Snyk home assignment

- Author: Dmitry Anoshin
- Postition: Data Engineer
- Date: 5 Oct 2021
- Time Start: 1pm
- Time End: 8pm

Part 1 - GitHub sample data set

I signed up to google cloud and got a BigQuery cluster. Added GitHub public data set and explored the data.

Usually, I am building things iteratively. For example, I make the 1st iteration and review it with stakeholders. It is a common problem that is requirements could be treated differently. I might misunderstand requirements and might produce the wrong output. But you can still see how I wrote SQL. If requires, I could rewrite.

Task 1. How many repos from the sample_repos table have any programming languages data on them?

```
select
count(distinct repo_name)
from  (
    select repo_name,
    count(distinct l.name)  as number_languages
    from bigquery-public-data.github_repos.languages, UNNEST(language)  as
l
    group  by repo_name
    having number_languages >  0
    order  by number_languages desc)
```

Answer: `2962596` (`89%`) repos among `3332011` repos has at least 1 language. 3332011 is number of unique repos in `sample_repos`.

Task 2.1 For the repos we found on section 1: What are the top 10 languages most commonly used?

```
select
rnk,
language_name
from  (with languages as
        (select
        l.name as language_name,
        sum(l.bytes)  as bytes
        from bigquery-public-data.github_repos.languages, UNNEST(language)
```

```
as l
        where repo_name in
            (select
            distinct repo_name
            from  (select repo_name,
                    count(distinct l.name)  as number_languages
                    from bigquery-public-data.github_repos.languages,
UNNEST(language)  as l
                    group  by repo_name
                    having number_languages >  0
                    order  by number_languages desc
                    )
            )
            group  by language_name)
            select
            language_name,
            rank()  over  (ORDER  BY bytes desc)  AS rnk
            from languages)
where rnk <=  10
order  by rnk;
```

The query will return TOP 10 languages among repo based on sum of bytes. In my example, they are: `'C','JavaScript','C++','PHP','HTML','Java','Python','CSS', 'Assembly','C#'` This is questioning the logic because I had an idea to find the top 10 languages based on a number of commits. In the 3rd exercise, it was exactly this task. But I couldn't join `sample_commits` with `languages`. They have in common only `repo_name` and one repo can have multiple languages. For the 3rd exercise, I used the `file extension` for the identification of language.

Task 2.2 How many repos use each language and what percent of the total repos does that represent?

First, I've created a view using the similar logic above using BigQuery UI. This query will give me TOP 10 languages at any point in time.

```
select  *  from dmitry.v_top_10_languages
```

The final query is:

```
with selected_repos as  (
    select
    distinct repo_name
    from bigquery-public-data.github_repos.languages, UNNEST(language)  as
l
    where l.name in  (select  distinct language_name from
dmitry.v_top_10_languages)
    group  by repo_name),
all_repos as  (
    select
```

```
    distinct repo_name
    from bigquery-public-data.github_repos.languages, UNNEST(language)  as
l
    group  by repo_name)
select
count(distinct s.repo_name)  as selected_repos,
count(distinct a.repo_name)  as all_repos,
count(distinct s.repo_name)/count(distinct a.repo_name)
from all_repos a left  join selected_repos s on a.repo_name = s.repo_name;
```

The result is `2391498` or (`81`%) repos have one or more languages from TOP 10 among the `2962596` repos in `language` table.

Task 3.1 Let's explore the sample_commits section. Find out the number of commits trend (YoY) for our top 10 most common languages:

In this task I just looked at YoY number of commits:

```
with yearly_commits as (
    select
    EXTRACT(YEAR FROM committer.date) as commit_year,
    #count(distinct commit) as number_commits,
    sum(1) as number_commits,
    from bigquery-public-data.github_repos.sample_commits
    group by commit_year
    order by commit_year asc)
select
commit_year,
number_commits as commits_current_year,
LAG(number_commits) OVER ( ORDER BY commit_year ) AS
commits_previous_year,
round(number_commits/LAG(number_commits) OVER ( ORDER BY commit_year ),2)
as yoy_difference
from yearly_commits
order by commit_year desc
```

Task 3.2 Let's explore the sample_commits section. Find out the number of commits trend (YoY) for our top 10 most common languages.

As I mentioned in task 2, I faced an issue with a lack of joins between `language` and `commit`. My close guess was to use file extension. I have several ideas. One of them is to use `CASE` and group file extensions:

```
     when language_extension in (".cc",".cpp","hxx") then "C++"
     when language_extension in (".py") then "Python"
     when language_extension in (".S",".lds.S") then "Assembly"
     when language_extension in (".swift") then "Swift"
     when language_extension in (".xml") then "XML"
     when language_extension in (".pl") then "Perl"
     when language_extension in (".sh") then "Shell"
```

```
     when language_extension in (".js") then "JavaScript" else
language_extension end as language
```

but we have too many of them and it is inefficient to do it manually. And I decided to use shortcat. I filtered out the popular values and keeped only values that started from `.`:

```
where SUBSTR(dif.new_path,STRPOS(dif.new_path,"."),length(dif.new_path))
not in
(".HEX",".ts",".txt",".dtsi",".dts",".debug","MAINTAINERS",".tmpl",".gitig
nore",".out.h")
and SUBSTR(dif.new_path,STRPOS(dif.new_path,"."),length(dif.new_path))
like '.%'
```

My final code is:

```
with commits as (
    select
    commit_year,
    case when language_extension in (".c",".h") then "C"
        when language_extension in (".cc",".cpp","hxx") then "C++"
        when language_extension in (".py") then "Python"
        when language_extension in (".S",".lds.S") then "Assembly"
        when language_extension in (".swift") then "Swift"
        when language_extension in (".xml") then "XML"
        when language_extension in (".pl") then "Perl"
        when language_extension in (".sh") then "Shell"
        when language_extension in (".js") then "JavaScript" else
language_extension end as language,
    sum(number_commits) as number_commits
    from (select
        extract(YEAR from committer.date) as commit_year,
        SUBSTR(dif.new_path,STRPOS(dif.new_path,"."),length(dif.new_path))
as language_extension,
        sum(1) as number_commits
        from bigquery-public-data.github_repos.sample_commits,
UNNEST(difference) dif
        where
SUBSTR(dif.new_path,STRPOS(dif.new_path,"."),length(dif.new_path)) not in
(".HEX",".ts",".txt",".dtsi",".dts",".debug","MAINTAINERS",".tmpl",".gitig
nore",".out.h")
        and
SUBSTR(dif.new_path,STRPOS(dif.new_path,"."),length(dif.new_path))   like
'.%'
        group by commit_year, language_extension)
    group by commit_year, language
)
select
commit_year,
language,
```

```
number_commits as current_number_commits,
LAG(number_commits) OVER (partition by language ORDER BY commit_year ) AS
commits_previous_year,
round(number_commits/LAG(number_commits) OVER (partition by language
ORDER BY commit_year ),2) - 1 as yoy_difference
from (
    select
    commit_year,
    language,
    number_commits,
    rank() over (partition by commit_year order by number_commits desc) AS
rnk
    from commits )
where rnk <= 3
order by commit_year, number_commits desc
```

This query will return the number of commits for the current and previous years. The only drawback is that I don't identify the language.

But the "a picture is worth a thousand words", so I visuazed the data using `Tableau`. I could connect BigQuery to Tabelau using native connector but I downloaded the `csv` file.

Commit Year	Language	Commits	PY Commits	YoY
2005	Assembly	9,809		
	C	295,184		
	Shell	157		
2006	Assembly	11,711	9,809	19%
	C	365,626	295,184	24%
	Shell	135	157	-14%
2007	Assembly	4,562	11,711	-61%
	C	188,486	365,626	-48%
	Shell	93	135	-31%
2008	Assembly	27,157	4,562	495%
	C	1,048,717	188,486	456%
	Perl	608		
2009	Assembly	22,608	27,157	-17%
	C	1,033,501	1,048,717	-1%
	XML	1,930		
2010	Assembly	18,103	22,608	-20%
	C	1,069,810	1,033,501	4%
	XML	986	1,930	-49%
2011	Assembly	22,500	18,103	24%
	C	1,048,095	1,069,810	-2%
	XML	6,876	986	597%
2012	Assembly	24,499	22,500	9%
	C	1,373,540	1,048,095	31%
	C++	4,768		
2013	Assembly	18,753	24,499	-23%
	C	985,256	1,373,540	-28%
	C++	12,229	4,768	156%
2014	C	909,865	985,256	-8%
	C++	14,745	12,229	21%
	Swift	18,397		
2015	C	933,778	909,865	3%
	C++	18,803	14,745	28%
	Swift	69,679	18,397	279%
2016	C	487,535	933,778	-48%
	C++	26,946	18,803	43%
	Swift	55,576	69,679	-20%

Task 4 Find one interesting insight from the 'stackoverflow' public dataset you think is worth mentioning.

I connected stackoverflow data set and generated the following hypothesis and query ideas. Sample query:

```
select *
from bigquery-public-data.stackoverflow.stackoverflow_posts
limit 10
```

My ideas:

- Number of posts by year. I can see how it grows. Ideally, I would like to see the trending tags.
- Avg time between `creation_date` and `last_activity_date`
- Percentage of answered questions vs not answered
- Top authors based on `view_count`
- Check the popularity of technologies based on `tags`. For example:

```
select sum(1) from bigquery-public-data.stackoverflow.stackoverflow_posts
where tags like '%snowflake%'
```

```
select sum(1) from bigquery-public-data.stackoverflow.stackoverflow_posts
where tags like '%spark%'
```

I clearly see, that Snowflake has `45` posts and spark has `16840`. It means that Snowflake developers doens't like stackoverlow. Maybe because it is low code application.