# Facebook Statistics API Specifications

| | |
|---|---|
| Version: | 1 |
| Date Modified: | 31 January 2018 |

## 1. The context

We are a company specialising in commodities trading. We are developing a complex system to provide an analytics platform for our traders. We wish to open our system to third party software houses, offering them ability to "plug-in" independent software modules that implement particular functions. We therefore request all interested companies to provide an independent software module that implements an API as specified in the rest of this document.

## 2. Function of the Facebook Statistics API

This service enables the customers to receive Facebook statistics related to any company or brand worldwide. The source of these data is the Facebook Graph API. The objective is to build an API that would easily enable a user to find all the social network statistics related to a particular brand or company.

### What the module does

The module aims to isolate some specific statistics of a brand or company according to the user's criteria.

The user has to input 3 main information: page name of the company or instrument ID, list of interested statistics, time period of interest.

The output would be a JSON file containing all the requested statistics for the company, in the specified time period.

## 3. Input Parameters for API

**The Company name or InstrumentId**

This input contains an string indicting the company name or instrument ID you want to get statistics about. Each company is identified by a unique code (Instrument ID). *See References for more details.*

For example: if you want to retrieve news related to Woolworth input should be :

- CompanyID = “WOW.AX,”

Alternatively, a user should be able to use company names:

- CompanyIDs = “Woolworths”

**The Period of Time:**

It’s composed of two inputs:

- *A start date*
- *An end date*

Both dates have the respect the following format: **“yyyy-MM-ddTHH:mm:ss.SSSZ”.**

These inputs can NOT be empty.

For example: if I want to retrieve all the statistics between the first of October 2015 at 08:45:10.295 (8:45am 10 seconds and 295 milliseconds) and the first of November 2015 at 19:37:12:193 (7:37pm 12 seconds and 193 milliseconds), my inputs will be:

- start_date = “2015-10-01T08:45:10.295Z”
- end_date = “2015-11-01T19:37:12.193Z”

The module will then filter the data according to the preferences of user and return a formatted output for user.

**List of interested statistics**

Regarding a company user should be able to select the information they need-

id, name, website, description ,category, fan_count, post_type, post_message, post_created_time, post_like_count, post_comment_count

## 4. Output Format for the API

The module should generate and return a JSON output file to users of the following format.

```
{
  "Facebook Statistic Data": [
  {
        "PageId"": <>,

         "InstrumentIDs": <>,

         “CompanyNames”:<>,

        "PageName": <>,

        "Website": <>,

        "Description ": <>,

        "Category": <>,

        "fan_count": <>,

        "posts [

        {

                "post_id": <>,

                "post_type": <>,

                "post_message": <>,

                "post_created_time": <>,

                "post_like_count": <>,

                "post_comment_coun": <>

        }

..

..

]
```

The fields related to posts should be returned for all posts published in the page within the user defined time duration.

Another output of the API is a log file must contain the following information:

- Developer team
- Module name and version
- Parameters passed
- An indication if execution has been successful or there is an error
- If error, indicate the nature of the error
- If successful, need to supply
  - ➢ Start date and time of execution
  - ➢ End date and time of execution
  - ➢ Elapsed time

- ➢ Output file name

## 5. Acquiring the data for API

Appendix 6.2 provides an overview on how to use Facebook REST API to acquire data. Within your implementation you need to translate user request into one or more Facebook Graphe API queries, acquire data and transform them into defined output format in section 4.

Particularly you may need to call Facebook graph API twice to get page related data and data related to posts within the page.

Within your API you need to translate the company ID into Facebook page id. You can do this by include a map between instrumentIDs and company names to translate user requests. Facebook page search (https://graph.facebook.com/v2.12/search?q=<keyword>&type=page) can also be helpful.

## 6. Appendix

### 6.1 Additional Information:

Teams have the choice of running their system on two different platforms:

- Standalone Program
  - ♣PC running Windows
  - ♣Unix/Linux platform
- Web service (accessible via a REST interface)

Throughout the workshop, each team will need to have a Web page. As a minimum, the page is showing:

- The team name and members
- Consecutive releases of their module. Each release page must include a link to download the module and information about:
  - The date and version of the release
  - What has been implemented so far
  - Differences with previous version
  - Clear instructions on how to run the module in standalone mode
  - Guidelines on how to integrate the module with other systems
  - Any test software or data

### 6.2 Quick overview of Facebook Graph API and the Graph API Explorer.

Facebook Graph API provide access to public and/or private Facebook data. Public data is typically available without users being prompted for authorization. However, private data require users to grant your application access. In this course, we are using Facebook Graph API to access publicly available data.

*Nodes, Fields and Connections*

Facebook entities are called nodes. Example of nodes are a page, a post or a user. Every node has fields describing it. Examples of fields of a post are its id, message, type, creation time, etc. In addition, a node can be connected to other nodes through “connections”. A connection is a set of nodes.

For example, a Facebook page (e.g. company page) has id, name, fan_count or website as fields. It is also connected to a set of posts, events, likes, etc. A post has id, message, from, type, creation time, shares, etc. as fields and it is also connected to a set of comments, likes or attachments. Every comment is a node on its own, connected to a list of replies (a set of comments on the comment). There are many more fields and connections described in the documentation of Facebook Graph API.

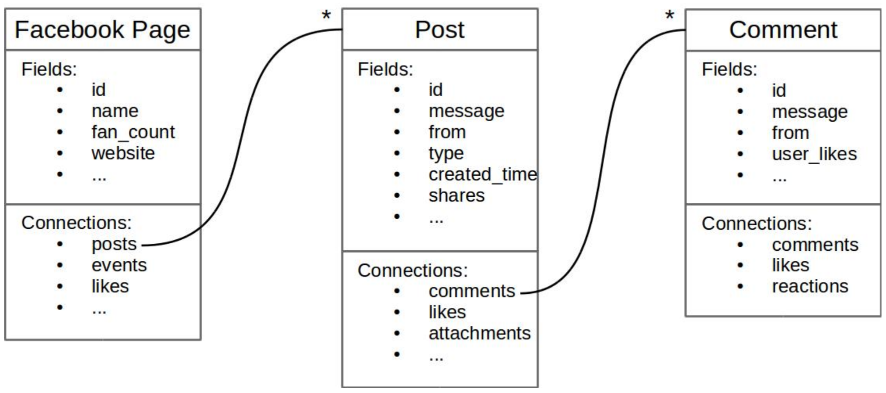


*Figure 0.1 Developer API data model*

Querying Facebook Graph API consists of querying nodes, their fields and their connections using a REST protocol. For example, given a node with an id={nodeid}, accessing the fields f1, f2 and f3 of the node id would be using the query:

```
GET {nodeid}?fields=f1,f2,f3
```

Accessing the connection c of the node would be using the query:

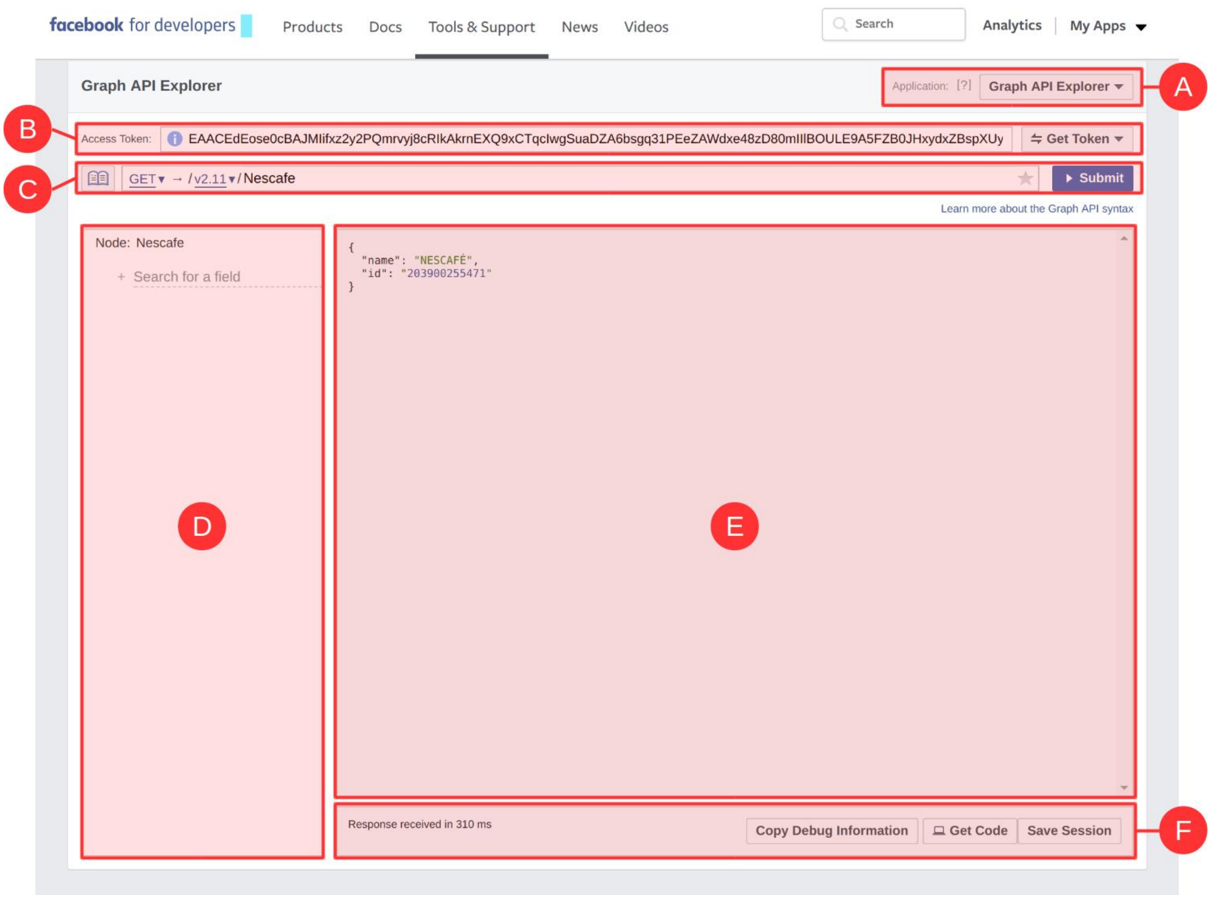
```
GET {nodeid}/c
```

Furthermore, accessing selected fields f6 and f7 from connection c of the node would be using the query:

```
GET {nodeid}/c}?fields=f6,f7
```

*Facebook Graph API Explorer*

Facebook provides a variety of tools to help developers access its data and interact with its API. Facebook's Graph API Explorer (https://developers.facebook.com/tools/explorer/) allows developers to test API calls and debug responses. The Graph API Explorer is composed of the following key elements:



Access Token: EAACEdEose0cBAJMIifxz2y2PQmrvyj8cRIkAkrnEXQ9xCTqclwgSuaDZA6bsgq31PEeZAWdxe48zD80mIlIBOULE9A5FZB0JHxydxZBspXUy

GET → /v2.11/Nescafe

Node: Nescafe

```
{
  "name": "NESCAFÉ",
  "id": "203900255471"
}
```

Response received in 310 ms

*Figure 0.2 Facebook Graph API Explorer*

A. Facebook Graph API is accessed by a Facebook Application. You can create your own Facebook Application and use it in Graph API Explorer. For testing purposes, the default application is called "Graph API Explorer". Every application has different settings which impact what kind of data it can access.
B. The Access Token is a temporary key used by the Facebook Application being used to access the API. It expires after some time. If you click on "Get Token", you would be prompted with the kind of data you would like to be granted access to. For the purpose of this course, we would like to access publicly available data from Facebook Graph API, so no need to request additional privileges.
C. A query is constructed and can be typed directly in the query bar (C). Queries follow the REST API, with GET, POST and DELETE operations. In this course, we are simply downloading data, so the GET method is used.
D. The Query Builder is a representation of the structure of the current node being accessed (including fields and connections). It allows you to select which fields and connections you would like to query. The Query Builder automatically append any selected fields and connections to the query in C.
E. The output of a query is displayed in JSON format in E.
F. Debugging, code generation and session management is available from this control panel.

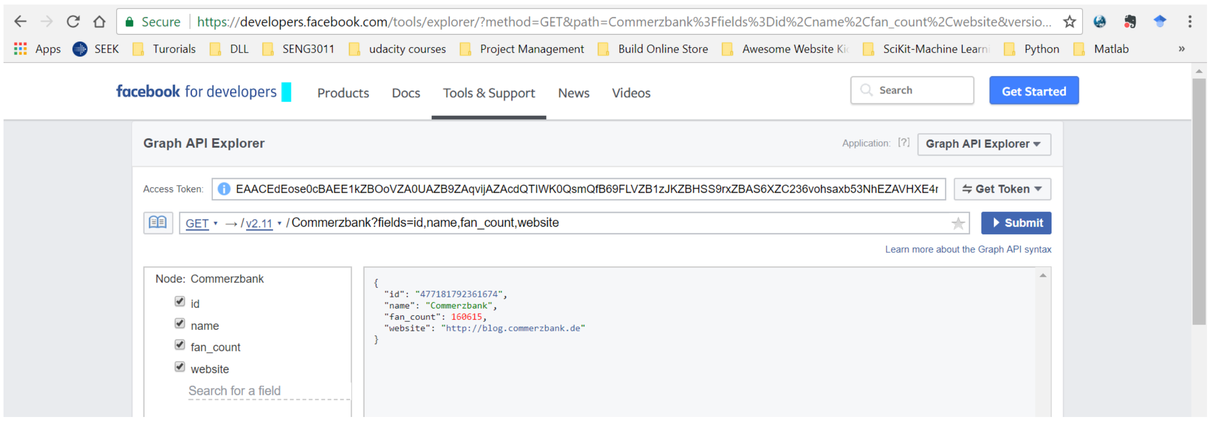
## Get data from Facebook Graph API for a specific company/brand page

On Facebook Graph API Explorer, type the id of company/brand page and hit enter. The following output is for the Commerzbank brand page. You can find the id of a brand page by searching for it on Facebook. A typical brand page would have a URL as follows:

```
https://www.facebook.com/pageid
```

A Facebook Graph API query for Commerzbank requesting fields id, name, fan_count and website would look like this

```
GET/Commerzbank?fields=id,name,fan_count,website
```



https://developers.facebook.com/tools/explorer/?method=GET&path=Commerzbank%3Ffields%3Did%2Cname%2Cfan_count%2Cwebsite&versio...

Graph API Explorer

Access Token: EAACEdEose0cBAEE1kZBOoVZA0UAZB9ZAqvijAZAcdQTIWK0QsmQfB69FLVZB1zJKZBHSS9rxZBAS6XZC236vohsaxb53NhEZAVHXE4r

GET → /v2.11 /Commerzbank?fields=id,name,fan_count,website

Node: Commerzbank
- ☑ id
- ☑ name
- ☑ fan_count
- ☑ website

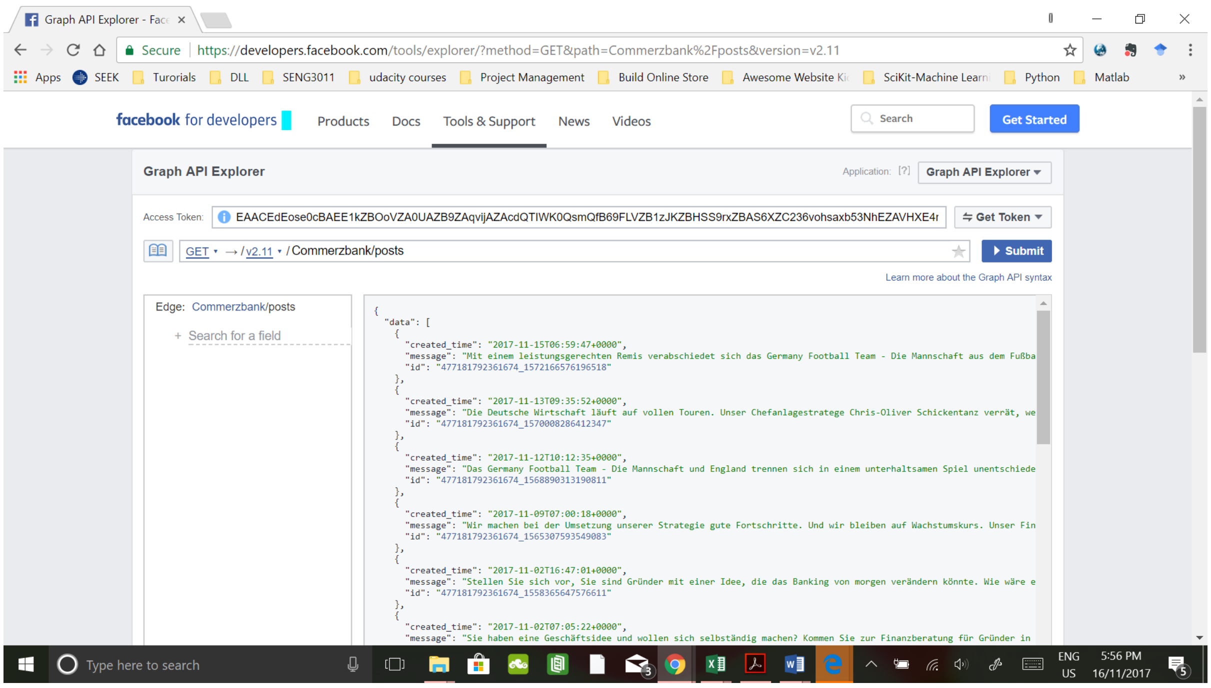
```
{
  "id": "477181792361674",
  "name": "Commerzbank",
  "fan_count": 160615,
  "website": "http://blog.commerzbank.de"
}
```

*Figure 0.3 CommerzBank fan page count*

Note that the output is in JSON format. Also note that the query builder reflects what has been specified in the query and vice-versa. Clicking on “Search for a field” in the query builder would list all available fields for this node (page in this example). After adding/remove fields and/or connections to the query using the query builder, click the Submit button to execute the query.

Now, let’s request the posts posted by the company. A Facebook Graph API query for Commerzbank requesting its posts would look like this:

```
GET /Commerzbank/posts
```


Graph API Explorer
Access Token: EAACEdEose0cBAEE1kZBOoVZA0UAZB9ZAqvijAZAcdQTIWK0QsmQfB69FLVZB1zJKZBHSS9rxZBAS6XZC236vohsaxb53NhEZAVHXE4r
Get Token
GET → /v2.11 /Commerzbank/posts
Submit
Edge: Commerzbank/posts

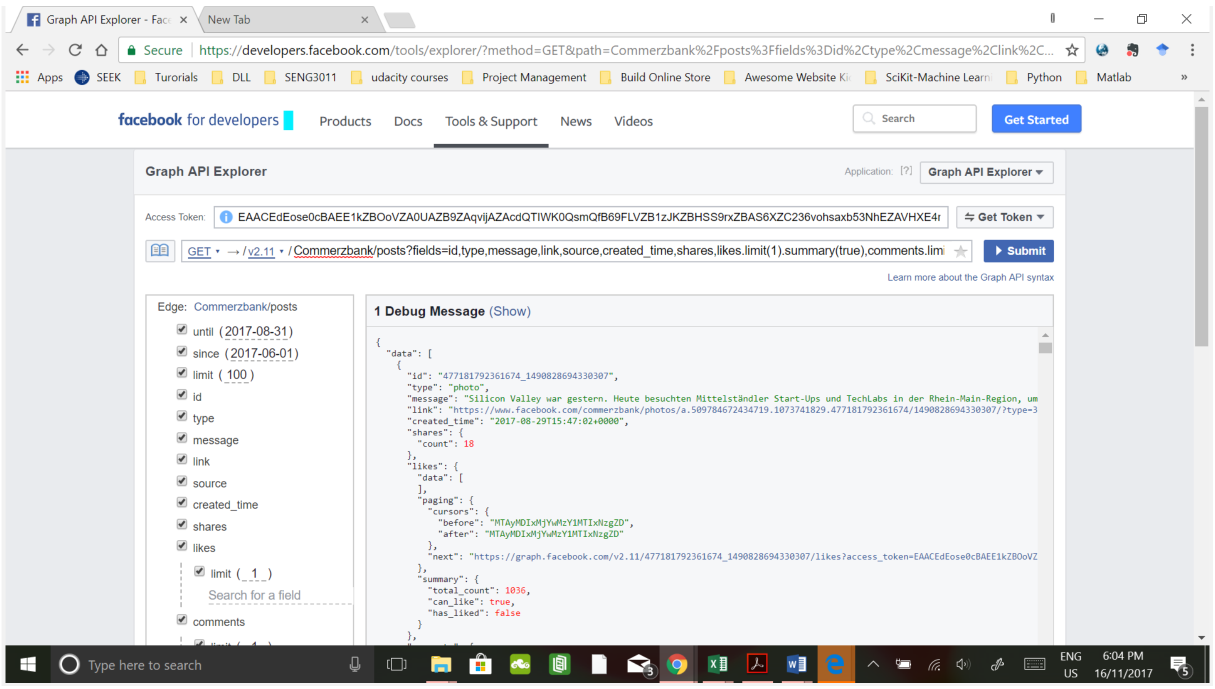
```
{
  "data": [
    {
      "created_time": "2017-11-15T06:59:47+0000",
      "message": "Mit einem leistungsgerechten Remis verabschiedet sich das Germany Football Team - Die Mannschaft aus dem Fußba
      "id": "477181792361674_1572166576196518"
    },
    {
      "created_time": "2017-11-13T09:35:52+0000",
      "message": "Die Deutsche Wirtschaft läuft auf vollen Touren. Unser Chefanlagestratege Chris-Oliver Schickentanz verrät, we
      "id": "477181792361674_1570008286412347"
    },
    {
      "created_time": "2017-11-12T10:12:35+0000",
      "message": "Das Germany Football Team - Die Mannschaft und England trennen sich in einem unterhaltsamen Spiel unentschiede
      "id": "477181792361674_1568890313190811"
    },
    {
      "created_time": "2017-11-09T07:00:18+0000",
      "message": "Wir machen bei der Umsetzung unserer Strategie gute Fortschritte. Und wir bleiben auf Wachstumskurs. Unser Fin
      "id": "477181792361674_1565307593549083"
    },
    {
      "created_time": "2017-11-02T16:47:01+0000",
      "message": "Stellen Sie sich vor, Sie sind Gründer mit einer Idee, die das Banking von morgen verändern könnte. Wie wäre e
      "id": "477181792361674_1558365647576611"
    },
    {
      "created_time": "2017-11-02T07:05:22+0000",
      "message": "Sie haben eine Geschäftsidee und wollen sich selbständig machen? Kommen Sie zur Finanzberatung für Gründer in
```

*Figure 0.4 CommerzBank posts*

The output of the above query lists the latest few posts and only provides, for each post, the creation time, the message and the id of the post. The JSON output contains a “data” array containing the posts.

Let’s request all posts posted by the company/brand between 1st June 2017 and 31st August 2017. We would like also to request for the following fields: id, type, message, link, source, and created_time. We would also like to get the count of likes and the count of comments (note that likes and comments are connections). Finally, we would like to limit the output to 100 posts (note that Facebook Graph API limits queries to up to 100 posts per call, so if you would like to download more data, you would have to split your query into multiple queries). The final Facebook Graph API query would look like this:

```
GET/Commerzbank/posts?fields=id,type,message,link,source,created_time,shares,likes.limit(1).summary(true),comments.limit(1).summary(true)&limit=100&since=2017-06-01&until=2017-08-31
```



Graph API Explorer

Access Token: EAACEdEose0cBAEE1kZBOoVZA0UAZB9ZAqvijAZAcdQTIWK0QsmQfB69FLVZB1zJKZBHSS9rxZBAS6XZC236vohsaxb53NhEZAVHXE4r

GET → /v2.11 /Commerzbank/posts?fields=id,type,message,link,source,created_time,shares,likes.limit(1).summary(true),comments.limi

Edge: Commerzbank/posts
- ✓ until (2017-08-31)
- ✓ since (2017-06-01)
- ✓ limit ( 100 )
- ✓ id
- ✓ type
- ✓ message
- ✓ link
- ✓ source
- ✓ created_time
- ✓ shares
- ✓ likes
  - ✓ limit ( 1 )
- ✓ comments

1 Debug Message (Show)

```
{
  "data": [
    {
      "id": "477181792361674_1490828694330307",
      "type": "photo",
      "message": "Silicon Valley war gestern. Heute besuchten Mittelständler Start-Ups und TechLabs in der Rhein-Main-Region, um
      "link": "https://www.facebook.com/commerzbank/photos/a.509784672434719.1073741829.477181792361674/1490828694330307/?type=3
      "created_time": "2017-08-29T15:47:02+0000",
      "shares": {
        "count": 18
      },
      "likes": {
        "data": [
        ],
        "paging": {
          "cursors": {
            "before": "MTAyMDIxMjYwMzY1MTIxNzgzZD",
            "after": "MTAyMDIxMjYwMzY1MTIxNzgzZD"
          },
          "next": "https://graph.facebook.com/v2.11/477181792361674_1490828694330307/likes?access_token=EAACEdEose0cBAEE1kZBOoVZ
        },
        "summary": {
          "total_count": 1036,
          "can_like": true,
          "has_liked": false
        }
      },
```

*Figure 0.5 CommerzBank filtering by a certain comment id*

The output of the above query is much bigger, given that we requested 100 posts. Scroll down the output to see all returned posts. Note that all fields requested are now included.

In order to get the comments generated by all those posts, multiple queries are required, as comments can be retrieved one post at a time. Let's take one post as an example (id= 477181792361674_1490828694330307) and run a query to retrieve all its comments (limited to 100 comments). We would like also to request for the following fields for each comment: id, message, created_time, and the count of likes and the count of replies (replies are comments on the comment). The Facebook Graph API query would look like this:

```
GET
/477181792361674_1490828694330307/comments?fields=id,message,created_time,likes.limit(1).summary(true),comments.limit(1).summary(true)&limit=100
```

## Save output as a JSON file

The output of Facebook Graph API Explorer is provided in JSON format. The same applies to programmatically calling the Graph API. From Facebook Graph API Explorer, saving the output consists of simply copy-paste the output content into a text file. Simply use notepad or any text editor to save the file. Another option is to run the query directly in a browser and save the output page. The URL for getting the posts would look like the following (replace {access token} with your access token):

https://graph.facebook.com/v2.11/Commerzbank/posts?fields=id%2Ctype%2Cmessage%2Clink%2Csource%2Ccreated_time%2Cshares%2Clikes.limit(1).summary(true)%2Ccomments.limit(1).summary(true)&limit=100&since=2017-06-01&until=2017-08-31&access_token={}

Note: substitute the curly brackets {} the access_token variable with the token value from panel B in Figure 1.

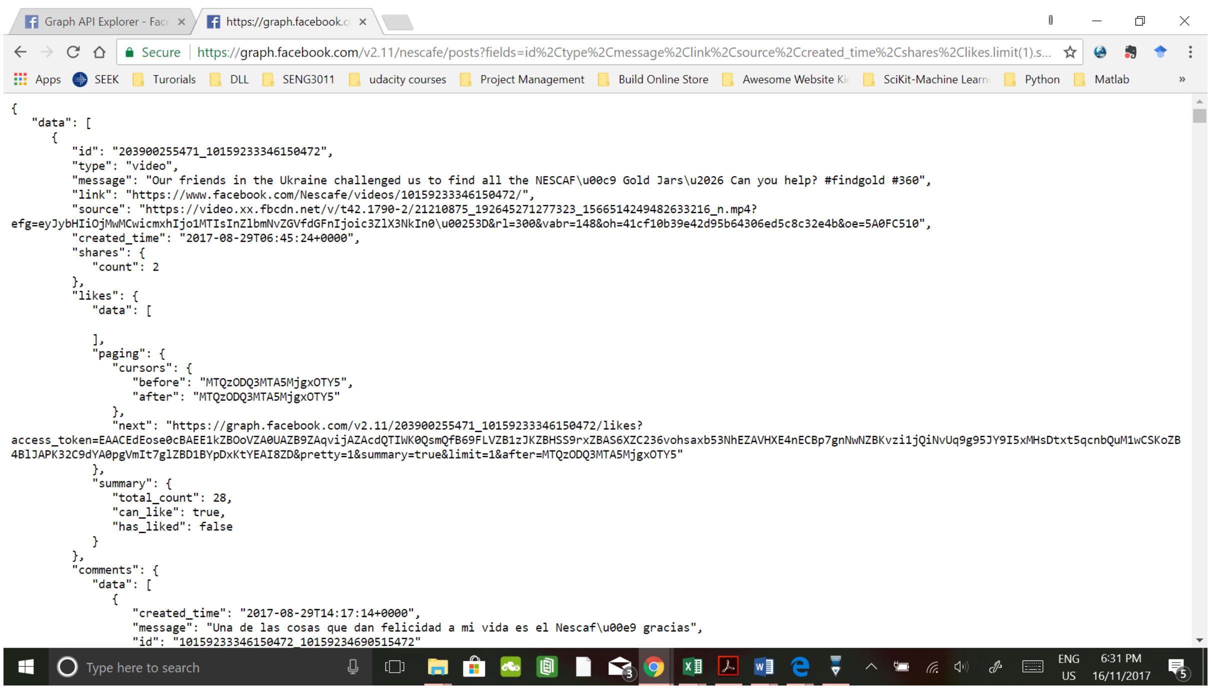
So the URL will be for example: (Note the text in red Is the token ID)

**https://graph.facebook.com/v2.11/Commerzbank/posts?fields=id%2Ctype%2Cmessage%2Clink%2Csource%2Ccreated_time%2Cshares%2Clikes.limit(1).summary(true)%2Ccomments.limit(1).summary(true)&limit=100&since=2017-06-01&until=2017-08-31&access_token=EAACEdEose0cBAGZC1aZC5e3dSUNnvFuFgGFIAS2GtKND8sQVmnd7R4mHX9OR5gMb5C4LT9vlUevq4SFcLfTvpcjYwuwQlY76uvS4s7NBuqwUum54ZCQofjM1aF4fDOOsAqyzl3fiPEsDYGCL8nJYm2Q995Ni6iUkOPXuxnvpkRxYv5P70x5UwEGFzsaWVkZD**

Note1 : if the URL didn't work, it could be the token has expired then go back to Panel B in Figure 2 and click on Get Token button to get new token, then copy the new token into the above URL substituting the old token id (shown in red font) with the new one.

Note2: This example gets the comments for **Commerzbank**, if you want to try another company then copy the Facebook ID from Table1 below for the company you are interested in.

The output of the above URL would like the following (JSON is displayed in the web browser and can be saved by hitting Ctl + s.



```
{
   "data": [
      {
         "id": "203900255471_10159233346150472",
         "type": "video",
         "message": "Our friends in the Ukraine challenged us to find all the NESCAF\u00c9 Gold Jars\u2026 Can you help? #findgold #360",
         "link": "https://www.facebook.com/Nescafe/videos/10159233346150472/",
         "source": "https://video.xx.fbcdn.net/v/t42.1790-2/21210875_192645271277323_1566514249482633216_n.mp4?
efg=eyJybHIiOjMwMCwicmxhIjo1MTIsInZlbmNvZGVfdGFnIjoic3ZlX3NkIn0\u00253D&rl=300&vabr=148&oh=41cf10b39e42d95b64306ed5c8c32e4b&oe=5A0FC510",
         "created_time": "2017-08-29T06:45:24+0000",
         "shares": {
            "count": 2
         },
         "likes": {
            "data": [

            ],
            "paging": {
               "cursors": {
                  "before": "MTQzODQ3MTA5MjgxOTY5",
                  "after": "MTQzODQ3MTA5MjgxOTY5"
               },
               "next": "https://graph.facebook.com/v2.11/203900255471_10159233346150472/likes?
access_token=EAACEdEose0cBAEE1kZBOoVZA0UAZB9ZAqvijAZAcdQTIWK0QsmQfB69FLVZB1zJKZBHSS9rxZBAS6XZC236vohsaxb53NhEZAVHXE4nECBp7gnNwNZBKvzi1jQiNvUq9g95JY9I5xMHsDtxt5qcnbQuM1wCSKoZB
4BlJAPK32C9dYA0pgVmIt7glZBD1BYpDxKtYEAI8ZD&pretty=1&summary=true&limit=1&after=MTQzODQ3MTA5MjgxOTY5"
            },
            "summary": {
               "total_count": 28,
               "can_like": true,
               "has_liked": false
            }
         },
         "comments": {
            "data": [
               {
                  "created_time": "2017-08-29T14:17:14+0000",
                  "message": "Una de las cosas que dan felicidad a mi vida es el Nescaf\u00e9 gracias",
                  "id": "10159233346150472_10159234690515472"
```

*Figure 0.6 Viewing JSON output on the web browser*