worklab on normalization

This document is a copy of the google form on normalization https://forms.gle/GBGffeAT1sLnYeTk9

The goal is to normalize the WorldHits dataset provided by Spotify and available from Kaggle https://www.kaggle.com/datasets/thebumpkin/300-world-music-tracks-with-spotify-data

This dataset is a curated collection of world music, featuring 326 tracks from 66 diverse artists spanning six decades, from 1958 to 2019. It offers a rich tapestry of global sounds, from traditional rhythms to contemporary fusions. Each track is meticulously tagged with Spotify audio features, providing insights into tempo, key, energy, and more. This dataset is ideal for exploring the evolution of world music, analyzing trends across different cultures, or even training machine learning models to recognize unique musical patterns.

Load the dataset

You can download the data csv file called WorldHits.csv from the github repo.

- if you have already cloned the repo, refresh it with git pull origin master
- if you haven't done so already:

 git clone git@github.com:SkatAI/epitadb.git and cd epitadb
- or simply go to https://github.com/SkatAl/epitadb/tree/master/data and click right on the WorldHits.csv file to download it

Start by creating the database and importing the data from the csv file.

- psql on your local server either using the command line
- psql (-U postgres) -d postgres
- or connect on your local server with pgAdmin andd open a PSQL window.

Create a new database called worldhitsdb:

```
-- make sure the database does not exist
DROP database if exists worldhitsdb WITH(force);

-- then create the database. Change the OWNER if needed.

CREATE DATABASE worldhitsdb

WITH

OWNER = postgres

ENCODING = 'UTF8'

LOCALE_PROVIDER = 'libc'

CONNECTION LIMIT = -1

IS_TEMPLATE = False;
```

Check that the database has been created with \1 and connect to worldhitsdb with \c worldhitsdb.

Create the main table called tracks:

```
SQL
CREATE TABLE tracks (
    id serial primary key,
   Track VARCHAR(255),
    Artist VARCHAR(255),
   Album VARCHAR(255),
   Year INT,
   Duration INT,
   Time_Signature INT,
   Danceability FLOAT,
    Energy FLOAT,
   Key INT,
   Loudness FLOAT,
   Mode INT,
   Speechiness FLOAT,
    Acousticness FLOAT,
   Instrumentalness FLOAT,
   Liveness FLOAT,
   Valence FLOAT,
   Tempo FLOAT,
    Popularity INT
);
```

Finally, copy the data into the table with

```
\COPY tracks FROM '<YOUR PATH to>/WorldHits.csv' WITH CSV HEADER DELIMI
```

Finally make sure you have correctly imported the data. This query should return 326 rows

```
select count(*) from tracks;
```

Add a primary key

```
alter table tracks add column id serial primary key;
```

Explore the data

Now let's run a few queries to understand the data.

some info:

The Key column represents a numeric value ranging from 0 to 11, with 0 corresponding to the key of C. The list of keys is:

```
['A', 'A#', 'B', 'C', 'C#', 'D', 'D#', 'E', 'F', 'F#', 'G', 'G#']
```

- The Mode column uses 0 for minor and 1 for major
- Valence in music refers to the musical positiveness of a track: good vibes

Write down the queries:

1. How many artists?

```
your answer
```

2. Number of tracks per artist?

```
your answer
```

3. Who are the artists with 1 track only?

```
your answer
```

4. Average track duration per artist?

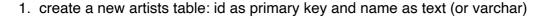
```
your answer
```

Normalization

First, let's ask the question: Why is the artist column a good candidate for normalization?

Why normalize the artist column?

The normalization process is the following



```
your answer
```

2. import the sorted artist names from tracks to the artists table

```
your answer
```

3. add a artist_id INT column in trees

```
your answer
```

4. reconcile both tables by updating the trees.artist_id with the correct artists.id

```
your answer
```

5. make the trees.artists_id a foreign key

```
your answer
```

6. check that there is no gap between the 2 tables: count the number of rows where trees.artist != artists.name

```
your answer
```

7. delete the artist column in the tracks table

```
your answer
```

For each step write the query in the answer box;

Finally, rewrite the queries

· how many artists?

```
your answer
```

· number of tracks per artists

```
your answer
```

• names of artists with 1 track only.

your answer

• average track duration per artist?

your answer