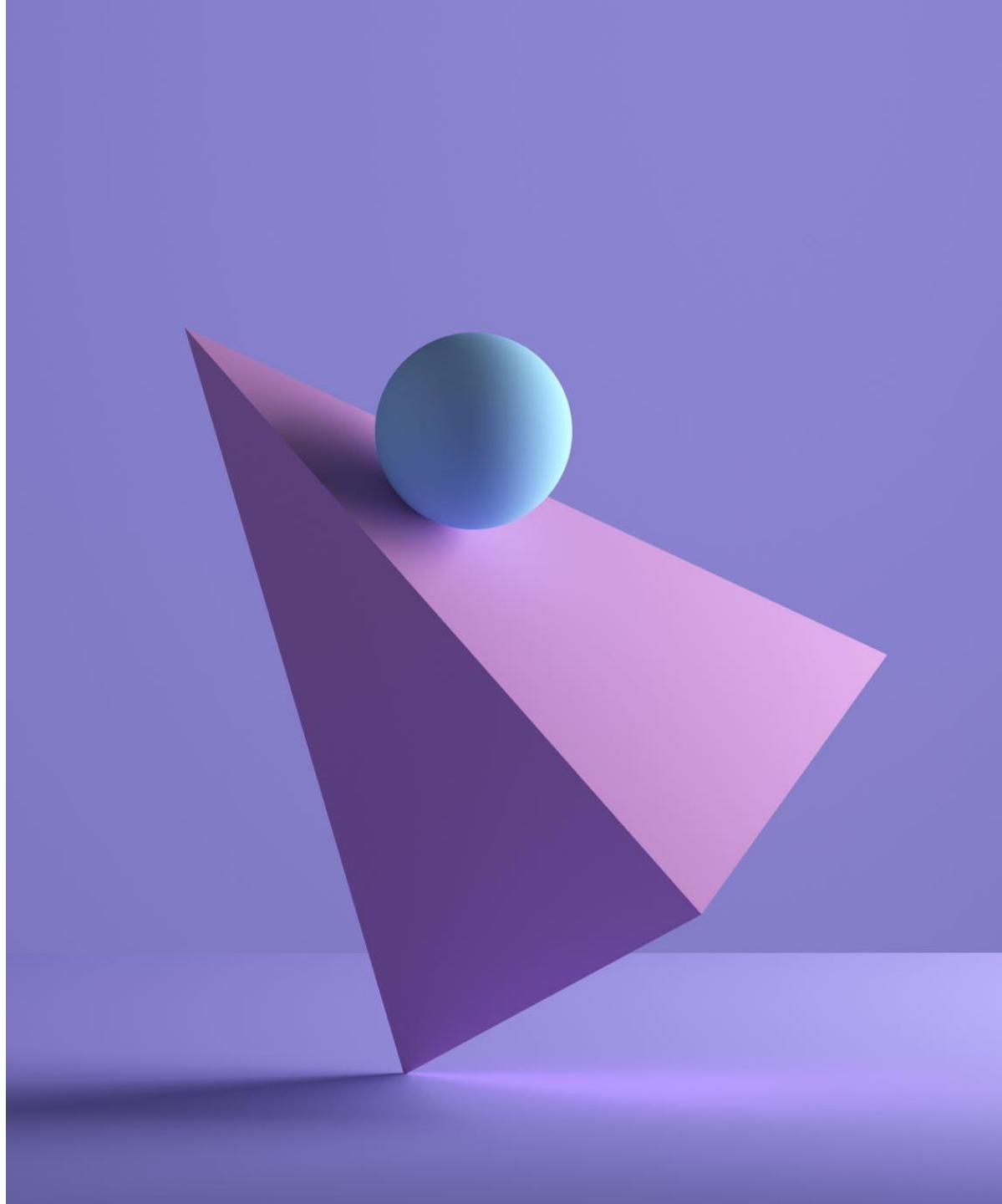
REDASH TO GOOGLE SHEETS CONNECTION

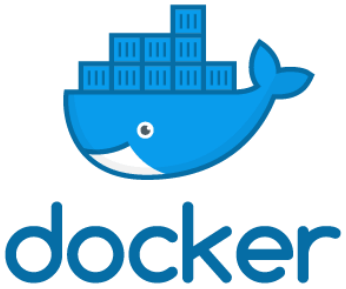
HOW-TO (WITH PYTHON)

Collin – Data Analyst Happy5

Est. Aug 2021



SOFTWARE REQUIREMENTS



Anaconda Navigator

(You can also use google colab)

Docker and WSL

(Kinda need to ask Backend Engineers for this, but it would be great to explore more about it!)

Any kind of Text Editor

(Visual Studio Code, Pycharm, Notepad++, Sublime text)

OVERVIEW

Google Sheets

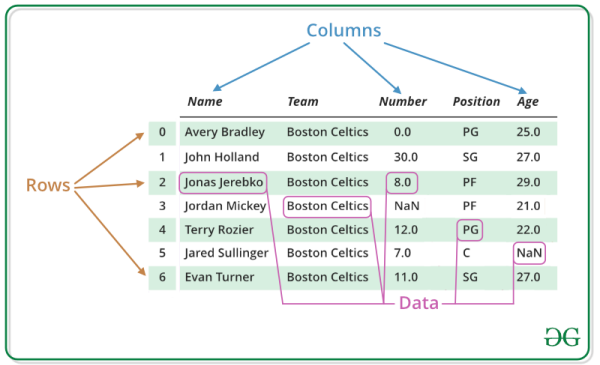
Write onto Google Sheet / Excel

redash

SQL Query

{JSON}

Retrieve with Python Scripts



Metabase

JSON File

Rearranged into Pandas DataFrame (akin to table)

REDASH

POSTGRESQL BASICS

We only need simple queries to retrieve data

Further advanced queries could be used for summaries, or complex data retrieval

I recently found a good website to practice basic to advanced SQL knowledge:

https://platform.stratascratch.com/coding

```
SELECT column1, column2, ...
FROM table
LEFT / INNER JOIN table2 on ...
WHERE conditions
```

REDASH TO GOOGLE SHEETS CONNECTION

In order to do this ETL (Extract Transform Load). We are going to use Python scripts.

There is a function created by Redash Owner, that allow us to Extract Queries from our redash in the form of JSON file.

If you want to explore more, his Github provides many scripts on API usage using Python, JS and many more

[Redash Refresh API usage example with parameters Raw · GitHub]

Basically the flow is:

1. Create or use an existing query.
2. Extract the Data using the function given
3. Transform the Data using Pandas / just give it raw.
4. Load the data to Google Sheet / Excel

RETRIEVE DATA FROM REDASH

```python
def poll_job(s, redash_url, job):
    # TODO: add timeout
    while job['status'] not in (3, 4):
        response = s.get('{}/api/jobs/{}'.format(redash_url, job['id']))
        job = response.json()['job']
        time.sleep(1)

    if job['status'] == 3:
        return job['query_result_id']

    return None


def get_fresh_query_result(redash_url, query_id, api_key, params):
    s = requests.Session()
    s.headers.update({'Authorization': 'Key {}'.format(api_key)})

    payload = dict(max_age=0, parameters=params)

    response = s.post('{}/api/queries/{}/results'.format(redash_url, query_id), data=json.dumps(payload))

    if response.status_code != 200:
        raise Exception('Refresh failed.')

    result_id = poll_job(s, redash_url, response.json()['job'])

    if result_id:
        response = s.get('{}/api/queries/{}/results/{}.json'.format(redash_url, query_id, result_id))
        if response.status_code != 200:
            raise Exception('Failed getting results.')
    else:
        raise Exception('Query execution failed.')

    return response.json()['query_result']['data']['rows']
```

The function that we will use is simply get_fresh_query_result.

In order to use it, we would need:

1. redash_url
2. query_id
3. api_key (you can retrieve this from your redash profile, just paste it here)
4. params

And call the function.

SORT AND ORDER THE DATA!

We can just convert the JSON data into pandas DataFrame, but the data is not sorted, therefore we need to sort the data manually (You can also choose the columns you wanna show and ignore the others).

NEXT STEP..

- You may choose to transform the data or simply put it raw onto google sheets. This tutorial won't give you the way to transform the data, but simply creating ETL from Redash to Google Sheets.
- As for the next step we will be starting with Loading our data to Google Sheets.

AND ITS DONE! HOORAY!

You have finally created an ETL from Redash to Gsheets.

We still have another step, which is automation but this is optional.

The automation are done using Docker, and a scheduler. But this was done by Backend Engineer team before.

THANK YOU

Pardon me if the tutorial is lacking. Good Luck!