

# 4\_pandas

October 17, 2023

## 1 Data Wrangling with Pandas

- A major part of computational social science is the storing, manipulation and reporting of data.
- Pandas is a powerful data management library specifically built for these kinds of tasks.
- It can handle very large amounts of data whilst remaining quick and responsive.
- We will be using Pandas throughout our practical sessions as a general purpose data management tool but this week we will focus on learning its features.

### Pandas Documentation

The first thing we need to do is `import` the pandas library. This ensures it is available for us to use in this environment.

Here we import the `pandas` module. We could simply use `import pandas` however `as` allows us to use a shorter name.

As social convention many modules are referred to with these short names.

```
[ ]: import pandas as pd
```

### 1.1 Loading the Data

Today we will be using data gathered from Spotify, the popular music streaming service. Spotify provides access to some of its data through their public API. This data has been collected and pre-prepared by your instructors.

```
[ ]: filename = 'spotify_top_songs.csv'
songs_df = pd.read_csv(filename)
type(songs_df)
```

```
[ ]: pandas.core.frame.DataFrame
```

```
[ ]: songs_df
```

```
[ ]:
```

	track_id	track_name	artists \
0	5mjYQaktjmjcMKcUIcqz4s	Strangers	Kenya Grace
1	56y1j0TK0XSvJzVv9vHQBK	Paint The Town Red	Doja Cat
2	1reEeZH9wNt4z1ePYLyC7p	greedy	Tate McRae

3	59NraMJsLaMCVtwXTSia8i	Prada	cassö
4	2FDTHlrBguDzQkp7PVj16Q	Sprinter	Dave
...	...	...	...
1275	07GtD0Cxmye5KDwsTSACPk	Chantilly Lace	The Big Bopper
1276	3SQhmctWrenMOX6Zkm2K5R	Maybellene	Chuck Berry
1277	7ycFNferNuk6wAjBa0vWv1	Little Darlin'	The Diamonds
1278	1xV0ttVNT27FBTD8iHj0fU	It's Only Make Believe	Conway Twitty
1279	3nUrhP3KuK4R1qdxRk2Kgo	Stupid Cupid	Connie Francis

	genre	release_year	explicit	popularity	duration_ms	\
0	singer-songwriter pop	2023	False	97	172964	
1	dance pop	2023	True	87	230480	
2	alt z	2023	True	56	131872	
3	***OOPS!***	2023	True	94	132359	
4	uk hip hop	2023	True	94	229133	
...	...	...	...	...	...	
1275	doo-wop	1958	False	57	145266	
1276	blues	1959	False	57	143240	
1277	rhythm and blues	1996	False	57	129333	
1278	arkansas country	1959	False	57	132026	
1279	adult standards	2005	False	57	133746	

	playlist_name	danceability	loudness	speechiness	\
0	Top 50 - United Kingdom	0.628	-8.307	NaN	
1	Top 50 - United Kingdom	0.864	-7.683	0.1940	
2	Top 50 - United Kingdom	0.750	-3.190	0.0322	
3	Top 50 - United Kingdom	0.638	-5.804	0.0375	
4	Top 50 - United Kingdom	0.916	-8.067	0.2410	
...	...	...	...	...	
1275	All Out 50s	0.489	-6.054	0.0858	
1276	All Out 50s	0.756	-10.701	0.1120	
1277	All Out 50s	0.631	-11.402	0.0420	
1278	All Out 50s	0.461	-9.627	0.0598	
1279	All Out 50s	0.609	-4.739	0.0389	

	playlist_type
0	mixed_pop
1	mixed_pop
2	mixed_pop
3	mixed_pop
4	mixed_pop
...	...
1275	all_out_decades
1276	all_out_decades
1277	all_out_decades
1278	all_out_decades
1279	all_out_decades

[1280 rows x 13 columns]

```
[ ]: # .head() shows us the top 5 rows
songs_df.head()
```

```
[ ]:
      track_id      track_name  artists \
0  5mjYQaktjmcMKcUIcqz4s      Strangers  Kenya Grace
1  56y1j0TKOXsvJzVv9vHQBK  Paint The Town Red      Doja Cat
2  1reEeZH9wNt4z1ePYLyC7p      greedy      Tate McRae
3  59NraMJsLaMCVtwXTSia8i      Prada      cassö
4  2FDTHlrBguDzQkp7PVj16Q      Sprinter      Dave

      genre  release_year  explicit  popularity  duration_ms \
0  singer-songwriter pop      2023      False      97      172964
1      dance pop      2023      True      87      230480
2      alt z      2023      True      56      131872
3  ***OOPS!***      2023      True      94      132359
4      uk hip hop      2023      True      94      229133

      playlist_name  danceability  loudness  speechiness  playlist_type
0  Top 50 - United Kingdom      0.628  -8.307      NaN      mixed_pop
1  Top 50 - United Kingdom      0.864  -7.683      0.1940      mixed_pop
2  Top 50 - United Kingdom      0.750  -3.190      0.0322      mixed_pop
3  Top 50 - United Kingdom      0.638  -5.804      0.0375      mixed_pop
4  Top 50 - United Kingdom      0.916  -8.067      0.2410      mixed_pop
```

```
[ ]: # .tail() shows us the last 5 rows
songs_df.tail()
```

```
[ ]:
      track_id      track_name  artists \
1275  07GtDOCxmye5KDWsTSACPk      Chantilly Lace  The Big Bopper
1276  3SQhmctWreNMOX6Zkm2K5R      Maybellene      Chuck Berry
1277  7ycFNferNuk6wAjBa0vWv1      Little Darlin'  The Diamonds
1278  1xV0ttVNT27FBTD8iHj0fU  It's Only Make Believe  Conway Twitty
1279  3nUrhP3KuK4R1qdxRk2Kgo      Stupid Cupid  Connie Francis

      genre  release_year  explicit  popularity  duration_ms \
1275  doo-wop      1958      False      57      145266
1276  blues      1959      False      57      143240
1277  rhythm and blues      1996      False      57      129333
1278  arkansas country      1959      False      57      132026
1279  adult standards      2005      False      57      133746

      playlist_name  danceability  loudness  speechiness  playlist_type
1275  All Out 50s      0.489  -6.054      0.0858  all_out_decades
```

1276	All Out 50s	0.756	-10.701	0.1120	all_out_decades
1277	All Out 50s	0.631	-11.402	0.0420	all_out_decades
1278	All Out 50s	0.461	-9.627	0.0598	all_out_decades
1279	All Out 50s	0.609	-4.739	0.0389	all_out_decades

```
[ ]: # You can specify the number of rows to return
```

```
songs_df.head(10)
```

```
[ ]:
      track_id      track_name \
0  5mjYQaktjmcMKcUIcqz4s      Strangers
1  56y1j0TK0XSvJzVv9vHQBK  Paint The Town Red
2  1reEeZH9wNt4z1ePYLyC7p      greedy
3  59NraMJsLaMCVtwXTSia8i      Prada
4  2FDTHlrBguDzQkp7PVj16Q      Sprinter
5  1BxfuPKGuaTgP7aMOBbdwr      Cruel Summer
6  5aIVCx5tnk0ntmdiinnYvw      Water
7  1kuGVB7EU95pJ0bxwvfwKS      vampire
8  3vkCue0mm7xQDoJ17W1Pm3      My Love Mine All Mine
9  2ZWmmrWUGDBcPSLihBMvhg  Baddadan (feat. IRAH, Flowdan, Trigga & Takura)
```

	artists	genre	release_year	explicit	popularity	\
0	Kenya Grace	singer-songwriter pop	2023	False	97	
1	Doja Cat	dance pop	2023	True	87	
2	Tate McRae	alt z	2023	True	56	
3	cassö	***OOPS!***	2023	True	94	
4	Dave	uk hip hop	2023	True	94	
5	Taylor Swift	pop	2019	False	99	
6	Tyla	***OOPS!***	2023	False	91	
7	Olivia Rodrigo	pop	2023	True	95	
8	Mitski	brooklyn indie	2023	False	93	
9	Chase & Status	dancefloor dnb	2023	False	81	

	duration_ms	playlist_name	danceability	loudness	speechiness	\
0	172964	Top 50 - United Kingdom	0.628	-8.307	NaN	
1	230480	Top 50 - United Kingdom	0.864	-7.683	0.1940	
2	131872	Top 50 - United Kingdom	0.750	-3.190	0.0322	
3	132359	Top 50 - United Kingdom	0.638	-5.804	0.0375	
4	229133	Top 50 - United Kingdom	0.916	-8.067	0.2410	
5	178426	Top 50 - United Kingdom	0.552	-5.707	0.1570	
6	200255	Top 50 - United Kingdom	0.673	-3.495	0.0755	
7	219724	Top 50 - United Kingdom	0.511	-5.745	0.0578	
8	137773	Top 50 - United Kingdom	0.504	-14.958	0.0321	
9	177291	Top 50 - United Kingdom	0.620	-0.504	0.2520	

```
playlist_type
0      mixed_pop
```

```

1    mixed_pop
2    mixed_pop
3    mixed_pop
4    mixed_pop
5    mixed_pop
6    mixed_pop
7    mixed_pop
8    mixed_pop
9    mixed_pop

```

The `.info()` method gives us an overview of our DataFrame, including... - A summary of the index labels - Information the columns - a 'Non-Null' Count. i.e. how many 'cells' in the column have a value in them. - The type (Dtype) of values that column holds. - Integer (int64) - e.g. 5 - Float (float64)- e.g. 5.3 - Boolean (bool) - e.g. True / False - Other (object) - Usually a string, but can also be any python object e.g. lists, dictionaries, classes. - A summary of how much computer memory the data needs.

```
[ ]: songs_df.info()
```

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1280 entries, 0 to 1279
Data columns (total 13 columns):
#   Column          Non-Null Count  Dtype
---  -
0   track_id        1280 non-null   object
1   track_name      1280 non-null   object
2   artists         1280 non-null   object
3   genre           1280 non-null   object
4   release_year    1280 non-null   int64
5   explicit        1280 non-null   bool
6   popularity      1280 non-null   int64
7   duration_ms     1280 non-null   int64
8   playlist_name   1280 non-null   object
9   danceability    1280 non-null   float64
10  loudness        1280 non-null   float64
11  speechiness     1279 non-null   float64
12  playlist_type   1280 non-null   object
dtypes: bool(1), float64(3), int64(3), object(6)
memory usage: 121.4+ KB

```

We can also get some of this information separately. For example the row and column names. Sometimes this is necessary if there are more columns than `.info()` wants to show.

```
[ ]: songs_df.index
```

```
[ ]: RangeIndex(start=0, stop=1187, step=1)
```

```
[ ]: songs_df.columns
```

```
[ ]: Index(['track_id', 'track_name', 'artists', 'genre', 'release_year',
          'explicit', 'popularity', 'duration_ms', 'playlist_name',
          'danceability', 'loudness', 'speechiness', 'playlist_type'],
          dtype='object')
```

## 1.2 Accessing columns, rows and cells

Being able to tell Pandas to provide us specific columns, specific rows, or even specific cells can be important when exploring data.

```
[ ]: # We can access a single column by
      # providing the name in square brackets.
      songs_df['track_name']
```

```
[ ]: 0      Running Up That Hill (A Deal With God)
      1              As It Was
      2      Afraid To Feel
      3      BREAK MY SOUL
      4      Glimpse of Us
      ...
      1182      Blue Suede Shoes
      1183      Cheek To Cheek
      1184      Diana
      1185      Just A Gigolo - Remastered
      1186      Bo Diddley
      Name: track_name, Length: 1187, dtype: object
```

```
[ ]: # or multiple names in a list if we want a few columns
      songs_df[['track_name', 'artists']]
```

```
[ ]:
      track_name      artists
0      Strangers      Kenya Grace
1      Paint The Town Red      Doja Cat
2      greedy      Tate McRae
3      Prada      cassö
4      Sprinter      Dave
...
1275      Chantilly Lace      The Big Bopper
1276      Maybellene      Chuck Berry
1277      Little Darlin'      The Diamonds
1278      It's Only Make Believe      Conway Twitty
1279      Stupid Cupid      Connie Francis

[1280 rows x 2 columns]
```

```
[ ]: # We can also assign this to a new variable if we want quick access.
      # Note: When you set columns to a variable like this, it is referencing
```

```
# the original DataFrame, not copying it.

songs_track_artist = songs_df[['track_name', 'artists']]
songs_track_artist.head()
```

```
[ ]:      track_name      artists
0      Strangers  Kenya Grace
1  Paint The Town Red    Doja Cat
2      greedy      Tate McRae
3      Prada        cassö
4      Sprinter      Dave
```

We can access specific rows... - By referring to their row label - `.loc` - By referring to their row index - `.iloc`

```
[ ]: songs_df.loc[1]
```

```
[ ]: track_id      56y1jOTK0XSvJzVv9vHQBK
track_name      Paint The Town Red
artists          Doja Cat
genre            dance pop
release_year      2023
explicit          True
popularity         87
duration_ms      230480
playlist_name    Top 50 - United Kingdom
danceability      0.864
loudness          -7.683
speechiness       0.194
playlist_type     mixed_pop
Name: 1, dtype: object
```

At the moment row labels and row indexes (the row's position in the data) are the same by default. However, say we messed up the order by sorting the data by year.

```
[ ]: by_year = songs_df.sort_values('release_year')
by_year
```

```
[ ]:      track_id      track_name \
1211  4I4aQGNJ2HufloNtB65nxR      That's Amore
1182  648TTtYB0bH0P8Hfy0FmkL      Unforgettable
1257  7GnMzVWOHLBPcfco4L1GtE      Earth Angel
1259  0x0ffSAP6PkdoDgH0froof      My Funny Valentine - Remastered
1204  1uRKT2LRANv4baowBWHfDS      (We're Gonna) Rock Around The Clock
...      ...      ...
254   651eXqfkdViSssEVN23uYL      Mama's Boy
255   0LzidBf7cUsnZnG340UPSF      Mosquito
256   2SXx70fa79CeJfio98aJcG      Eat Your Young
```

258	26vDr5jgWQoJ0TH4Bu3KCQ	WORTHLESS
0	5mjYQaktjmjcMKcUIcqz4s	Strangers

	artists	genre	release_year	\
1211	Dean Martin	adult standards	1954	
1182	Nat King Cole	adult standards	1954	
1257	The Penguins	doo-wop	1954	
1259	Frank Sinatra	adult standards	1954	
1204	Bill Haley & His Comets	rock-and-roll	1955	
...	...	...	...	
254	Dominic Fike	alternative pop rock	2023	
255	PinkPantheress	bedroom pop	2023	
256	Hozier	irish singer-songwriter	2023	
258	d4vd	bedroom pop	2023	
0	Kenya Grace	singer-songwriter pop	2023	

	explicit	popularity	duration_ms	playlist_name	\
1211	False	64	190400	All Out 50s	
1182	False	71	191973	All Out 50s	
1257	False	58	171066	All Out 50s	
1259	False	58	150666	All Out 50s	
1204	False	66	129893	All Out 50s	
...	...	...	...	...	
254	False	81	155721	alt/pop	
255	False	76	146240	alt/pop	
256	False	73	243946	alt/pop	
258	False	69	163049	alt/pop	
0	False	97	172964	Top 50 - United Kingdom	

	danceability	loudness	speechiness	playlist_type
1211	0.471	-13.600	0.0309	all_out_decades
1182	0.349	-13.507	0.0310	all_out_decades
1257	0.487	-11.121	0.0271	all_out_decades
1259	0.257	-14.267	0.0332	all_out_decades
1204	0.811	-6.317	0.1680	all_out_decades
...	...	...	...	...
254	0.807	-7.554	0.0321	mixed_pop
255	0.710	-4.032	0.1180	mixed_pop
256	0.546	-6.442	0.0286	mixed_pop
258	0.644	-6.299	0.0545	mixed_pop
0	0.628	-8.307	NaN	mixed_pop

[1280 rows x 13 columns]

To access the very first row of this dataset we either need to know the row label...

```
[ ]: by_year.loc[1211]
```



```
[ ]: track_id      4I4aQGNJ2HufloNtB65nxR
     track_name    That's Amore
     artists       Dean Martin
     genre          adult standards
     release_year   1954
     explicit       False
     popularity     64
     duration_ms    190400
     playlist_name  All Out 50s
     danceability   0.471
     loudness       -13.6
     speechiness    0.0309
     playlist_type  all_out_decades
     Name: 1211, dtype: object
```

Or simply ask for the first row by index location like we would a list.

```
[ ]: by_year.iloc[0]
```

```
[ ]: track_id      4I4aQGNJ2HufloNtB65nxR
     track_name    That's Amore
     artists       Dean Martin
     genre          adult standards
     release_year   1954
     explicit       False
     popularity     64
     duration_ms    190400
     playlist_name  All Out 50s
     danceability   0.471
     loudness       -13.6
     speechiness    0.0309
     playlist_type  all_out_decades
     Name: 1211, dtype: object
```

Or the very last row, again like a list.

```
[ ]: by_year.iloc[-1]
```

```
[ ]: track_id      5mjYQaktjnmjcmKcUIcqz4s
     track_name    Strangers
     artists       Kenya Grace
     genre          singer-songwriter pop
     release_year   2023
     explicit       False
     popularity     97
     duration_ms    172964
     playlist_name  Top 50 - United Kingdom
     danceability   0.628
```

```
loudness                -8.307
speechiness              NaN
playlist_type           mixed_pop
Name: 0, dtype: object
```

The index labels given to rows stick to them like an ID, however an index position can change.

index	label	name
0	0	Arthur
1	1	Betty
2	2	Carole

After reversing the order of `name`

index	label	name
0	2	Carole
1	1	Betty
2	0	Arthur

Finally we can access specific cells, or collections of cells using `.loc` and `.iloc` as well.

```
[ ]: # Row named 4, column artists
songs_df.loc[2, 'artists']
```

```
[ ]: 'Tate McRae'
```

```
[ ]: # Rows named 2 to 4, column artists and track_name
songs_df.loc[2:4, ['artists', 'track_name']]
```

```
[ ]:      artists track_name
2  Tate McRae    greedy
3      cassö    Prada
4      Dave    Sprinter
```

```
[ ]: # Rows at position -5 to -1, second column
songs_df.iloc[-5:-1, 1 ]
```

```
[ ]: 1275      Chantilly Lace
1276      Maybellene
1277      Little Darlin'
1278      It's Only Make Believe
Name: track_name, dtype: object
```

Often if you are accessing specific rows, you'll use `.loc` but it is helpful to know the difference from `.iloc` as sometimes you won't know the index names, but you'll know the position.

### 1.3 Exercises 1

Take a look at section 1 of the exercises sheet. Complete the tasks before moving on.

### 1.4 Filtering Data

Filtering allows you to select multiple rows based on particular criteria.

A filter is essentially a list of **True** or **False** values that is the same length as our dataset.

If the first item in our list is **True** then that means we want to **keep** the first row, if the second item is **False** that means we **drop** the second row and so on.

```
[ ]: toy_example = pd.read_csv('toy_example.csv')
toy_example
```

```
[ ]:      name  age
0   Arya   20
1    Bob   25
2  Chloe   30
```

```
[ ]: toy_filter = [True, False, True]
toy_example[toy_filter]
```

```
[ ]:      name  age
0   Arya   20
2  Chloe   30
```

Rather than manually make our filters, we use Pandas to generate them based on conditional expressions.

```
[ ]: age_filter = toy_example['age'] > 20
age_filter
```

```
[ ]: 0    False
1     True
2     True
Name: age, dtype: bool
```

```
[ ]: toy_example[age_filter]
```

```
[ ]:      name  age
1    Bob   25
2  Chloe   30
```

We can also combine conditions for more complex questions using the **&** operator which means **and**.

We should wrap our different conditions in parentheses ( ) to make sure we're clear where a condition begins and ends.

```
[ ]: (toy_example['age'] > 20) & (toy_example['age'] < 30)
```

```
[ ]: 0    False
      1     True
      2    False
      Name: age, dtype: bool
```

For readability's sake it is worth either making filters as their own variables, or laying them out on separate lines.

```
[ ]: filter_above_20 = toy_example['age'] > 20
      filter_below_30 = toy_example['age'] < 30
      toy_example[filter_above_20 & filter_below_30]
```

```
[ ]:   name  age
      1  Bob   25
```

```
[ ]: toy_example[
      (toy_example['age'] > 20) &
      (toy_example['age'] < 30)
      ]
```

```
[ ]:   name  age
      1  Bob   25
```

Let's move filter our spotify data to answer a specific question.

Q: What are the most popular explicit songs from the UK Top 50?

Break it down, we'll need three filters - Songs from the UK Top 50 playlist - Songs marked as explicit - Songs above a certain popularity score

Spotify scores songs 0-100, so we'll say 90, which is the top 10%.

```
[ ]: # we check our columns to see what we'll need
      songs_df.columns
```

```
[ ]: Index(['track_id', 'track_name', 'artists', 'genre', 'release_year',
           'explicit', 'popularity', 'duration_ms', 'playlist_name',
           'danceability', 'loudness', 'speechiness', 'playlist_type'],
          dtype='object')
```

Looks like we will be using playlist\_name, explicit and popularity.

#### 1.4.1 Filter 1: By playlist

We check to see what the values of the playlist\_name column look like, what type of values they are and then we can decide how to use them.

```
[ ]: songs_df['playlist_name']
```

```
[ ]: 0    Top 50 - United Kingdom
      1    Top 50 - United Kingdom
```

```

2      Top 50 - United Kingdom
3      Top 50 - United Kingdom
4      Top 50 - United Kingdom
...
1275      All Out 50s
1276      All Out 50s
1277      All Out 50s
1278      All Out 50s
1279      All Out 50s
Name: playlist_name, Length: 1280, dtype: object

```

```
[ ]: songs_df['playlist_name'].unique()
```

```
[ ]: array(['Top 50 - United Kingdom', 'The Pop List', "Today's Top Hits ",
        'Cheesy Hits!', 'alt/pop', 'Hit Rewind',
        'Every Official UK Number 1 Ever', 'Every UK Number One: 2023',
        'All Out 2000s', 'All Out 2010s', 'All Out 90s', 'All Out 80s',
        'All Out 70s', 'All Out 60s', 'All Out 50s'], dtype=object)

```

```
[ ]: playlist_filter = songs_df['playlist_name'] == 'Top 50 - United Kingdom'
```

#### 1.4.2 Filter 2: By Explicit

We don't know exactly how Spotify categorises this so let's check the column values like before, and then create our filter.

```
[ ]: songs_df['explicit']
```

```
[ ]: 0      False
      1      True
      2      True
      3      True
      4      True
...
1275  False
1276  False
1277  False
1278  False
1279  False
Name: explicit, Length: 1280, dtype: bool

```

```
[ ]: explicit_filter = songs_df['explicit'] == True
```

#### 1.4.3 Filter 3: By Popularity

If we check the popularity column we can get a sense of how it works.

```
[ ]: songs_df['popularity']
```

```
[ ]: 0      97
      1      87
      2      56
      3      94
      4      94
      ..
     1275    57
     1276    57
     1277    57
     1278    57
     1279    57
     Name: popularity, Length: 1280, dtype: int64
```

We can check the minimum and maximum values to confirm the range they use, though the API documentation would also tell us this.

The `.describe()` method provides a range of summary information about a column including that... - The **count** of records is 1,280 - The **mean** or average value is 75.49 - The **minimum** value is 0 - The **maximum** value is 100

```
[ ]: # t
     songs_df['popularity'].describe()
```

```
[ ]: count      1280.000000
     mean        75.489844
     std         15.585720
     min          0.000000
     25%         71.000000
     50%         79.000000
     75%         85.000000
     max        100.000000
     Name: popularity, dtype: float64
```

We can also get each of these values independently by using the associated method. For example we can get the `mean()`

```
[ ]: avg_popularity = songs_df['popularity'].mean()
     avg_popularity
```

```
[ ]: 75.48984375
```

```
[ ]: popularity_filter = songs_df['popularity'] > avg_popularity
```

#### 1.4.4 Combining them all together

```
[ ]: uk_popular_explicit = songs_df[playlist_filter & explicit_filter &
     popularity_filter]
     uk_popular_explicit
```

	track_id	track_name \
1	56y1j0TK0XSvJzVv9vHQBK	Paint The Town Red
3	59NraMJsLaMCVtwXTSia8i	Prada
4	2FDTHlrBguDzQkp7PVj16Q	Sprinter
7	1kuGVB7EU95pJ0bxwvfwKS	vampire
11	2YSzYUF3jWqb9YP9VXmpjE	IDGAF (feat. Yeat)
14	4rXLjWdF2ZZpXCVTfWcshS	fukumean
15	3IX0yuEVvDbnqUwMBB3ouC	bad idea right?
16	2gyxAWHebV7xPYVxqoi86f	get him back!
17	602d2gJewoiF1FivuOMMwE	Disconnect
22	7aqfrAY2p9BUSiupwk3svU	First Person Shooter (feat. J. Cole)
28	4RoKNqyZ9622tcAeNPNv5k	City Boys
31	3JvKfv6T31z00ini8iNI0	Another Love
33	6WzRpISELf3YglGAh7TXcG	Popular (with Playboi Carti & Madonna) - Music...
39	1yeB8MUNeLo9Ek1UEpsyz6	Rich Baby Daddy (feat. Sexyy Red & SZA)
40	5mHdCZtVyby4DcJw8799hZp	Escapism.

	artists	genre	release_year	explicit \
1	Doja Cat	dance pop	2023	True
3	cassö	***OOPS!***	2023	True
4	Dave	uk hip hop	2023	True
7	Olivia Rodrigo	pop	2023	True
11	Drake	canadian hip hop	2023	True
14	Gunna	atl hip hop	2023	True
15	Olivia Rodrigo	pop	2023	True
16	Olivia Rodrigo	pop	2023	True
17	Becky Hill	pop dance	2023	True
22	Drake	canadian hip hop	2023	True
28	Burna Boy	afrobeats	2023	True
31	Tom Odell	chill pop	2013	True
33	The Weeknd	canadian contemporary r&b	2023	True
39	Drake	canadian hip hop	2023	True
40	RAYE	uk contemporary r&b	2023	True

	popularity	duration_ms	playlist_name	danceability	loudness \
1	87	230480	Top 50 - United Kingdom	0.864	-7.683
3	94	132359	Top 50 - United Kingdom	0.638	-5.804
4	94	229133	Top 50 - United Kingdom	0.916	-8.067
7	95	219724	Top 50 - United Kingdom	0.511	-5.745
11	90	260111	Top 50 - United Kingdom	0.663	-8.399
14	95	125040	Top 50 - United Kingdom	0.847	-6.747
15	93	184783	Top 50 - United Kingdom	0.627	-3.446
16	92	211141	Top 50 - United Kingdom	0.546	-5.719
17	80	164914	Top 50 - United Kingdom	0.504	-0.653
22	89	247444	Top 50 - United Kingdom	0.470	-7.779
28	84	153346	Top 50 - United Kingdom	0.808	-6.764
31	92	244360	Top 50 - United Kingdom	0.445	-8.532

33	93	215466	Top 50 - United Kingdom	0.855	-6.276
39	85	319191	Top 50 - United Kingdom	0.645	-4.560
40	86	272373	Top 50 - United Kingdom	0.538	-5.355

	speechiness	playlist_type
1	0.1940	mixed_pop
3	0.0375	mixed_pop
4	0.2410	mixed_pop
7	0.0578	mixed_pop
11	0.2710	mixed_pop
14	0.0903	mixed_pop
15	0.0955	mixed_pop
16	0.1810	mixed_pop
17	0.0946	mixed_pop
22	0.3200	mixed_pop
28	0.1720	mixed_pop
31	0.0400	mixed_pop
33	0.1890	mixed_pop
39	0.0528	mixed_pop
40	0.1140	mixed_pop

```
[ ]: len(uk_popular_explicit)
```

```
[ ]: 15
```

```
[ ]: uk_popular_explicit.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 15 entries, 1 to 40
Data columns (total 13 columns):
#   Column          Non-Null Count  Dtype
---  -
0   track_id        15 non-null    object
1   track_name      15 non-null    object
2   artists         15 non-null    object
3   genre           15 non-null    object
4   release_year    15 non-null    int64
5   explicit        15 non-null    bool
6   popularity      15 non-null    int64
7   duration_ms     15 non-null    int64
8   playlist_name   15 non-null    object
9   danceability    15 non-null    float64
10  loudness        15 non-null    float64
11  speechiness     15 non-null    float64
12  playlist_type   15 non-null    object
dtypes: bool(1), float64(3), int64(3), object(6)
memory usage: 1.5+ KB
```



The same result could be achieved all in one go, but exploring the data and understanding each column is an important part of the process.

```
[ ]: songs_df[
    (songs_df['playlist_name'] == 'Top 50 - United Kingdom') &
    (songs_df['explicit'] == True) &
    (songs_df['popularity'] > avg_popularity)
].sort_values('popularity', ascending=False)
```

```
[ ]:
      track_id      track_name \
7    1kuGVB7EU95pJ0bxwvfwKS      vampire
14   4rXLjWdF2ZZpXCVTfWcshS      fukumean
3    59NraMJsLaMCVtwXTSia8i      Prada
4    2FDTHlrBguDzQkp7PVj16Q      Sprinter
15   3IX0yuEVvDbnqUwMBB3ouC      bad idea right?
33   6WzRpISELf3YglGAh7TXcG  Popular (with Playboi Carti & Madonna) - Music...
16   2gyxAWHebV7xPYVxqoi86f      get him back!
31   3JvKfv6T31z00ini8iNI0      Another Love
11   2YSzYUF3jWqb9YP9VXmpjE      IDGAF (feat. Yeat)
22   7aqfrAY2p9BUSiupwk3svU      First Person Shooter (feat. J. Cole)
1    56y1j0TK0XSvJzVv9vHQBK      Paint The Town Red
40   5mHdCZtVyb4DcJw8799hZp      Escapism.
39   1yeB8MUNeLo9Ek1UEpsyz6      Rich Baby Daddy (feat. Sexyy Red & SZA)
28   4RoKNqyZ9622tcAeNPNv5k      City Boys
17   602d2gJewoiF1FivuOMMwE      Disconnect
```

```

      artists      genre  release_year  explicit \
7    Olivia Rodrigo      pop      2023      True
14      Gunna      atl hip hop      2023      True
3      cassö      ***OOPS!***      2023      True
4      Dave      uk hip hop      2023      True
15   Olivia Rodrigo      pop      2023      True
33      The Weeknd  canadian contemporary r&b      2023      True
16   Olivia Rodrigo      pop      2023      True
31      Tom Odell      chill pop      2013      True
11      Drake      canadian hip hop      2023      True
22      Drake      canadian hip hop      2023      True
1      Doja Cat      dance pop      2023      True
40      RAYE      uk contemporary r&b      2023      True
39      Drake      canadian hip hop      2023      True
28      Burna Boy      afrobeats      2023      True
17      Becky Hill      pop dance      2023      True
```

```

      popularity  duration_ms      playlist_name  danceability  loudness \
7              95      219724  Top 50 - United Kingdom      0.511      -5.745
14             95      125040  Top 50 - United Kingdom      0.847      -6.747
3              94      132359  Top 50 - United Kingdom      0.638      -5.804
```

4	94	229133	Top 50 - United Kingdom	0.916	-8.067
15	93	184783	Top 50 - United Kingdom	0.627	-3.446
33	93	215466	Top 50 - United Kingdom	0.855	-6.276
16	92	211141	Top 50 - United Kingdom	0.546	-5.719
31	92	244360	Top 50 - United Kingdom	0.445	-8.532
11	90	260111	Top 50 - United Kingdom	0.663	-8.399
22	89	247444	Top 50 - United Kingdom	0.470	-7.779
1	87	230480	Top 50 - United Kingdom	0.864	-7.683
40	86	272373	Top 50 - United Kingdom	0.538	-5.355
39	85	319191	Top 50 - United Kingdom	0.645	-4.560
28	84	153346	Top 50 - United Kingdom	0.808	-6.764
17	80	164914	Top 50 - United Kingdom	0.504	-0.653

	speechiness	playlist_type
7	0.0578	mixed_pop
14	0.0903	mixed_pop
3	0.0375	mixed_pop
4	0.2410	mixed_pop
15	0.0955	mixed_pop
33	0.1890	mixed_pop
16	0.1810	mixed_pop
31	0.0400	mixed_pop
11	0.2710	mixed_pop
22	0.3200	mixed_pop
1	0.1940	mixed_pop
40	0.1140	mixed_pop
39	0.0528	mixed_pop
28	0.1720	mixed_pop
17	0.0946	mixed_pop

## 1.5 Exercises 2

Take a look at section 2 of the exercises sheet. Complete the tasks.

If there is time, work through the appropriate chapter of the McLevey textbook OR the recommended DataCamp course.

See Moodle for details.