In [ ]:

```python
from config import views
from spark import createSession

from typing import List, Tuple

from matplotlib import pyplot as plt
from pyspark.sql.dataframe import DataFrame

import pyspark.sql.functions as F
import pyspark.sql.types as T

from IPython.display import display
```

In [ ]:

```python
def get_columns_of_type(data_frame: DataFrame, type: str) -> List[str]:
    return [column[0] for column in data_frame.dtypes if column[1] == type]
```

In [ ]:

```python
LENGTH = 80
def show_table_name(table: str) -> None:
    print('=' * LENGTH)
    print(' ' * ((LENGTH - len(table)) // 2), table.upper())
    print('=' * LENGTH)

def show_column_name(column: str) -> None:
    print(column.upper())
```

In [ ]:

```python
VERSION = 'v2'

VIEWS = views(VERSION)
spark = createSession()

for view, file in VIEWS.items():
    df = spark.read.json(file)
    for column in get_columns_of_type(df, 'boolean'):
        df = df.withColumn(column, F.col(column).cast(T.IntegerType()))

    for column in df.columns:
        if column in ['timestamp', 'release_date']:
            df = df.withColumn(f'{column}_s', F.unix_timestamp(column, "yyyy[-MM[-dd[['T'][' ']HH:mm[:ss[.SSSSSS]]]]]"))

    df.createOrReplaceTempView(view)
```

```
your 131072x1 screen size is bogus. expect trouble
23/04/01 21:45:20 WARN Utils: Your hostname, LAPTOP-7KCON786 resolves to a loopback address: 127.0.1.1; using 192.168.18.206 instead (on interface eth0)
23/04/01 21:45:20 WARN Utils: Set SPARK_LOCAL_IP if you need to bind to another address
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
23/04/01 21:45:21 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
```

In [ ]:

```python
DATA_FRAMES = list(zip(VIEWS.keys(), [spark.sql(f"SELECT * FROM {view}") for view in VIEWS.keys()]))
```

In [ ]:

```python
for view, df in DATA_FRAMES:
    show_table_name(view)
    for column, type in df.dtypes:
        print(column.upper(), '-', type)

    try:
        dfp = df.limit(100_000).toPandas()
        display(dfp)
    except Exception as e:
        df.show()
        print(df.count(), 'rows')
```

```
================================================================================
                                     ARTISTS
================================================================================
GENRES - array<string>
ID - string
NAME - string
```

	genres	id	name
0	[filmi, indian folk, indian rock, kannada pop]	72578usTM6Cj5qWsi471Nc	Raghu Dixit
1	[desi pop, hindi indie, indian indie, indian r...	7b6Ui7JVaBDEfZB9k6nHL0	The Local Train
2	[indian folk]	4bvGDTEPFnllKiJaEZGuXk	Achint
3	[opm, pinoy hip hop, pinoy r&b, pinoy trap, ta...	0n4a5imdLBN24fIrBWoqrv	Because
4	[hindi indie, indian indie, indian singer-song...	4gdMJYnopf2nEUcanAwstx	Anuv Jain
...	...	...	...
27519	[italian hip hop]	2My6j5BEgOi8VHi5WGVyfw	Apocalypshit Army
27520	[belgian pop]	0bzW9kGcTyMxXuG9dUdj7E	GRANDGEORGE
27521	[thai indie]	4iS19hLpsgRd8jLPKI4Ni3	Blissonic
27522	[thai indie]	3JGC3LkYrwlrTscixVwY72	พราว
27523	[indie folk]	7AhCTepWX7n4dQFh3Ro3YG	Haroula Rose

27524 rows × 3 columns

```
================================================================================
                                     SESSIONS
================================================================================
EVENT_TYPE - string
SESSION_ID - bigint
TIMESTAMP - string
TRACK_ID - string
USER_ID - bigint
TIMESTAMP_S - bigint
```

	event_type	session_id	timestamp	track_id	user_id	timestamp_s
0	PLAY	124	2020-04-17T16:43:09	5EmL6IbswQGhfH9AX7ezWd	101	1587134589
1	LIKE	124	2020-04-17T16:43:55.237000	5EmL6IbswQGhfH9AX7ezWd	101	1587134635
2	PLAY	124	2020-04-17T16:45:44.733000	67ov0nL5eR7zdx0JfXDqro	101	1587134744
3	SKIP	124	2020-04-17T16:48:26.836000	67ov0nL5eR7zdx0JfXDqro	101	1587134906
4	ADVERTISEMENT	124	2020-04-17T16:48:26.836000		101	1587134906
...	...	...	...	...	...	...
99995	SKIP	2796	2022-07-02T14:43:05.686000	50oXqDFyjbuGLzdfCwYWRu	301	1656765785
99996	PLAY	2796	2022-07-02T14:43:05.686000	3MT6rJBU7VUAPWtQsowIQv	301	1656765785
99997	SKIP	2796	2022-07-02T14:45:38.948000	3MT6rJBU7VUAPWtQsowIQv	301	1656765938
99998	PLAY	2796	2022-07-02T14:45:38.948000	04grddSQnpTQKkzeM6ri54	301	1656765938
99999	SKIP	2796	2022-07-02T14:48:54.523000	04grddSQnpTQKkzeM6ri54	301	1656766134

100000 rows × 6 columns

```
================================================================================
                                  TRACK_STORAGE
================================================================================
DAILY_COST - double
STORAGE_CLASS - string
TRACK_ID - string
```

	daily_cost	storage_class	track_id
0	0.003752	SLOW	708ZiYL3ydBWHS2a7gvJB3
1	0.014561	SLOW	48SFtLr5URCI97X2Ynfdnc
2	0.008304	SLOW	1y0U0HAe5QfTRzOsz74bOt
3	0.012207	SLOW	2TlbZ8JhF9ORa7lJylxABw
4	0.011799	SLOW	7ij5kN8jwXr8fZD54M0xb6
...	...	...	...
99995	0.012688	SLOW	3flurnTXJlSjMa9yj2uvY0
99996	0.010389	SLOW	6UjVlcCLMmwfyZfumUhsgN
99997	0.011977	SLOW	2OXAWAySnYPJHLvgLX5fFT
99998	0.008842	SLOW	1hQreq8n3jTwLWD1sjVb3t
99999	0.011849	SLOW	6DVY3lXlOgbu0iD5BhkWXj

100000 rows × 3 columns

```
================================================================================
                                      TRACKS
================================================================================
ACOUSTICNESS - double
DANCEABILITY - double
DURATION_MS - bigint
ENERGY - double
EXPLICIT - bigint
ID - string
ID_ARTIST - string
INSTRUMENTALNESS - double
KEY - bigint
LIVENESS - double
LOUDNESS - double
NAME - string
POPULARITY - bigint
RELEASE_DATE - string
SPEECHINESS - double
TEMPO - double
VALENCE - double
RELEASE_DATE_S - bigint
```

	acousticness	danceability	duration_ms	energy	explicit	id	id_artist	instrumentalness	key	liveness	loudness	name	popularity	release_date	speechiness	tempo	valence	release_date_s
0	0.8390	0.740	75040	0.8910	0	708ZiYL3ydBWHS2a7gvJB3	0PCtW4w0RN89andUBQ3TVv	0.000000	7	0.869	-7.480	031 - Der Schatz im Silbersee I - Teil 39	13	1968-09-11	0.8920	51.496	0.557	-41216400
1	0.6950	0.603	291227	0.5170	0	48SFtLr5URCI97X2Ynfdnc	2yTUYhIf8fxptTly3KLuJD	0.000003	6	0.744	-8.504	Par Avion (Live) (2014 - Remaster) - Live; 20...	0	2014	0.0235	96.181	0.327	1388530800
2	0.9530	0.313	166080	0.1160	0	1y0U0HAe5QfTRzOsz74bOt	338mC0yGyX0C9of8QMJ5hK	0.331000	0	0.161	-12.645	My Foolish Heart	25	1950-01-01	0.0319	74.071	0.255	-631155600
3	0.1670	0.958	244133	0.6350	0	2TlbZ8JhF9ORa7lJylxABw	5A4ExW2nMBFRy2JDoYUcUE	0.000000	11	0.362	-7.853	Kathysterisi	14	1998	0.2590	108.024	0.866	883609200
4	0.1200	0.684	235974	0.8390	0	7ij5kN8jwXr8fZD54M0xb6	48CUA59SDed3IdCctKndud	0.000000	4	0.354	-6.457	Aleni Aleni	51	2015	0.0658	128.051	0.580	1420066800
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
99995	0.4180	0.874	253755	0.6250	1	3flurnTXJlSjMa9yj2uvY0	2QDHxmDObOuv9MCeBYiFtq	0.000136	5	0.131	-8.277	Şampanya	60	2019-07-19	0.0656	117.094	0.461	1563487200
99996	0.7090	0.610	207771	0.5380	0	6UjVlcCLMmwfyZfumUhsgN	3iVIrcJmrV7GawrxVWsBUF	0.002230	7	0.302	-11.594	Başıma Gelenler	20	1978	0.0379	105.682	0.677	252457200
99997	0.0469	0.693	239533	0.9050	1	2OXAWAySnYPJHLvgLX5fFT	4oLZx5FplbgfM8DEe9U8LB	0.000000	0	0.268	-8.701	Luchini Aka This Is It	45	1990-01-01	0.3030	82.911	0.832	631148400
99998	0.9940	0.462	176842	0.0444	0	1hQreq8n3jTwLWD1sjVb3t	2e42axkOGHNvACKRN4MfDU	0.874000	7	0.148	-21.646	Agg Lagi	0	1946-01-01	0.0381	128.364	0.314	-757386000
99999	0.1030	0.737	236987	0.5750	1	6DVY3lXlOgbu0iD5BhkWXj	1cUNRt3Ha4lnnNvPTJAIa8	0.000016	11	0.655	-7.001	My Lady - P-Money Mix	36	2003-01-01	0.2010	82.549	0.671	1041375600

100000 rows × 18 columns

```
================================================================================
                                      USERS
================================================================================
CITY - string
FAVOURITE_GENRES - array<string>
NAME - string
PREMIUM_USER - int
STREET - string
USER_ID - bigint
```

	city	favourite_genres	name	premium_user	street	user_id
0	Warszawa	[motown, soul, regional mexican]	Marika Pilipczuk	1	ul. Księżycowa 31	101
1	Gdynia	[regional mexican, psychedelic rock, new roman...	Anita Pioch	0	plac Sadowa 527	102
2	Kraków	[soul, mellow gold, blues rock]	Jan Gryga	0	plac Wyspiańskiego 73/43	103
3	Wrocław	[permanent wave, post-teen pop, mandopop]	Ksawery Klus	1	ulica Długosza 71/06	104
4	Gdynia	[metal, new wave, argentine rock]	Maciej Bandyk	0	ul. Rybacka 07	105
...	...	...	...	...	...	...
19995	Warszawa	[latin rock, lounge, alternative metal]	Ernest Mikoda	0	plac Mieszka I 25/28	20096
19996	Szczecin	[new wave, soft rock, regional mexican]	Leonard Wrochna	1	ulica Tysiąclecia 25	20097
19997	Szczecin	[alternative rock, tropical, rock en espanol]	Kornel Ernst	0	plac Morska 87	20098
19998	Warszawa	[album rock, latin rock, dance pop]	Olga Miąsik	0	plac Opolska 61/80	20099
19999	Wrocław	[pop rock, latin, tropical]	Marcin Łosiak	1	aleja Lubelska 662	20100

20000 rows × 6 columns

In [ ]:

```python
for view, data_frame in DATA_FRAMES:
    show_table_name(view)
    for column, type in data_frame.dtypes:
        show_column_name(column)
        group_by_column = f"""--sql
            SELECT
                {column},
                COUNT(*) AS length
            FROM {view}
            GROUP BY {column}
            ORDER BY {column} IS NULL DESC, length DESC, {column} NULLS FIRST
        """
        df = spark.sql(group_by_column)
        display(df.limit(100_000).toPandas())

        count_distinct = f"""--sql
            SELECT
                COUNT(DISTINCT {column})
            FROM {view}
        """
        df = spark.sql(count_distinct)
        display(df.toPandas())
```

```
================================================================================
                                     ARTISTS
================================================================================
GENRES
```

	genres	length
0	[indonesian pop]	78
1	[classic thai pop]	74
2	[thai pop]	63
3	[classic turkish pop]	59
4	[classic israeli pop]	58
...	...	...
13577	[yiddish folk]	1
13578	[yoga]	1
13579	[yugoslav new wave]	1
13580	[zhongguo feng]	1
13581	[zolo]	1

13582 rows × 2 columns

	count(DISTINCT genres)
0	13582

```
ID
```

	id	length
0	0001wHqxbF2YYRQxGdbyER	1
1	000p4jMMhpEHq1h6PFCyO1	1
2	001aJOc7CSQVo3XzoLG4DK	1
3	0027wHZDQXpRll4ckwDGad	1
4	002oyMRzxTzEsBRLzACi8d	1
...	...	...
27519	7zup4xIPjtv50lM7x3n4qW	1
27520	7zw8gWmNncuk2QZHIc70So	1
27521	7zwF847GE2hY5ApGSOLmBG	1
27522	7zwiFdY90oXzLh1Wz22oEq	1
27523	7zzsdcNemyhcNk2wpNsXZt	1

27524 rows × 2 columns

	count(DISTINCT id)
0	27524

```
NAME
```

	name	length
0	TNT	4
1	Kali	3
2	Sebastian	3
3	Akcent	2
4	Alice	2
...	...	...
27411	黃韻玲	1
27412	黑豹	1
27413	龍飄飄	1
27414	龔秋霞	1
27415	龔詩嘉	1

27416 rows × 2 columns

	count(DISTINCT name)
0	27416

```
================================================================================
                                     SESSIONS
================================================================================
EVENT_TYPE
```

	event_type	length
0	PLAY	5618760
1	SKIP	1672489
2	LIKE	1612195
3	ADVERTISEMENT	1279933
4	BUY_PREMIUM	8385

	count(DISTINCT event_type)
0	5

```
SESSION_ID
```

	session_id	length
0	250589	107
1	230533	104
2	131400	102
3	148756	102
4	176182	102
...	...	...
99995	116075	46
99996	116203	46
99997	116280	46
99998	116504	46
99999	116635	46

100000 rows × 2 columns

	count(DISTINCT session_id)
0	249530

```
TIMESTAMP
```

	timestamp	length
0	2020-01-29T19:25:30.488000	4
1	2020-02-20T10:39:42.713000	4
2	2021-01-07T16:44:07.775000	4
3	2021-03-13T07:02:31.763000	4
4	2021-06-26T13:27:28.552000	4
...	...	...
99995	2020-01-23T14:58:21.213000	2
99996	2020-01-23T14:58:51.287000	2
99997	2020-01-23T14:59:45.772000	2
99998	2020-01-23T15:01:21.274000	2
99999	2020-01-23T15:02:38.246000	2

100000 rows × 2 columns

	count(DISTINCT timestamp)
0	8576422

```
TRACK_ID
```

	track_id	length
0		1288318
1	2RSHsoi04658QL5xgQVov3	37722
2	7lPN2DXiMsVn7XUKtOW1CS	37132
3	3ee8Jmje8o58CHK66QrVC2	37112
4	1daDRI9ahBonbWD8YcxOIB	37097
...	...	...
10704	6iS1qciFCYHM7vjY0pAKQC	276
10705	0Q2S7WezdxOedwVO2jYv7V	274
10706	301p9XBvsYen2aKNgSWfgE	273
10707	6p44R8rCmmpc2pSUVBqEpm	266
10708	45QyGXbqTWaFUrIKe2ugs3	263

10709 rows × 2 columns

	count(DISTINCT track_id)
0	10709

```
USER_ID
```

	user_id	length
0	7323	1259
1	4662	1238
2	2427	1216
3	2203	1209
4	12257	1201
...	...	...
19995	12413	80
19996	14391	74
19997	1693	72
19998	784	67
19999	1387	61

20000 rows × 2 columns

	count(DISTINCT user_id)
0	20000

```
TIMESTAMP_S
```

	timestamp_s	length
0	1607549836	8
1	1623648113	8
2	1602955210	7
3	1614202131	7
4	1622393793	7
...	...	...
99995	1655532051	3
99996	1655532670	3
99997	1655534404	3
99998	1655536617	3
99999	1655537447	3

100000 rows × 2 columns

	count(DISTINCT timestamp_s)
0	8232921

```
================================================================================
                                  TRACK_STORAGE
================================================================================
DAILY_COST
```

	daily_cost	length
0	0.009600	44
1	0.011700	41
2	0.008000	39
3	0.010000	39
4	0.010800	38
...	...	...
47433	0.229282	1
47434	0.236263	1
47435	0.239629	1
47436	0.239863	1
47437	0.249754	1

47438 rows × 2 columns

	count(DISTINCT daily_cost)
0	47438

```
STORAGE_CLASS
```

	storage_class	length
0	SLOW	128369
1	MEDIUM	1275
2	FAST	4

	count(DISTINCT storage_class)
0	3

```
TRACK_ID
```

	track_id	length
0	000jBcNljWTnyjB4YO7ojf	1
1	000u1dTg7y1XCDXi80hbBX	1
2	0017A6SJgTbfQVU2EtsPNo	1
3	001UI3J6PKAEnBgqrwGGQC	1
4	001gx41rQo0bKh063TrC1l	1
...	...	...
99995	5ye1yhnGkhvf4G5yDIP6fq	1
99996	5yeBQ7ll2Qi9Ez0ZBDCYgT	1
99997	5yeCt0MReP9i652S9l1fOa	1
99998	5yeXw1L7CqKXkHaJ0W4RrT	1
99999	5yeoAPpSg8eD4MRRojxtpY	1

100000 rows × 2 columns

	count(DISTINCT track_id)
0	129648

```
================================================================================
                                     TRACKS
================================================================================
ACOUSTICNESS
```

	acousticness	length
0	0.99500	525
1	0.99400	426
2	0.99300	355
3	0.99200	317
4	0.99100	312
...	...	...
4535	0.00853	1
4536	0.00868	1
4537	0.00926	1
4538	0.00960	1
4539	0.00986	1

4540 rows × 2 columns

	count(DISTINCT acousticness)
0	4540

```
DANCEABILITY
```

	danceability	length
0	0.629	359
1	0.565	350
2	0.549	348
3	0.652	348
4	0.611	345
...	...	...
1023	0.980	1
1024	0.982	1
1025	0.984	1
1026	0.985	1
1027	0.988	1

1028 rows × 2 columns

	count(DISTINCT danceability)
0	1028

```
DURATION_MS
```

	duration_ms	length
0	192000	44
1	234000	41
2	160000	39
3	200000	39
4	224000	39
...	...	...
46735	4585640	1
46736	4725264	1
46737	4792587	1
46738	4797258	1
46739	4995083	1

46740 rows × 2 columns

	count(DISTINCT duration_ms)
0	46740

```
ENERGY
```

	energy	length
0	0.5380	230
1	0.4990	227
2	0.6340	217
3	0.4840	212
4	0.7160	211
...	...	...
1873	0.0920	1
1874	0.0957	1
1875	0.0960	1
1876	0.0987	1
1877	0.0996	1

1878 rows × 2 columns

	count(DISTINCT energy)
0	1878

```
EXPLICIT
```

	explicit	length
0	0	124929
1	1	4719

	count(DISTINCT explicit)
0	2

```
ID
```

	id	length
0	000jBcNljWTnyjB4YO7ojf	1
1	000u1dTg7y1XCDXi80hbBX	1
2	0017A6SJgTbfQVU2EtsPNo	1
3	001UI3J6PKAEnBgqrwGGQC	1
4	001gx41rQo0bKh063TrC1I	1
...	...	...
99995	5ye1yhnGkhvf4G5yDIP6fq	1
99996	5yeBQ7Il2Qi9Ez0ZBDCYgT	1
99997	5yeCt0MReP9i652S9l1fOa	1
99998	5yeXw1L7CqKXkHaJ0W4RrT	1
99999	5yeoAPpSg8eD4MRRojxtpY	1

100000 rows × 2 columns

	count(DISTINCT id)
0	129648

```
ID_ARTIST
```

	id_artist	length
0	3meJIgRw7YleJrmbpbJK6S	1106
1	0i38tQX5j4gZ0KS3eCMoll	575
2	1l6d0RIxTL3JytlLGvWzYe	458
3	3t2iKODSDyzoDJw7AsD99u	453
4	61JrslREXq98hurYL2hYoc	435
...	...	...
27519	7zjX652bWyemXyFFVhBnch	1
27520	7zlWN2A8mV2thjdvAyMrEJ	1
27521	7zmk5lkmCMVvfvwF3H8FWC	1
27522	7zpw4vmlZNCUlwbdnFwxwO	1
27523	7zw8gWmNncuk2QZHIc70So	1

27524 rows × 2 columns

	count(DISTINCT id_artist)
0	27524

```
INSTRUMENTALNESS
```

	instrumentalness	length
0	0.000000	46190
1	0.000010	83
2	0.897000	74
3	0.000012	73
4	0.000104	72
...	...	...
5392	0.099100	1
5393	0.099900	1
5394	0.993000	1
5395	0.994000	1
5396	0.995000	1

5397 rows × 2 columns

	count(DISTINCT instrumentalness)
0	5397

```
KEY
```

	key	length
0	0	16686
1	7	16466
2	9	15219
3	2	15118
4	5	11655
5	4	11090
6	11	8781
7	1	8522
8	10	7921
9	8	7182
10	6	6607
11	3	4401

	count(DISTINCT key)
0	12

```
LIVENESS
```

	liveness	length
0	0.1110	1209
1	0.1080	1178
2	0.1100	1164
3	0.1070	1116
4	0.1090	1113
...	...	...
1735	0.0239	1
1736	0.0250	1
1737	0.0262	1
1738	0.0284	1
1739	0.9990	1

1740 rows × 2 columns

	count(DISTINCT liveness)
0	1740

```
LOUDNESS
```

	loudness	length
0	-8.026	36
1	-5.797	32
2	-7.679	28
3	-7.338	26
4	-12.502	25
...	...	...
20356	2.534	1
20357	2.639	1
20358	2.695	1
20359	3.273	1
20360	4.362	1

20361 rows × 2 columns

	count(DISTINCT loudness)
0	20361

```
NAME
```

	name	length
0	Hold On	42
1	Summertime	23
2	Home	21
3	99 Year Blues	20
4	Intro	19
...	...	...
99995	Xtabay - Alternate Version	1
99996	Xxplosive - Instrumental	1
99997	Xymeronei Pali - Live	1
99998	Xácara das mulheres amadas	1
99999	Xô Satanás	1

100000 rows × 2 columns

	count(DISTINCT name)
0	114159

```
POPULARITY
```

	popularity	length
0	0	4465
1	35	3066
2	36	3026
3	23	2995
4	34	2824
...	...	...
90	89	2
91	91	1
92	92	1
93	97	1
94	99	1

95 rows × 2 columns

	count(DISTINCT popularity)
0	95

```
RELEASE_DATE
```

	release_date	length
0	1998-01-01	750
1	1997-01-01	738
2	1998	720
3	1995	718
4	1996	692
...	...	...
14936	2021-03-23	1
14937	2021-03-27	1
14938	2021-03-28	1
14939	2021-04-03	1
14940	2021-04-04	1

14941 rows × 2 columns

	count(DISTINCT release_date)
0	14941

```
SPEECHINESS
```

	speechiness	length
0	0.0315	531
1	0.0312	514
2	0.0310	510
3	0.0308	502
4	0.0309	501
...	...	...
1632	0.8040	1
1633	0.8240	1
1634	0.8470	1
1635	0.9680	1
1636	0.9690	1

1637 rows × 2 columns

	count(DISTINCT speechiness)
0	1637

```
TEMPO
```

	tempo	length
0	0.000	48
1	139.980	29
2	119.996	22
3	127.997	22
4	130.022	22
...	...	...
70580	233.013	1
70581	236.134	1
70582	238.895	1
70583	239.906	1
70584	243.507	1

70585 rows × 2 columns

	count(DISTINCT tempo)
0	70585

```
VALENCE
```

	valence	length
0	0.9610	614
1	0.9620	536
2	0.9630	469
3	0.9640	445
4	0.9600	387
...	...	...
1623	0.0888	1
1624	0.0891	1
1625	0.0919	1
1626	0.0939	1
1627	0.0979	1

1628 rows × 2 columns

	count(DISTINCT valence)
0	1628

```
RELEASE_DATE_S
```

	release_date_s	length
0	883609200	1470
1	852073200	1418
2	820450800	1351
3	788914800	1349
4	631148400	1288
...	...	...
14678	1616454000	1
14679	1616799600	1
14680	1616886000	1
14681	1617400800	1
14682	1617487200	1

14683 rows × 2 columns

	count(DISTINCT release_date_s)
0	14683

```
================================================================================
                                     USERS
================================================================================
CITY
```

	city	length
0	Kraków	2924
1	Wrocław	2880
2	Gdynia	2864
3	Radom	2861
4	Warszawa	2847
5	Szczecin	2820
6	Poznań	2804

	count(DISTINCT city)
0	7

```
FAVOURITE_GENRES
```

	favourite_genres	length
0	[c-pop, lounge, rock en espanol]	4
1	[post-teen pop, mellow gold, regional mexican]	4
2	[adult standards, europop, mellow gold]	3
3	[adult standards, folk, hoerspiel]	3
4	[adult standards, latin rock, folk rock]	3
...	...	...
18544	[vocal jazz, vocal jazz, latin pop]	1
18545	[vocal jazz, vocal jazz, modern rock]	1
18546	[vocal jazz, vocal jazz, mpb]	1
18547	[vocal jazz, vocal jazz, permanent wave]	1
18548	[vocal jazz, vocal jazz, soft rock]	1

18549 rows × 2 columns

	count(DISTINCT favourite_genres)
0	18549

```
NAME
```

	name	length
0	Nataniel Duszkiewicz	4
1	Albert Smykała	3
2	Anita Pompa	3
3	Apolonia Bazylewicz	3
4	Aurelia Kuliberda	3
...	...	...
19612	Łukasz Węgrzyniak	1
19613	Łukasz Świętoń	1
19614	Łukasz Żbik	1
19615	Łukasz Żero	1
19616	Łukasz Żyto	1

19617 rows × 2 columns

	count(DISTINCT name)
0	19617

```
PREMIUM_USER
```

	premium_user	length
0	0	11615
1	1	8385

	count(DISTINCT premium_user)
0	2

```
STREET
```

	street	length
0	ulica Jagodowa 15	3
1	al. Boczna 88	2
2	al. Daleka 25	2
3	al. Daleka 64	2
4	al. Jarzębinowa 25	2
...	...	...
19906	ulica Żytnia 312	1
19907	ulica Żytnia 44/76	1
19908	ulica Żytnia 55/39	1
19909	ulica Żytnia 721	1
19910	ulica Żytnia 928	1

19911 rows × 2 columns

	count(DISTINCT street)
0	19911

```
USER_ID
```

	user_id	length
0	101	1
1	102	1
2	103	1
3	104	1
4	105	1
...	...	...
19995	20096	1
19996	20097	1
19997	20098	1
19998	20099	1
19999	20100	1

20000 rows × 2 columns

	count(DISTINCT user_id)
0	20000

In [ ]:

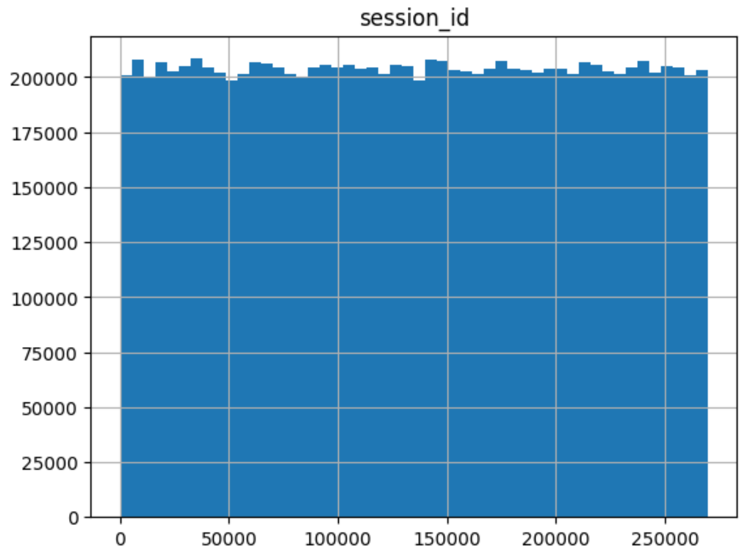
```
def aggregate_numeric_column(view: str, column: str) -> str:
    return f"""--sql
            SELECT
                "{column}" AS name,
                COUNT({column}) AS count,
                MIN({column}) AS min,
                MAX({column}) AS max,
                AVG({column}) AS average,
                SUM({column}) AS sum,
                SUM(DISTINCT {column}) AS sum_distinct,
                KURTOSIS({column}) AS kurtosis,
                SKEWNESS({column}) AS skewness,
                STDDEV({column}) AS standard_deviation,
                STDDEV_POP({column}) AS population_standard_deviation,
                VARIANCE({column}) AS variance,
                VAR_POP({column}) AS population_variance
            FROM {view}
            WHERE {column} IS NOT NULL
        """


for view, data_frame in DATA_FRAMES:
    show_table_name(view)
    for column, type in data_frame.dtypes:
        if type in ['double', 'bigint']:
            show_column_name(column)
            df = spark.sql(aggregate_numeric_column(view, column))
            display(df.toPandas())

            dfp = spark.sql(f"SELECT {column} FROM {view}").toPandas()
            dfp.hist(bins=50)
            plt.show()
```

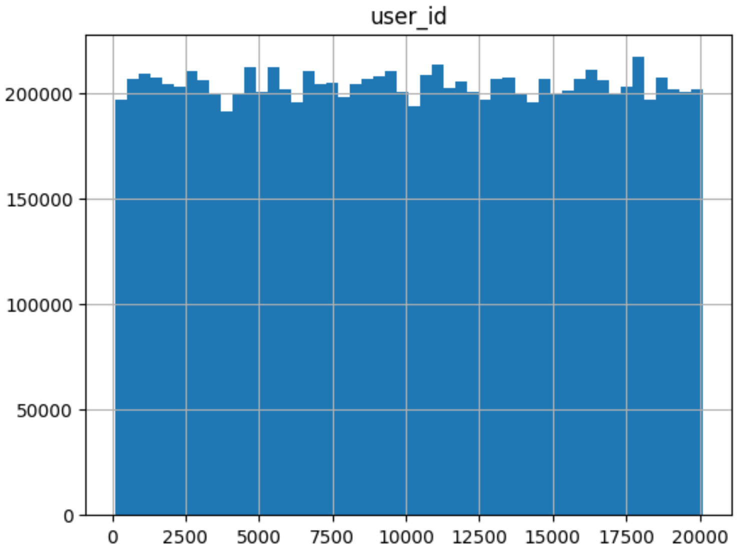
```
================================================================================
                                    ARTISTS
================================================================================
================================================================================
                                    SESSIONS
================================================================================
SESSION_ID
```

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	session_id	10191762	124	269652	134862.941757	1374491005008	33658388722	-1.199749	-0.000136	77790.366875	77790.363059	6.051341e+09	6.051341e+09



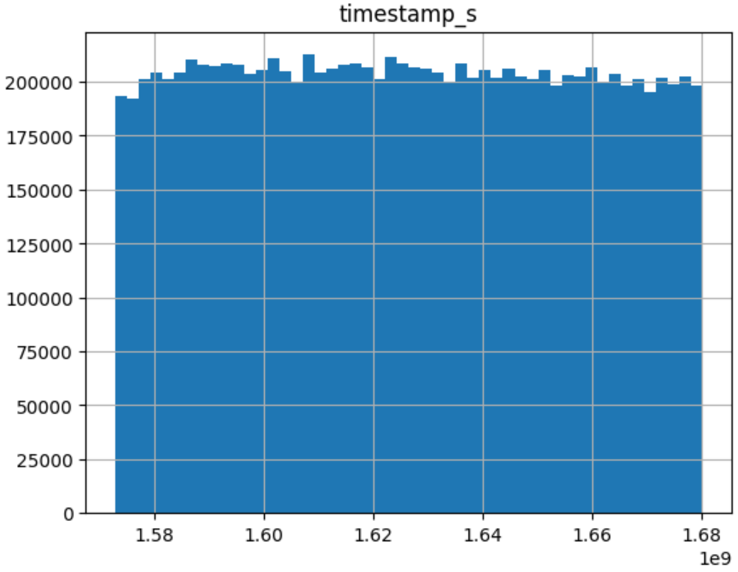
```
USER_ID
```

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	user_id	10191762	101	20100	10097.862532	102915011633	202010000	-1.200667	0.000433	5773.002178	5773.001894	3.332755e+07	3.332755e+07



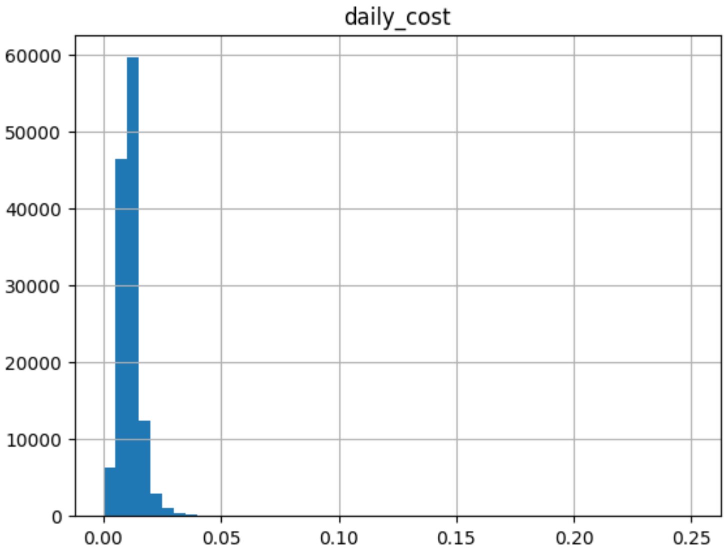
```
TIMESTAMP_S
```

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	timestamp_s	10191762	1572822218	1680270885	1.626383e+09	16575712200199489	13388733788041283	-1.192379	0.012246	3.085547e+07	3.085547e+07	9.520598e+14	9.520597e+14



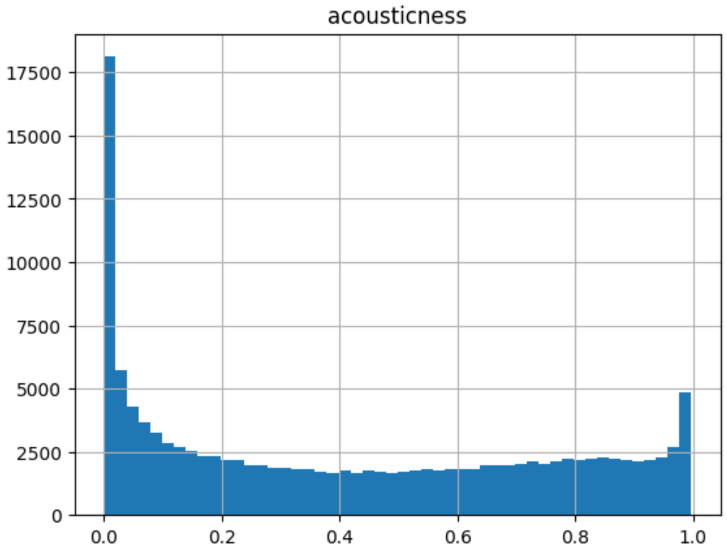
```
================================================================================
                                 TRACK_STORAGE
================================================================================
DAILY_COST
```

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	daily_cost	129648	0.000167	0.249754	0.011535	1495.508148	591.933795	259.234276	10.35695	0.005815	0.005815	0.000034	0.000034



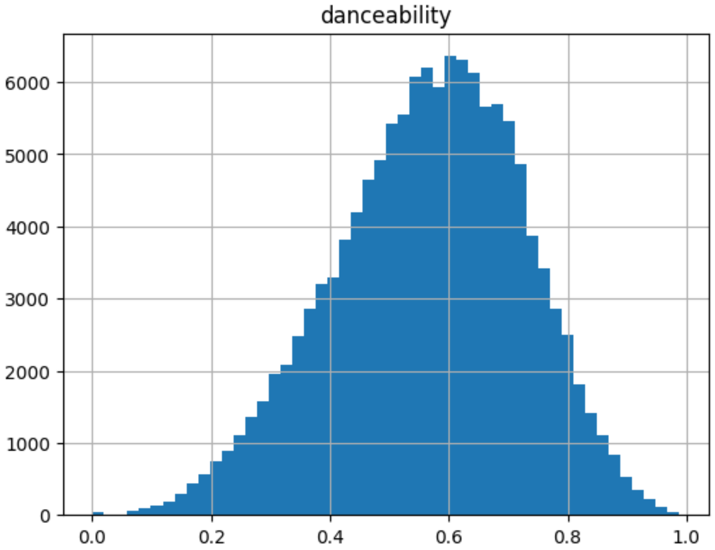
```
================================================================================
                                     TRACKS
================================================================================
ACOUSTICNESS
```

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	acousticness	129648	0.0	0.996	0.41755	54134.576468	546.440307	-1.383039	0.250805	0.335652	0.335651	0.112662	0.112661



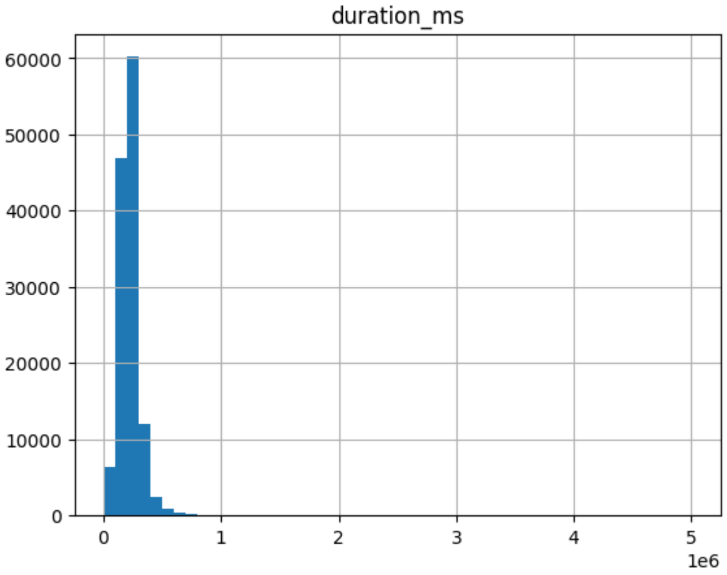
```
DANCEABILITY
```

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	danceability	129648	0.0	0.988	0.564894	73237.4093	491.2168	-0.258259	-0.28432	0.159114	0.159113	0.025317	0.025317



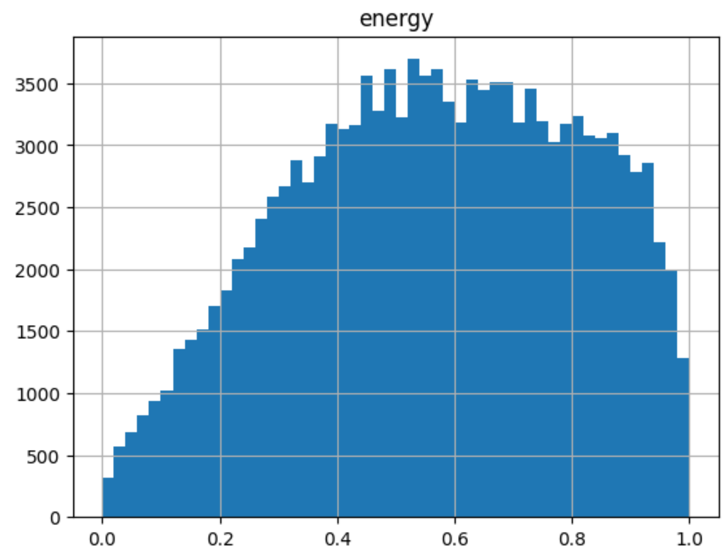
```
DURATION_MS
```

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	duration_ms	129648	3344	4995083	228526.632274	29628020821	11430854470	281.491889	10.884919	113801.507474	113801.068587	1.295078e+10	1.295068e+10



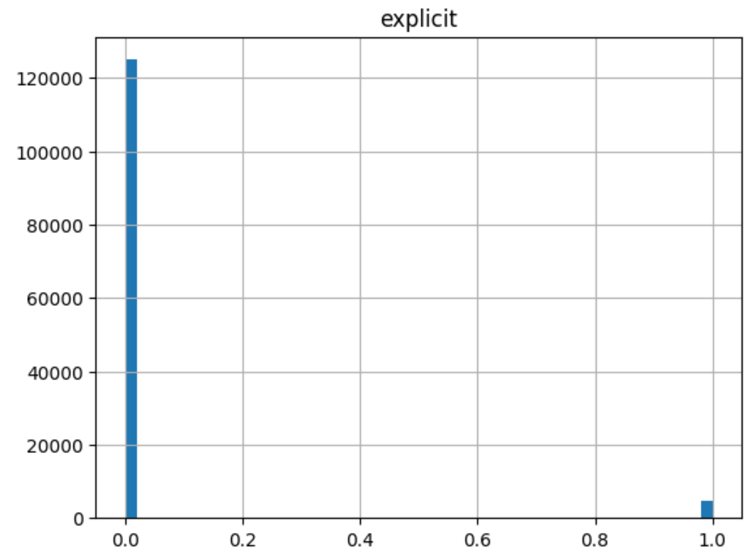
```
ENERGY
```

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	energy	129648	0.0	1.0	0.562776	72962.72439	543.752618	-0.899073	-0.168391	0.241957	0.241956	0.058543	0.058543



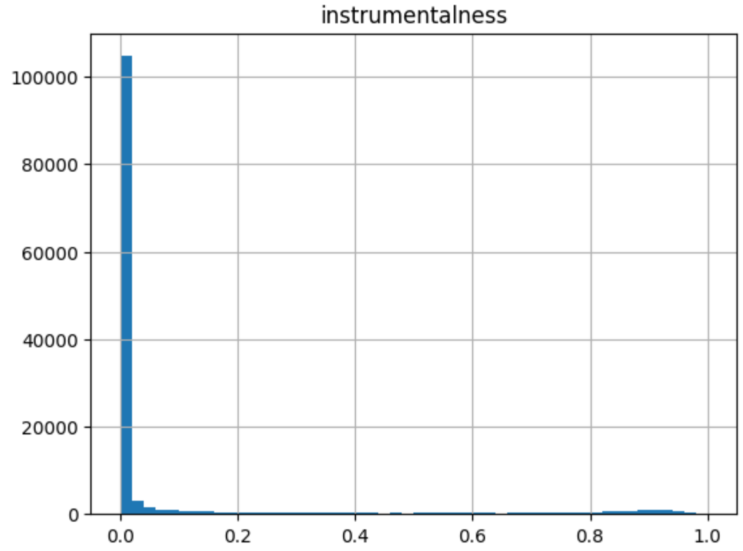
EXPLICIT

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	explicit	129648	0	1	0.036399	4719	1	22.511391	4.950898	0.18728	0.18728	0.035074	0.035074



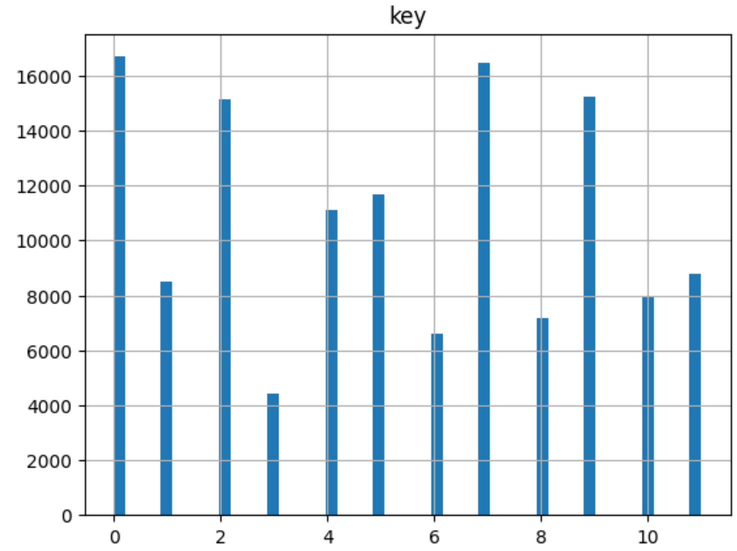
INSTRUMENTALNESS

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	instrumentalness	129648	0.0	1.0	0.086754	11247.463381	549.236231	6.200105	2.759591	0.232285	0.232284	0.053956	0.053956



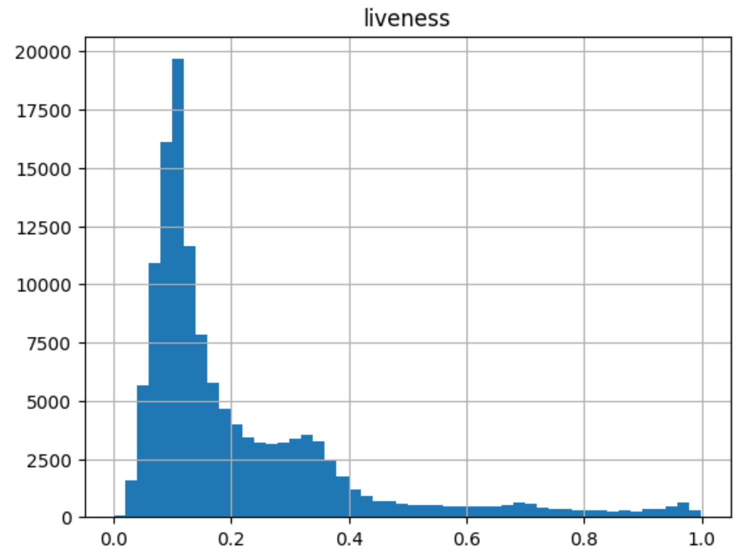
```
KEY
```

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	key	129648	0	11	5.242873	679728	66	-1.265013	-0.011349	3.518889	3.518876	12.382581	12.382485



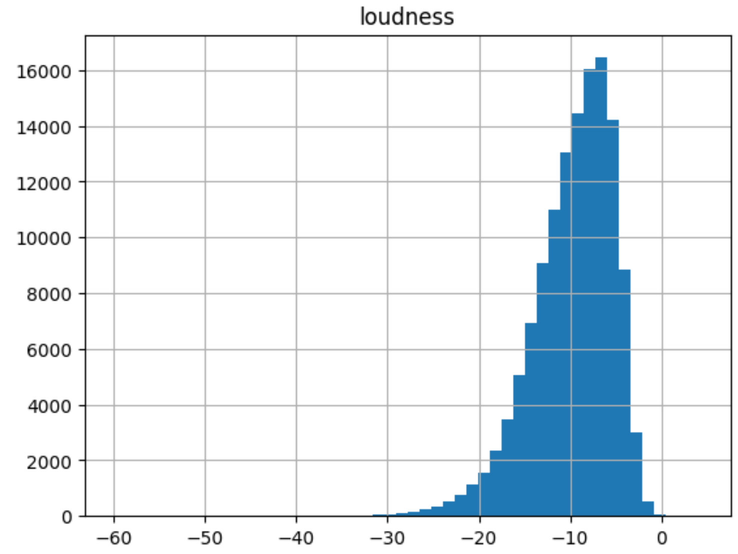
```
LIVENESS
```

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	liveness	129648	0.0	0.999	0.21406	27752.50933	543.09323	4.380976	2.072202	0.186901	0.1869	0.034932	0.034932



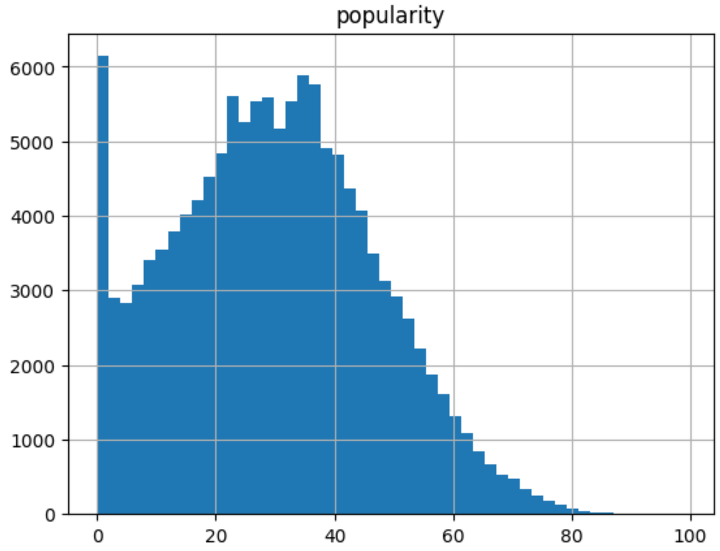
LOUDNESS

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	loudness	129648	-60.0	4.362	-9.734177	-1262016.64	-252312.279	2.778514	-1.104693	4.5213	4.521283	20.442158	20.442



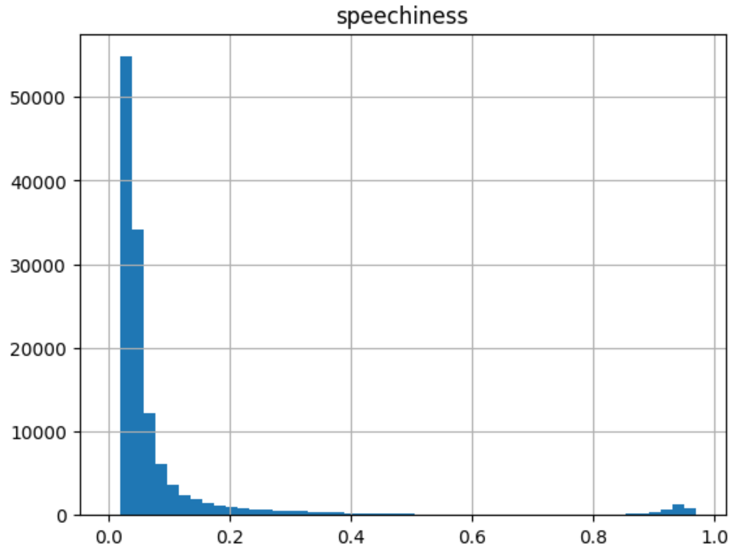
POPULARITY

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	popularity	129648	0	99	29.671241	3846817	4474	-0.484103	0.223677	17.1278	17.127734	293.361545	293.359283



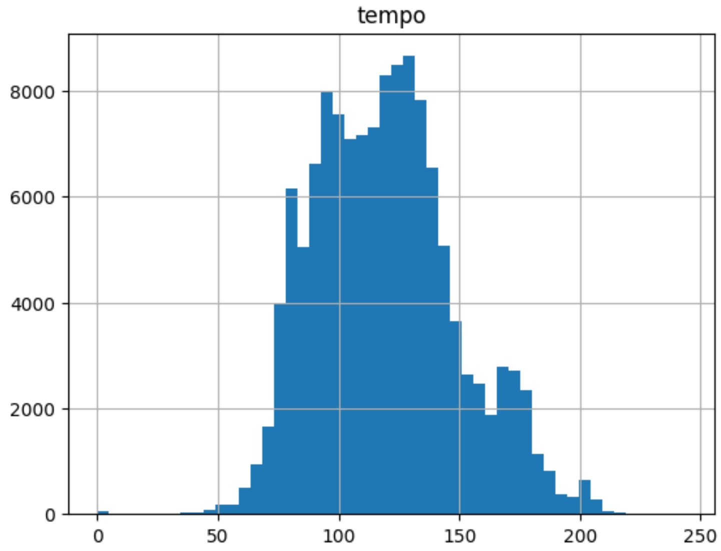
SPEECHINESS

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	speechiness	129648	0.0	0.969	0.095068	12325.3914	503.1898	16.456687	4.045176	0.166167	0.166166	0.027611	0.027611



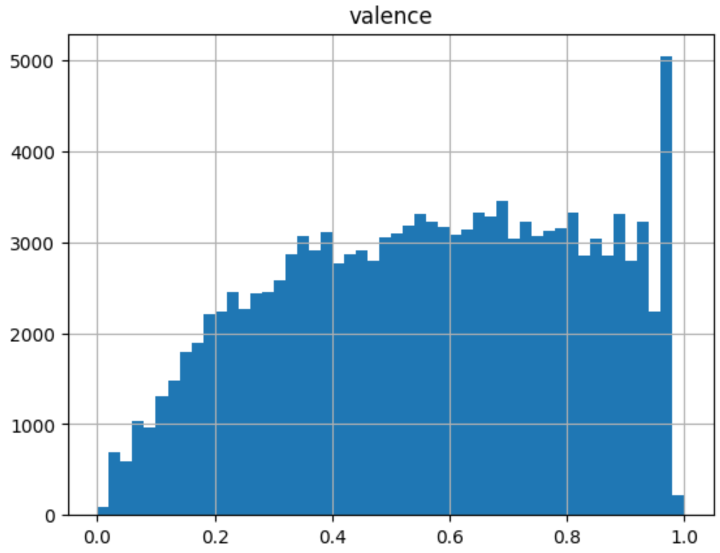
TEMPO

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	tempo	129648	0.0	243.507	119.53864	1.549795e+07	8607442.191	-0.106043	0.402869	29.653393	29.653278	879.323707	879.316925



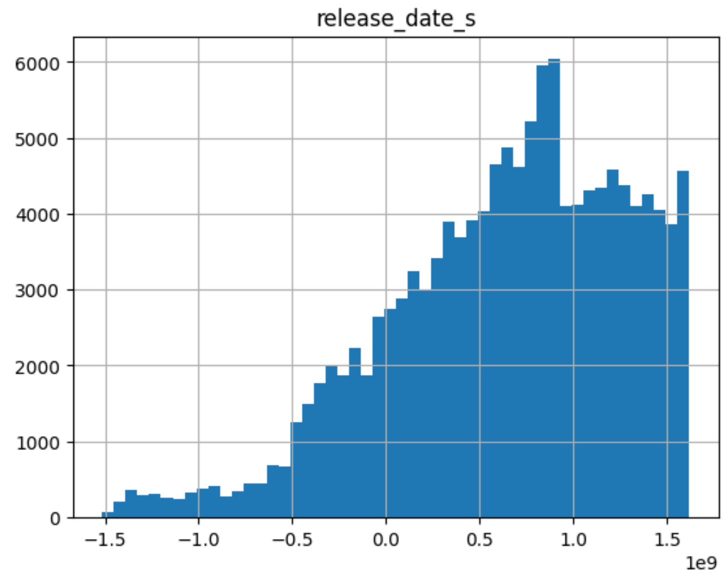
VALENCE

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	valence	129648	0.0	1.0	0.563443	73049.2694	537.05768	-1.035815	-0.154964	0.252581	0.25258	0.063797	0.063796



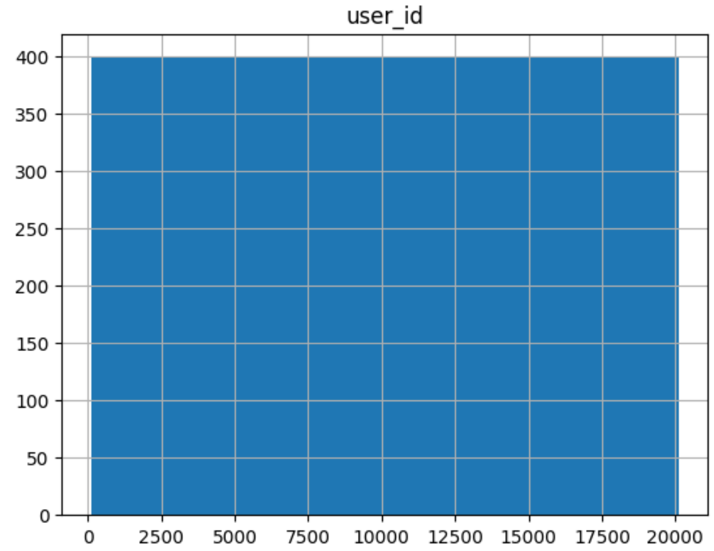
RELEASE_DATE_S

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	release_date_s	129648	-1514772000	1618524000	6.407151e+08	83067436238400	10982866910400	0.075787	-0.656014	6.358551e+08	6.358526e+08	4.043117e+17	4.043086e+17



```
================================================================================
                                     USERS
================================================================================
USER_ID
```

	name	count	min	max	average	sum	sum_distinct	kurtosis	skewness	standard_deviation	population_standard_deviation	variance	population_variance
0	user_id	20000	101	20100	10100.5	202010000	202010000	-1.2	5.753064e-17	5773.647028	5773.502685	33335000.0	33333333.25



In [ ]:

```python
def explode_column(view: str, column: str) -> str:
    return f"""--sql
            SELECT
                DISTINCT EXPLODE({column}) AS distinct_{column}
            FROM {view}
            ORDER BY distinct_{column} NULLS FIRST
        """


def count_exploded_column(view: str, column: str) -> str:
    exploded = f"""--sql
        SELECT
            DISTINCT EXPLODE({column}) AS {column}
        FROM {view}
    """

    return f"""--sql
            SELECT
                COUNT(*) AS length
            FROM ({exploded})
        """

for view, data_frame in DATA_FRAMES:
    show_table_name(view)
    for column, type in data_frame.dtypes:
        if type.startswith('array'):
            show_column_name(column)
            df = spark.sql(explode_column(view, column))
            display(df.toPandas())
            df = spark.sql(count_exploded_column(view, column))
            display(df.toPandas())
```

```
================================================================================
                                     ARTISTS
================================================================================
GENRES
```

	distinct_genres
0	48g
1	a cappella
2	abstract
3	abstract hip hop
4	accordeon
...	...
3907	zolo
3908	zouglou
3909	zouk
3910	zouk riddim
3911	zydeco

3912 rows × 1 columns

	length
0	3912

```
================================================================================
                                     SESSIONS
================================================================================
================================================================================
                                  TRACK_STORAGE
================================================================================
================================================================================
                                      TRACKS
================================================================================
================================================================================
                                      USERS
================================================================================
FAVOURITE_GENRES
```

	distinct_favourite_genres
0	adult standards
1	album rock
2	alternative metal
3	alternative rock
4	argentine rock
5	art rock
6	blues rock
7	brill building pop
8	c-pop
9	classic rock
10	country rock
11	dance pop
12	europop
13	folk
14	folk rock
15	funk
16	hard rock
17	hoerspiel
18	italian adult pop
19	j-pop
20	latin
21	latin alternative
22	latin pop
23	latin rock
24	lounge
25	mandopop
26	mellow gold
27	metal
28	modern rock
29	motown
30	mpb
31	new romantic
32	new wave
33	new wave pop
34	permanent wave
35	pop
36	pop rock
37	post-teen pop
38	psychedelic rock
39	quiet storm
40	ranchera
41	regional mexican

	distinct_favourite_genres
42	rock
43	rock en espanol
44	roots rock
45	singer-songwriter
46	soft rock
47	soul
48	tropical
49	vocal jazz

	length
0	50

In [ ]:

```
JOINS = {
    ('artists', 'tracks') : ('id', 'id_artist'),
    ('tracks', 'track_storage') : ('id', 'track_id'),
    ('tracks', 'sessions') : ('id', 'track_id'),
    ('users', 'sessions') : ('user_id', 'user_id'),
}
```

In [ ]:

```
def count_everything(table: str) -> str:
    return f"""--sql
        SELECT
            COUNT(*) AS length_{table}
        FROM {table}
    """

def count_joined(tables: Tuple[str, str], ids: Tuple[str, str]) -> str:
    return f"""--sql
        SELECT
            COUNT(*) AS length_{tables[0]}_{tables[1]}
        FROM {tables[0]} AS first
        INNER JOIN {tables[1]} AS second ON first.{ids[0]} == second.{ids[1]}
    """

def count_joined_distinct(tables: Tuple[str, str], ids: Tuple[str, str]) -> str:
    return f"""--sql
        SELECT
            COUNT(DISTINCT first.{ids[0]}) AS length_{tables[0]}_{tables[1]}_distinct
        FROM {tables[0]} AS first
        INNER JOIN {tables[1]} AS second ON first.{ids[0]} == second.{ids[1]}
    """

for tables, ids in JOINS.items():
    print(tables[0].upper(), '-', tables[1].upper())
    df = spark.sql(count_everything(tables[0]))
    display(df.toPandas())
    df = spark.sql(count_everything(tables[1]))
    display(df.toPandas())
    df = spark.sql(count_joined(tables, ids))
    display(df.toPandas())
    df = spark.sql(count_joined_distinct(tables, ids))
    display(df.toPandas())
```

```
ARTISTS - TRACKS
```

	length_artists
0	27524

	length_tracks
0	129648

	length_artists_tracks
0	129648

	length_artists_tracks_distinct
0	27524

```
TRACKS - TRACK_STORAGE
```

	length_tracks
0	129648

	length_track_storage
0	129648

	length_tracks_track_storage
0	129648

	length_tracks_track_storage_distinct
0	129648

```
TRACKS - SESSIONS
```

	length_tracks
0	129648

	length_sessions
0	10191762

	length_tracks_sessions
0	8903444

	length_tracks_sessions_distinct
0	10708

```
USERS - SESSIONS
```

	length_users
0	20000

	length_sessions
0	10191762

	length_users_sessions
0	10191762

	length_users_sessions_distinct
0	20000

```
In [ ]: def select_unknown(tables: Tuple[str, str], ids: Tuple[str, str]) -> str:
            spark.sql(f'SELECT DISTINCT {ids[1]} AS id FROM {tables[1]}') \
                .createOrReplaceTempView('temporary')

            return f"""--sql
                SELECT
                    *
                FROM {tables[0]}
                WHERE {ids[0]} NOT IN (SELECT id FROM temporary)
            """

        for tables, ids in JOINS.items():
            print(tables[0].upper(), '-', tables[1].upper())
            df = spark.sql(select_unknown(tables, ids))
            display(df.toPandas())
            df = spark.sql(select_unknown(tables[::-1], ids[::-1]))
            display(df.toPandas())
```

```
ARTISTS - TRACKS
```

	genres	id	name

	acousticness	danceability	duration_ms	energy	explicit	id	id_artist	instrumentalness	key	liveness	loudness	name	popularity	release_date	speechiness	tempo	valence	release_date_s

```
TRACKS - TRACK_STORAGE
```

	acousticness	danceability	duration_ms	energy	explicit	id	id_artist	instrumentalness	key	liveness	loudness	name	popularity	release_date	speechiness	tempo	valence	release_date_s

	daily_cost	storage_class	track_id

```
TRACKS - SESSIONS
```

	acousticness	danceability	duration_ms	energy	explicit	id	id_artist	instrumentalness	key	liveness	loudness	name	popularity	release_date	speechiness	tempo	valence	release_date_s
0	0.8390	0.740	75040	0.891	0	708ZiYL3ydBWHS2a7gvJB3	0PCtW4w0RN89andUBQ3TVv	0.000000	7	0.8690	-7.480	031 - Der Schatz im Silbersee I - Teil 39	13	1968-09-11	0.8920	51.496	0.557	-41216400
1	0.6950	0.603	291227	0.517	0	48SFtLr5URCI97X2Ynfdnc	2yTUYhIf8fxptTly3KLuJD	0.000003	6	0.7440	-8.504	Par Avion (Live) (2014 - Remaster) - Live; 20...	0	2014	0.0235	96.181	0.327	1388530800
2	0.9530	0.313	166080	0.116	0	1y0U0HAe5QfTRzOsz74bOt	338mC0yGyX0C9of8QMJ5hK	0.331000	0	0.1610	-12.645	My Foolish Heart	25	1950-01-01	0.0319	74.071	0.255	-631155600
3	0.1670	0.958	244133	0.635	0	2TlbZ8JhF9ORa7lJylxABw	5A4ExW2nMBFRy2JDoYUcUE	0.000000	11	0.3620	-7.853	Kathysterisi	14	1998	0.2590	108.024	0.866	883609200
4	0.1200	0.684	235974	0.839	0	7ij5kN8jwXr8fZD54M0xb6	48CUA59SDed3IdCctKndud	0.000000	4	0.3540	-6.457	Aleni Aleni	51	2015	0.0658	128.051	0.580	1420066800
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
118935	0.4110	0.633	214773	0.345	0	59nszNIEDpnOS0prsKudPb	6wcIBaOvA9XNGgPujYZZ7L	0.000028	4	0.3610	-15.231	最真的夢	16	1990-02-05	0.0291	132.691	0.368	634172400
118936	0.2220	0.295	213667	0.417	0	0xiHNGGiSfrFfOJZGpxpJY	04u3fc37nHFKN7GJTSIwI8	0.000006	6	0.1480	-8.002	By My Side	61	2017-08-11	0.0307	64.687	0.135	1502402400
118937	0.6720	0.347	208467	0.216	0	4peXvhLT61oP9leXdPQ36B	4etuCZVdP8yiNPn4xf0ie5	0.000118	8	0.0738	-15.215	Cu Cu Rru Cu Cu Paloma	49	1978	0.0315	108.566	0.478	252457200
118938	0.0229	0.784	214827	0.821	0	2pS2IdtMXpvaEONreUlSAo	6IE6z7DcZIT4MI3Fh5Ivch	0.000007	0	0.1760	-7.621	No Quiero Saber - 2000 Mix	26	1990	0.0423	119.609	0.885	631148400
118939	0.7200	0.701	139691	0.715	0	5m5g55OSy0kQnaxKU4lZ11	7FsRH5bw8iWpSbMX1G7xf1	0.000000	9	0.2970	-5.876	Ojitos De Golondrina	52	1991-12-19	0.0305	104.061	0.970	693097200

118940 rows × 18 columns

	event_type	session_id	timestamp	track_id	user_id	timestamp_s
0	ADVERTISEMENT	124	2020-04-17T16:48:26.836000		101	1587134906
1	ADVERTISEMENT	124	2020-04-17T16:55:35.031000		101	1587135335
2	ADVERTISEMENT	124	2020-04-17T17:13:11.269000		101	1587136391
3	ADVERTISEMENT	124	2020-04-17T17:16:39.747000		101	1587136599
4	ADVERTISEMENT	124	2020-04-17T17:28:35.461000		101	1587137315
...	...	...	...	...	...	...
1288313	ADVERTISEMENT	269649	2021-09-06T17:13:18.086000		20100	1630941198
1288314	ADVERTISEMENT	269649	2021-09-06T17:19:25.038000		20100	1630941565
1288315	ADVERTISEMENT	269649	2021-09-06T17:22:12.632000		20100	1630941732
1288316	ADVERTISEMENT	269649	2021-09-06T17:24:52.352000		20100	1630941892
1288317	BUY_PREMIUM	269649	2021-09-06T17:25:17.352000		20100	1630941917

1288318 rows × 6 columns

```
USERS - SESSIONS
```

city	favourite_genres	name	premium_user	street	user_id

event_type	session_id	timestamp	track_id	user_id	timestamp_s

```
In [ ]: premium_user_comparison = f"""--sql
            SELECT
                COUNT_IF(premium_user == 1) AS premium_users,
                COUNT_IF(premium_user == 0) AS non_premium_users,
                COUNT_IF(premium_user == 0) / COUNT(*) * 100 AS non_premium_users_percentage,
                COUNT_IF(premium_user == 1) / COUNT(*) * 100 AS premium_users_percentage
            FROM users
        """
        df = spark.sql(premium_user_comparison)
        display(df.toPandas())
```

	premium_users	non_premium_users	non_premium_users_percentage	premium_users_percentage
0	8385	11615	58.075	41.925