Contents

1 Executive Summary

In this notebook, I am going to connect my python jupyte notebook to the BigQuery API.
The database that I select is the "Stack Overflow" which consists of technical questions and answers of newbies or experts in a variety of programming language or technologies. Using SQL queries I want to discover this database and identify some interesting patterns in the data.

2 Introduction

In the "Stack Overflow" database, we have two seperate tables which keep the track of submitted questions and answers.
Each submitted question has a "tag" attached to it which shows the relevancy of the question with a certain technology or topic.
In this notebook, I aim to look at the number of questions which are relevant to two programming languages, "Java" and "Python".
I also would like to observe the trend of changes over years.

3 Importing the Libraries

In [2]:
```
#importing the OS library↔
```

In [1]:
```
# Importing the libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
%matplotlib inline
warnings.filterwarnings('ignore')
```

4 Connection with the BigQuery API

In [3]:
```
# Importing BigQuery API
import bq_helper
from bq_helper import BigQueryHelper
from google.cloud import bigquery
# Create a "Client" object
client = bigquery.Client()
# calling the "stackoverflow" databse from the bigquery public project
db = client.dataset("stackoverflow", project="bigquery-public-data")
# extracting the dataset into python
dataset = client.get_dataset(db)
# this will show if I got all the tables imported
tables_list = list(client.list_tables(dataset))
```

5 Database Tables

Let us have a look at the number of tables in this database

In [7]:
```
len(tables_list)
```

Out[7]: 16

This database consists of 16 tables

In [9]:
```
# Let us look at the name of the tables
stackOverflow = bq_helper.BigQueryHelper(active_project="bigquery-public-data",
                                         dataset_name="stackoverflow")
bqa = BigQueryHelper("bigquery-public-data", "stackoverflow")
bqa.list_tables()
```

Out[9]:
```
['badges',
 'comments',
 'post_history',
 'post_links',
 'posts_answers',
 'posts_moderator_nomination',
 'posts_orphaned_tag_wiki',
 'posts_privilege_wiki',
 'posts_questions',
 'posts_tag_wiki',
 'posts_tag_wiki_excerpt',
 'posts_wiki_placeholder',
 'stackoverflow_posts',
 'tags',
 'users',
 'votes']
```

Contents

In [10]:
```
#Let us have a look at a sample table which is called "posts_answers"
# First we construct a reference to this table
table1_ref = dataset.table("posts_answers")

# I send the request to fetch the table
table1_ref = client.get_table(table1_ref)

# Preview the table as a dataframe
client.list_rows(table1_ref, max_results=2).to_dataframe()
```

Out[10]:

	id	title	body	accepted_answer_id	answer_count	comment_count	community_owned_date	creation_date	favo
0	63172172	None	<p>Up to now 7 answers. But I am really surpri...	None	None	0	NaT	2020-07-30 11:26:18.453000+00:00	
1	63172205	None	<p>I know this is an old one, but here's a bit...	None	None	0	NaT	2020-07-30 11:28:08.737000+00:00	

In this table, owner_user_id corresponds to the Id of the person who answered the question which its Id has been set as parent_id. The original questions are located at the "posts_questions" table.So let us have a look at this table.

In [11]:
```
#Let us have a look at the "posts_questions" Table
# First we construct a reference to this table
table2_ref = dataset.table("posts_questions")

# I send the request to fetch the table
table2_ref = client.get_table(table2_ref)

# Preview the table as a dataframe
client.list_rows(table2_ref, max_results=2).to_dataframe()
```

Out[11]:

	id	title	body	accepted_answer_id	answer_count	comment_count	community_owned_date	creation_date
0	9028004	(android)How can do the package name don't sho...	<p>How can do the package name don't shown on ...	NaN	3	5	NaT	2012-01-27 01:44:30.933000+00:00
1	9043121	What type of activity animation view is this?	<p>In the picture below, what type of view is ...	9043369.0	4	0	NaT	2012-01-28 05:15:51.087000+00:00

There is a "tag" column in this table which shows the relevancy of the question with certain technologies. For example the first question is re to the Android packages. We can also see the view count for each question.

We need to make several queries to fetch the questions which are relevant to the java or python categories.

In [20]:
```
# Composing our 1st Query (Python related questions)
query1 = """
SELECT
  EXTRACT(YEAR FROM creation_date) AS Year,
  count (ques.view_count) AS total_view
FROM
  `bigquery-public-data.stackoverflow.posts_questions` AS ques
WHERE
  tags LIKE '%python%'
  GROUP BY
  Year
  order by Year
  """
```

In [21]:
```
# submitting the 1st query
safe_config = bigquery.QueryJobConfig(maximum_bytes_billed = 10**10)
query_job = client.query(query1, job_config=safe_config)
df = query_job.to_dataframe()
df
```

Out[21]:

	Year	total_view
0	2008	2105
1	2009	13212
2	2010	27559
3	2011	43065
4	2012	66671
5	2013	102053
6	2014	123878
7	2015	147526
8	2016	173607
9	2017	213610
10	2018	232584
11	2019	273458
12	2020	266561

In [13]:
```
# Fetching the python related tags
query1b = """
SELECT
  tags
FROM
  `bigquery-public-data.stackoverflow.posts_questions` AS ques
WHERE
  tags LIKE '%python%'
  """
```

In [16]:
```
# let us have a look at the "tags" column values for Python
safe_config = bigquery.QueryJobConfig(maximum_bytes_billed = 10**10)
query_job = client.query(query1b, job_config=safe_config)
dfb = query_job.to_dataframe()
dfb
```

Out[16]:

	tags
0	python|xml
1	python|winapi|active-directory-group
2	python|if-statement|python-3.x|while-loop
3	python|python-2.6
4	python|linux|xmpp|openfire
...	...
1685884	python|pandas|dataframe
1685885	python|pandas|dataframe
1685886	python|pandas|dataframe
1685887	python|pandas|dataframe
1685888	python|pandas|dataframe

1685889 rows × 1 columns

In [18]:
```
# Composing our 2nd Query (Java related questions)
query2 = """
SELECT
  EXTRACT(YEAR FROM creation_date) AS Year,
  count (ques.view_count) AS total_view,
FROM
  `bigquery-public-data.stackoverflow.posts_questions` AS ques
WHERE
  tags LIKE '%java%'
  GROUP BY
  Year
  order by Year
  """
```

In [19]:
```
# submitting the 2nd query
safe_config = bigquery.QueryJobConfig(maximum_bytes_billed = 10**10)
query_job = client.query(query2, job_config=safe_config)
df2 = query_job.to_dataframe()
df2
```

Out[19]:

	Year	total_view
0	2008	7093
1	2009	43205
2	2010	98824
3	2011	191563
4	2012	284974
5	2013	390903
6	2014	457521
7	2015	476151
8	2016	466207
9	2017	430341
10	2018	360481
11	2019	336790
12	2020	288856

In [22]:
```
# Fetching the java related tags
query2b = """
SELECT
  tags
FROM
  `bigquery-public-data.stackoverflow.posts_questions` AS ques
WHERE
  tags LIKE '%java%'
  """
```

In [23]:
```
# Let us have a look at the "tags" column values for Java
safe_config = bigquery.QueryJobConfig(maximum_bytes_billed = 10**10)
query_job = client.query(query2b, job_config=safe_config)
dfb2 = query_job.to_dataframe()
dfb2
```

Out[23]:

	tags
0	java|hibernate|spring|transactions
1	java|text-mining|svd|lsa|jama
2	javascript|opengl-es|matrix|webgl|perspective
3	java|google-app-engine|jdo
4	javascript|object|coding-style
...	...
3832904	javascript|jquery|html|css
3832905	javascript|jquery|html|css
3832906	javascript|jquery|html|css
3832907	javascript|jquery|html|css
3832908	javascript|jquery|html|css

3832909 rows × 1 columns

In [24]:
```
# For python the total count is
np.sum(df['total_view'])
```

Out[24]: 1685889

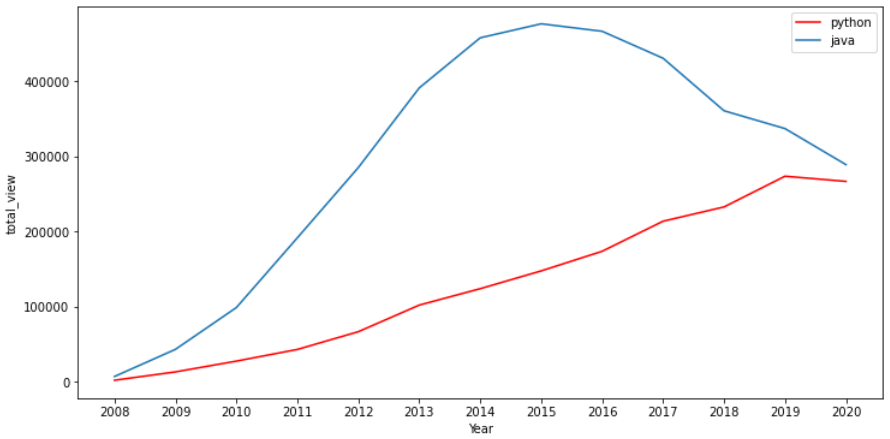
In [25]:
```
#for Java the total count is
np.sum(df2['total_view'])
```

Out[25]: 3832909

It is obvious that the number of java related questions are way higher than python ones but let us look at the trend by year

In [38]:
```
f=pd.Series(df.Year)
ax = df[['total_view']].plot(color="red", figsize=(12,6))
df2[['total_view']].plot(ax=ax)
ax.set_xlabel("Year", fontsize=10)
ax.set_ylabel("total_view", fontsize=10)
ax.legend(["python","java"])
ax.set_xticks(range(len(df2["Year"])))
ax.set_xticklabels(d)
```

Out[38]:
```
[Text(0, 0, '2008'),
 Text(0, 0, '2009'),
 Text(0, 0, '2010'),
 Text(0, 0, '2011'),
 Text(0, 0, '2012'),
 Text(0, 0, '2013'),
 Text(0, 0, '2014'),
 Text(0, 0, '2015'),
 Text(0, 0, '2016'),
 Text(0, 0, '2017'),
 Text(0, 0, '2018'),
 Text(0, 0, '2019'),
 Text(0, 0, '2020')]
```



The trend is quite surprising. The trends show an increasing number of posted question which are python related. However the trend shows a turning point for Java related questions. More specifically, after the year 2015 the number java related posts follows a decreasing pattern. The cause is unclear and should be further investigated, however, from my point of know this can be an indicatio that the interest has been increased in considering Python as a programmiung language by developers. Knowing the history of java, there is no doubt in its capabilities but honestly it has a syntaxthat sometimes drives the programmers especially newbies crazy qucikly :-).

Ok now it is time to save the results of our quesries as a labled csv file for further use.

6 Saving the Result as a CSV File

In [40]:
```
#Let us append the dataframes and make a labeled dataset out of that
dfb["label"]= 1 # python tags
dfb2["label"]= 0 # java tags
dftotal= dfb.append(dfb2, ignore_index = True)
dftotal
```

Out[40]:

	tags	label
0	python|xml	1
1	python|winapi|active-directory-group	1
2	python|if-statement|python-3.x|while-loop	1
3	python|python-2.6	1
4	python|linux|xmpp|openfire	1
...	...	...
5518793	javascript|jquery|html|css	0
5518794	javascript|jquery|html|css	0
5518795	javascript|jquery|html|css	0
5518796	javascript|jquery|html|css	0
5518797	javascript|jquery|html|css	0

5518798 rows × 2 columns

In [41]:
```
# removing the '|' among the keywords
dftotal['tags'] = dftotal['tags'].str.replace('|',' ')
```

In [ ]:
```
dftotal.to_csv("TotalDataset.csv")
```