Data Engineering Task - Questions

1. Who are the 10 most common artists in this dataset? Please answer with a SQL query (include query in your answer)

The most common artists are Pablo Picasso, Andy Warhol, Marc Chagall.

Query 1: Show most common artists

```
query = CleanedArtist.select(CleanedArtist.artist, CleanedArtist.count) \
.order_by(CleanedArtist.count.desc()).limit(10)
for row in query.dicts():
    print(row)
```

Here are the most common artists:

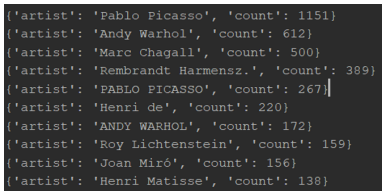

Figure 1: Most common artists.

2. How many artists were born before 1900? Include any python/SQL code used to answer this question.

1674 artists were born before 1900.

Query 2: Retrieve artists born before 1900

```
query = CleanedArtist.select(CleanedArtist.artist) \
.where(CleanedArtist.birth < 1900)
print(len(list(query.dicts())))
```

3. How many artists appear once in the dataset? How confident are you in this number?

Query 3: Show number of unique artists

```
query = CleanedArtist.select(CleanedArtist.artist).distinct()
print(len(list(query.dicts())))
```

There are 2098 unique artists in this dataset. However I wouldn't trust this value given that the data isn't purely cleaned.

1223 rows have a count <= 1, it represents over 50% of the unique artists (58,3%). It would mean that 50% of the artists only have one art piece. This could be discussed whether these lines are correct or if the come from uncleaned data. It is likely that is actually doesnt correspond to an artist however it could also be that the dataset only has one entry of this artist. I believe more data cleaning should be done to validate a 100% confidence in this number.

4. How did you handle entries where the artist name was not actually a person?

I noticed that some names weren't representing anything such as 'w '. Given the short amount of time I applied a first basic step: I considered that records where the artist name was less than two it was probably an error. Hence I deleted records when the artist name was shorter than two.

Given more time I would have added the following:

1. Fine tune the way I look for the artist name pattern in the Source field. I noticed that in most cases the name could be found at the beginning of the field and in most cases a name is composed of a first and last. So I kept the first two elements of the field. However I didn't deal with last names composed of more than one word (eg: Henri de)

2. Sometimes the name is not located at the beginning of the field but in the middle. I would design a more complex logic with regex or NLP analysis to look for the name pattern in the Source Field.

5. Given more time, what else would you have explored? Did you find any interesting insights along the way?

I think the data cleaning part is essential and I think adding some NLP in the logic would make sense in this case to extract the name of the artist and the dates.

I think the birth and death fields should be integers instead of Double since the year is always an int and is maximum 4 characters so we could set up less memory.

I would also look for invalid death and birth date like in the case of Picasso, he has many birth dates. For this maybe it would be more efficient and reliable if we fetched this information from outside of the database. For example either have a reference database with information about artists or either by making a query over to Wikipedia or another information website.

6. Was this fun? Which sections / questions were the most difficult and which were the easiest?

What I enjoyed less was the peewee interface, but it is like all python package it just takes a bit of time to get use to the syntax. I had some troubles when formatting to insert into the database. For instance, it only accepts a specific input that is done using the argument orient=index in the pandas function. And I had some problems in the configuration of the database when trying to insert for the first time. But after I got used to it quite fast.

I really enjoyed doing this exercise, I thought it was well prepared and covered many aspects. Moreover, the data provided was interesting to work with. I am also glad I got to learn about Pipenv. Finally doing this exercise really confirmed my motivation to join Arthena:)