# Creating the Schema

For the database schema, we identified all the primary keys and foreign keys within each table. We also identified each field’s datatype. Then we used QuickDBD to create the schema. From there we were able to export four files, a pdf, a PNG image file, a SQL schema file, and a text file of the physical schema.

A screenshot of a computer screen

Description automatically generated

# Creating the database within Postgres

We created a new database named “crowdfunding” within Postgres. With the SQL schema file that we had already created, we were able to create four tables within the database: category\_df, subcategory\_df, contacts\_df, and campaign\_df.

After the tables were created, it was time to load the data into the tables. For this process, we use Python code from upload\_csv file.



Below is a screenshot of our SELECT statement and corresponding output to test that the data was loaded into the campaign\_df table. We repeated this process for the other 3 tables.

A screenshot of a computer

Description automatically generated

# Data visualization: Treemap

A screenshot of a computer

Description automatically generated

We created a treemap that visualized the different outcomes grouped by category and subcategory. We used plotly express for the treemap. Currently, the campaign\_df only displays category ids and subcategory ids. Instead, we prefer to display the category name and subcategory name, i.e., theater, plays, documentary…etc. In order to achieve this, we had to perform two joins with the category.csv and the subcategory.csv. After the joins were completed, we were able to create a new dataframe which we used for the treemap. We used plotly express for the plot.