_acous)
```

```
-0.00354 5.0
audio_features.acousticness
3      -0.000537
6      -0.003540
audio_features.acousticness
1         5.0
527        1.5
535        2.0
```

```
1 # artist_popularity
2 valores_unicos = data['artist_popularity'].unique()
3
4 print("Valores únicos en 'artist_popularity':")
5 print(valores_unicos)
6
```

```
Valores únicos en 'artist_popularity':
[120]
```

```
1 # album_release_date
2 _release_date = data.groupby('album_release_date')['album_release_date'].count()
3 _release_date_ord = _release_date.sort_values(ascending=False).head(30)
4
5 print(_release_date_ord, True)
```

```
album_release_date
2017-11-09      46
2019-08-23      36
2008-11-11      35
2020-11-25      34
2014-01-01      32
2010-10-25      30
2021-11-12      30
2021-04-09      26
2027-05-26      24
2010-01-01      22
2012-10-22      22
2023-10-27      22
2023-07-07      22
2023-10-26      21
2022-10-22      20
2020-08-18      17
2021-01-07      17
2020-07-24      16
2017-11-10      16
2020-12-11      15
1989-10-24      15
2022-10-21      13
2008-06-28       8
Name: album_release_date, dtype: int64 True
```

```
1 # album_total_tracks
2 album_total_tracks = data.groupby('album_total_tracks')['album_total_tracks'].count()
3 album_total_tracks_ord = album_total_tracks.sort_values(ascending=False).head(30)
4
5 print(album_total_tracks_ord, True)
```

```
album_total_tracks
22      88
34      64
16      48
46      46
19      38
18      36
```

```

17      34
13      26
26      26
24      24
21      21
20      20
15      16
10      15
Thirteen 15
14      14
8        8
Name: album_total_tracks, dtype: int64 True

```

1

Aspectos destacados

- En la columna "duration_ms" existen 5 registros inadecuados en relación al campo de tiempo (ms). Los valores negativos no tienen representación en el tiempo y los valores menores a 60000, fueron 10, 1000 y 3000 ms, que representan menos de 3 segundos en una canción, lo cual no es adecuado para el ámbito musical.
- Para la columna "track_popularity" existen valores negativos y mayores a 100. El rango que ofrece la documentación de Spotify va de los 0 a 100, todo valor por fuera, se considerará anormal.
- Para "track_name" se encontraron valores como: This Love (Taylor's Version) que contienen "â€™" caracteres extraños, por problemas de comas, tildes, acentos y no tenerlo en cuenta al llevarlos de json-csv-pandas por el formato. Estos cambios deben realizarse, para tener calidad en el nombre de la canción.
- Para "audio_features.acousticness" existen dos valores -0.000537,-0.003540 por debajo del rango de 0 y 3 por encima de 1 (1.5, 2 y 5.0)
- Para "audio_features.instrumentalness" 7.28x-06, valor que no esta en formato tipo numero para su notación científica.
- Para "artist_id" con valor "06HL4z0CvFAxyc27GX" este dato, no responde a la API para la cantante Taylor Swift (ver imagen posterior). Este es un hecho relevante, dado que es la principal llave para consultar a la cantante en la API de Spotify. El valor que llama a su nombre es: "06HL4z0CvFAxyc27GXpf02"
- La columna de "artist_popularity" en la documentación de la API de Spotify el rango va de 0 a 100. El unico valor que trae los datos es de 120, valor que no corresponde, de hecho taylor tiene un artist_popularity de: 100. En este caso, tocaría modifica el valor o truncarlo por encima de este.
- En la columna "album_release_date" la fecha "2027-05-26" para el album: "Midnights (The Til Dawn Edition)" es incorrecta, dado que nos encontramos en 2024, la fecha correcta es: "2023-05-26". Aparte la fecha "1989-10-24" del album "Taylor Swift" fue lanzado el "2006-10-24", claramente teniendo un error en la casilla del año de 1989 a 2006.
- La columna "album_total_tracks" la documentación recomienda valores enteros, pero se encuentra un valor diferente "Thirteen". Este dato, no concuerda con el tipo de dato usados en esta columna.
- Valores nulos se pueden revisar en el encabezado de "Completeness"

Status 400 error por no encontrar el artist_id de Taylor Swift

Web API • References / Artists / Get Artist

Get Artist OAuth 2.0

Get Spotify catalog information for a single artist identified by their unique Spotify ID.

Important policy notes

- ▶ Spotify content may not be downloaded
- ▶ Keep visual content in its original form
- ▶ Ensure content attribution

Request

GET `/artists/{id}`**id** string RequiredThe **Spotify ID** of the artist.

Example: 0TnOYISbd1XYR8k9myaseg

ENDPOINT `https://api.spotify.com/v1/artists/{id}`**id** `06HL4z0CvFAxyc27GX`

Try it

REQUEST SAMPLE

cURL Wget HTTPie

```
1 curl --request GET \
2   --url https://api.spotify.com/v1/artists/06HL4z0CvFAxyc27GX \
3   --header 'Authorization: Bearer 1PodFZRZbvb...qqillR0w2z'
```

RESPONSE SAMPLE

```
1 {
2   "error": {
3     "status": 400,
4     "message": "Invalid base62 id"
5   }
6 }
```

Uniqueness

```
1 # Valores unicos
2
3 # Valores duplicados
4 duplicated_values = data.duplicated()
5
6 # Numero de duplicados
7 print("El número de duplicados en el df es {}".format(duplicated_values.sum()))
```

El número de duplicados en el df es 18

Aspectos destacados

- Existe 18 registros duplicados, los cuales modifican campos como el numero de album_total_tracks. El porcentaje del total de registros no duplicados es del 96.6%.

Validity

```
1 # Campo explicit (BOOL)
2 explicit = data.groupby('explicit')['explicit'].count()
3 explicit_ord = explicit.sort_values(ascending=False).head(30)
4
5 print(explicit_ord, True)
```

```
explicit
False    480
True      54
No         4
```

```

Si      1
Name: explicit, dtype: int64 True

1 # Campo album_release formato date
2
3 date_album = data.groupby('album_release_date')['album_release_date'].count()
4 album_date_ord = date_album.sort_values(ascending=False).head(50)
5 print(album_date_ord, True)

```

```

album_release_date
2017-11-09      46
2019-08-23      36
2008-11-11      35
2020-11-25      34
2014-01-01      32
2010-10-25      30
2021-11-12      30
2021-04-09      26
2027-05-26      24
2010-01-01      22
2012-10-22      22
2023-10-27      22
2023-07-07      22
2023-10-26      21
2022-10-22      20
2020-08-18      17
2021-01-07      17
2020-07-24      16
2017-11-10      16
2020-12-11      15
1989-10-24      15
2022-10-21      13
2008-06-28       8
Name: album_release_date, dtype: int64 True

```

Aspectos destacados

- El campo explicit que debe ser booleano (TRUE/FALSE) pero en la revisión existen 4 casos con el valor No y 1 con Si, se debe realizar un reemplazo de los valores y dejarlos en su formato deseado.
- El formato de "release_date" en los datos se encuentra como tipo entero y la API sugiere tipo string. Los datos se encuentran con formato tipo AAAA-MM-DD, pero existe una diferencia entre la documentación de la API y los datos que entrega, dado que recomienda tener formato tipo AAAA-MM, y los datos estan en AAAA-MM-DD, en este caso tocaría encontrar un camino para estandarizar y alinearse con la documentación de la API y los datos recibidos (ver foto posterior)

Campo Album[release_date] API Spotify



✓ Consistency

```
1 # Cantidad de filas y columnas
2 print(data.shape)
```

```
(539, 27)
```

```
1 print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 539 entries, 0 to 538
Data columns (total 27 columns):
#   Column                                Non-Null Count  Dtype
---  -
0   disc_number                          539 non-null    int64
1   duration_ms                          539 non-null    int64
2   explicit                             539 non-null    object
3   track_number                         539 non-null    int64
4   track_popularity                     539 non-null    int64
5   track_id                             531 non-null    object
6   track_name                           532 non-null    object
7   audio_features.danceability          537 non-null    float64
8   audio_features.energy                537 non-null    float64
9   audio_features.key                   538 non-null    float64
10  audio_features.loudness               537 non-null    float64
11  audio_features.mode                   539 non-null    int64
12  audio_features.speechiness            538 non-null    float64
13  audio_features.acousticness           538 non-null    float64
14  audio_features.instrumentalness       539 non-null    object
15  audio_features.liveness               538 non-null    float64
16  audio_features.valence                 539 non-null    float64
17  audio_features.tempo                  538 non-null    float64
18  audio_features.id                     539 non-null    object
19  audio_features.time_signature         538 non-null    float64
20  artist_id                             539 non-null    object
21  artist_name                           539 non-null    object
22  artist_popularity                     539 non-null    int64
23  album_id                              539 non-null    object
24  album_name                            477 non-null    object
25  album_release_date                    539 non-null    object
26  album_total_tracks                    539 non-null    object
dtypes: float64(10), int64(6), object(11)
memory usage: 113.8+ KB
None
```

Coherencia del tipo de datos:

- df Type: asignación del tipo de datos al leer el csv.
- API Type: definición a partir de documentación de la API de Spotify del tipo de dato

Nombre columna	df Type	API Type
0 disc_number	int64	integer
1 duration_ms	int64	integer
2 explicit	object	bool
3 track_number	int64	integer
4 track_popularity	int64	integer
5 track_id	object	string
6 track_name	object	string
7 audio_features.danceability	float64	float
8 audio_features.energy	float64	float
9 audio_features.key	float64	integer
10 audio_features.loudness	float64	float
11 audio_features.mode	float64	integer
12 audio_features.speechiness	float64	float
13 audio_features.acousticness	float64	float
14 audio_features.instrumentalness	object	float
15 audio_features.liveness	float64	float
16 audio_features.valence	float64	float
17 audio_features.tempo	float64	float
18 audio_features.id	object	string
19 audio_features.time_signature	float64	integer
20 artist_id	object	string
21 artist_name	object	string
22 artist_popularity	int64	integer
23 album_id	object	string
24 album_name	object	string
25 album_release_date	object	string
26 album_total_tracks	object	integer

▼ Timeliness

No existe un campo en el registro de datos que permita ver la actualización de datos por parte de Spotify sobre registros como popularidad que puede variar con el input de los usuarios. Este campo, podría permitir dentro de la calidad de datos, ver en cuanto tiempo fue su última actualización.

▼ Otros aspectos

1. Se encontraron dentro del json, las claves Unicas para crear topologias y construcción de un data modelling a partir de las diferentes tipo de request a la API de spotify.

- track_id
- audio_features.id

- album_id

2. Ordenar las columnas mejor y con nombres ajustados al diccionario. Seguir la secuencia:

- artista.[*] > album.[*] > track.[*] > audio_features.[*]

De esa misma forma el orden de las columnas de mayor a menor importancia, pasando primero por artista y terminando en audio_features

3. Columnas que probablemente no generen valor

En audio_features se encuentran audio_features.liveness y audio_features.valence, datos que son muy específicos para un contexto de entender algunos comportamientos de la canción sobre algo puntual. Sobre un posterior análisis, se puede considerar tener o no, algunas columnas de audio_features, que entrega el dataset.