2016312761 여혁수 Homework 3 report

1-(a)

```
                                Name   AlbumId  ...       Bytes  UnitPrice
TrackId                                         ...
610          My Funny Valentine (Live)      49  ...    29416781       0.99
620                     Space Truckin'      50  ...    39267613       0.99
621          Going Down / Highway Star      50  ...    29846063       0.99
1581                Dazed And Confused     127  ...    36052247       0.99
1666                Dazed And Confused     137  ...    52490554       0.99
2429    We've Got To Get Together/Jingo    198  ...    34618222       0.99
2432                       Funky Piano     198  ...    30200730       0.99
2819  Battlestar Galactica: The Story So Far   226  ...   490750393       1.99
2820             Occupation / Precipice    227  ...  1054423946       1.99
2821                      Exodus, Pt. 1    227  ...   475079441       1.99
```

1-(b)

Puja Srivastava

1-(c)

```
Led Zeppelin 14
Metallica 10
Deep Purple 11
Iron Maiden 21
Ozzy Osbourne 6
U2 10
```

1-(d)

```
The World of Classical Favourites 6

English Renaissance 6
```

2-(a)

```
          ArtistId  ... PlaylistId
0                1  ...        1.0
1                1  ...        8.0
2                1  ...       17.0
3                1  ...        1.0
4                1  ...        8.0
...            ...  ...        ...
8781           249  ...        1.0
8782           249  ...        5.0
8783           249  ...        8.0
8784           249  ...       12.0
8785           249  ...       14.0
[8786 rows x 14 columns]
```

```
Columns of dataframe: ['ArtistId', 'ArtistName', 'AlbumId', 'Title', 'TrackId', 'TrackName',
       'MediaTypeId', 'GenreId', 'Composer', 'Milliseconds', 'Bytes',
       'UnitPrice', 'GenreName', 'PlaylistId']
```

2-(b)

```
['AC/DC', 'Accept', 'Led Zeppelin', 'Queen', 'Kiss', 'Deep Purple', 'Santana', 'Creedence Clearwater
Revival', 'Foo Fighters', "Guns N' Roses", 'Iron Maiden', 'Nirvana', 'Ozzy Osbourne', 'Pearl Jam', 'Red
```

Hot Chili Peppers', 'Skank', 'The Cult', 'The Rolling Stones', 'U2', 'Van Halen', 'Spyro Gyra', 'Miles Davis', 'Antônio Carlos Jobim', 'Caetano Veloso', 'Chico Science & Nação Zumbi', 'Various Artists', 'Gilberto Gil', 'Milton Nascimento', 'Cássia Eller', 'Djavan', 'Legião Urbana', 'Lulu Santos', 'Os Paralamas Do Sucesso', 'Tim Maia', 'Black Label Society', 'Black Sabbath', 'Metallica', 'Audioslave', 'Green Day', 'Faith No More', 'R.E.M.', 'Smashing Pumpkins', 'The Tea Party', 'Titãs', 'Eric Clapton', 'The Black Crowes', 'Cidade Negra', 'Jamiroquai', 'Amy Winehouse', 'Battlestar Galactica', 'Lost', 'The Office', 'English Concert & Trevor Pinnock', 'Eugene Ormandy', 'Michael Tilson Thomas & San Francisco Symphony', 'Berliner Philharmoniker & Herbert Von Karajan']

2-(c)

Top 7 genres:

```
   GenreId  COUNT(*)
0        1      1297
1        7       579
2        3       374
3        4       332
4        2       130
5       19        93
6        6        81
```

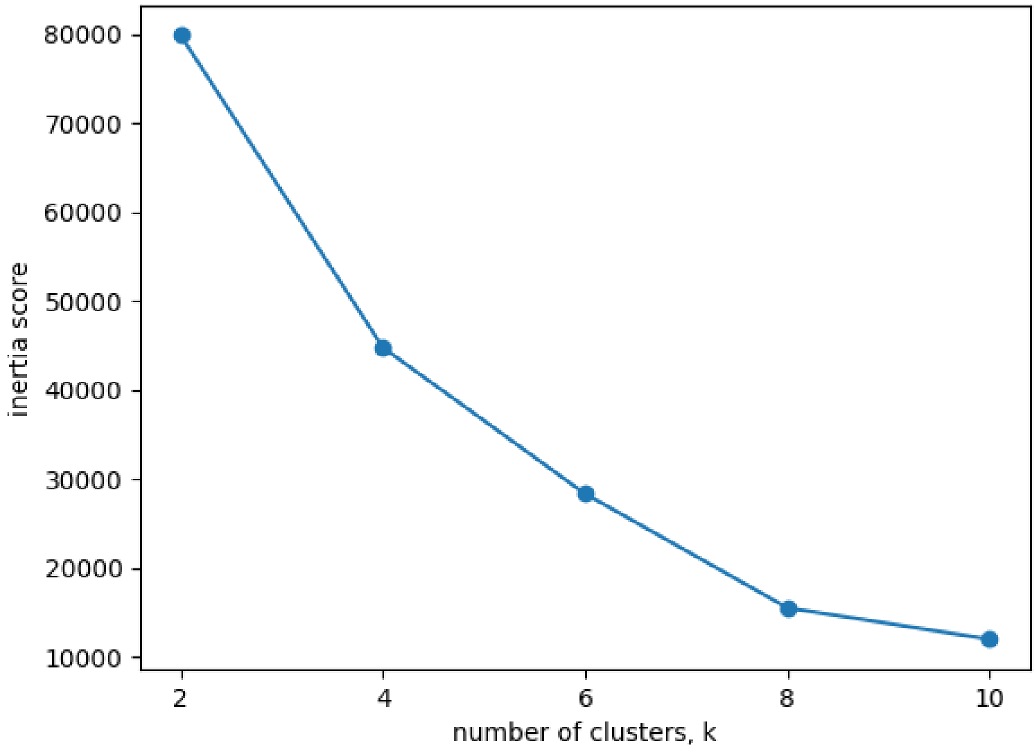
Construct a set of ten features for each artist:

```
                          Rock  ...     Number of playlists
AC/DC                       18  ...                       3
Accept                       4  ...                       4
Led Zeppelin               114  ...                       3
Queen                       45  ...                       3
Kiss                        35  ...                       3
Deep Purple                 92  ...                       3
```

Santana	27	...	3
Creedence Clearwater Revival	40	...	2
Foo Fighters	33	...	3
Guns N' Roses	28	...	3
Iron Maiden	81	...	4
Nirvana	29	...	4
Ozzy Osbourne	18	...	4
Pearl Jam	54	...	4
Red Hot Chili Peppers	31	...	3
Skank	23	...	3
The Cult	30	...	3
The Rolling Stones	41	...	3
U2	112	...	3
Van Halen	52	...	3
Spyro Gyra	0	...	3
Miles Davis	0	...	3
Antônio Carlos Jobim	0	...	3
Caetano Veloso	0	...	4
Chico Science & Nação Zumbi	0	...	3
Various Artists	0	...	2
Gilberto Gil	0	...	4
Milton Nascimento	0	...	3
Cássia Eller	0	...	3
Djavan	0	...	4
Legião Urbana	0	...	3
Lulu Santos	0	...	2

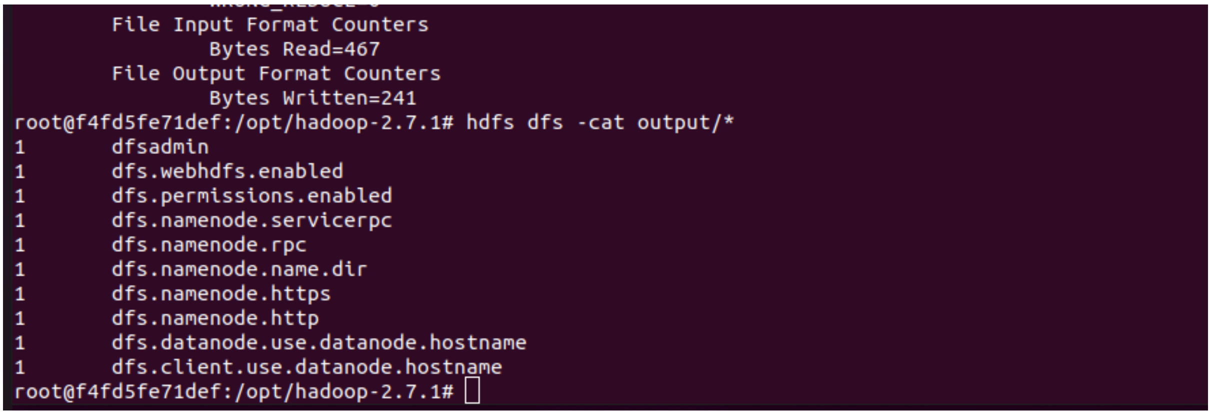
```
Os Paralamas Do Sucesso                            0  ...     3
Tim Maia                                           0  ...     3
Black Label Society                                0  ...     2
Black Sabbath                                      0  ...     3
Metallica                                          0  ...     4
Audioslave                                        14  ...     3
Green Day                                          0  ...     3
Faith No More                                     15  ...     3
R.E.M.                                            14  ...     3
Smashing Pumpkins                                  0  ...     3
The Tea Party                                      0  ...     3
Titãs                                              0  ...     3
Eric Clapton                                       0  ...     4
The Black Crowes                                   0  ...     2
Cidade Negra                                       0  ...     3
Jamiroquai                                        10  ...     3
Amy Winehouse                                      0  ...     2
Battlestar Galactica                               0  ...     2
Lost                                               0  ...     2
The Office                                         0  ...     2
English Concert & Trevor Pinnock                   0  ...     6
Eugene Ormandy                                     0  ...     7
Michael Tilson Thomas & San Francisco Symphony     0  ...     5
Berliner Philharmoniker & Herbert Von Karajan      0  ...     6

[56 rows x 10 columns]
```

2-(d)



As you see the plot, the degree of cohesion goes higher. Based on 8, you can see that the slope of inertia score changes much gentler. It can be determined that the point 8 is the elbow point. So I think appropriate value of k is 8.

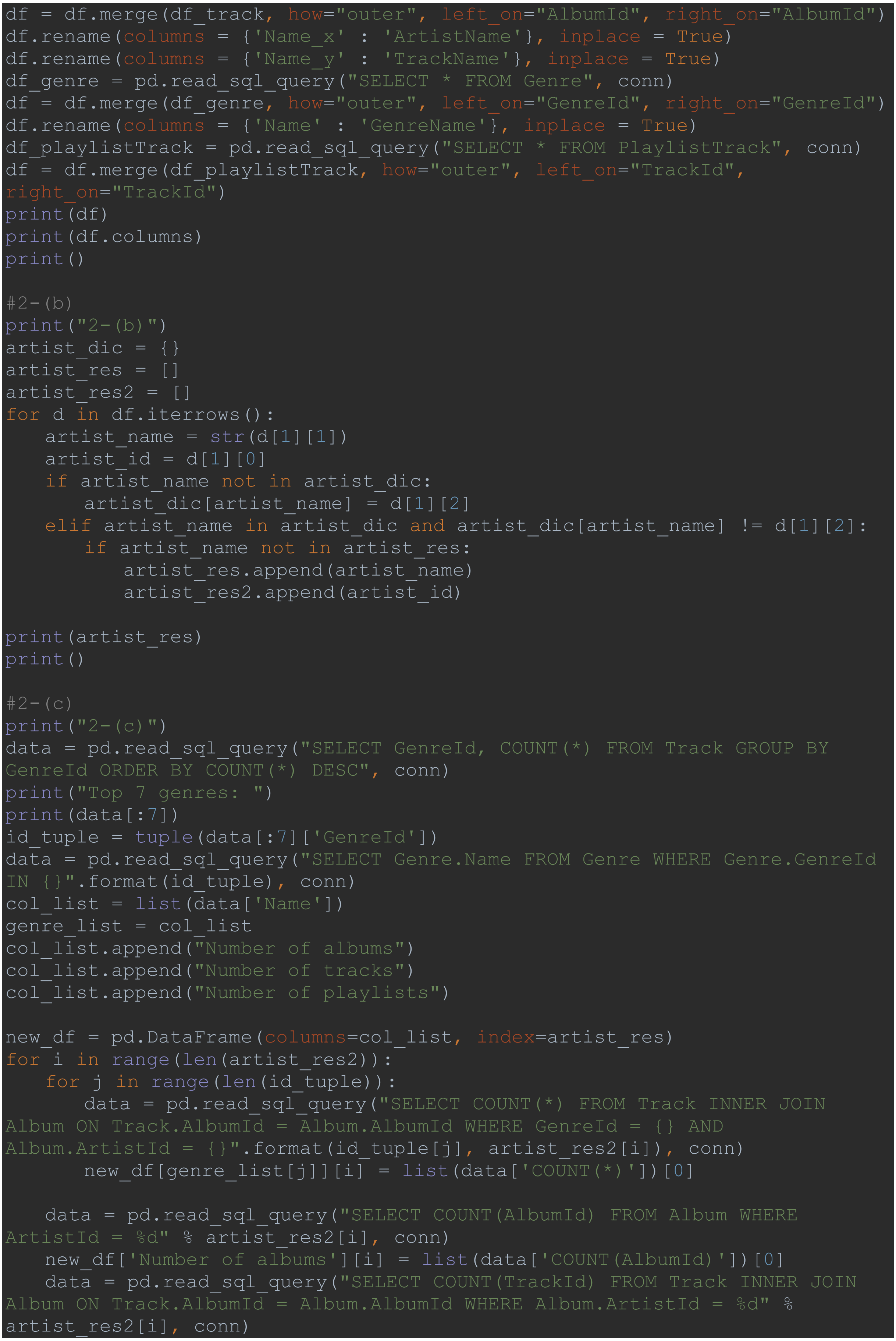
3



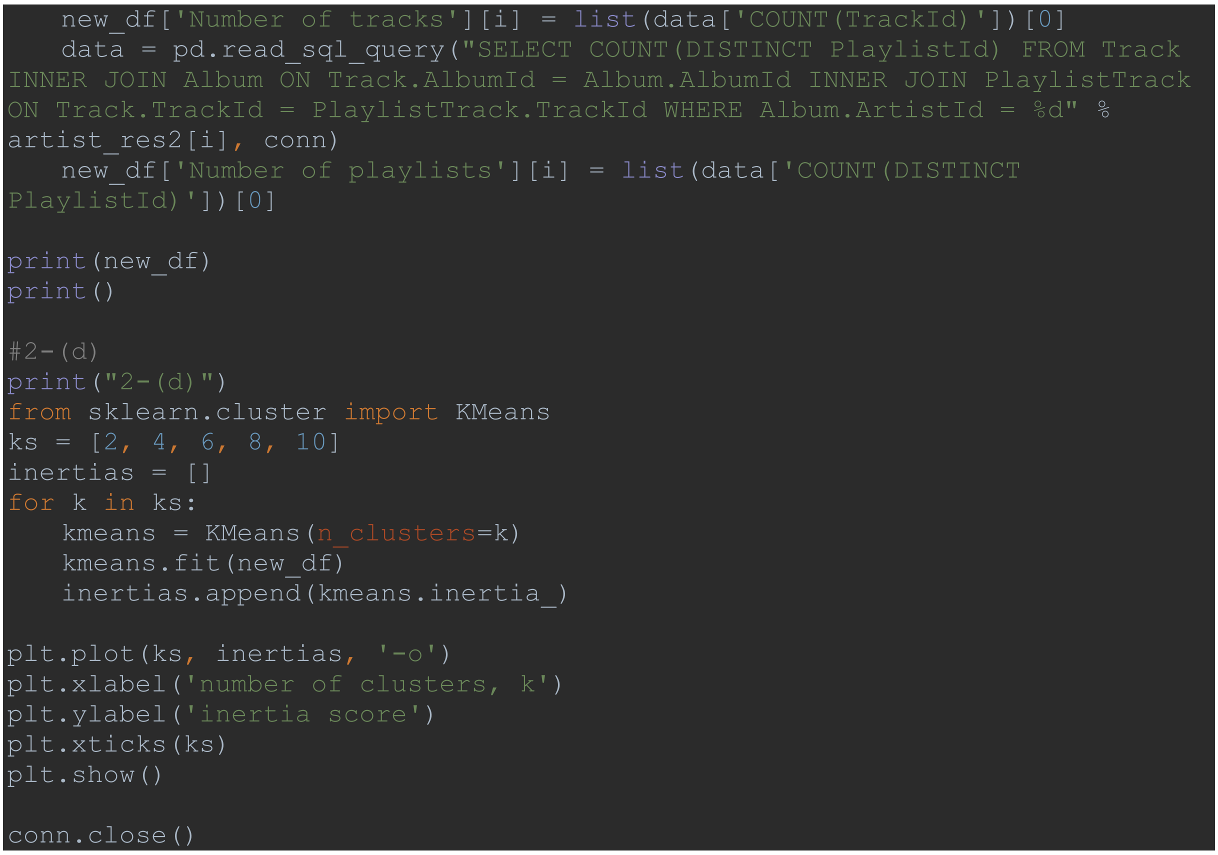
```
        File Input Format Counters
                Bytes Read=467
        File Output Format Counters
                Bytes Written=241
root@f4fd5fe71def:/opt/hadoop-2.7.1# hdfs dfs -cat output/*
1       dfsadmin
1       dfs.webhdfs.enabled
1       dfs.permissions.enabled
1       dfs.namenode.servicerpc
1       dfs.namenode.rpc
1       dfs.namenode.name.dir
1       dfs.namenode.https
1       dfs.namenode.http
1       dfs.datanode.use.datanode.hostname
1       dfs.client.use.datanode.hostname
root@f4fd5fe71def:/opt/hadoop-2.7.1# 
```

Code)

```
import sqlite3
import pandas as pd
import matplotlib.pyplot as plt

conn = sqlite3.connect("Chinook_Sqlite.sqlite")
cursor = conn.cursor()

#1-(a)
print("1-(a)")
data = pd.read_sql_query("SELECT * FROM Track WHERE Milliseconds >
900000;", conn, index_col="TrackId")
print(data[:10])
print()

#1-(b)
print("1-(b)")
data = pd.read_sql_query("SELECT Invoice.CustomerId FROM Invoice INNER JOIN
InvoiceLine ON Invoice.InvoiceId = InvoiceLine.InvoiceId GROUP BY
CustomerId ORDER BY COUNT(InvoiceLine.TrackId);", conn,
index_col="CustomerId")
min_amount = list(data.index)[0]
data = pd.read_sql_query("SELECT FirstName, LastName FROM Customer WHERE
CustomerId='%d';" % min_amount, conn)
print(data['FirstName'][0], data['LastName'][0])
print()

#1-(c)
print("1-(c)")
data = pd.read_sql_query("SELECT COUNT(*), * FROM Album INNER JOIN Artist
ON Album.ArtistId = Artist.ArtistId GROUP BY Album.ArtistId HAVING COUNT(*)
> 5;", conn)
count_list = list(data['COUNT(*)'])
name_list = list(data['Name'])
for i in range(len(count_list)):
    print(str(name_list[i]) + ' ' + str(count_list[i]))
print()

#1-(d)
print("1-(d)")
data = pd.read_sql_query("SELECT Track.AlbumId, COUNT(DISTINCT
PlaylistTrack.PlaylistId) FROM Track INNER JOIN PlaylistTrack ON
Track.TrackId = PlaylistTrack.TrackId GROUP BY Track.AlbumId HAVING
COUNT(DISTINCT PlaylistTrack.PlaylistId) > 5;", conn)
id_tuple = tuple(data['AlbumId'])
data2 = pd.read_sql_query("SELECT Album.Title FROM Album WHERE
Album.AlbumId IN {}".format(id_tuple), conn)
for i in range(len(data)):
    print(data2['Title'][i], data['COUNT(DISTINCT
PlaylistTrack.PlaylistId)'][i])
print()

#2-(a)
print("2-(a)")
df = pd.read_sql_query("SELECT * FROM Artist", conn)
df_album = pd.read_sql_query("SELECT * FROM Album", conn)
df = df.merge(df_album, how="outer", left_on="ArtistId",
right_on="ArtistId")
df_track = pd.read_sql_query("SELECT * FROM Track", conn)
```



```
df = df.merge(df_track, how="outer", left_on="AlbumId", right_on="AlbumId")
df.rename(columns = {'Name_x' : 'ArtistName'}, inplace = True)
df.rename(columns = {'Name_y' : 'TrackName'}, inplace = True)
df_genre = pd.read_sql_query("SELECT * FROM Genre", conn)
df = df.merge(df_genre, how="outer", left_on="GenreId", right_on="GenreId")
df.rename(columns = {'Name' : 'GenreName'}, inplace = True)
df_playlistTrack = pd.read_sql_query("SELECT * FROM PlaylistTrack", conn)
df = df.merge(df_playlistTrack, how="outer", left_on="TrackId",
right_on="TrackId")
print(df)
print(df.columns)
print()

#2-(b)
print("2-(b)")
artist_dic = {}
artist_res = []
artist_res2 = []
for d in df.iterrows():
    artist_name = str(d[1][1])
    artist_id = d[1][0]
    if artist_name not in artist_dic:
        artist_dic[artist_name] = d[1][2]
    elif artist_name in artist_dic and artist_dic[artist_name] != d[1][2]:
        if artist_name not in artist_res:
            artist_res.append(artist_name)
            artist_res2.append(artist_id)

print(artist_res)
print()

#2-(c)
print("2-(c)")
data = pd.read_sql_query("SELECT GenreId, COUNT(*) FROM Track GROUP BY
GenreId ORDER BY COUNT(*) DESC", conn)
print("Top 7 genres: ")
print(data[:7])
id_tuple = tuple(data[:7]['GenreId'])
data = pd.read_sql_query("SELECT Genre.Name FROM Genre WHERE Genre.GenreId
IN {}".format(id_tuple), conn)
col_list = list(data['Name'])
genre_list = col_list
col_list.append("Number of albums")
col_list.append("Number of tracks")
col_list.append("Number of playlists")

new_df = pd.DataFrame(columns=col_list, index=artist_res)
for i in range(len(artist_res2)):
    for j in range(len(id_tuple)):
        data = pd.read_sql_query("SELECT COUNT(*) FROM Track INNER JOIN
Album ON Track.AlbumId = Album.AlbumId WHERE GenreId = {} AND
Album.ArtistId = {}".format(id_tuple[j], artist_res2[i]), conn)
        new_df[genre_list[j]][i] = list(data['COUNT(*)'])[0]

    data = pd.read_sql_query("SELECT COUNT(AlbumId) FROM Album WHERE
ArtistId = %d" % artist_res2[i], conn)
    new_df['Number of albums'][i] = list(data['COUNT(AlbumId)'])[0]
    data = pd.read_sql_query("SELECT COUNT(TrackId) FROM Track INNER JOIN
Album ON Track.AlbumId = Album.AlbumId WHERE Album.ArtistId = %d" %
artist_res2[i], conn)
```



```
    new_df['Number of tracks'][i] = list(data['COUNT(TrackId)'])[0]
    data = pd.read_sql_query("SELECT COUNT(DISTINCT PlaylistId) FROM Track
INNER JOIN Album ON Track.AlbumId = Album.AlbumId INNER JOIN PlaylistTrack
ON Track.TrackId = PlaylistTrack.TrackId WHERE Album.ArtistId = %d" %
artist_res2[i], conn)
    new_df['Number of playlists'][i] = list(data['COUNT(DISTINCT
PlaylistId)'])[0]

print(new_df)
print()

#2-(d)
print("2-(d)")
from sklearn.cluster import KMeans
ks = [2, 4, 6, 8, 10]
inertias = []
for k in ks:
    kmeans = KMeans(n_clusters=k)
    kmeans.fit(new_df)
    inertias.append(kmeans.inertia_)

plt.plot(ks, inertias, '-o')
plt.xlabel('number of clusters, k')
plt.ylabel('inertia score')
plt.xticks(ks)
plt.show()

conn.close()
```