

Project Two Crowdfunding – Written Report

The purpose of this project was to build an ETL (Extract, Transform, Load) pipeline using Python and Pandas to transform crowdfunding data from Excel files. Four different CSV files were created and then used to generate an ERD (Entity Relationship Diagram) and Table Schema. This information was then extracted and loaded into a Postgres database to analyze the data and get insight information.

For the cleaning portion of the project, we had to create new columns to show category and subcategory ids to have unique identifiers for the data and be able to have primary and foreign keys in the new tables that were created. In addition to this, we also had to split a column that included both the category and subcategory items together. After we split the information, the combined column was deleted and therefore we ended up with two clean columns with single items. Another step in the cleaning process was to transform the data into the correct formatting such as dates and integers. The main reason why we had to create four different tables is to eliminate data redundancy by creating primary keys outside the main data table.

Our Crowdfunding Data Base included four different tables (campaign, contacts, category, and subcategory). Using these four tables we were able to get insight information by making joins and aggregations to understand the relationship between datasets. The campaign table has the majority of the data. It shows the Crowdfunding data such as pledge amounts, goals, countries, and company name among other information. We used the foreign keys category_id, subcategory_id, and contact_id to connect to the category, subcategory, and contacts tables. Some of the insight information we were able to discover

includes the category with the highest single pledged which turned out to be “Film & Video”. Another interesting finding was that the US had both the minimum and maximum pledged amounts. Also that Canada has the highest average pledged (Fig .1) amount with \$63,927 and that there were 201 more successful outcomes compared to failing ones.

For future work on this data base we suggest using Python to update CSV files into Postgres in order to automate the process of updating new information.

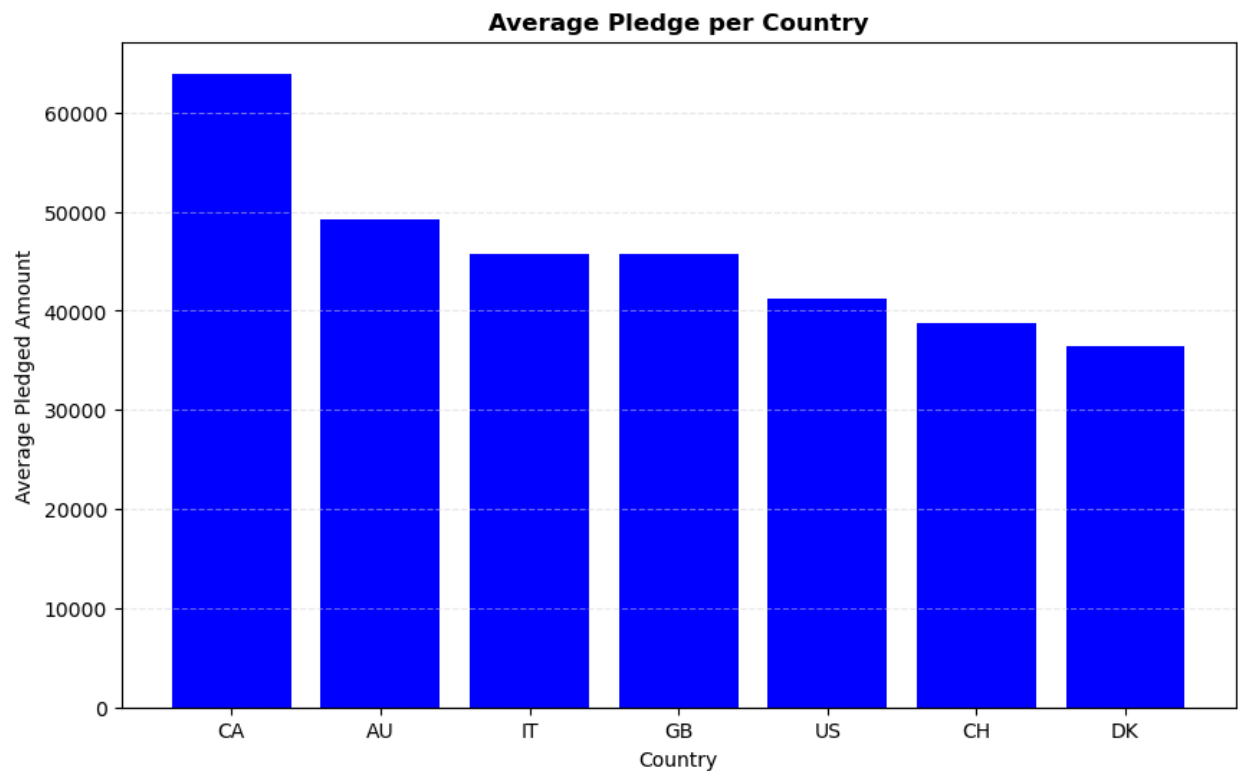
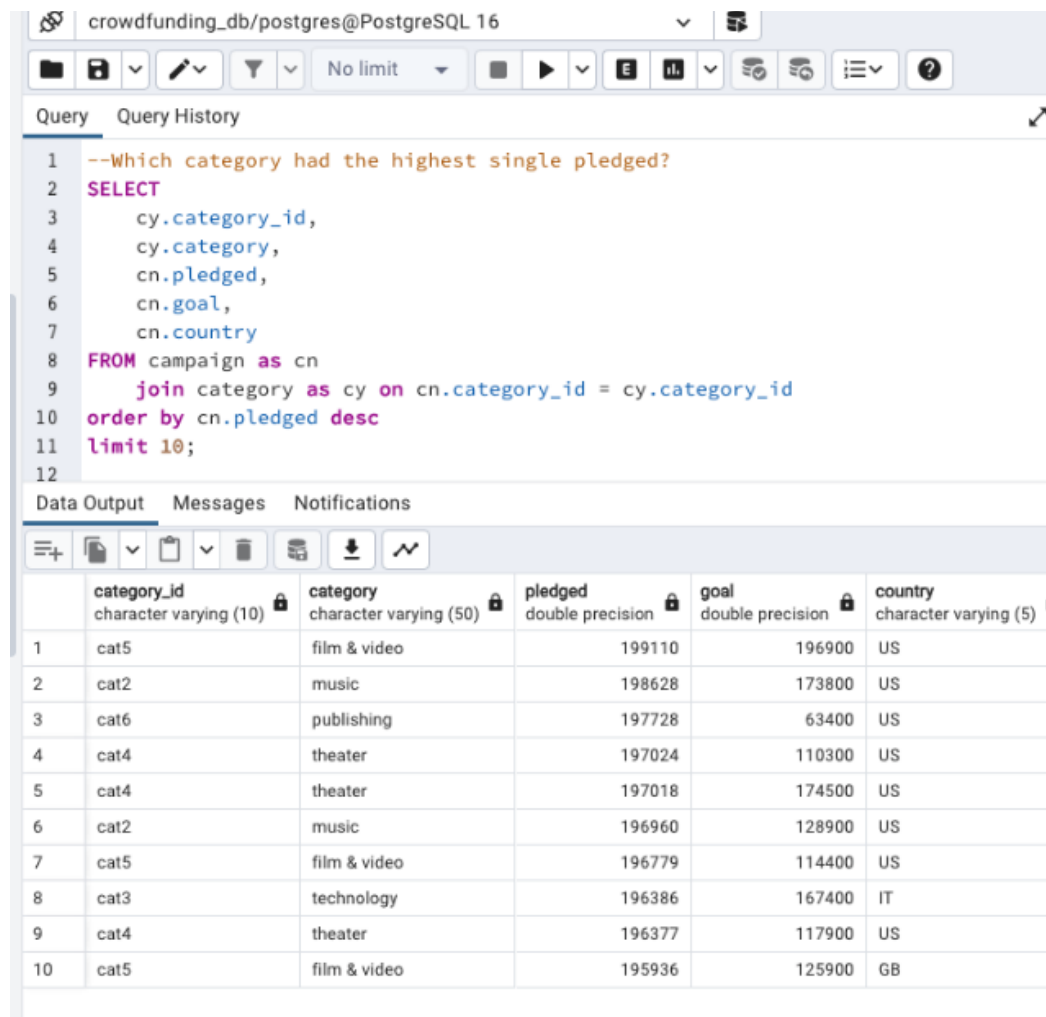


Fig. 1

Query Summary:

Query 1: Which category had the highest single pledged?

A: "Film & Video"



The screenshot shows a PostgreSQL query editor interface. The query is as follows:

```
1 --Which category had the highest single pledged?
2 SELECT
3     cy.category_id,
4     cy.category,
5     cn.pledged,
6     cn.goal,
7     cn.country
8 FROM campaign as cn
9     join category as cy on cn.category_id = cy.category_id
10 order by cn.pledged desc
11 limit 10;
```

The results are displayed in a table with the following columns: category_id, category, pledged, goal, and country. The table shows the top 10 results, with 'film & video' being the category with the highest pledged amount.

	category_id character varying (10)	category character varying (50)	pledged double precision	goal double precision	country character varying (5)
1	cat5	film & video	199110	196900	US
2	cat2	music	198628	173800	US
3	cat6	publishing	197728	63400	US
4	cat4	theater	197024	110300	US
5	cat4	theater	197018	174500	US
6	cat2	music	196960	128900	US
7	cat5	film & video	196779	114400	US
8	cat3	technology	196386	167400	IT
9	cat4	theater	196377	117900	US
10	cat5	film & video	195936	125900	GB

Query 2: What is the average pledge amount for each country?

A: AU \$49K, CA \$64K, DK \$36K, CH \$39K, US \$41K, GB \$46K, and IT \$46K

The screenshot shows a PostgreSQL query editor interface. The query editor displays the following SQL query:

```
--What is the average pledged amount for each country?
SELECT
  AVG(pledged)::NUMERIC(10,2) AS avg_pledged,
  country
FROM
  campaign
group by country;
```

Below the query editor, the "Data Output" tab is active, showing a table with the results of the query. The table has two columns: "avg_pledged" (numeric(10,2)) and "country" (character varying(5)). The results are as follows:

	avg_pledged numeric(10,2)	country character varying(5)
1	49175.37	AU
2	63927.00	CA
3	36421.55	DK
4	38757.17	CH
5	41165.58	US
6	45681.35	GB
7	45795.88	IT

Query 3: Which country pledged the highest amount?

A: US with \$199,110

The screenshot shows a PostgreSQL query editor interface. The top navigation bar includes tabs for Dashboard, Properties, SQL, Statistics, Dependencies, Dependents, and Processes. The current connection is 'crowdfunding_db/postgres@PostgreSQL 16'. The SQL editor contains the following query:

```
40
41 --What country had the maximum pledged amount?
42 SELECT
43     MAX(pledged) AS max_pledged,
44     country
45 FROM
46     campaign
47 group by country
48 order by max(pledged) DESC;
49
50
51
```

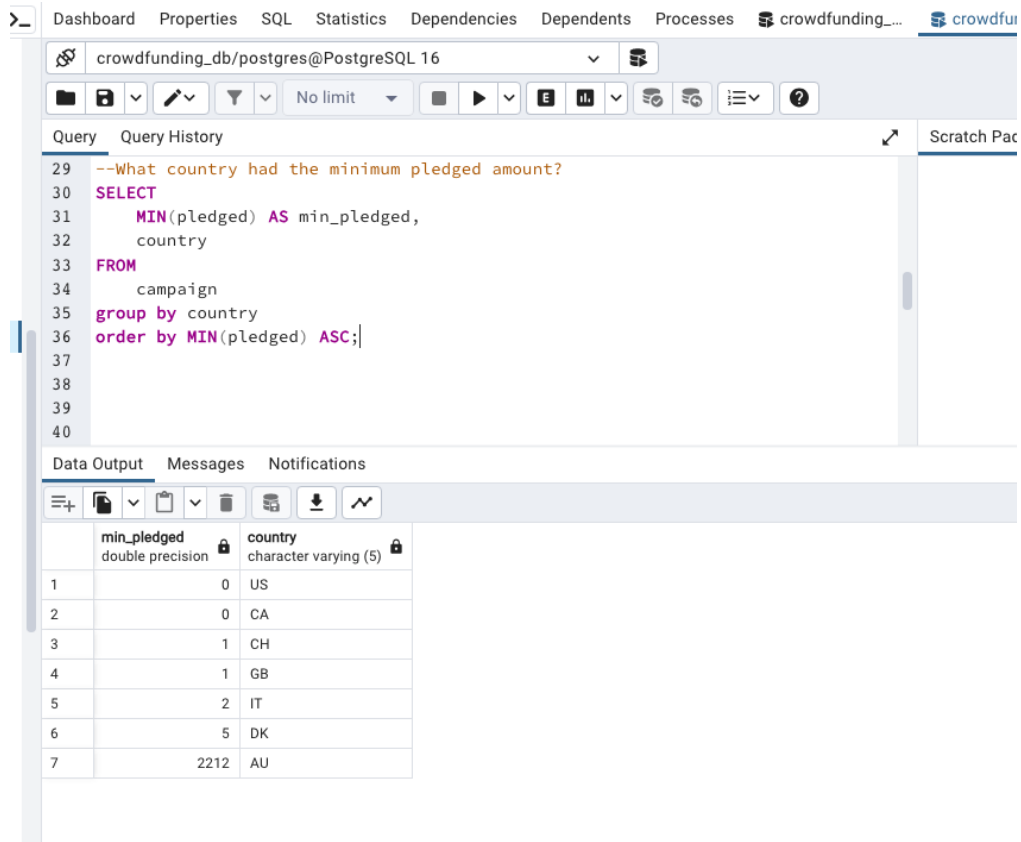
Below the query editor, the 'Data Output' tab is active, displaying the results of the query in a table format:

	max_pledged double precision	country character varying (5)
1	199110	US
2	196386	IT
3	195936	GB
4	195750	DK
5	194309	CA
6	193413	AU
7	178338	CH

At the bottom of the interface, it indicates 'Total rows: 7 of 7' and 'Query complete 00:00:00.092'.

Query 4: Which country pledged the lowest amount?

A: US and Canada with \$0



The screenshot shows a PostgreSQL query editor interface. The top navigation bar includes tabs for Dashboard, Properties, SQL, Statistics, Dependencies, Dependents, Processes, and a connection named 'crowdfunding_...'. The main editor area displays a SQL query to find the minimum pledged amount by country. The query is as follows:

```
--What country had the minimum pledged amount?
SELECT
  MIN(pledged) AS min_pledged,
  country
FROM
  campaign
group by country
order by MIN(pledged) ASC;
```

Below the query editor, the 'Data Output' tab is active, showing the results of the query. The results are displayed in a table with two columns: 'min_pledged' (double precision) and 'country' (character varying (5)). The table contains 7 rows of data, sorted by the minimum pledged amount in ascending order.

	min_pledged double precision	country character varying (5)
1	0	US
2	0	CA
3	1	CH
4	1	GB
5	2	IT
6	5	DK
7	2212	AU

Query 5: How many successful outcomes were there?

A: 565

The screenshot shows a PostgreSQL query editor interface. The top bar indicates the connection is to 'crowdfunding_db/postgres@PostgreSQL 16'. The query editor contains the following SQL code:

```
51
52 --How many successful outcomes were there?
53 SELECT
54     count(outcome) as count_successful
55 FROM
56     campaign
57 WHERE outcome = 'successful';
58
59
60
61
62
```

Below the query editor, the 'Data Output' tab is active, showing a single row of results:

	count_successful bigint
1	565

At the bottom of the interface, a status bar shows 'Total rows: 1 of 1' and 'Query complete 00:00:00.077'.

Query 6: How many failed outcomes were there?

A: 364

The screenshot shows a PostgreSQL query editor interface. The top navigation bar includes tabs for Dashboard, Properties, SQL, Statistics, Dependencies, Dependents, and Processes. The current database is 'crowdfunding_db/postgres@PostgreSQL 16'. The query editor shows the following SQL query:

```
--How many failed outcomes were there?
SELECT
    count(outcome) as count_failed
FROM
    campaign
WHERE outcome = 'failed';
```

The query is executed, and the results are displayed in the 'Data Output' tab. The results show a single row with the column 'count_failed' of type 'bigint' and the value 364.

count_failed
364

At the bottom of the interface, a status bar indicates 'Total rows: 1 of 1' and 'Query complete 00:00:00.046'. A green notification box on the right states '✓ Successfully run. Total query r'.

ERD:

