SQL & Relational Databases - Final Report
Group #2: TED Talks

Introduction & Project Overview:

We are a small consulting team working with a well-known research university to better understand how they can improve their marketing effectiveness by using digital channels and posting educational videos on the web. The client ultimately wants to help its professors and researchers land speaking gigs at prestigious conferences and gain status in higher academic circles. The client wants to use the TED Talk dataset to better understand how different qualities of videos (e.g. length, title, descriptions, topics) affect their views and ratings numbers. The client can apply these insights to improve the qualities of their own videos in hopes of gaining more views and getting better ratings.

TED Talks are currently one of the main ways that novel research ideas are shared on the internet. TED, which has been available for free online since 2006, has hundreds of millions of views on its videos and is known for spreading new ideas in technology, education, design. Our client's goal is to post their own videos online that can spread learning to a greater number of people and help the university market itself through new channels. If professionals and researchers associated with the university are able to gain popularity online, this will translate into better opportunities for them to expand their prestige in the academic world and land speaking gigs at conferences such as TED, which in turn will improve the reputation of the university. This dataset can help us in this task by informing the client on how they should present the videos online, which factors to focus on to maximize views and ratings, and which topics to discuss that are most popular amongst viewers. Whether they are posting through TED or another online channel, this information should help the organization understand how to improve its digital presence.

The decision to use this dataset for our client engagement was made after some research about TED Talks was conducted. Up to date, there are about 3000 TED Talks free online, creating 11 billion total number of views on all TED Talks and about 3.2 billion views last year alone. TED.com originated from an annual conference in the 1990s and its early emphasis was on technology and design, consistent with its Silicon Valley origins. As a result, many ideas shared on the platform involve cutting edge ideas and technology, as it is quite common for researchers to share their recent research achievements through TED Talks. With so many views on each TED talk and ample feedback from the users, such as comments and ratings, we have a great opportunity to help our client understand how different aspects of online educational videos are related to the quality of ratings and number of views.

The first step for our team will be to better understand the data at hand by performing some exploratory data analyses. This process should highlight various features of the data including the data structure, any outliers, trends across variables, missing data, amongst others. We may also use feature engineering, in addition to text mining on the character fields, to come up with new variables that could be beneficial to our analysis and provide deeper insights. Based on these insights gained from our preliminary analyses, we will attempt to narrow down the number of variables to include those that are most relevant to the business problem, as well as the optimal way to store our data in a single schema. We will then use various modeling techniques to determine which features better predict higher user ratings and views.

The insights derived from this project will enable the client identify key features that will influence audience engagement with the content they put out. They will be able to put out more targeted content that will increase professors recognition in the academic and related industries as well as the university's public image. This will in turn create additional revenue for both parties as professors would potentially be invited to other paid global conferences and speaking engagements, and the university would have created a more prestigious image for the caliber of education provided which could increase student enrollment.

Team Contract:

The following tasks were completed by the team in previous weeks as part of the checkpoints:

- Build an ER diagram for the finalized database schema **(Leader: Nally)**
- Develop SQL, python or R code for assigned segment **(Leader: Casey)**
- Develop specific plans for transforming and entering the data to the database system **(Leader: Junyi)**
- Deliver plans for client interaction with database **(Leader: Echo)**
- Develop implementation for the research university in order to increase student enrollment **(Leader: Agnes)**
- Develop analytical processes, interactive dashboard redundancy, and performance **(Leader: TBD)**
    - Specific tasks will depend on results from previous stages
    - Multiple co-leaders assigned to focus on different aspects of task
- Draft report and presentation (sections broken up based on individual tasks below)

Individual Task Delegations

The information below details the tasks for our project that each team member will lead and co-lead in future weeks. Leaders will have primary responsibilities for their given tasks, and co-leaders will work alongside the leaders to ensure these tasks are completed on time. Note that all group members will contribute to building the tables for our finalized schema in Checkpoint 3. Also note that all group members will be heavily involved in the final report and presentation, with each person contributing mostly to the sections that pertain to the checkpoint tasks completed.

Agnes:

- Explore the dataset and build 3 of the tables from the 3NF schema shown below
- Lead: Develop implementation for the research university in order to increase student enrollment
- Co-Lead: Develop specific plans for transforming and entering the data to the database system
- Co-Lead: Develop analytical processes, interactive dashboard redundancy, and performance
- Draft report and presentation

Casey:

- Explore the dataset and build 3 of the tables from the 3NF schema shown below
- Lead: Develop SQL, python or R code for assigned segment
- Co-lead: Develop specific plans for transforming and entering the data to the database system
- Co-lead: Build ER diagram for finalized database schema
- Draft report and presentation

Echo:

- Explore the dataset and build 3 of the tables from the 3NF schema shown below
- Lead: Deliver plans for client interaction with the database
- Co-lead: Develop implementation for the research university in order to increase student enrollment
- Co-lead: Develop analytical processes, interactive dashboard redundancy, and performance
- Draft report and presentation

Junyi:

- Explore the dataset and build 3 of the tables from the 3NF schema shown below

- Lead: Develop specific plans for transforming and entering the data to the database system
- Co-lead: Develop SQL, python or R code for assigned segment
- Co-lead: Develop analytical processes, interactive dashboard redundancy, and performance
- Draft report and presentation

Nally:

- Explore the dataset and build 3 of the tables from the 3NF schema shown below
- Lead: Build an ER diagram for finalized database schema
- Co-lead: Develop implementation for the research university in order to increase student enrollment
- Co-lead: Develop SQL, python or R code for assigned segment
- Draft report and presentation

Timeline:

We will have a zoom/in-person group meeting once a week to check up on where we are in our assigned tasks, discuss any challenges, or modifications that need to be made going forward, any new insights, and finalize deliverables for Checkpoints. We will use the group chat to keep each other updated in between group meetings.

Below is the list of items we should have completed by each meeting.

Date	**To be completed**
11/10/19	● Checkpoint 2 ○ Finalize target dataset ○ Final normalization plan ○ Delegation of tasks

11/17/19	• Checkpoint 3
	  ○ Database schema
	  ○ ER Diagram
	• EDA
	• Ideas for feature engineering

11/24/19	• Checkpoint 4 ○ Data transformation / ETL using Python and/or R for tables
12/01/19	• Checkpoint 5 ○ Plan for customer interaction with database • Draft/outline for presentation report and slides
12/08/19	• Deliver final report and slides

Team Expectations:

- Complete assigned tasks by deadline and inform group members of any delays or challenges prior well in advance
- If a group member cannot complete their assigned task, they must make up this work on other sections to ensure that each person takes on a similar workload.

Our group fully acknowledges our commitment to the project, the tasks described above, and the expected timeline for completion. We agree to abide by these guidelines to achieve success in our final project.

Data Overview & Sample:

Dataset:
· **Description**: This data contains information about all audio-video recordings of TED Talks uploaded to the official TED.com website until September 21st, 2017. The TED main dataset contains information about all talks including number of views, number of comments, descriptions, speakers and titles. The TED transcripts dataset contains the transcripts for all talks available on TED.com. For the purposes of this project, these datasets will be combined in the same database schema.
Source: The dataset was sourced from Kaggle at the link below:
https://www.kaggle.com/rounakbanik/ted-talks (Links to an external site.)

Type: See below for a brief description of each column in the data.

- comments: The number of first level comments made on the talk
- description: A blurb of what the talk is about
- duration: The duration of the talk in seconds
- event: The TED/TEDx event where the talk took place
- film_date: The Unix timestamp of the filming
- languages: The number of languages in which the talk is available
- main_speaker: The first named speaker of the talk
- name: The official name of the TED Talk. Includes the title and the speaker.
- num_speaker: The number of speakers in the talk
- published_date: The Unix timestamp for the publication of the talk on TED.com
- ratings: A stringified dictionary of the various ratings given to the talk (inspiring, fascinating, jaw dropping, etc.)
- related_talks: A list of dictionaries of recommended talks to watch next
- speaker_occupation: The occupation of the main speaker
- tags: The themes associated with the talk
- title: The title of the talk
- url: The URL of the talk
- views: The number of views on the talk
- transcript: The official English transcript of the talk

Extent:

- Main dataset has 2550 tuples and 17 attributes
- Transcripts table is smaller at 2464 tuples and 2 attributes
- Transcripts joins with main dataset on URL (18 distinct features in total)

** Please see final page of report for ER Diagram from Lucidchart **

Normalization Plan:

The main goal in our normalization plan was to break up the tables in the original data set into a 3NF database, as reflected in our ER diagram. We combined the transcript data set to our original one and with our data in 3NF, we ended up with 15 tables. To achieve 1NF, we first ensured there were no repeating rows in the original dataset so that every record was unique and each cell only contained one value. Since all videos in the dataset were unique there were no repeating rows so we created a videos table and assigned each video a unique identifier. The ratings and related_talks tables contained nested json objects that we had to extract which resulted in duplicate rows as we had to ensure there was only one value in each cell for each

record. We removed all duplicates, assigned unique identifiers, and created a table that held only unique records to increase database performance and avoid wasting space. The tags and speaker_occupation columns contained lists that we also had to extract, assign unique identifiers to, and remove duplicates from to make the data look cleaner and ensure easier querying.

To get the data in 2NF we made sure all tables had a unique identifier that every other attribute in that table was fully dependent on, and also made sure that for all tables with composite keys, all other attributes in that table were fully dependent on the entire composite key and not just part of it. We transformed tables such as main_speakers to include information that was more relevant to our business case and that would make writing queries easier such as assigned them an id and including personal information like occupations. We merged comments, views, and duration to create a more insightful table called video_stats that contained information about each video's performance and audience engagement. We clearly identified foreign keys and made sure they related back to the right tables to avoid any errors. Once all this was put in place, we had also achieved 3NF.

ETL Description:

Once we merged the transcripts and main datasets together to create a main data frame, and ensured there were no duplicates, we cleaned up the data a little using Python. We changed date formats from Unix timestamps to DD-MM-YY format, checked for null values and made all of them blank instead. The code below shows this process.

```
from datetime import datetime
data['film_date'] = data['film_date'].apply(lambda x: datetime.fromtimestamp( int(x)).strftime('%d-%m-%Y'))
data['published_date'] = data['published_date'].apply(lambda x: datetime.fromtimestamp(
int(x)).strftime('%d-%m-%Y'))

data["transcript"][data["transcript"].isnull()] = ""
```

We first created the videos table and dropped all duplicates so we could easily index the table which would serve as unique identifiers for each video. We then joined this table back to the main data so that the creation of all other tables will reference the right video. For each table we created, we joined it back to the main data frame to have all data in a central location and make it easier to create subsequent tables. For the events table, we had to create start and end dates attributes for each event and assign an id. A sample of this table is depicted below.

	id	event	start_date	end_date
1	1	AORN Congress	14-03-2009	14-03-2009
2	2	Arbejdsglaede Live	04-05-2009	04-05-2009
3	3	BBC TV	08-07-1983	08-07-1983
4	4	Bowery Poetry Club	12-11-2005	12-11-2005
5	5	Business Innovation Factory	06-10-2009	09-10-2009

For the occupations table, we extracted all unique occupations from the main table, deleted all duplicates, and assign each occupation an id. Since we did not have individual languages for each video but only the language count, we created the language table using only the video_id and languages attributes. Creating the main_speakers table resulted in duplicate rows so they all had to be dropped before an id could be assigned to each speaker. The presentations, url_links, transcripts, dates, and video_stats table were more straightforward and made by extracting all pre-decided attributes from the main data frame, with all duplicate rows dropped.

To create the video_tags table, all individual tags had to be extracted from their list. Doing so resulted in duplicates as the table contained every tag for each video as a new record so in creating the tags table. To use this data to create the tags table, all duplicates had to be dropped and then ids were assigned to each tag. These ids were then merged with the video_tags table, the columns were renamed, and the tag column was dropped as it was no longer relevant.

The nested jsons in related_talks were extracted and transformed into a data frame and all duplicates were dropped. We followed the same process for ratings where each rating was extracted from the json object and all duplicates were dropped so the table only contained unique ratings and their assigned ids. Video_ratings was then created matching video ids to their various rating ids. The tables and data were then pushed into the database using the sample code below. The detailed code for the ETL process can be found here.

```
connection = engine.connect()
stmt = '''
CREATE TABLE videos (
    id integer,
    title varchar(150) NOT NULL,
    description text,
    PRIMARY KEY (id)
);
```

Analytical Procedures:

Since our client is interested in understanding the aspects of our project that relate to video views, we have taken preliminary steps to conduct an exploratory data analysis for features that may be related to the outcome variable. To simulate how a real database user would conduct an analysis, we are using a combination of Postgres and Python, including the SQL Alchemy, Pandas, Matplotlib, and Seaborn packages. Data is queried from the database using SQL Alchemy, stored in a Pandas dataframe, and visualized with Matplotlib and/or Seaborn. Specific parts of the EDA are detailed below:

Procedure 1: What is the distribution of the outcome variable – views?

To help the client with their goal of producing popular online learning videos, we must first understand how many views the TED Talks in our database generated. Let's examine the views variable from the video_stats table.

Data retrieval:

```
# query videos and video stats

stmt = '''SELECT * FROM videos, video_stats
          WHERE videos.id = video_stats.video_id;'''
```

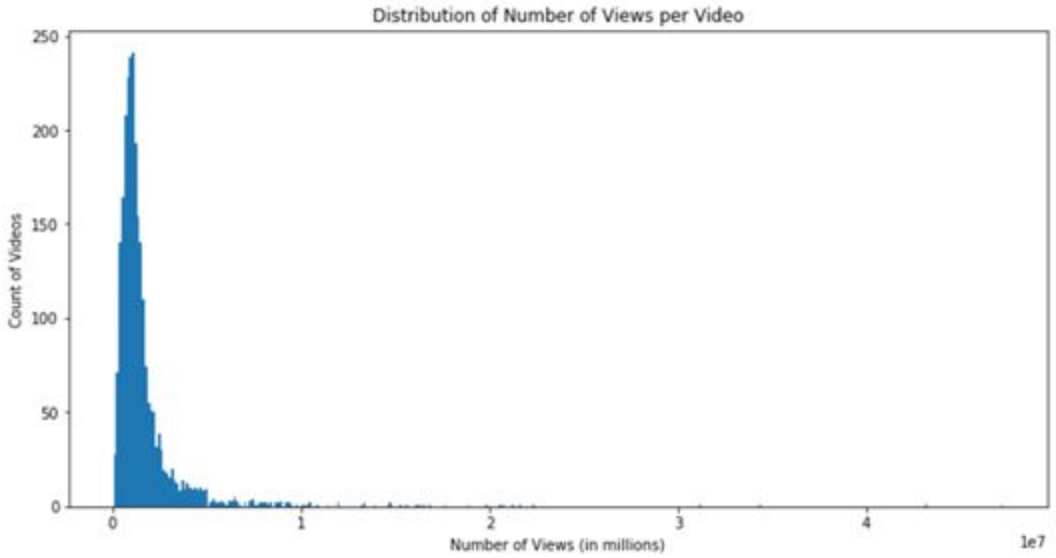
```
# create dataframe

video_data = pd.read_sql(stmt, engine)
```

```
video_data.head()
```

	id	title	description	video_id	duration	num_speaker	views	comments
0	0	Do schools kill creativity?	Sir Ken Robinson makes an entertaining and pro...	0	1164	1	47227110	4553
1	1	Averting the climate crisis	With the same humor and humanity he exuded in ...	1	977	1	3200520	265
2	2	Simplicity sells	New York Times columnist David Pogue takes aim...	2	1286	1	1636292	124
3	3	Greening the ghetto	In an emotionally charged talk, MacArthur-winn...	3	1116	1	1697550	200
4	4	The best stats you've ever seen	You've never seen data presented like this. Wi...	4	1190	1	12005869	593

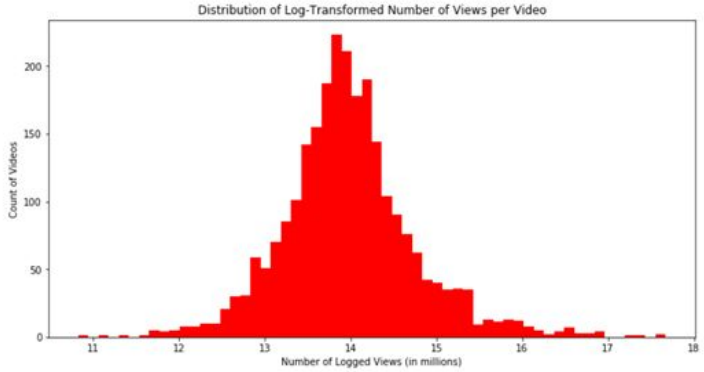
Data visualization:

```
plt.figure(figsize = (12,6))
plt.hist(video_data['views'], bins = 'auto')
plt.title('Distribution of Number of Views per Video')
plt.xlabel('Number of Views (in millions)')
plt.ylabel('Count of Videos')
plt.show()
```



Procedure 2: The outcome variable looks highly skewed to the right – would it look better under a log transformation?

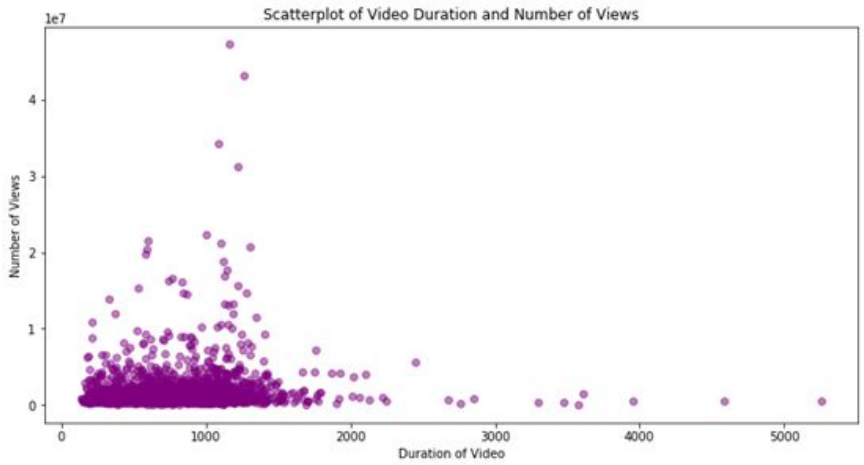
```
# data heavily skewed right - lets try a log transformation
plt.figure(figsize = (12,6))
plt.hist(np.log(video_data['views']), bins = 'auto', color = 'r')
plt.title('Distribution of Log-Transformed Number of Views per Video')
plt.xlabel('Number of Logged Views (in millions)')
plt.ylabel('Count of Videos')
plt.show()
```



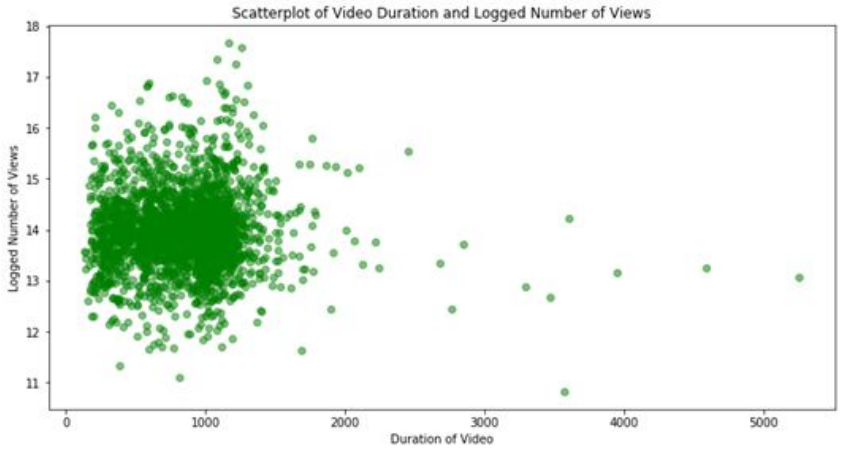
The distribution of the outcome variable takes on a much more Gaussian form when we apply a log transformation. If we decide to use a parametric method to build a machine learning model, we should probably apply this transformation before we do so. This will help to eliminate outliers from the training examples and ultimately generate predictions about video view statistics that are more accurate. Let's now check out some of the predictor variables at our disposal.

Procedure 3: How does the duration of a video affect its number of views?

One of the other features in the queried table is video duration. Let's check it out in a scatterplot to see if we can observe any relationships in the data.



Unfortunately, most observations are bunched in the bottom right and we cannot easily see a relationship in the data. Does it look any different under a log transformation as performed in Procedure 2?



Not really – the observations are certainly more spread out than they were previously, but there are no easily identifiable trends that could benefit us for predictive modeling.

Let's do some feature engineering to create new variables.

Procedure 4: Feature Engineering on Video Data

The flexibility of Python and Pandas allows us to easily create new columns using a few short lines of code. In this case, we are interested in seeing whether the length of a video title or description have an effect on its view count. In this part, we go back to the video_data table from previous procedures to make new columns for the length of their titles and descriptions (measured in number of characters).

```
# create title and description length columns in video data

video_data['title_length'] = video_data['title'].apply(lambda x: len(x))

video_data['description_length'] = video_data['description'].apply(lambda x: len(x))
```

```
video_data.head()
```

	id	title	description	video_id	duration	num_speaker	views	comments	title_length	description_length
0	0	Do schools kill creativity?	Sir Ken Robinson makes an entertaining and pro...	0	1164	1	47227110	4553	27	149
1	1	Averting the climate crisis	With the same humor and humanity he exuded in ...	1	977	1	3200520	265	27	233
2	2	Simplicity sells	New York Times columnist David Pogue takes aim...	2	1286	1	1636292	124	16	202
3	3	Greening the ghetto	In an emotionally charged talk, MacArthur-winn...	3	1116	1	1697550	200	19	213
4	4	The best stats you've ever seen	You've never seen data presented like this. Wi...	4	1190	1	12005869	593	31	172

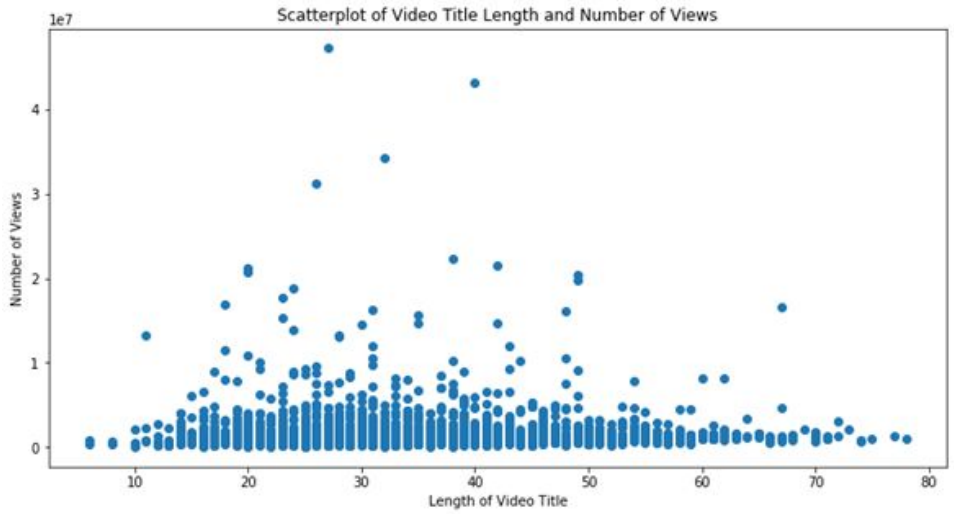
Looking at the current data, it may be helpful to also include new features that count the number of words, not characters, that appear in each field. We update the table below:

```
video_data['title_length_words'] = video_data['title'].apply(lambda x: len(x.split(" ")))

video_data['description_length_words'] = video_data['description'].apply(lambda x: len(x.split(" ")))
```

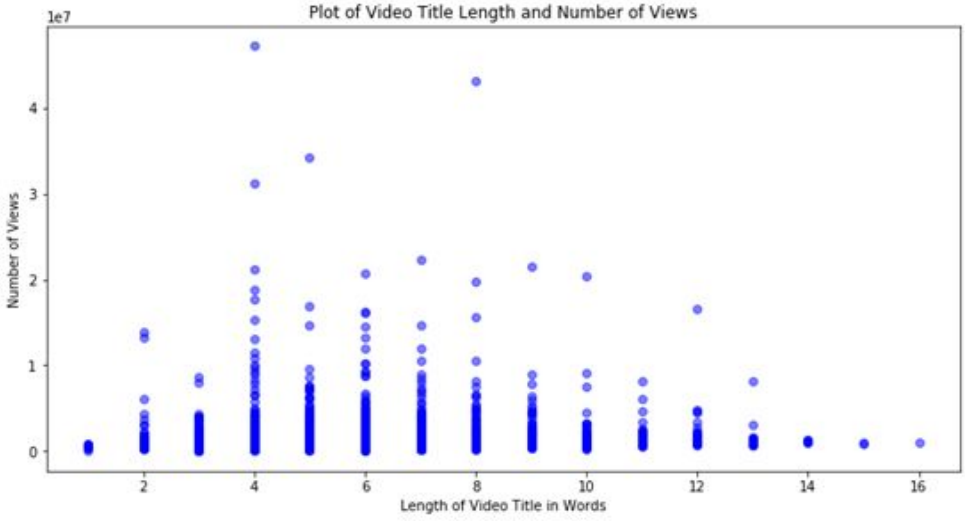
Now that the new features are ready, we can continue to explore them through plots and graphs.

Procedure 5: Visualizing relationship between title length and view count

We seek to know if the length of a video's title has any significant effect on its view count. To better understand the relationship, we can once again use a scatter plot in Matplotlib to visualize the data.

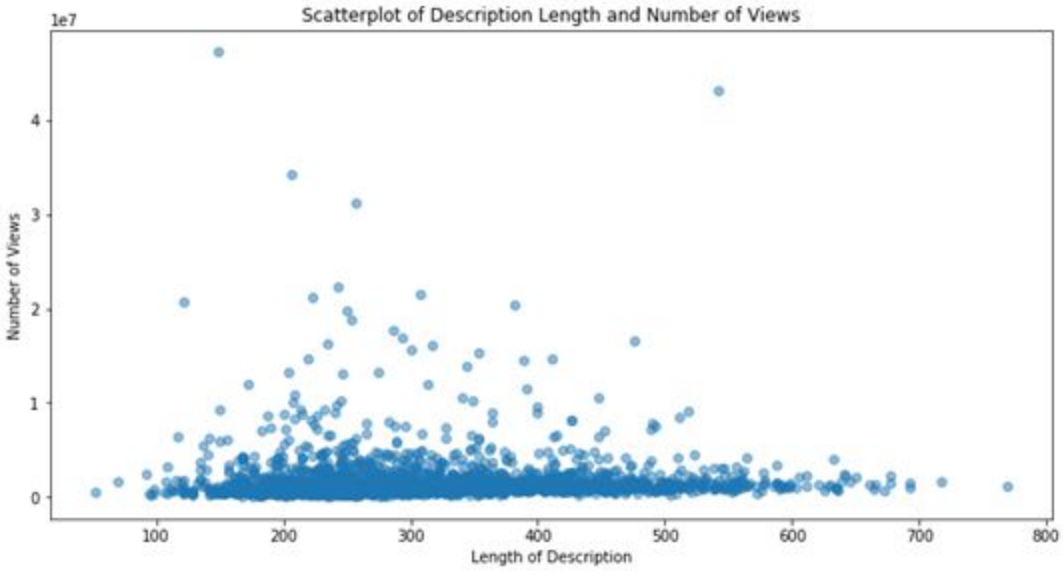


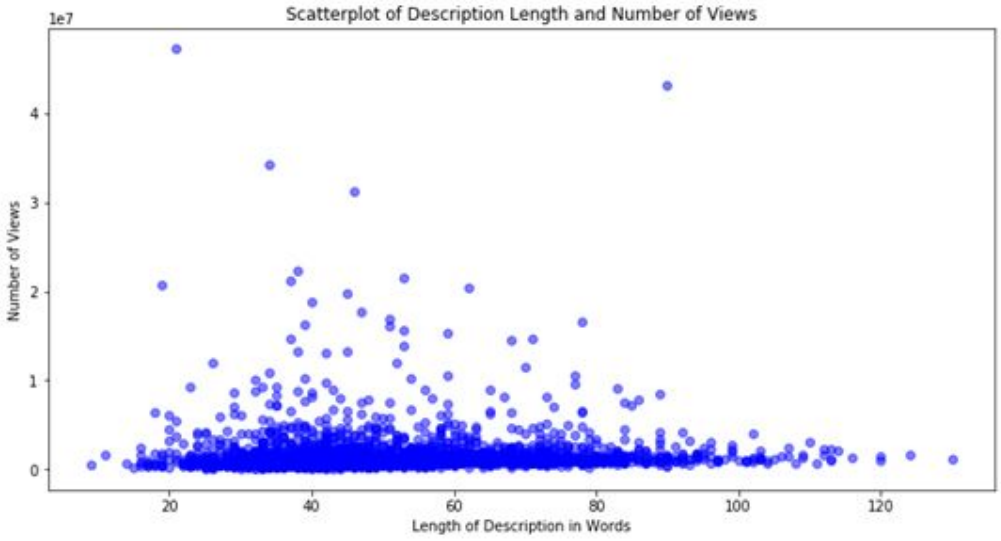
We cannot see any significant increase or decrease in the outcome variable at title length increases. Let's try again with the new feature that measures the number of words used instead of characters:



Despite the new feature engineering efforts, the plot still shows no strong relationship between title length and number of views. The videos that show the most views here are also in the range of values that are most common for observations in our database, so it is not entirely clear if these would be useful predictors for modeling endeavors. Let's check out the descriptions to see if we get better results.

Procedure 6: Visualizing relationship between description length and view count





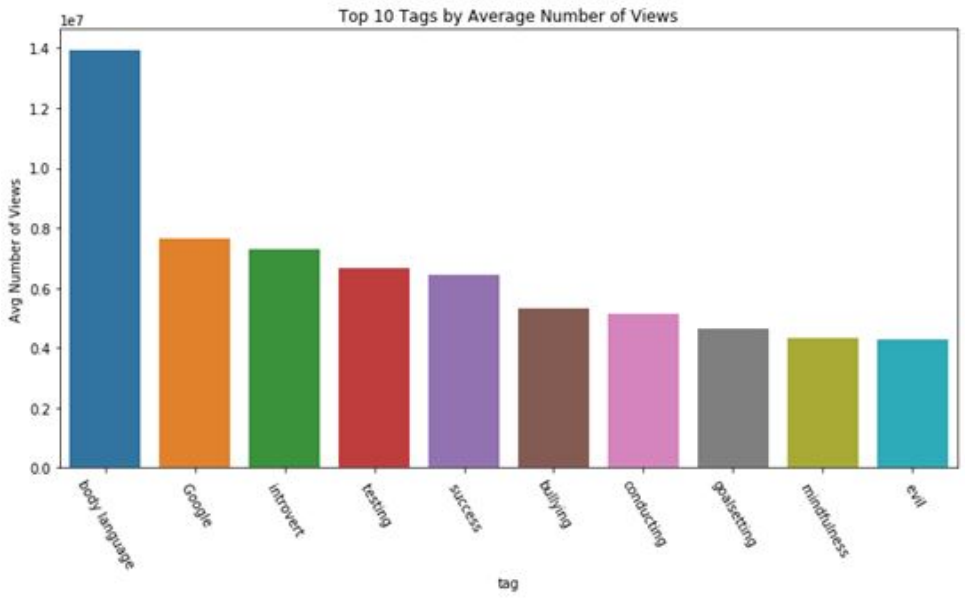
The plots above show a slight relationship between description lengths and view counts. Through the similar distributions in both plots, we can see that videos with the highest views tended to have description lengths between 25 and 60 words, or alternatively, between 150 and 400 characters. These insights could be actionable for our client and help generate some initial best practices for the videos they post.

Let's continue to explore the data and check out the tags on each video.

Procedure 7: Which tags were included on the videos with the highest average number of views?

One of the features captured in our database is tags, which has 594 distinct values. We can combine the video stats and tags data to generate summary statistics about each tag and see which has the highest average view counts.

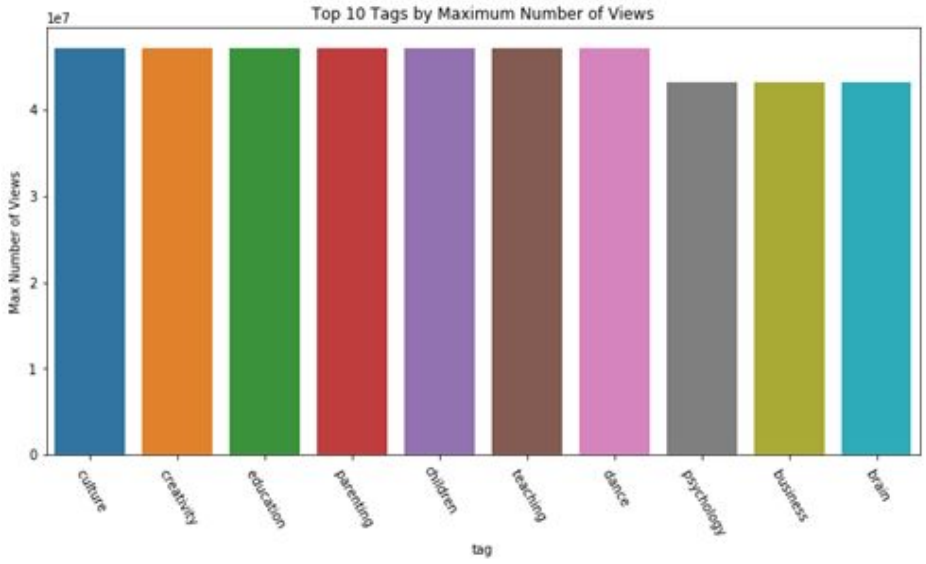
```
stmt = '''SELECT tag, AVG(views) AS avg_views, MAX(views) AS max_views FROM video_stats, tags, video_tags
          WHERE video_tags.video_id = video_stats.video_id
              AND video_tags.tag_id = tags.id
          GROUP BY tag;'''
```



Interestingly, body language had an exceptionally high average view count at ~1.4 million. The next highest topic, Google, was significantly lower at ~800k. It is worthwhile to note that body language is a rather unique tag, so a few outlier videos may be pulling this number up significantly compared to other tags.

Procedure 8: Which tags had the highest maximum number of views?

We can take a similar approach as we did on the above visual to examine which tags were included on the videos with the most views in our database.



This visual is less informative than the summary by average, since all it tells us is that the most viewed video had the seven tags on the left included on it. Nonetheless, it tells us that certain unique topics, like children and dance, may have the potential to generate lots of interest in the online learning community.

Procedure 9: Clean transcripts text data

One of the most important features in our database is the transcripts table, which tells us exactly what was said during each TED Talk. We can pull this info in with videos and video stats to make a new data frame in Python.

```
stmt = '''SELECT videos.id, videos.title, transcripts.transcript, video_stats.views
          FROM videos, video_stats, transcripts
          WHERE videos.id = video_stats.video_id
              AND video_stats.video_id = transcripts.video_id;'''
```

	id	title	transcript	views
0	0	Do schools kill creativity?	Good morning. How are you?(Laughter)It's been ...	47227110
1	1	Averting the climate crisis	Thank you so much, Chris. And it's truly a gre...	3200520
2	2	Simplicity sells	(Music: "The Sound of Silence," Simon & Garfun...	1636292
3	3	Greening the ghetto	If you're here today — and I'm very happy that...	1697550
4	4	The best stats you've ever seen	About 10 years ago, I took on the task to teac...	12005869

Now that the important information is all in one place, we need to clean the transcript data. Every NLP project is at least slightly different, so our team decided to take a few major steps to

clean the data in a way that is usually successful in other similar projects. The steps to clean the text data included:

- Removing stop words using NLTK
- Lemmatizing words using NLTK's Wordnet Lemmatizer
- Making all letters lower-case
- Splitting words into a list

This was all accomplished in one function, as shown below:

```
def clean_text(t):

    text = ''.join(x for x in t if x not in string.punctuation)

    tokens = re.split('\W+', text)

    text = [wn.lemmatize(word.lower()) for word in tokens if word not in stops]

    return text
```

Once ready, this function was applied to the transcripts column to make a new column – clean transcripts. A sample of this column is shown below next to the original title and transcript columns:

```
text_data[['title', 'transcript', 'clean_transcript']].head()
```

	title	transcript	clean_transcript
0	Do schools kill creativity?	Good morning. How are you?(Laughter)It's been ...	[good, morning, how, youlaughterits, great, ha...
1	Averting the climate crisis	Thank you so much, Chris. And it's truly a gre...	[thank, much, chris, and, truly, great, honor,...
2	Simplicity sells	(Music: "The Sound of Silence," Simon & Garfun...	[music, the, sound, silence, simon, garfunkelh...
3	Greening the ghetto	If you're here today — and I'm very happy that...	[if, youre, today, im, happy, youve, heard, su...
4	The best stats you've ever seen	About 10 years ago, I took on the task to teac...	[about, 10, year, ago, i, took, task, teach, g...

Procedure 10: Vectorizing transcripts using Scikit Learn

Another task that would be highly relevant to our project goal would be vectorizing the words in the transcript. Vectorization is a process that converts text data into sparse matrices by recording where certain words exist in each observation. In our case, we used a Count Vectorizer from the sklearn package to vectorize the transcripts in our text data object. The initial parameters specified that transcripts would be cleaned with our clean_text function and that words would only be included if they showed up in at least 3 videos; 26217 distinct words were included in the output matrix:

```
# create instance of CountVectorizer()

# only using words that show up in at least 3 different transcripts

count_vec = CountVectorizer(analyzer=clean_text, min_df=3)
```

```
# fit and transform the data

X_counts = count_vec.fit_transform(text_data['transcript'])
```

```
# shape of document term matrix - 26217 unique words counted

X_counts.shape
```

```
(2550, 26217)
```

Since we probably don't want to include all 26000+ columns in a predictive model, we increased the threshold to 10 and examined how much that reduced the size of the matrix.

```
# only using words that show up in at least 10 different transcripts

count_vec = CountVectorizer(analyzer=clean_text, min_df=10)
```

```
# fit and transform the data

X_counts = count_vec.fit_transform(text_data['transcript'])
```

```
# shape of document term matrix - 11763 unique words counted

X_counts.shape
```

```
(2550, 11763)
```

Our threshold increase was successful, as the number of columns (words) was reduced to 11763. Now that we have a more manageable count vectorized document term matrix, we can convert it to a Pandas dataframe for further analysis:

```
xcounts_df = pd.DataFrame(X_counts.toarray(), columns = count_vec.get_feature_names())
```

```
xcounts_df.head()
```

	0	01	05	1	10	100	1000	10000	100000	...	zerosum	zimbabwe	zip	zombie	zone	zoning	zoo	zoom	zooming	zurich
0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	1	0	0	0	...	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	2	...	0	0	1	0	0	2	0	0	0	0
4	0	0	0	0	0	4	3	1	0	...	0	0	0	0	0	0	0	0	0	0

5 rows × 11763 columns

With our data vectorized for each transcript, predictive models may now be run on this data. Other types of vectorization, such as TF-IDF, should also be explored in the future to see how the results compare against each other.

Stakeholder Presentation:

To make our project efforts interpretable and valuable for the client, there are a number of steps the team has taken to make the system highly usable for all stakeholders. For direct querying, our plan is to implement Metabase. Metabase is an open-source business intelligence tool that can help stakeholders at all levels better understand the contents of our new TED Talk database. It leverages the simplicity of data querying for the analysts as they now can simply type their questions about the database and save them for later usage. For instance, in our case, if our client wishes to explore the relationship between the duration of video and its number of views, analysts can simply type a question such as: "How many videos have ratings higher than 4.0 with number of views less than 10,000?" Instead of the complicated SQL sentence, human-readable questions saved in Metabase will better help the analyst understand their database. Metabase will answer users' questions in the form of dashboards and intuitive charts/graphs so an analyst can gain better insights from it. The data analyst will also have access to the raw data with SQL inside python to manipulate the data.

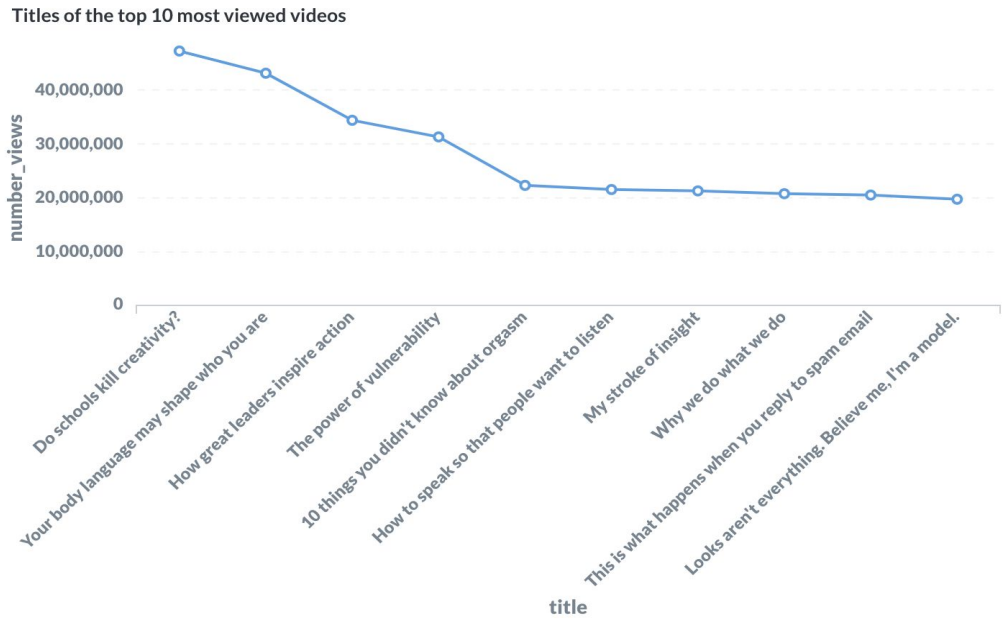
Metabase is suitable not only for data analysts but also for "C level" executives, who have different needs than analysts. To make company-wide decisions, the information conveyed by data needs to be comprehensive yet easily accessible. Metabase is capable of doing this job because it allows users to create different dashboards that show various aspects of the data. Each dashboard can contain different charts/graphs that can be easily understood with notes and comments. In our case, the executive board might be interested in how both the length of the title and the occupation of the speaker will affect the rating of the video. Different dashboards with an appropriate title can be created by the data analysts to report to the board. As a result, the report for the board can consist of many of these illustrations and are indeed comprehensive.

Planning for Redundancy and Performance:

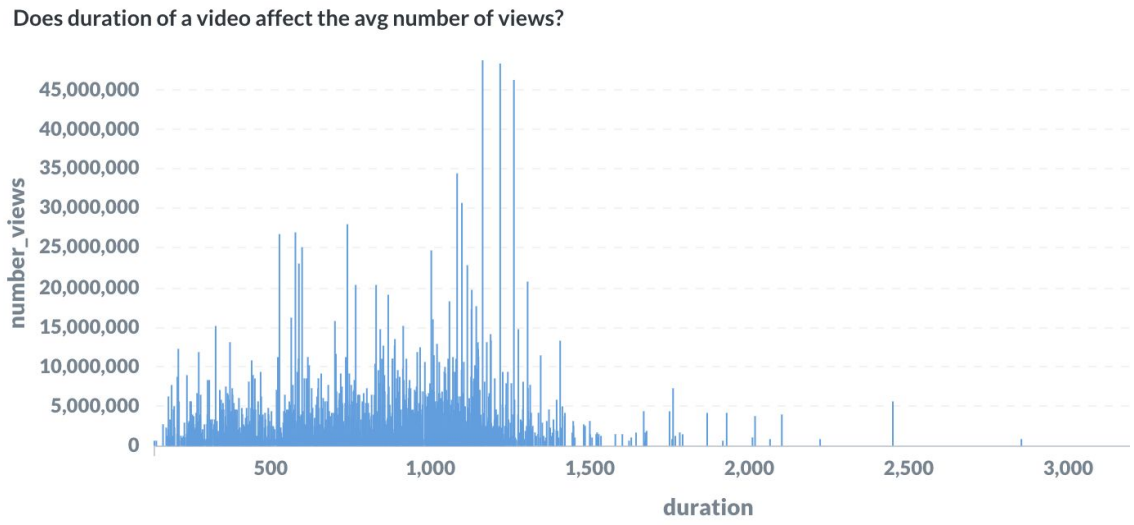
Data redundancy occurs when data appears multiple times in a database, which is a common issue in data storage and database systems. It may increase the size of the database unnecessarily, cause data inconsistency, and data corruption. To prevent data redundancy, we use data normalizations that includes creating tables and establishing relationships between those tables according to rules designed both to protect the data and to make the database more flexible. To optimize database performance, we cleaned data duplicates and removed unnecessary symbols, making data analysis more efficient and effective. In addition, we formed different table segmentations to provide insights on how our client should present the videos online.

BI Dashboards in Metabase:

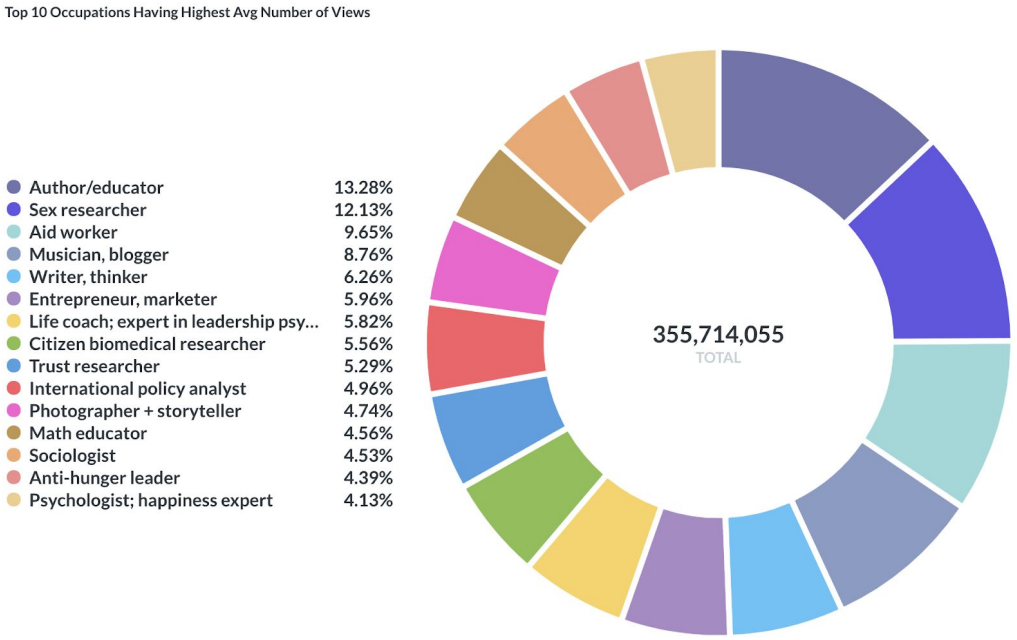
To better help C-level understand the factors influencing number of views, we have created a few visuals on Metabase. The first plot is presented below::



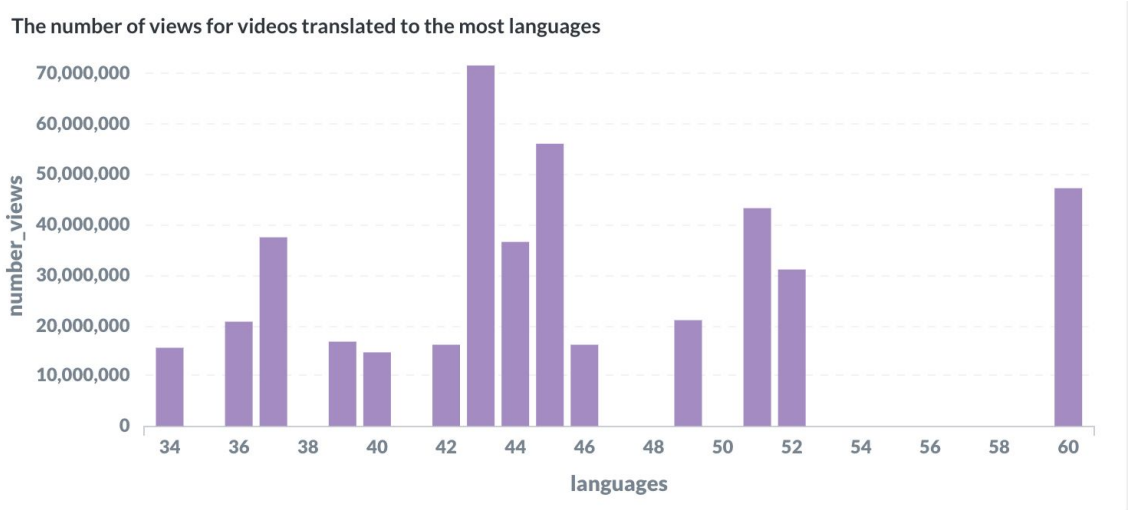
This graph explores the titles of the Top 10 most viewed videos. We found out that all these titles are less than 10 words and all of them are concise and eye-catching. As you can see in the graph, the title of the most viewed video is "Do schools kill creativity". These four words simply describe the content of the video in the most intriguing way. Thus, for a TED talk video to be successful, having an easy and interesting title is the premise.



Next, we wanted to know if the duration of a video affects the number of views. The x-axis of the histogram above reveals the durations of the videos in seconds, and the y-axis represents the number of views. The duration of the most viewed videos is between 1000-1500 seconds, and a few of the other most viewed videos have duration between 500-1000 seconds. Hence, a TED talk having duration between 500-1500 seconds is likely to become a popular one.

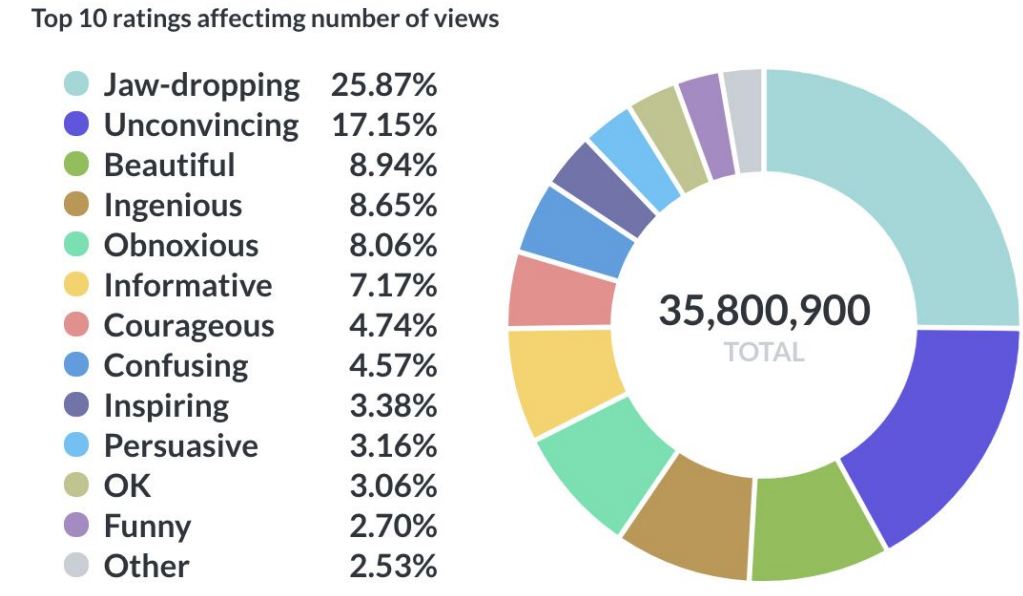


Then, we dug in what occupations of presenters tend to attract more viewers. Author, sex researcher, musician, entrepreneur and psychologist are among the top 10 occupations of speaker that produce the most popular TED Talk content. In the future, we can emphasize presenters' occupation in the introduction or title of the video to increase the audience's views.



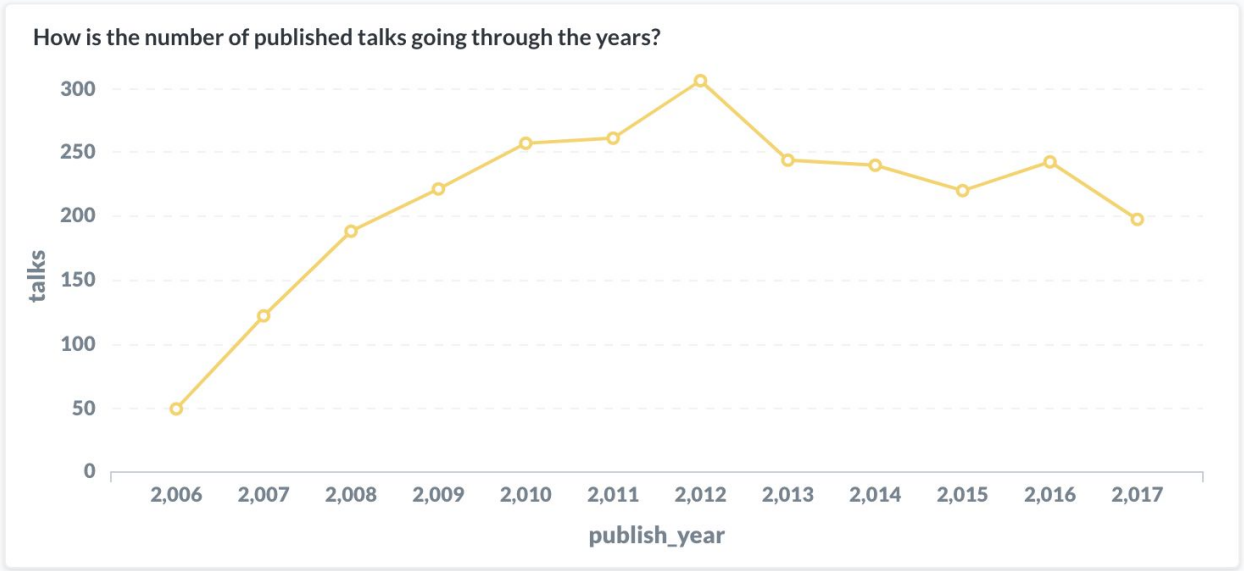
Our team was also curious whether the number of languages in which the talk was available could affect the views. The above graph shows that the number of languages available for the top 20 viewed videos. The range of number of languages is between 34 and 60. To create a hit TED talk, our team believes that it is necessary to translate to at least 42 languages which is the average number of languages translated for the top 20 viewed TED talk.

This graph tells us which event attract more viewers to watch our videos. TED EQChCh, Business Innovation Factory, TED New York, RSA Animate, Royal Institution and TED Maui are the most popular events. We should suggest these branches to host more talks in the future.

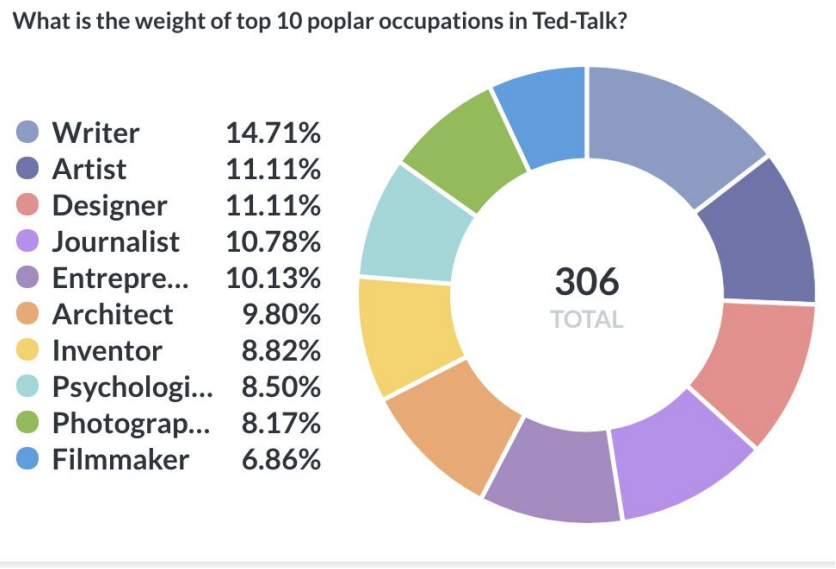


The above graph shows the Top 10 ratings contributing to the number of views. The rating of Jaw-dropping attracts the most percentage of views which is 25.87%. Surprisingly, the rating of unconvincing contributes to the second highest percentage of views which is 17.87% which informs us that videos with negative ratings could also be popular. Among the top 10 ratings, there are three negative ratings which share almost 30% of the top 10 popular TED talk views.

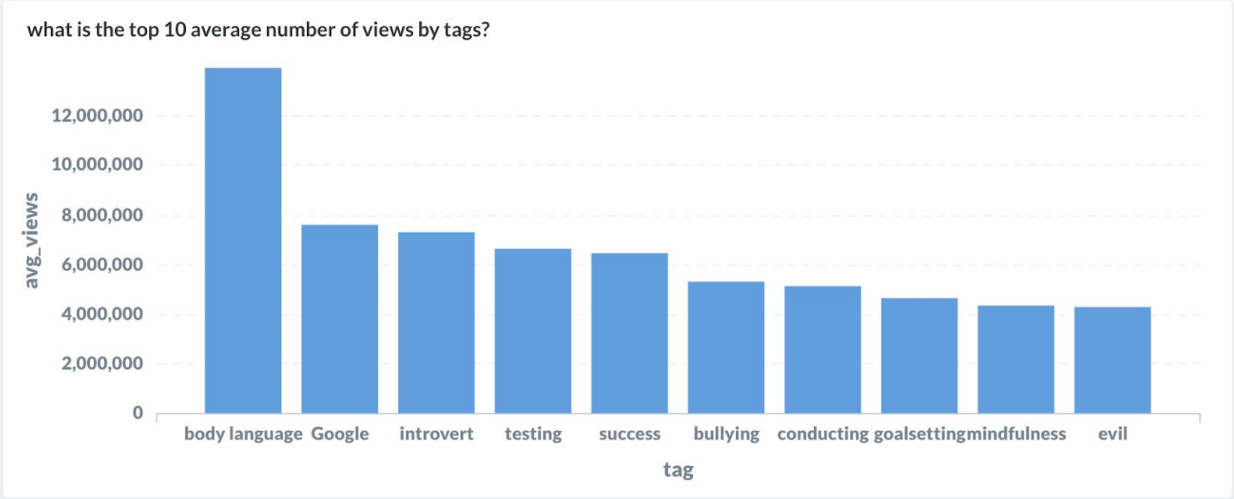
We also created a dashboard for the analysts to have a better understanding of what kind of videos are viewed by audiences.



The chart above shows the trend of the number of talks over the year. We can see that in 2012, there are around 300 talks published, which is the highest number between the year of 2006 and 2017. We can conclude that the number was increasing starting from 2006, and after 2012, the number has a descending trend. Analysts can have some further exploration of the data after 2012 to find out the reason why the number of talks was decreased.



The pie chart above shows the weight of the top 10 occupations of the speakers who have been invited to the Ted-Talk. With the chart, we can easily identify that writers, artists, and designers are the most popular groups to be invited to become Ted speakers.



The bar chart above shows the average top 10 number of views by the tags, which shows the popularity of the theme among the Ted-Talk history. This graph is consistent with the one built in Python using the same information. According to the chart, the videos tagged with body language are most viewed by audiences, followed by content about Google and introverts.

The following are the links to the dashboards:
C-levels:
http://f19server.apan5310.com:3202/dashboard/5

Analysts:
http://f19server.apan5310.com:3202/dashboard/2

Conclusion:

The steps that our team has detailed in this report show the effectiveness of the approaches taken and the dedication we have towards our client's success. Using a PostgreSQL database and a Python ETL script, our team was able to develop a 3NF database out of two messy CSV files from Kaggle. From here, we performed a wide range of analytical procedures on the data to understand which parts of the data may impact view counts. We even took our analysis a step further by employing advanced analytics techniques like count vectorization to prepare the data for predictive modeling. These advancements directly relate to the goals of the client and would set our team up for success based on the current project scope.

As an additional step towards user-friendliness, our team also took the time to connect this data to Metabase and create dashboards for key stakeholders. For executives, dashboards with visuals about key metrics allow them to focus on the information that matters most for the decisions they will make on a regular or semi-regular basis. For analysts, the easy querying capabilities allow for easy access to any TED Talk data that might be required for an analysis. This setup will allow the client to be self-sufficient even after our engagement has ended and their internal clients are the only ones still interacting with the system.

Through our new database, it is possible to examine and investigate a multitude of variables to gain an understanding of the factors that relate to video view counts. With this information at hand, we should be able to complete the objective set out by our clients. We can deem that the project has been a success thus far, but next steps must also be taken to ensure that all aspects of the project are properly addressed. These may include predictive modeling on existing features, additional feature engineering and data manipulation, or some other step that would allow us to better comprehend the drivers of online learning video views.

Appendix: ER Diagram