Database Project
URL Dashboard

Spring 2023

Abolfazl Arab Ansari
Amirkabir University of Technology
Polytechnic Tehran

URL Dashboard Project

Introduction

The purpose of designing and implementing a link-shortening project along with a dashboard is to display statistics.
Implementation For implementing this project, you can use any programming language, library, and technology that supports working with SQL Server. However, it is mandatory to use SQL Server for storing information. By referring to this link, you can see the information about SQL Server communication libraries in suitable programming languages.

Requirements

Create a new link The system receives an input string (e.g. www.averylongrul.com). If this link has already been shortened, it returns the previous value. Otherwise, it generates and maps a new random string of 6 characters, consisting of lowercase letters and numbers (e.g., n2r1tw).
Reference to a shortened link The system receives a string containing the shortened form of a link. If the equivalent link is registered in the system, it is returned; otherwise, an error is thrown.
Utilizing at least two functions, initiators, processes, and all the concepts covered during the term in the design and implementation of the project is necessary.
Note: To reduce the storage size, any link that has not been referred to within a week should expire.

Dashboard Information

- Displaying charts of the number of newly registered links and the number of referrals to shortened links per day.

- Displaying the top 3 links that have received the most referrals.
- Displaying all mappings in the system along with the time remaining until expiration and the number of referrals made.

Link Shortening System Project Delivery

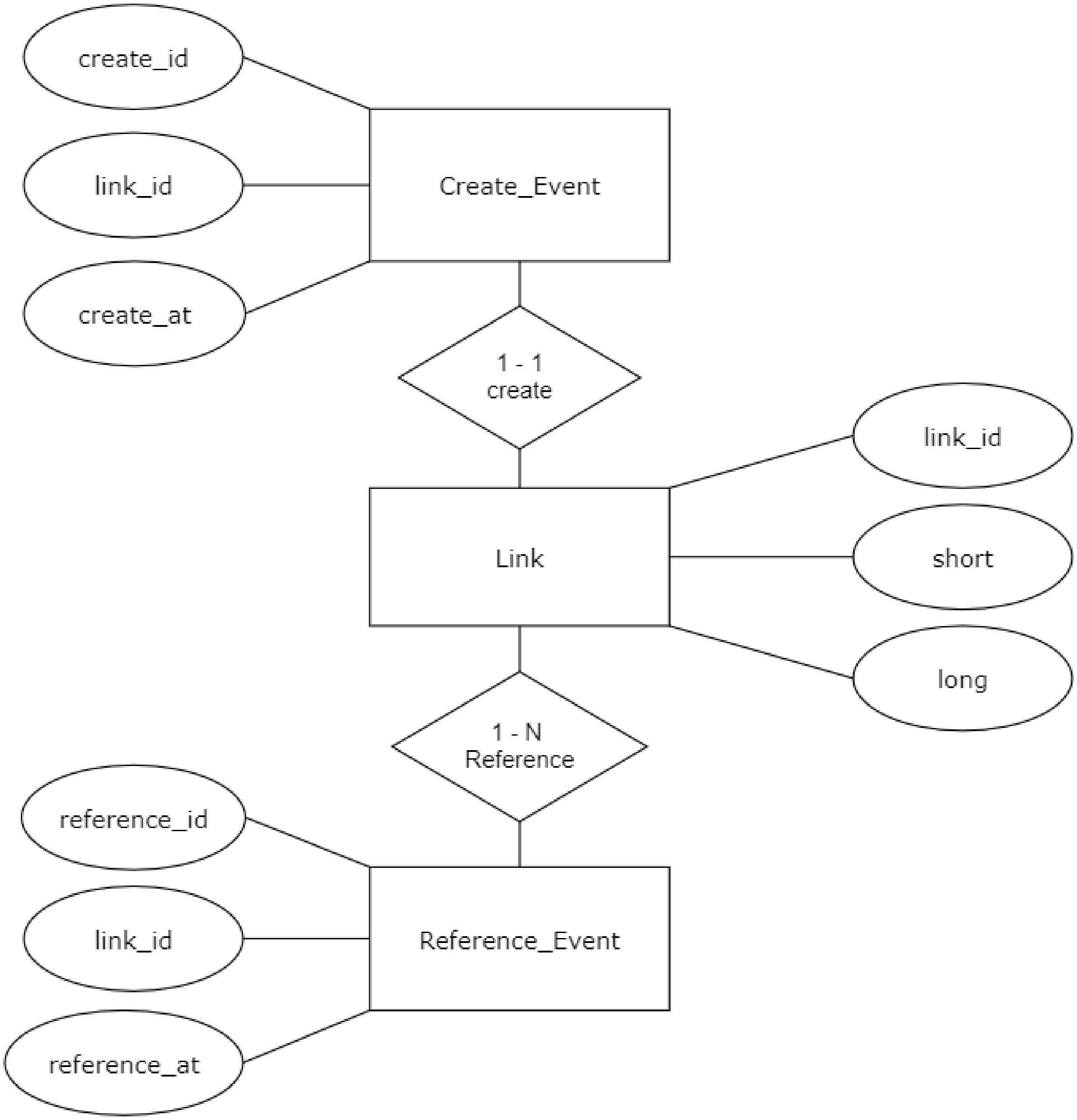
- The above project should be done individually.
- The method of project delivery will be agreed upon with the students in class.
- Creativity in implementing the above requirements or considering new requirements will receive positive grading.

Best of luck!

Report:

First Step: detecting the entities, relations and attributes of the problem

We can implement different databases base on different schemas but we behave according this Entity Relationship Diagrams that we propose:

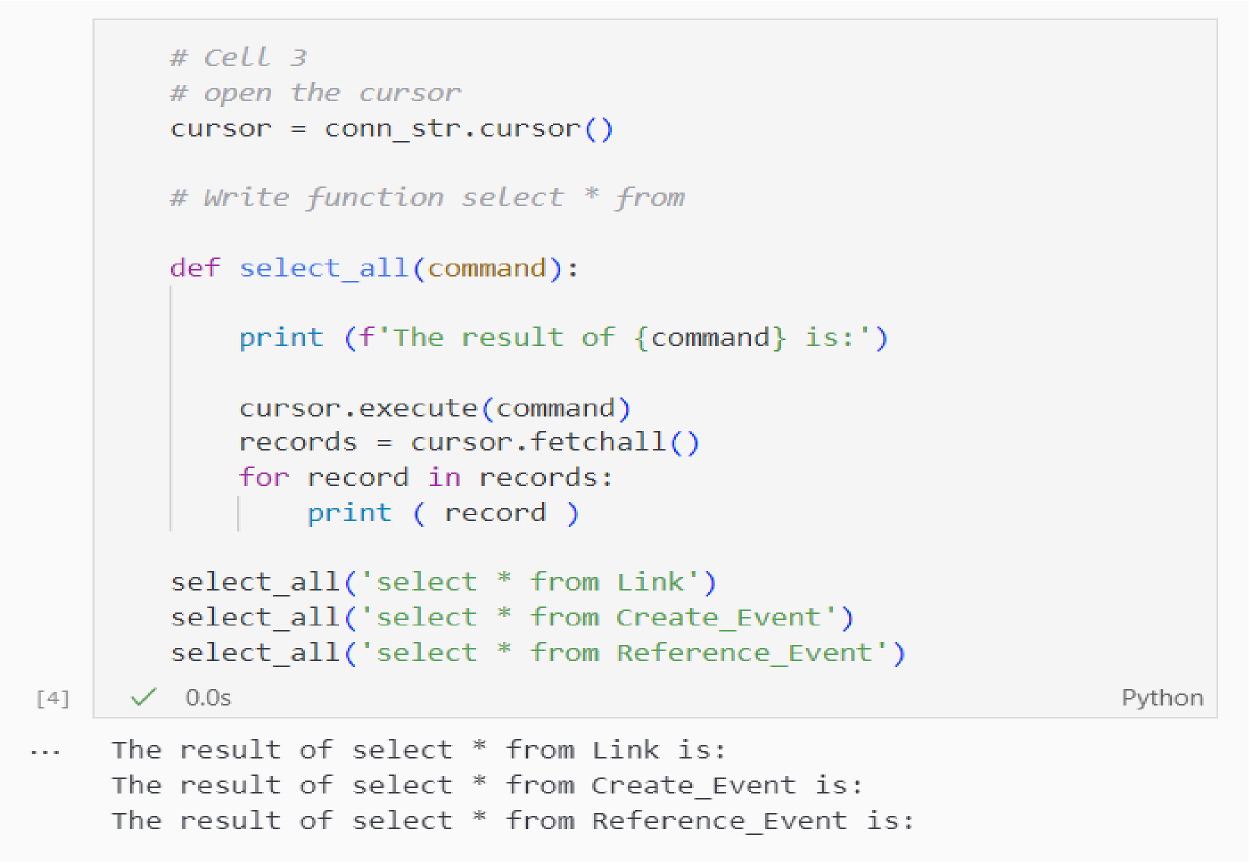


The code for implementing the database and all the follow-up tasks with related comments are available on main.ipynb

Process:

- Cell 1: Import needed packages

  Such as podbc
- Cell 1: Connect to SQL Server and create connection cursor
- Cell 2: Write Schema of Database, Execute and Commit to shaping our database
- Cell 3: A snapshot of the database when all tables are empty

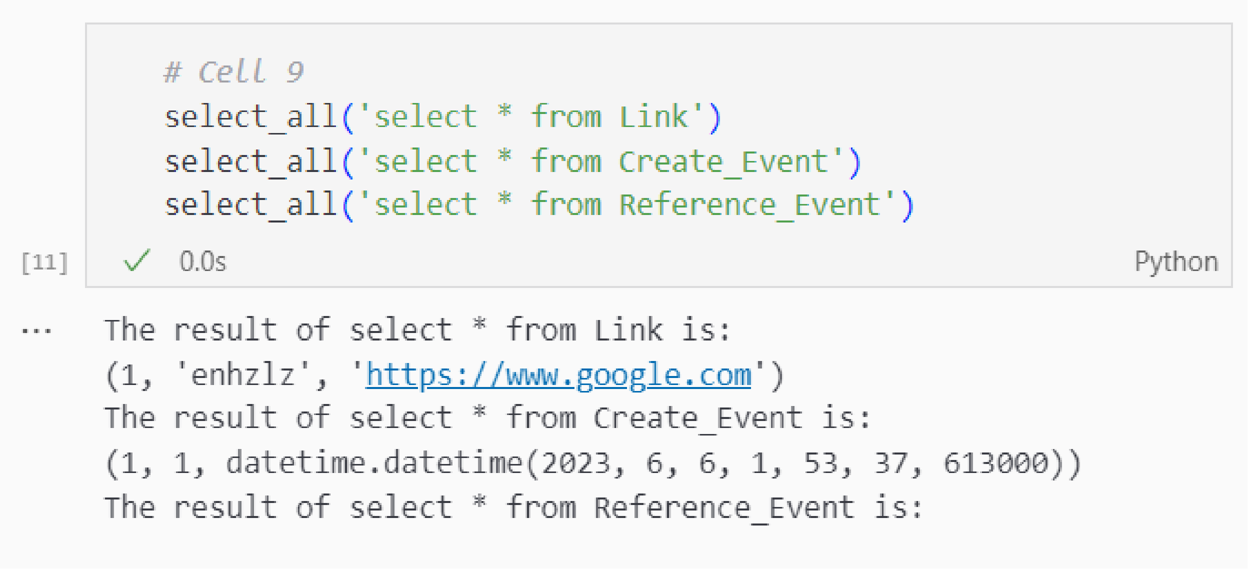


```python
# Cell 3
# open the cursor
cursor = conn_str.cursor()

# Write function select * from

def select_all(command):

    print (f'The result of {command} is:')

    cursor.execute(command)
    records = cursor.fetchall()
    for record in records:
        print ( record )

select_all('select * from Link')
select_all('select * from Create_Event')
select_all('select * from Reference_Event')
```

```
The result of select * from Link is:
The result of select * from Create_Event is:
The result of select * from Reference_Event is:
```

- Cell 6: Define a trigger to insert into Create_Event Table after inserting into Link Table

- Cell 7: Define the insertion function, get a long URL, and add it to the Link table (note: the previous trigger is on)
- Cell 8 & 9: insert one Link and a snapshot of all three tables

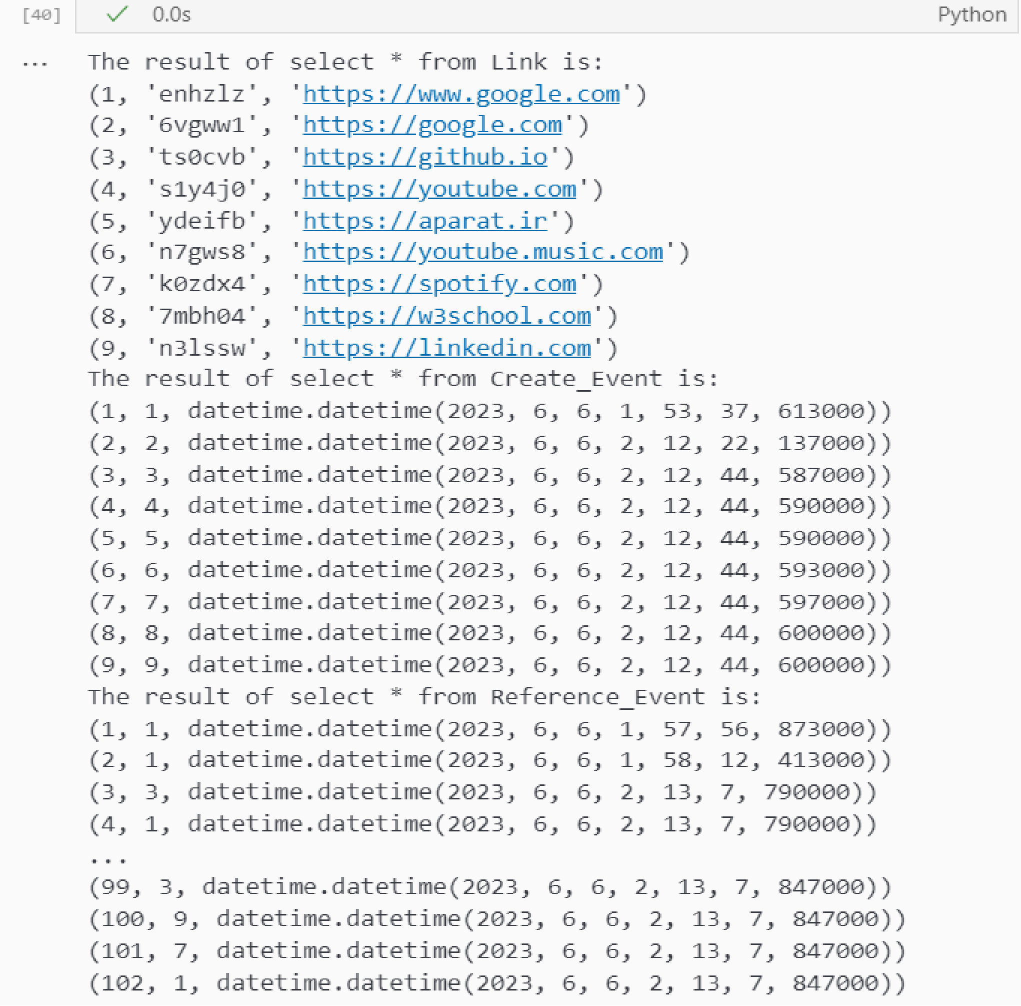


```
# Cell 9
select_all('select * from Link')
select_all('select * from Create_Event')
select_all('select * from Reference_Event')
```

```
The result of select * from Link is:
(1, 'enhzlz', 'https://www.google.com')
The result of select * from Create_Event is:
(1, 1, datetime.datetime(2023, 6, 6, 1, 53, 37, 613000))
The result of select * from Reference_Event is:
```

- Cell 12 & 13: Define Reference functionality and insert it into Reference_Event Table as well
- Cell 14: Raise Error when short form does not exist
- Cell 15: Alter dependent tables on delete cascade on
- Cell 16: Write a procedure in order to remove expired links (links with an age of more than 7 days)

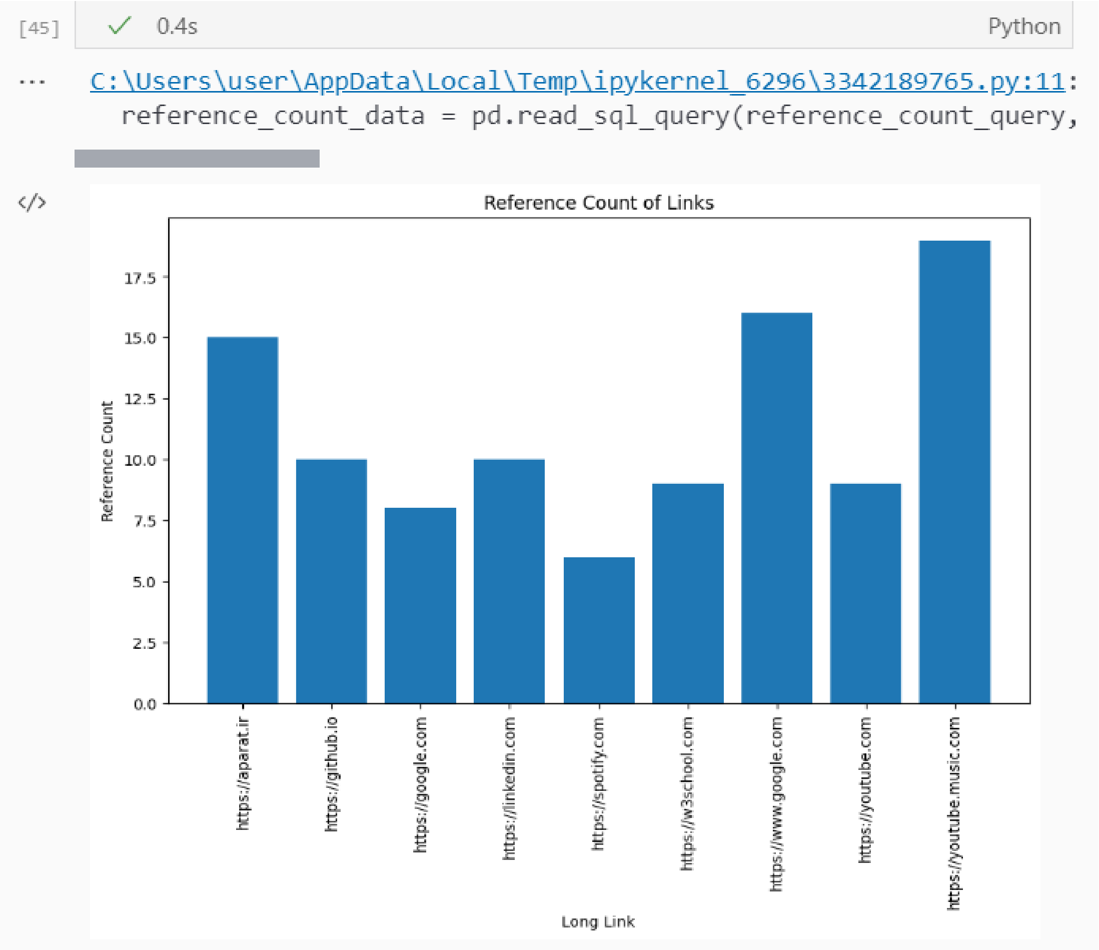
- Cell 17 - 22: add some records to all tables



```
The result of select * from Link is:
(1, 'enhzlz', 'https://www.google.com')
(2, '6vgww1', 'https://google.com')
(3, 'ts0cvb', 'https://github.io')
(4, 's1y4j0', 'https://youtube.com')
(5, 'ydeifb', 'https://aparat.ir')
(6, 'n7gws8', 'https://youtube.music.com')
(7, 'k0zdx4', 'https://spotify.com')
(8, '7mbh04', 'https://w3school.com')
(9, 'n3lssw', 'https://linkedin.com')
The result of select * from Create_Event is:
(1, 1, datetime.datetime(2023, 6, 6, 1, 53, 37, 613000))
(2, 2, datetime.datetime(2023, 6, 6, 2, 12, 22, 137000))
(3, 3, datetime.datetime(2023, 6, 6, 2, 12, 44, 587000))
(4, 4, datetime.datetime(2023, 6, 6, 2, 12, 44, 590000))
(5, 5, datetime.datetime(2023, 6, 6, 2, 12, 44, 590000))
(6, 6, datetime.datetime(2023, 6, 6, 2, 12, 44, 593000))
(7, 7, datetime.datetime(2023, 6, 6, 2, 12, 44, 597000))
(8, 8, datetime.datetime(2023, 6, 6, 2, 12, 44, 600000))
(9, 9, datetime.datetime(2023, 6, 6, 2, 12, 44, 600000))
The result of select * from Reference_Event is:
(1, 1, datetime.datetime(2023, 6, 6, 1, 57, 56, 873000))
(2, 1, datetime.datetime(2023, 6, 6, 1, 58, 12, 413000))
(3, 3, datetime.datetime(2023, 6, 6, 2, 13, 7, 790000))
(4, 1, datetime.datetime(2023, 6, 6, 2, 13, 7, 790000))
...
(99, 3, datetime.datetime(2023, 6, 6, 2, 13, 7, 847000))
(100, 9, datetime.datetime(2023, 6, 6, 2, 13, 7, 847000))
(101, 7, datetime.datetime(2023, 6, 6, 2, 13, 7, 847000))
(102, 1, datetime.datetime(2023, 6, 6, 2, 13, 7, 847000))
```

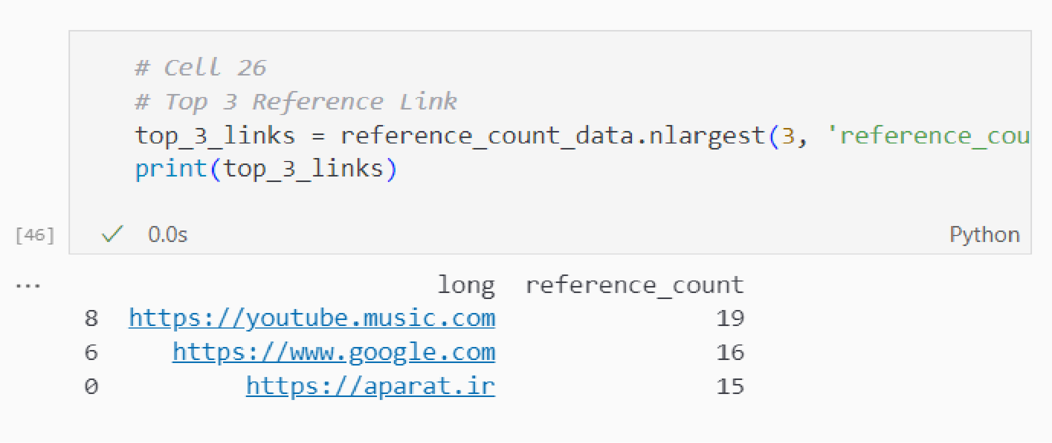
- Cell 23: Draw a plot for records of Create_Event of the last day

- Cell 25: Draw a plot of reference counts on the Reference_Event table for the last day



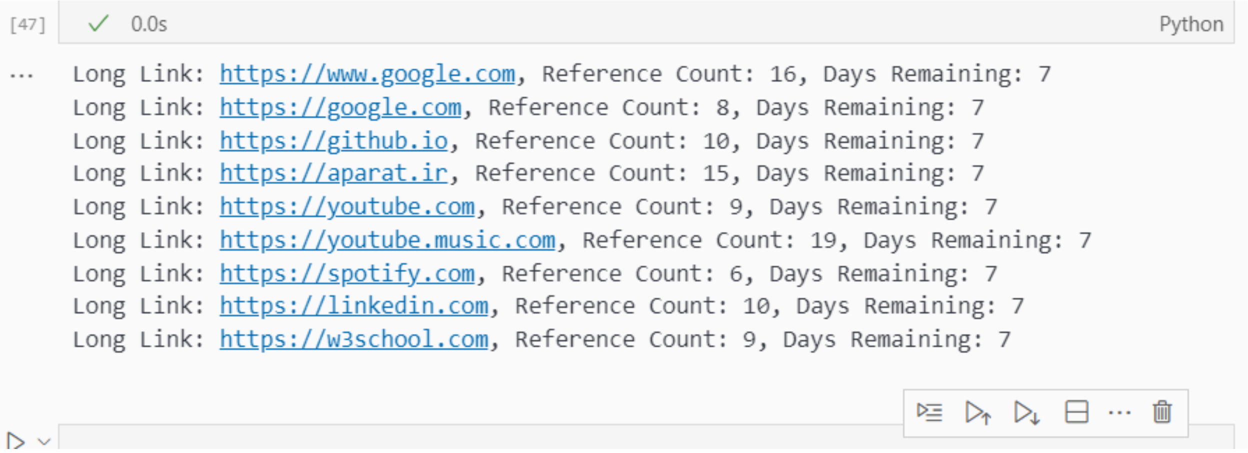
```
[45]   ✓ 0.4s                                                        Python

...    C:\Users\user\AppData\Local\Temp\ipykernel_6296\3342189765.py:11:
         reference_count_data = pd.read_sql_query(reference_count_query,
```

- Cell 26: Top 3 most favorite links



```
       # Cell 26
       # Top 3 Reference Link
       top_3_links = reference_count_data.nlargest(3, 'reference_cou
       print(top_3_links)

[46]   ✓ 0.0s                                                    Python

...                       long  reference_count
    8  https://youtube.music.com               19
    6     https://www.google.com               16
    0          https://aparat.ir               15
```

- Cell 27: Links with Reference count and Days Remaining until Expiration



```
Long Link: https://www.google.com, Reference Count: 16, Days Remaining: 7
Long Link: https://google.com, Reference Count: 8, Days Remaining: 7
Long Link: https://github.io, Reference Count: 10, Days Remaining: 7
Long Link: https://aparat.ir, Reference Count: 15, Days Remaining: 7
Long Link: https://youtube.com, Reference Count: 9, Days Remaining: 7
Long Link: https://youtube.music.com, Reference Count: 19, Days Remaining: 7
Long Link: https://spotify.com, Reference Count: 6, Days Remaining: 7
Long Link: https://linkedin.com, Reference Count: 10, Days Remaining: 7
Long Link: https://w3school.com, Reference Count: 9, Days Remaining: 7
```

- Cell 28: Commit the cursor and close it

Thanks,
Abolfazl