Priority Task Script – Tableau Server

Documentation

# Scope

Change the priority tasks (extract refreshes).

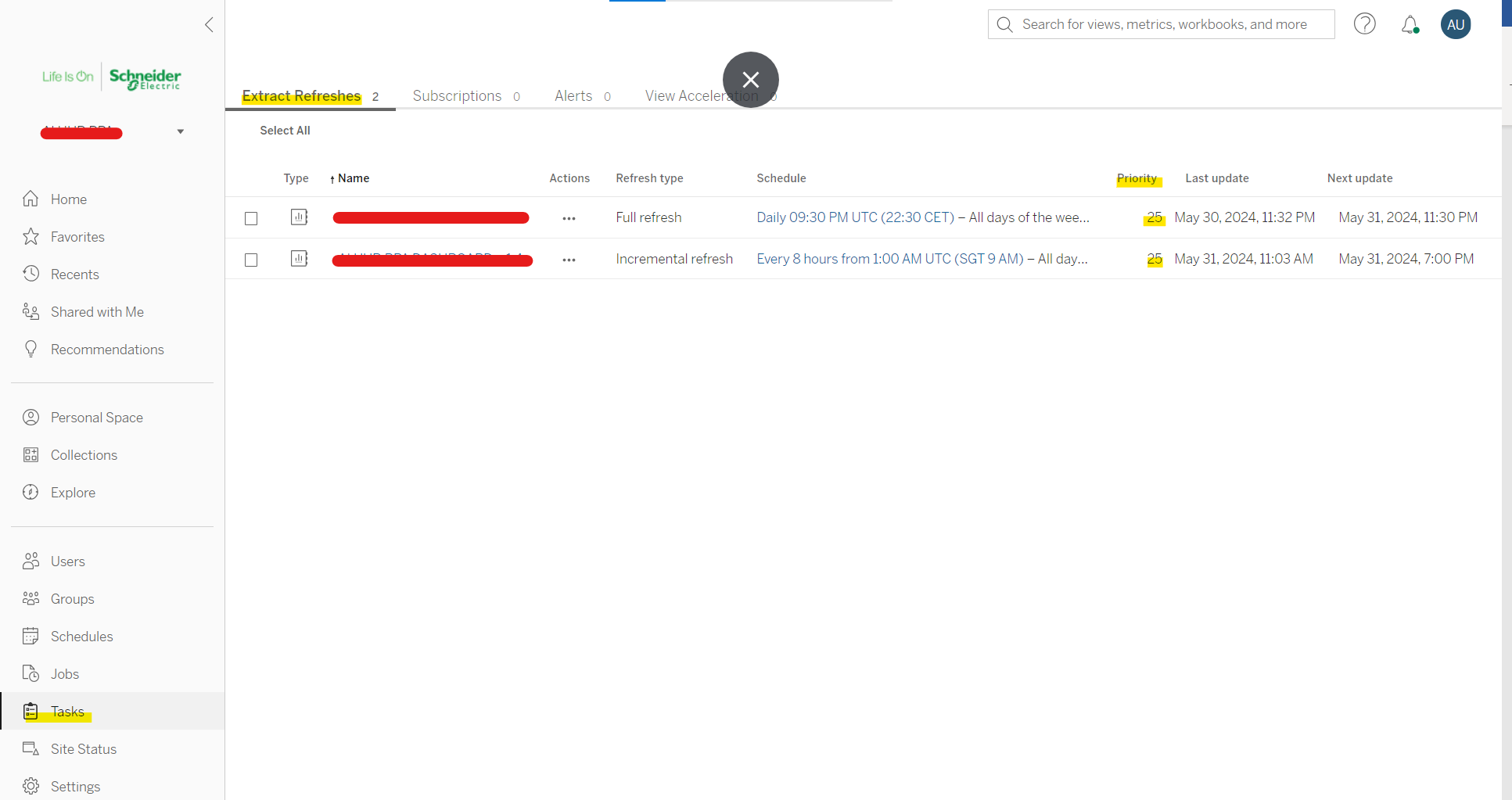


Figure – Landing Page

Based on the following logic

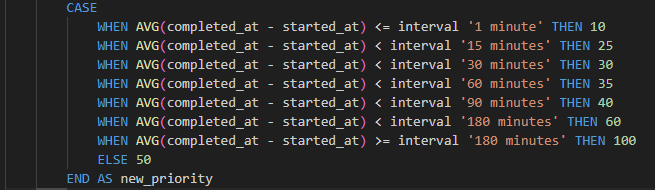


Figure - Logic

i.e., we are looking at the average duration of each task, to determine the new appropriate priority.

# How it works

We exploit the Python library Selenium to map a few clicks in each extract refresh content page.

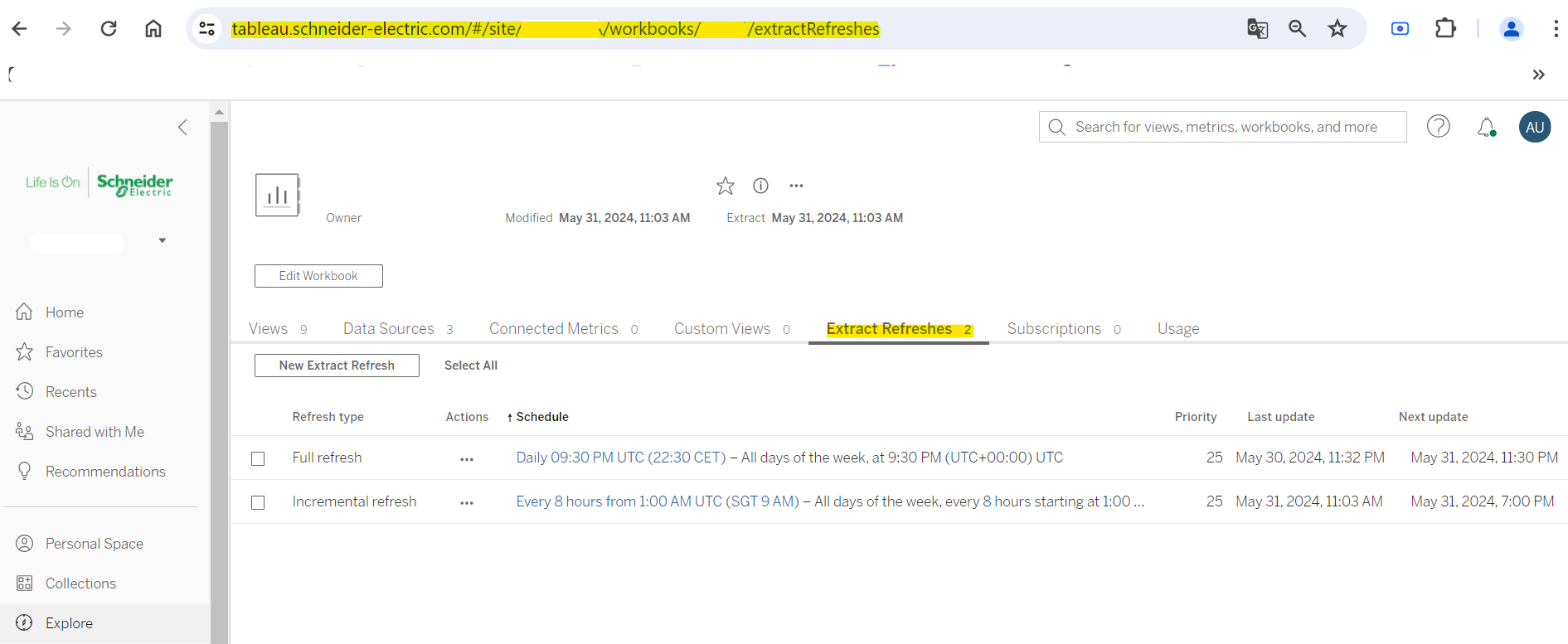


Figure - Landing Page

## Clicks

Once we land on the page of the extract refresh, we look for the corresponding Schedule to change (Daily 09:30 PM UTC (22:30 CET) in our case) and we perform the following clicks:

1. Option menu
2. Change Priority
3. Click in the box.
   * Delete the current (old) priority.
   * Insert the new priority.
4. Change priority (save)

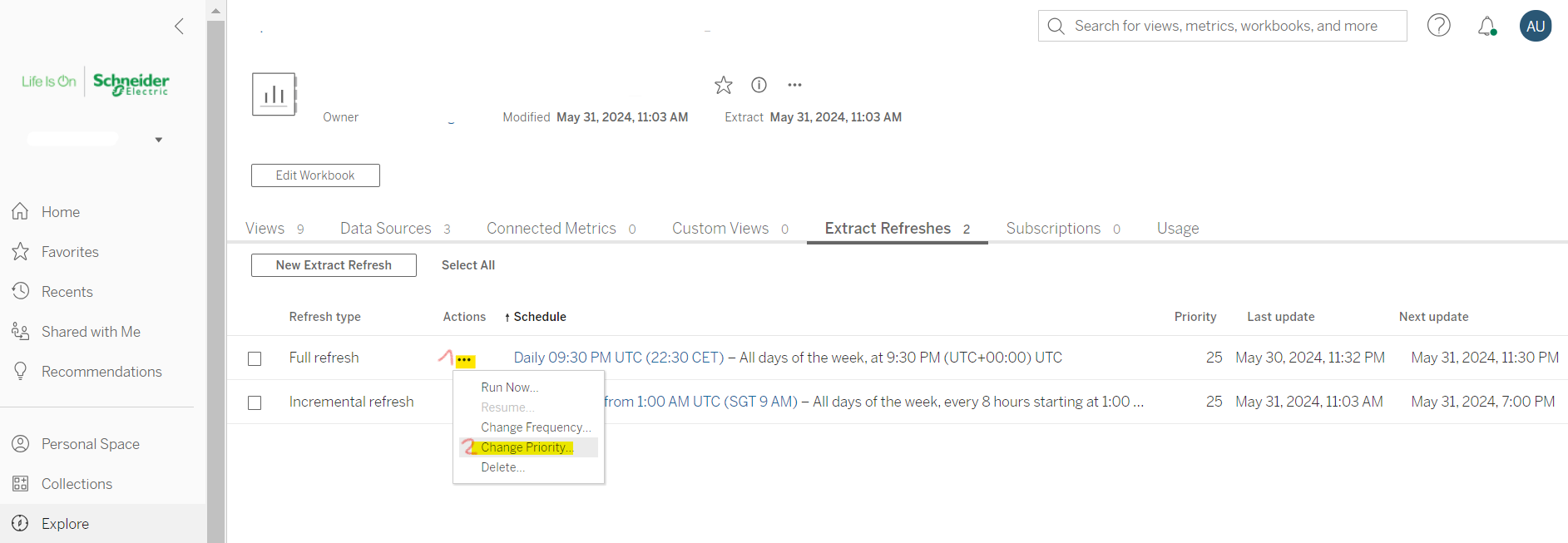


Figure - Click1

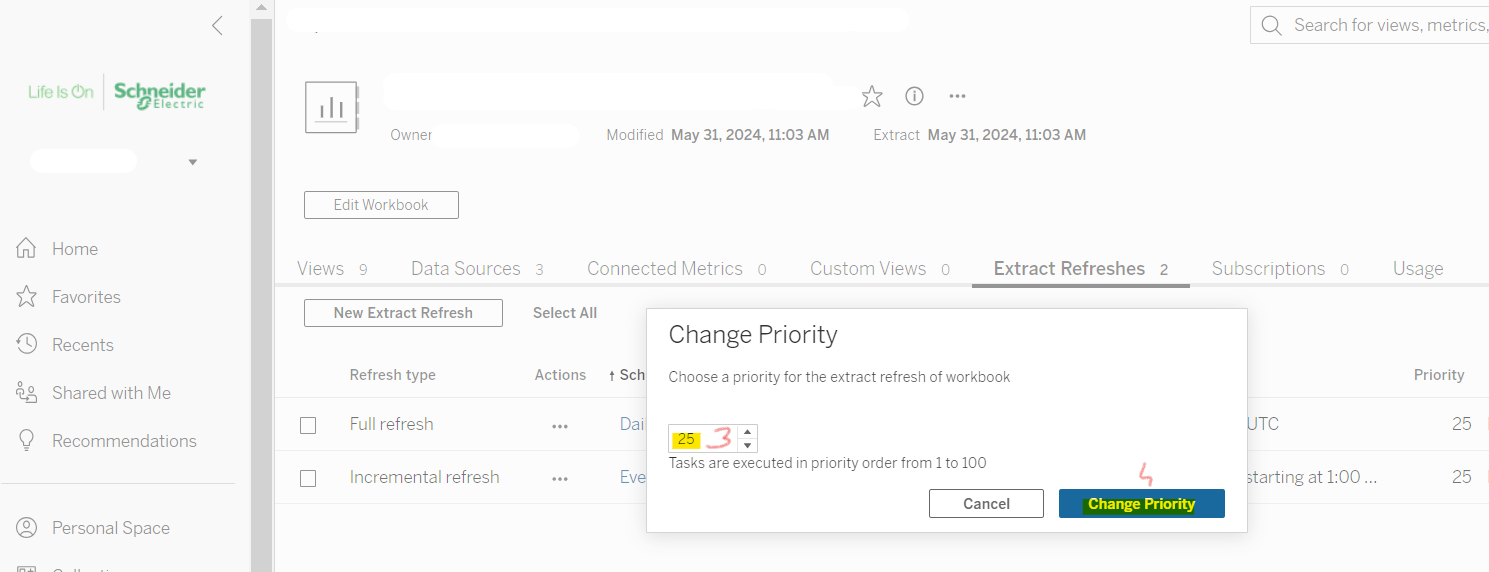


Figure - Click2

### API

Please note that no API method is available so far.

## How the url has been built

As you can see in Figure 3, we directly land to the page of each extract refresh.

So we create the URL based on the following logic

* If the site is NOT Enterprise BI

https://tableau.schneider-electric.com/#/site/<site\_urlname>/workbooks/<id\_content>/extractRefreshes

https://tableau.schneider-electric.com/#/site/<site\_urlname>/datasources/<id\_content>/extractRefreshes

* If the site is Enterprise BI

https://tableau.schneider-electric.com/#/datasources/<id\_content>/ extractRefreshes

https://tableau.schneider-electric.com/#/workbooks/<id\_content>/extractRefreshes

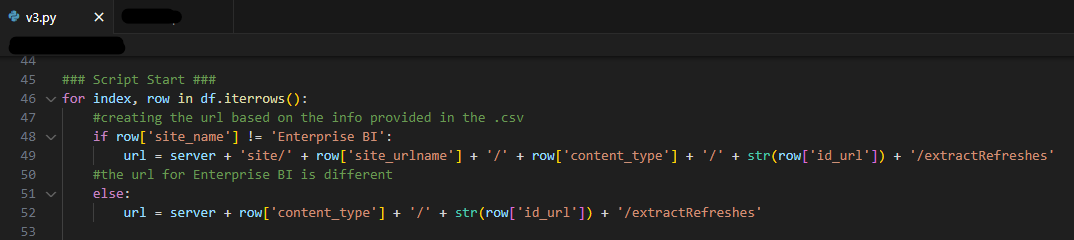
This is the part of the script where the URL is built, please have a look at the following chapter about the .csv file to better understand 

Figure - Building URL in the script

## The .csv file

So, to make the script work properly, we need a .csv file structured as follows:

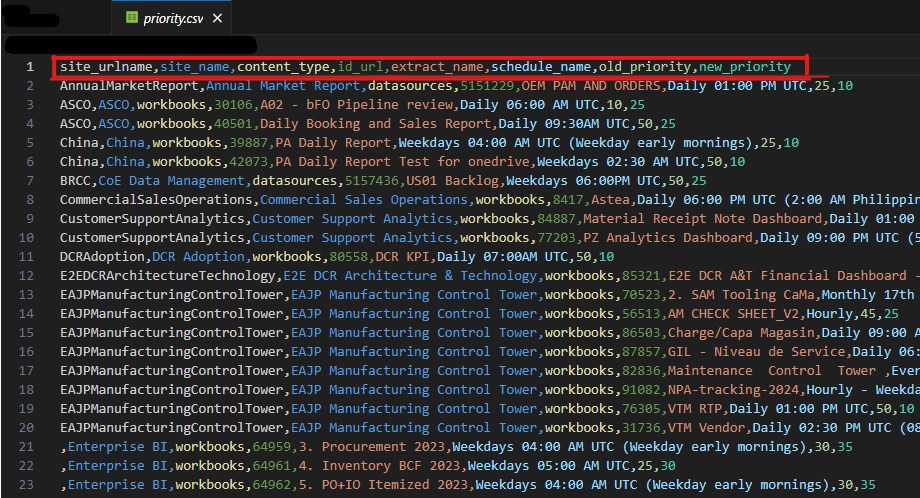


Figure - .csv

# Custom Query

To extract the above .csv file we perform the following query:

WITH

-- union between \_workbooks and \_datasources

CombinedResources as (

    SELECT

        ws.name,

        ws.id as content\_id,

        ws.id as id\_url, --i need a second one for the union all

        'workbooks' as content\_type

    FROM \_workbooks ws

    group by ws.name, content\_id, id\_url, content\_type

    union all

    SELECT

        ds.name,

        ds.id as content\_id,

        /\* Some values of \_datasources.id are associated to multiple values of data\_connections.id,

        because a data source can contain multiple data connections.

        However, each data\_connections.id land to the same page of the datasource,

        for this reason we use the max of data\_connections.id

        (also the min would work) \*/

        max(dcs.id) as id\_url,

        'datasources' as content\_type

    FROM \_datasources ds

    /\* the id of the url is not \_datasources.id,

    but data\_connections.id \*/

    inner join data\_connections dcs on ds.id = dcs.datasource\_id

    group by ds.name, content\_id, content\_type

),

-- create the average durations grouped by task id and assign them a new value

PriorityAdjustments AS (

    select

        task\_id,

        CASE

            WHEN AVG(completed\_at - started\_at) <= interval '1 minute' THEN 10

            WHEN AVG(completed\_at - started\_at) < interval '15 minutes' THEN 25

            WHEN AVG(completed\_at - started\_at) < interval '30 minutes' THEN 30

            WHEN AVG(completed\_at - started\_at) < interval '60 minutes' THEN 35

            WHEN AVG(completed\_at - started\_at) < interval '90 minutes' THEN 40

            WHEN AVG(completed\_at - started\_at) < interval '180 minutes' THEN 60

            WHEN AVG(completed\_at - started\_at) >= interval '180 minutes' THEN 100

            ELSE 50

        END AS new\_priority

    FROM

        background\_jobs

    GROUP BY

        task\_id

),

-- let's join the tables: tasks, \_schedules and \_sites

-- then, we can join this result with the above PriorityAdjustments

CombinedPriorities AS (

    SELECT

        ss.url\_namespace AS site\_urlname,

        ss.name AS site\_name,

        sc.name AS schedule\_name,

        ts.priority AS old\_priority,

        pa.new\_priority,

        ts.obj\_id

    FROM

        tasks ts

    INNER JOIN \_schedules sc ON ts.schedule\_id = sc.id

    INNER JOIN \_sites ss ON ts.site\_id = ss.id

    INNER JOIN PriorityAdjustments pa ON ts.id = pa.task\_id

    WHERE

        (ts.type = 'IncrementExtractTask' OR ts.type = 'RefreshExtractTask')

    GROUP BY

        ss.url\_namespace,

        ss.name,

        sc.name,

        ts.priority,

        pa.new\_priority,

        ts.obj\_id

),

-- where we join CombinedPriorities and CombinedResources (union between \_workbooks and \_datasources)

FinalQuery as (

    select

        cp.site\_urlname,

        cp.site\_name,

        cr.content\_type,

        cr.id\_url,

        cr.name as "extract\_name",

        cp.schedule\_name,

        cp.old\_priority,

        cp.new\_priority

    from

        CombinedPriorities cp

    inner join

        CombinedResources cr on cp.obj\_id = cr.content\_id

)

SELECT \* FROM FinalQuery

where

    -- let's keep only changes in priority

    old\_priority <> new\_priority

ORDER BY

    site\_name, extract\_name, schedule\_name

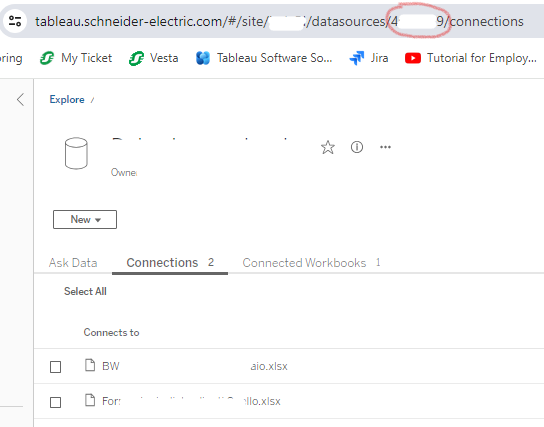
## Deep dive into the query

### **CombinedResources**

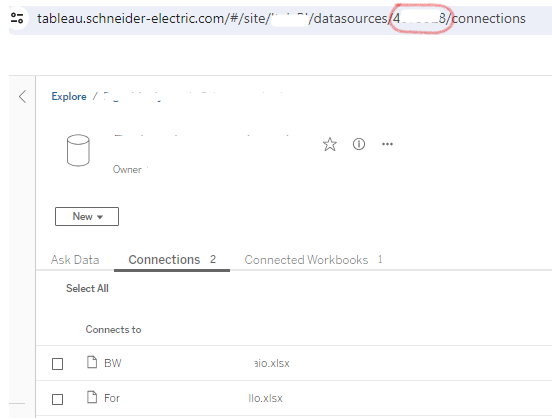
It’s a union between \_*datasource* and \_*workbook*.

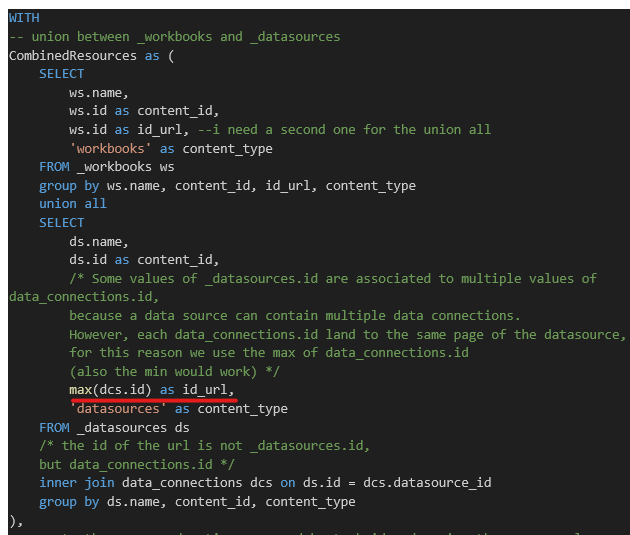
Unfortunately the \_*datasource.id* is not the right ID to build the URL (see chapter “How the URL has been built”), so we need the join with *data\_connections* to extract *data\_connections.id*.

Some values of *\_datasources.id* are associated to multiple values of *data\_connections.id*, because a data source can contain multiple data connections like in the example below (2 connections here).



However, this is not a problem, each *data\_connections.id* lands to the same datasource



So we can just pick up the max(*data\_connections.id*) 

Finally, we group by *ds.name, content\_id, content\_type* to avoid duplicates.

### **PriorityAdjustments**

From the *background\_jobs* table we compute the average duration of each *background\_jobs.task\_id* and we assign a new priority

### **CombinedPriorities**

* Joining *tasks* with *schedules, \_sites and PriorityAdjustmetns.*
* Filteringtasks.type = 'IncrementExtractTask' OR tasks.type = 'RefreshExtractTask'
* Extracting: *site\_urlname, site\_name, schedule\_name, old\_priority, new\_priority, tasks.obj\_id* (content\_id)

### **FinalQuery**

Where we join CombinedPriorities and CombinedResources

inner join CombinedResources cr on CombinedPriorities.obj\_id = cr.content\_id

Keeping only rows with *old\_priority* different from *new\_priority*

Alphabetical ordering

ORDER BY

    site\_name, extract\_name, schedule\_name

# Python Script

## Requirements

agate==1.6.3

atomicwrites==1.4.1

attrs==21.4.0

Automat==22.10.0

Babel==2.11.0

beautifulsoup4==4.11.1

blinker==1.6.2

cachetools==5.2.0

certifi==2022.6.15

cffi==1.15.1

charset-normalizer==2.1.0

click==8.1.3

cloudpickle==2.2.1

colorama==0.4.5

configparser==5.3.0

constantly==15.1.0

coverage==6.5.0

coveralls==3.3.1

cryptography==39.0.2

dbt-core==1.3.1

dbt-extractor==0.4.1

dbt-postgres==1.3.1

docopt==0.6.2

et-xmlfile==1.1.0

exceptiongroup==1.1.0

Flask==2.3.2

fsspec==2022.5.0

future==0.18.2

genson==1.2.2

google==3.0.0

google-api-core==2.8.2

google-api-python-client==2.52.0

google-auth==2.9.0

google-auth-httplib2==0.1.0

google-auth-oauthlib==0.5.2

googleapis-common-protos==1.56.3

greenlet==3.0.1

hologram==0.0.15

httplib2==0.20.4

hyperlink==21.0.0

hypothesis==6.68.2

idna==3.3

incremental==22.10.0

iniconfig==1.1.1

isodate==0.6.1

itsdangerous==2.1.2

Jinja2==3.1.2

joblib==1.2.0

jsonschema==3.2.0

leather==0.3.4

Logbook==1.5.3

MarkupSafe==2.1.1

mashumaro==3.0.4

minimal-snowplow-tracker==0.0.2

mock==5.0.1

MouseInfo==0.1.3

msgpack==1.0.4

mysql-connector-python==8.2.0

networkx==2.8.8

nltk==3.8.1

numpy==1.23.0

oauth2client==4.1.3

oauthlib==3.2.0

openpyxl==3.1.2

packaging==21.3

pandas==1.5.3

parsedatetime==2.4

pathspec==0.9.0

Pillow==9.2.0

pluggy==1.0.0

protobuf==4.21.2

psycopg2-binary==2.9.5

py==1.11.0

py4j==0.10.9.7

pyasn1==0.4.8

pyasn1-modules==0.2.8

PyAutoGUI==0.9.53

pycparser==2.21

PyGetWindow==0.0.9

PyMsgBox==1.0.9

PyMySQL==1.1.0

pyOpenSSL==23.0.0

pyparsing==3.0.9

pyperclip==1.8.2

PyRect==0.2.0

pyrsistent==0.19.2

PyScreeze==0.1.28

pytest==7.1.2

pytest-cov==4.0.0

pytest-warnings==0.3.1

python-dateutil==2.8.2

python-slugify==7.0.0

pytimeparse==1.1.8

pytweening==1.0.4

pytz==2022.1

PyYAML==6.0

regex==2022.10.31

requests==2.28.1

requests-oauthlib==1.3.1

rsa==4.8

scikit-learn==1.2.1

scipy==1.10.1

simplejson==3.18.3

six==1.16.0

sortedcontainers==2.4.0

soupsieve==2.3.2.post1

SQLAlchemy==2.0.22

sqlparse==0.4.3

tableauhyperapi==0.0.15305

tabpy==2.6.0

text-unidecode==1.3

textblob==0.17.1

threadpoolctl==3.1.0

tomli==2.0.1

tornado==6.2

tqdm==4.65.0

Twisted==22.10.0

twisted-iocpsupport==1.0.2

typing\_extensions==4.4.0

uritemplate==4.1.1

urllib3==1.26.9

Werkzeug==2.3.6

zope.interface==5.5.2

## Code

import os

from selenium import webdriver

from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as EC

import time

import pandas as pd

import logging

import datetime as datetime

# Logger Configuration

today\_date = datetime.datetime.now().strftime(f'%Y-%m-%d')

log\_filename = f'logs/logV3\_{today\_date}.log'

logging.basicConfig(filename=log\_filename,

                    level=logging.INFO,

                    format='%(asctime)s - %(levelname)s - %(message)s')

logging.info('''

             SCRIPT STARTED...

             ''')

#Reading the .csv file

fname = '\priority.csv'

try:

    excel\_file\_path = '.\priority\_files' + fname # Path

    df = pd.read\_csv(excel\_file\_path)

    logging.info('csv file imported')

except Exception as e:

    logging.error('error with csv file: {0}'.format(e))

# Open the Tableau Server website

server = 'https://tableau.schneider-electric.com/#/'

#Set the cheromedriver.exe

try:

    driver\_path = '.\chromedriver.exe' #path

    # Set the PATH environment variable to include the directory of the ChromeDriver executable

    os.environ['PATH'] += os.pathsep + os.path.dirname(driver\_path)

    driver = webdriver.Chrome() # Create a Chrome webdriver instance

    logging.info('chromedriver is working fine')

except Exception as e:

    logging.error('error with chromedriver: {0}'.format(e))

### Script Start ###

for index, row in df.iterrows():

    #creating the url based on the info provided in the .csv

    if row['site\_name'] != 'Enterprise BI':

        url = server + 'site/' + row['site\_urlname'] + '/' + row['content\_type'] + '/' + str(row['id\_url']) + '/extractRefreshes'

    #the url for Enterprise BI is different

    else:

        url = server + row['content\_type'] + '/' + str(row['id\_url']) + '/extractRefreshes'

    #costant values for the loop

    schedule\_name = row['schedule\_name']

    priority\_new = row['new\_priority']

    #lands to the extract refresh page

    try:

        driver.get(url)

        time.sleep(2)

        logging.info('row {0} - landed to: {1}'.format(index+1, url))

        #click on the actionsmenu-cell button (3 dots)

        try:

            actions\_button = WebDriverWait(driver, 10).until(

                EC.visibility\_of\_element\_located(

                    (By.XPATH, "//div[@aria-colindex='3' and contains(@data-tb-test-id, 'actionsmenu-cell') and following-sibling::div[@aria-colindex='4' and contains(., '{0}')]]".format(schedule\_name)

                    )

                )

            )

            actions\_button.click()

            time.sleep(1)

            logging.info('cliked on the actionsmenu-cell'.format(index+1))

            #click on the fourth element of the menu: action-menu-tasks-priority-MenuItem

            try:

                # waiting for the menu to be visible

                dropdown\_menu = WebDriverWait(driver, 10).until(

                    EC.visibility\_of\_element\_located(

                        (By.XPATH, "//div[@data-tb-test-id='action-menu-TextMenuItem']"

                            )

                    )

                )

                # click on the forth element of the list (change priority)

                fourth\_item = dropdown\_menu.find\_element(

                    By.XPATH, "//div[@data-tb-test-id='action-menu-tasks-priority-MenuItem']"

                    )

                fourth\_item.click()

                time.sleep(1)

                logging.info('cliked on the action-menu-tasks-priority-MenuItem')

                #delete the old priority

                try:

                    # find the box

                    input\_element  = WebDriverWait(driver, 10).until(

                        EC.visibility\_of\_element\_located(

                            (By.XPATH, "//input[@data-tb-test-id='-IntegerStepperWidget-TextInput']")

                        )

                    )

                    # JavaScript to drop the existing element

                    driver.execute\_script("arguments[0].value = '';", input\_element)

                    time.sleep(1)

                    logging.info('old priority deleted')

                    #insert the new priority

                    try:

                        input\_element.send\_keys('{0}'.format(priority\_new))

                        time.sleep(1)

                        logging.info('priority changed, waiting for confirming click')

                        # find and click on the botton "change priority"

                        try:

                            confirm\_button = WebDriverWait(driver, 10).until(

                                EC.visibility\_of\_element\_located(

                                    (By.XPATH, "//button[@data-tb-test-id='confirm-action-dialog-confirm-Button']")

                                )

                            )

                            confirm\_button.click()

                            logging.info('priority changed, confirming click done')

                        except Exception as e:

                            logging.error('priority changed, confirming click NOT done')

                    except Exception as e:

                        logging.error('priority NOT changed, error {0}'.format(e))

                except Exception as e:

                    logging.error('old priority NOT deleted, error {0}'.format(e))

            except Exception as e:

                logging.error('NOT cliked on the action-menu-tasks-priority-MenuItem, error {0}'.format(e))

        except Exception as e:

            logging.error('NOT cliked on the actionsmenu-cell button, error {0}'.format(e))

    except Exception as e:

        logging.error('row {0} - url not reached, error: {1}'.format(index+1, e))

try:

    # Close the browser

    driver.quit()

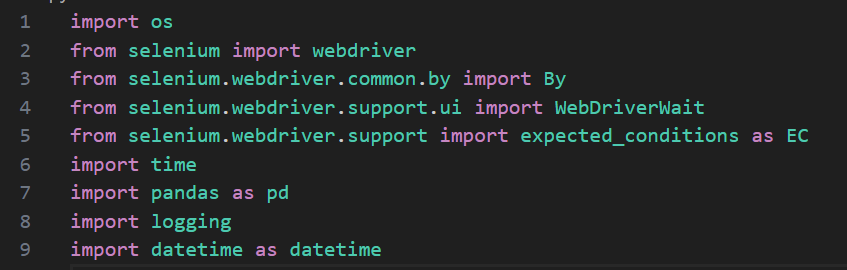
    logging.info('browser closed, script ends')

except Exception as e:

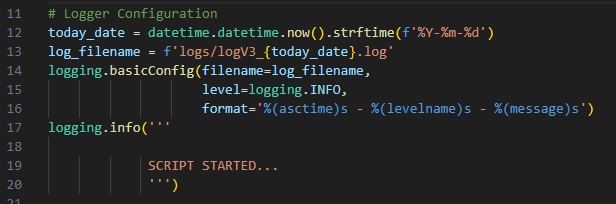
    logging.error('browser NOT closed')

## Code Explanation

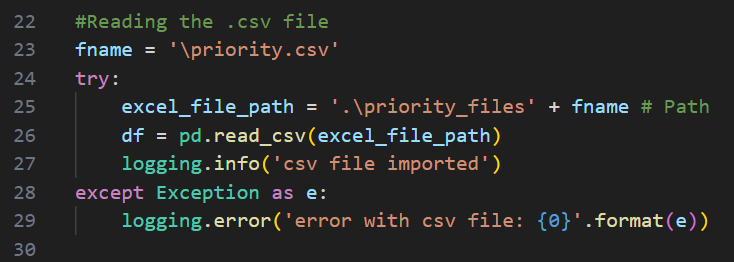
Importing the necessary packages



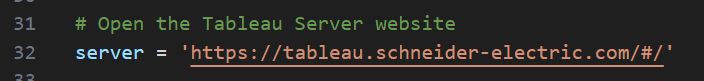
Set a logger



Reading the .csv file

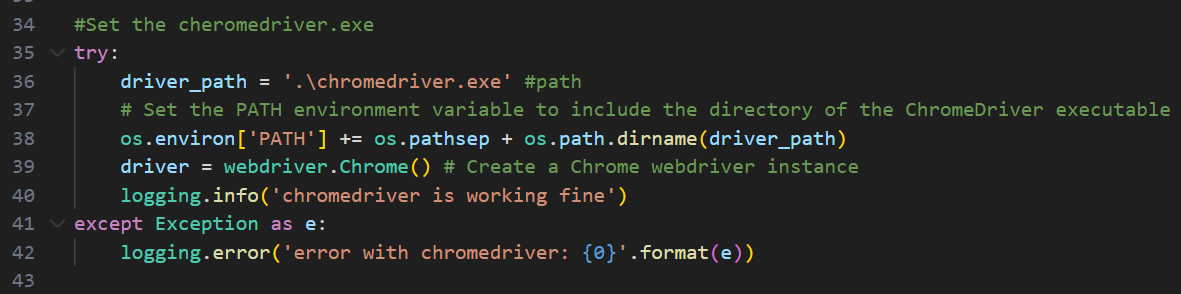


Set the basic URL



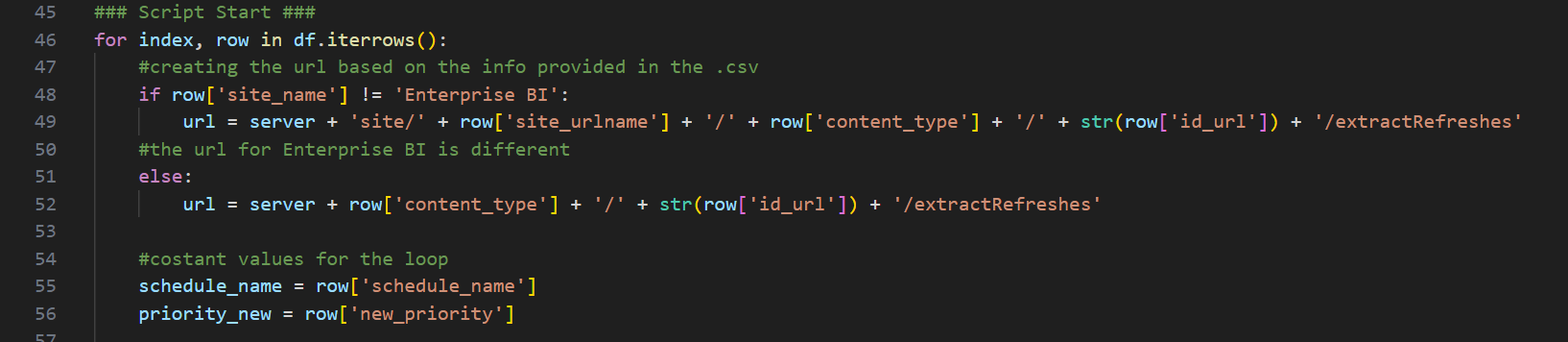
Chromedriver

Pay attention! This must be always aligned with your Google Chrome version

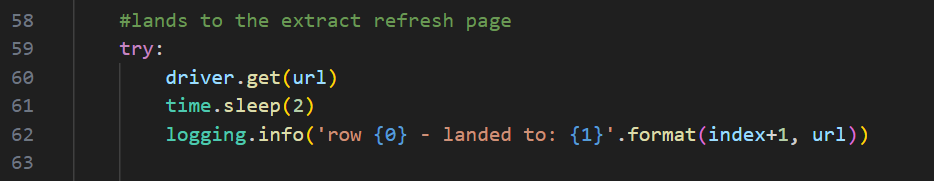


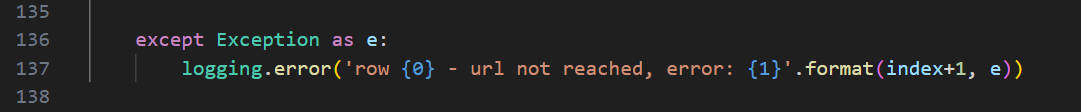
Defining the URL for each row of the .csv file

Defining the schedule\_name and the priority\_new for each row of the .csv file



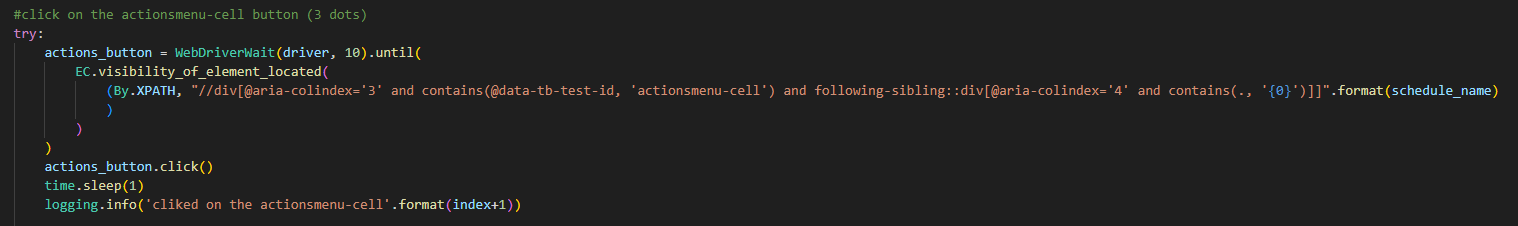
Lands to the extract refresh page with a GET call

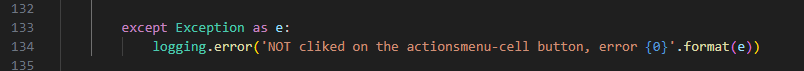




Click on the actionsmenu-cell button (3 dots)

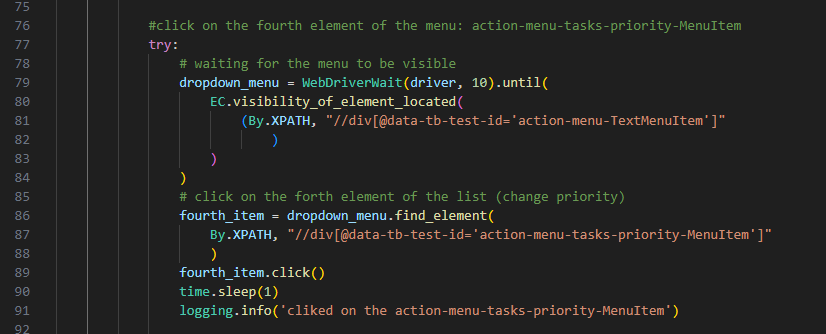
Point 1 of the “2.1 CLICKS” chapter

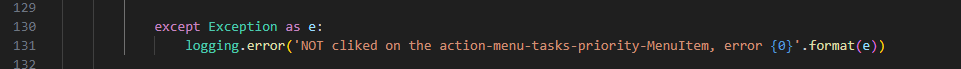




Click on the fourth element of the menu (Change Priority

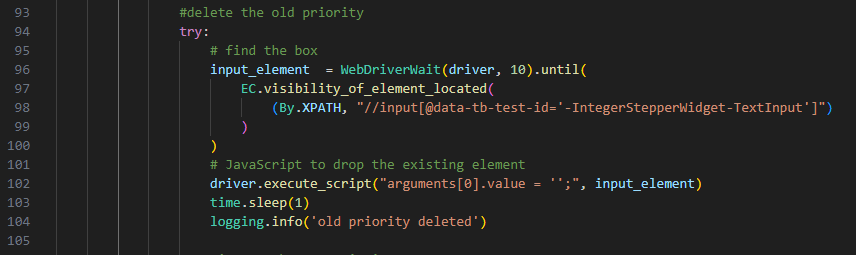
Point 2 of the “2.1 CLICKS” chapter





Delete the old priority

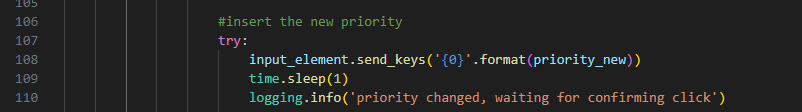
Point 3 of the “2.1 CLICKS” chapter





Insert the new priority

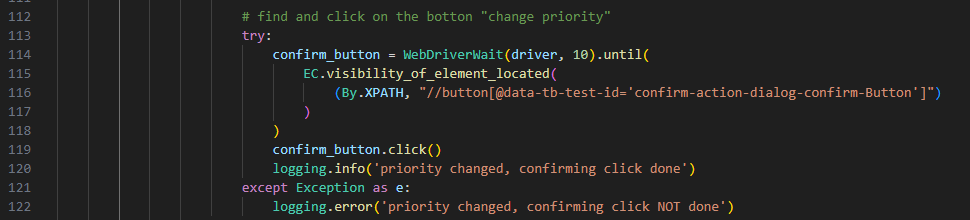
Point 3 of the “2.1 CLICKS” chapter



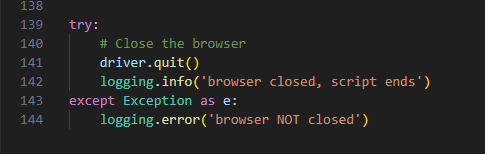


Confirm

Point 4 of the “2.1 CLICKS” chapter



Finally, close the instance



# Important Note

* why we don't have the corresponding tasks.id in background\_jobs.task\_id? because the task never runned, or data are not yet written in the postgres
* viceversa: sometimes, we don't have the corresponding background\_jobs.task\_id in tasks.id. because that tasks was deleted

**LOGIN**

from version v2.py onward is deleted (automatic with SAML)