Formation-Facepager et extraction de données

Vissého Adjiwanou & Félix Alain

13/01/2021

Contents

Introduction

Suite à la démocratisation de l'internet, l'explosion des plates-formes de types réseau sociaux a amené de nouveaux endroits de partage d'information, d'implication sociale et de socialisation. Ainsi, il semble pertinent pour les sciences sociales de développer des outils permettant une analyse de l'information circulant sur ces plateformes. Considérant que l'information sur ces plateformes est des données, ce qui devient intéressant est le fait que celles-ci peuvent être extraites. Ce qui permet la collecte d'un grand nombre d'informations sur divers sujets précis. Ces données peuvent être à la fois qualitatives et quantitatives. D'un point de vue qualitatif, il est possible de penser aux commentaires et publications des utilisateurs des plateformes. Pour ce qui est des données quantitatives, on peut parler de la fréquence des publications, du nombre d'interaction avec les publications et même quels types de publication amènent le plus d'interaction (Vidéo, photo, les mots clefs présents dans le texte des publications). Toutefois, pour se faire il faut plonger un petit peu dans le domaine de l'informatique, de l'internet et des analystes de l'information. Pour bien connaitre ce domaine, une simple formation ne semble pas suffisante. Heureusement, plusieurs individus, qui ont fait de ce domaine leur spécialité, ont créé des outils pour faciliter notre travail. Permettant à des individus possédant uniquement des connaissances de base d'extraire des données.

Cette formation focusera sur l'outil *Facepager*, qui facilite l'extraction de données de sites Internet. De plus, cette formation aura pour objectif d'utiliser cet outil avec Facebook. Considérant que Facebook est l'une des plateformes de réseaux sociaux les plus utilisés, apprendre à extraire de l'information de cette plateforme semble pertinent pour les sciences sociales. De plus, cette formation présentera certains concepts de base de l'extraction de données sur internet avec un accent *Facepager*, qui donnera les bases nécessaires pour permettre aux individus d'expérimenter par eux même avec d'autres plateformes, puisque *Facepager* permet d'être utilisé avec la majorité des grandes plateformes.

Dans ce document vous apprendrez:

1. Ce qu'est une URL, comment il est composé et son importance

2. Ce qu'est un API (_Application Programing Interface_)

3. Les différentes manières de collecter l'information (ajout de Vissého)

3. Comment _Facepager_ utilise ses informations pour extraire les données

4. Comment utiliser _Facepager_ pour extraire d'une page Facebook:

   i) Les publications
   ii) Les réactions et nombres de commentaires des publications
   iii) Les commentaires sur les publications

5. Comment exporter notre base de données :

   i) Ce qu'est un format CSV
   ii) Les différents séparateurs possibles pour notre base de données

6. Comment utiliser Rstudio pour rendre cette base de données utilisable

Notions de base

Avant de commencer à extraire des données provenant de l'internet, il semble important de comprendre comment un site internet est structuré. Ceci permettra de savoir comment il est possible de "parler" avec le site pour lui demander l'information désirée.

URL

L'URL ou *Uniform Resource Locator* correspond à l'adresse d'une ressource sur un site internet. Ainsi, l'URL est l'information principale nécessaire pour donner les directives permettant d'aller chercher l'information désirée. L'URL représente les coordonnées géographiques d'un endroit sur internet. Celui-ci est principalement trouvé dans la barre où l'on inscrit les sites web. Pour voguer sur le web, on utilise des URL. Voici comment une URL est construite:

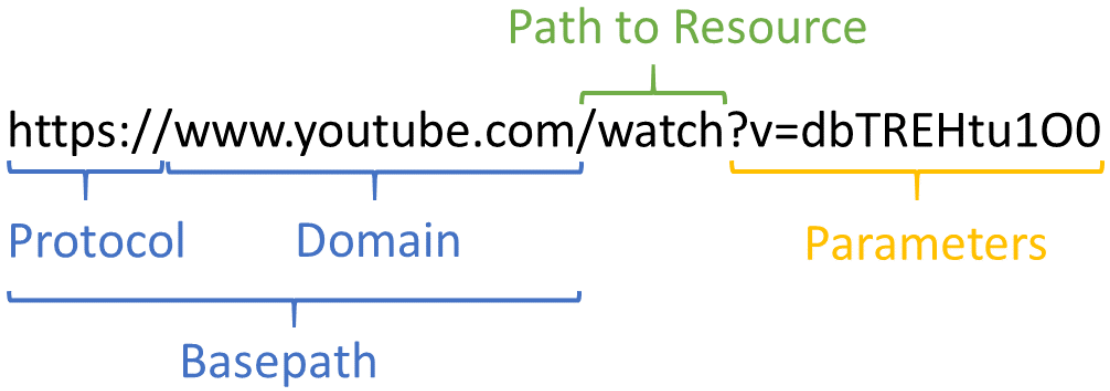


Figure 1: Comment une URL est construite

- **Protocol (Protocole)**: Le protocole indique comment le transfert d'information doit être exécuté sur internet. C'est un peu le format que prend l'information de la page déterminée. En informatique, il existe plusieurs formats de fichier, par exemple: PDF, Word, PNG (Photo), etc. Le protocole indique en quel format de fichier la page Web correspond. Le format HTTP ou *Hypertext Transfert Protocol* est le format standard sur internet. De sorte que vous ne risquerez pas de rencontrer un autre format.
- **Domain (Domaine)**: Le domaine correspond au serveur web qui héberge le site. Un site internet doit toujours être accueilli sur un serveur. Un serveur est un ordinateur physique qui héberge les données nécessaires au fonctionnement du site internet. Le domaine corespond au nom de celui-ci. Le domaine est divisé en plusieurs parties. Dans cet exemple, "www" correspond à ce qu'on appelle le *subdomain*, "youtube.com" correspond au *domain* et ".com" correspond au *top-level domain*.
- **Basepath (Chemin de base)**: Le *BasePath* est la combinaison du protocole et du domaine. Ceci est considéré comme étant la base d'un site internet. La porte d'entrée vers le serveur désiré.
- **Path to Resource (Chemin vers la ressource)**: Le *Path to ressource* correspond à une ressource, un fichier ou un dossier sur le serveur du site internet. Le *Path to Resource* est structuré à l'aide de séparateur "/". Dans notre exemple un seul séparateur est utilisé, mais il est possible d'avoir plusieurs séparateurs. Chaque séparateur représente l'ouverture d'un nouveau dossier dans le serveur du site. Par exemple: "https://www.github.com/strohne/Facepager/wiki/URLs" Il est possible de noter plusieurs *Path* vers la ressource. Cela fonctionne exactement comme les dossiers sur un ordinateur. Il est possible de voir le *Path to Ressource* comme le chemin à prendre pour trouver notre information.
- **Parameters (Paramètre)**: Les informations additionnelles sont principalement indiquées par les *parameters* ou paramètres. Dans notre exemple, ceci correspond au type de fichier et à l'identifiant (ID) du fichier. Ils commencent toujours par un "?" et sont

structuré comme-ci: “key=value” (“clef=valeur”). Lorsqu'il y a plusieurs paramètres, chaque paire de *key* et *value* sont séparées par le caractère “&”. Dans notre exemple, l'information rechercher correspondait à un vidéo YouTube, la valeur “dbTREHtu1O0”, qui correspond à l'ID du vidéo (*value*), est associé à la *key* “V” qui correspond au type vidéo.

Une URL est comme une carte permettant de naviguer et voyager. Le protocole correspondrait au dictionnaire de la langue utilisée dans le pays, permettant de comprendre les indications et les individus du pays. Le *Basepath* ou plus généralement appelé domaine serait la ville que l'on désire visiter et utilisé uniquement celui-ci, nous amène a la porte de la ville. Ensuite pour se diriger dans la ville, le *Path to Ressource* nous donne les indications sur les quartiers, les boulevards et les rues. Une fois sur la bonne rue, les paramètres nous indiquent l'adresse du bâtiment (ID) et le style du bâtiment (Commercial, résidentiel, historique, gouvernemental. Ce qui correspondrait à la *key*). L'URL va donc du plus large au plus précis. Ainsi, si l'on désire uniquement visiter une “rue” (page Facebook), il est possible de n'avoir aucun paramètre. Il y aura uniquement le domaine et le *Path to Ressource*. Il est important de noter que chaque site internet est différent dans son architecture. Ainsi le nombre de “rue” à prendre avant d'arriver de la porte d'entrée du site (domaine) à notre destination peut varier.

API (Interface de programmation d'application)

Comme nous l'avons vue l'URL permet d'indiquer qu'elles informations extraire, toutefois pour extraire l'information plusieurs sites mettent à la disposition des utilisateurs ce qu'on appelle un *Application Programming interfaces* (API) ou interface de programmation d'application en français. Un API, pour simplifier, est un ensemble d'information normalisé qui agit comme une façade permettant d'offrir des services à d'autre logiciel. Par normalisé on entend que les informations sont condensées dans différents formats permettant la lecture de celles-ci. Par exemple *Facepager* permet de lire l'information des API produisant des formats JSON ou WML. Un format correspond à une manière d'écrire un document. C'est comme un langage de communication (Français, Anglais, etc.). Pour pouvoir lire le format, il faut comprendre la langue. Principalement se connecter sur l'API d'un logiciel permet de communiquer de façons efficaces avec celui-ci et permet d'utiliser les services du logiciel. Dans notre cas, ce service est l'extraction de données.

Les demandes possibles sur un API sont spécifiques à chaque API et peuvent évoluer à travers le temps. Ainsi, il est important de lire la documentation de l'API du site pour déterminer s'il est possible de faire les sortes de demandes que nous désirons. Par exemple, il y a quelque années il était possible de faire des demandes sur l'API de Facebook sur des pages personnelles des individues. De nos jours Facebook a restreint sont API et il n'est plus possible de faire se genre de demande. L'API agit donc comme intermédiaire entre les demandeurs d'informations et l'information, mais il agit aussi comme agent régulateur des demandes. Autant dans le types de demandes possibles, que dans la manière dont ses demandes sont effectué.

Access Token (Jeton d'accès)

Un *Access Token*, ou *Token* pour faire plus court, est une clef permettant de se connecter à un API. Cette clef est différente pour chaque personne, pour chaque API et n'est pas fixe. L'objectif de cette clef est de permettre à des utilisateurs de se connecter à l'API sans donner les informations de compte à une tierce plateforme. Par exemple, si l'on se connecte à notre jeu mobile sur notre cellulaire avec notre compte Facebook, il serait dommage que cette application enregistre notre mot de passe. Alors Facebook produit un *Acess Token* qui permet d'identifier l'individu sans donner de l'information confidentielle. Les *Tokens* sont une autre forme d'identificateur de compte utilisé par les API.

Ainsi, il est facile de comprendre que pour avoir un *Token*, il faut avoir un compte sur la plateforme. De plus, Certaine plateforme demande un compte avec un accès développeur pour accéder directement et voir son *Token*. Pour avoir cet accès, il faut passer au travers d'une procédure de validation de son compte. Cette procédure est généralement très accessible, mais varie de plateforme à plateforme. Puisque nous utiliserons *Facepager* comme outils d'extraction de données, cet aspect ne nous affectera point, puisqu'il n'Est pas requis de posséder un compte développeur pour utiliser *Facepager*.

Rate limit (Limite de vitesse)

Une autre contrainte des API est ce qu'on appelle le *Rate limit*. Ceci correspond à la limite de demande qu'un individu peut faire à l'API sur une période déterminée. Chaque API a leur propre règle pour le *Rate limit*. De plus, cette limite n'est pas associée à un *Token*, mais bien au compte de l'individu. Encore une fois cette limite dépend des plateformes. Toutefois, il existe des façons d'augmenter notre limite.

Si l'on prend l'exemple de Facebook, l'API met une restriction non divulguée sur le nombre de demandes qu'un individu peut faire sur sa plateforme. Toutefois, sur Facebook il existe se qu'on appelle un *application access token* (Jeton d'accès d'application), ce qui est le même principe qu'un *acess Token* mais pour les applications. L'avantage est que le nombre de demandes est calculé ainsi:

Nombre de demandes par heure = 200 * Nombres d'utilisateurs

De plus, cette limite n'est pas individuelle, mais collective. Un individu peut dépasser 200 demandes par heure, si le reste des individus fonds moins de 200 demandes par heure. Ainsi, les utilisateurs plus actifs peuvent profiter du fait qu'il existe des utilisateurs moins actifs. Voici un des avantages de *Facepager*. Celui-ci permet de dépasser le *Rate limit* individuel.

L'exemple Facebook

Pour mieux comprendre l'interaction entre ces différents concepts, regardons comment faire une demande d'information sur Facebook fonctionne:

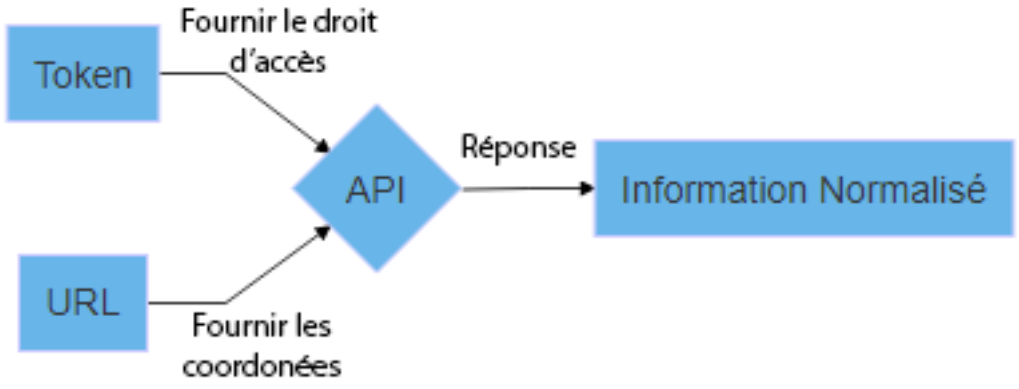


Figure 2: Comment un API fonctionne

La première façon de comprendre une demande est la suivante (*figure 2*). L'API a besoin de deux composantes pour renvoyer de l'information. Une autorisation et la position où se trouve cette information. En termes plus techniques, un *Acess Token* et une URL. Par la suite l'API répond en envoyant l'information demandée sous format normalisé. Pour Facebook ceci est en format JSON.

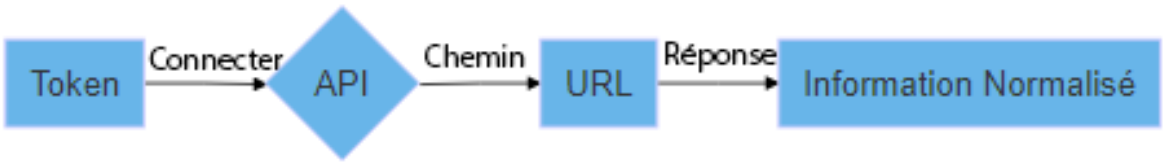


Figure 3: La procédure

Quoiqu'en théorie ceci est une bonne façon de comprendre comment les demandes sur les API fonctionnent, dans la pratique il est possible de voir une procédure assez linéaire (*figure 3*):

1. Se connecter à l'API avec un *Token*
2. Indiquer le chemin vers les données à l'aide d'une URL
3. L'API renvoie l'information normalisée

Pour l'extraction de données sur Facebook, il faut se connecter,à l'aide d'un *Token*, à l'API de Facebook. Une fois connecté, il est possible de faire des demandes d'informations en

utilisant l'URL comme chemin vers celle-ci. Les limites sur le type d'information et le nombre d'informations (*Rate limit*) qu'il est possible d'extraire sont dictés par les règles spécifiques de l'API Facebook, qui sont disponibles sur internet. Une fois le *Token* et l'URL fournis à l'API, celui-ci revoit l'information désirée sous un format normalisé. Heureusement, comme il sera possible de voir plus tard, *Facepager* facilite cette procédure pour nous.

Facepager les bases

Facepager est une application *Open source* (code source ouvert), ce qui signifie qu'elle est libre de distribution et que l'accès au code source est disponible. L'objectif de *Facepager* est de faciliter l'extraction de données sur le web. Dépendamment du site, l'application permet trois options:

1. Interagir avec les API
2. Web scraping (prélèvement web)
3. Télécharger des fichiers média

Ce document est principalement axé sur la première option soit l'interaction de *Facepager* avec les API. L'application offre des options préprogrammées pour les plateformes YouTube, Twitter, Twitter Streaming, Amazon et Facebook. Il y a aussi une fonction générique, qui permet de personnaliser des recherches sur d'autres sites. Il est important de souligner que *Facepager* n'est pas une application: un clic = résultat. De sorte, que comprendre les mécanismes derrière son fonctionnement permet de jouer avec les réglages pour arriver au résultat désiré.

Il est possible d'extraire des données sans l'utilisation d'une application comme *Facepager*, toutefois il n'existe plus de *package* (paquet) en R permettant de communiquer directement avec la plateforme Facebook. En fait, R n'est pas considéré comme un langage orienté internet. C'est pourquoi plusieurs *data analyst* (analyste de données) de formation jumelle le R avec un autre langage informatique plus adapté à l'internet comme le Python. L'avantage avec *Facepager* est que ce logiciel permet d'extraire sans avoir une connaissance approfondie des langages informatiques. Ce qui évite d'apprendre un langage comme le Python. Toutefois, il est recommandé d'avoir une connaissance d'au moins un langage informatique comme le R pour le traitement de la base de données par la suite.

En résumé, *Facepager* facilite l'extraction de données, permet de créer des bases de données rudimentaires et de les sauvegarder en CSV sur son ordinateur. Il faut comprendre que ces bases de données ne sont pas prêtes à l'analyse à la sortit de *Facepager*. Il faut donc les "nettoyer" à l'aide de Rstudio ou un autre programme.

Un autre avantage de cette application est le fait qu'il permet de dépasser le *Rate limit* individuel. Comme vue plus tôt, un des avantages des *applications acces Token*, est le fait que leur limite de demande est déterminée par le nombre d'utilisateurs (voir la section *Rate limit*). De sorte que même s'il était possible de connecter R avec Facebook, notre limite serait déterminée par notre *Token* individuel. Il deviendrait difficile et frustrant d'extraire une quantité importante de données. Passer par une application resterait le meilleur choix.

Les limites de *Facepager* sont déterminées principalement par les limites de l'API dont nous tentons de communiquer avec. De plus, lorsque l'application fait une demande, cette demande est cristallisée dans le temps. C'est comme prendre une photo du moment où la demande a été prise. Ce qui veut dire que si nous effectuons une demande sur le nombre de commentaires sur une publication X et que le lendemain de cette demande 50 nouvelles personnes ont commenté la publication, ses 50 nouvelles personnes ne seront pas comptées dans la base de données. *Facepager* ne permet pas de collecter continuellement et de façons automatiques les données. Cela ne veut pas dire qu'il est impossible de collecter un nombre faramineux d'informations. Par exemple, il m'a été possible de recueillir une base de données de plus de 2 millions de commentaires Facebook sur une cinquantaine de pages différentes.

La procédure générale pour utiliser FacePager est relativement simple:

1. Télécharger FacePager sur GitHub: https://github.com/strohne/Facepager
2. Se connecter sur son compte Facebook
3. Faire les demandes en suivant l'ordre des priorités des demandes (ex: page —> publications —> commentaire)

Le processus de travail en utilisant *Facepager* est assez simple. *Facepager* est utilisé pour faire des demandes sur l'API, celui-ci renvoie des données, qui sont par la suite convertit en fichier CSV et finalement se fichier est traité en utilisant un outil comme Rstudio.

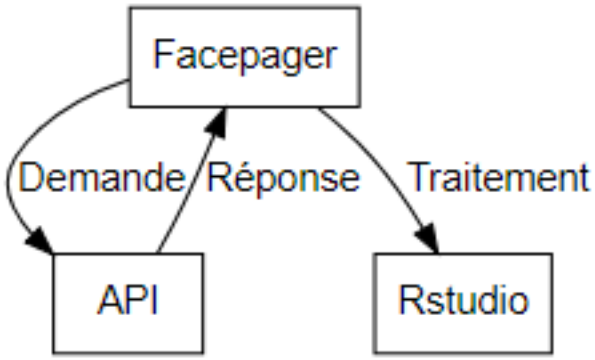


Figure 4: Processus de travail

Node (Nodule)

Facepager est un système qui collecte les informations et les conserves dans un système de *nodes*. Les *nodes* correspondent aux objets de notre collecte d'information. Un commentaire, une publication ou une page Facebook sont tous des exemples d'information entreposés dans des *nodes*. Chaque *node* correspond normalement à un objet de la demande et chaque rangée dans notre interface *Facepager* correspond à une *node*. Puisque le terme *node* réfère à un

ensemble d'information sur un objet de recherche recueilli par l'application et qu'elles sont organisées en rangé le terme rangé, objet et *node* sont utilisé de façons interchangeables dans se document. *Facepager* utilise quatre types de *nodes* différent et le type peut être retrouvé dans la colonne *Object type* (type d'objet) de notre base de données:

Object Type	Explication
`seed` (graine)	Représente des données manuellement ajoutées par l'utilisateur. Principalement le point de départ de la recherche. Par exemple, l'identifiant (ID) d'une page Facebook.
`data` (information)	L'information renvoyée par l'API divisé en nodes pour chaque information. Par exemple, l'information d'un commentaire sur une publication Facebook sera storé dans une *node* de type *data*.
`offcut` (reste)	Ceci est le résidu d'information de notre requête. Une fois l'information coupée et organisée par *Facepager*, le surplus d'information non utilisé est entreposé dans une *node* de type *offcut*. Par exemple, une requête pour obtenir les commentaires sur une publication Facebook retournera une *node* de *data* par commentaire et entreposera l'information comme le nombre de commentaires recueillit dans une *node offcut*.
`unpacked` (déballé)	Il est possible de découper par la suite nos *nodes* encore plus avec l'option `Extract data`. Les nodes créées par ce processus ont comme type *unpacked*.

Facepager organise donc l'information en rangé de *nodes*, mais utilise aussi un classement en *level* (niveau). Ainsi, un objet peut contenir un autre objet et celui-ci peut contenir d'autres objets, ainsi de suite. À chaque fois qu'un objet contient un autre objet, la nouvelle *node* est enregistrée dans le niveau suivant. Les premières nodes sont sur le niveau un, les nodes de ses nodes sont sur le niveau deux et ainsi de suite. La position d'une *nodes* dans se système de classification correspond au *node level* de celle-ci. Une *node* sur le premier niveau est appelée *parent node* (nodule parent) et les *nodes* sur les autres niveaux sont appelés *childs nodes* (Nodule enfant). Il est à noter qu'une *child node* peut contenir d'autre *child node* qui lui sont subordiné ou dépendante. Voici un exemple de comment *Facepager* organise les *nodes* dans son interface:

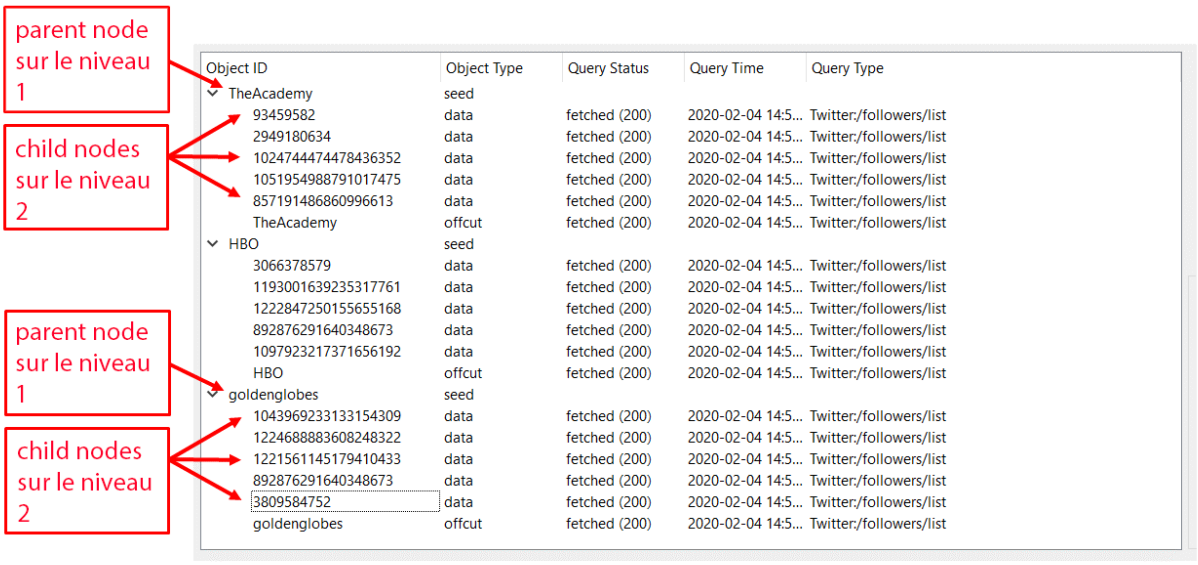


Figure 5: Les niveaux de *nodes*

Placeholders (espace réservé)

Pour collecter l'information, *Facepager* utilise des *Placeholders* dans la formation des URL. Un *Placeholder*, littéralement espace réservé en français, est une variable qui agit comme une donnée pouvant être remplacée par n'importe quelle autre information. Ce système sert à faciliter l'utilisation des URL et permet d'utiliser une ou plusieurs nodes pour faire des demandes. Ainsi, *Facepager* peut passer chaque *node* l'une après l'autre pour faire la même demande sur toutes les *nodes*.

Par exemple, l'URL d'une page Facebook est composée ainsi : https://www.facebook.com/Object-ID. En connaissant l'architecture de l'URL, il est possible de remplacer l'*object ID* par un *placeholder*: https://www.facebook.com/. Ensuite, en utilisant des *nodes* contenant différents ID de pages, *Facepager* remplacera le *placeholder* par la valeur ID de chaque node. Évitant de réécrire à chaque fois une URL différente et de faire des demandes automatisées sur plusieurs node en même temps.

Les *Placeholder* sont variés dans *Facepager*, mais ils sont tous encadré par les symboles "< >". Il existe trois sortes de *placeholders*:

1. Le *placeholder* est toujours remplacé par l'Object ID de la *node* active. L'*object ID* correspond à l'information dans la première colonne de l'interface de visualisation des *nodes* Facepager.
2. Un placeholder peut contenir différente *key* (clef) qui sont associés à diverse information contenue dans la *node* (voir la section *Key*).
3. *Facepager* possède aussi des placeholders définies dans les paramètres d'une demande. Ceux-ci, contrairement au précédent, ne sont pas des informations prises sur les *nodes*,

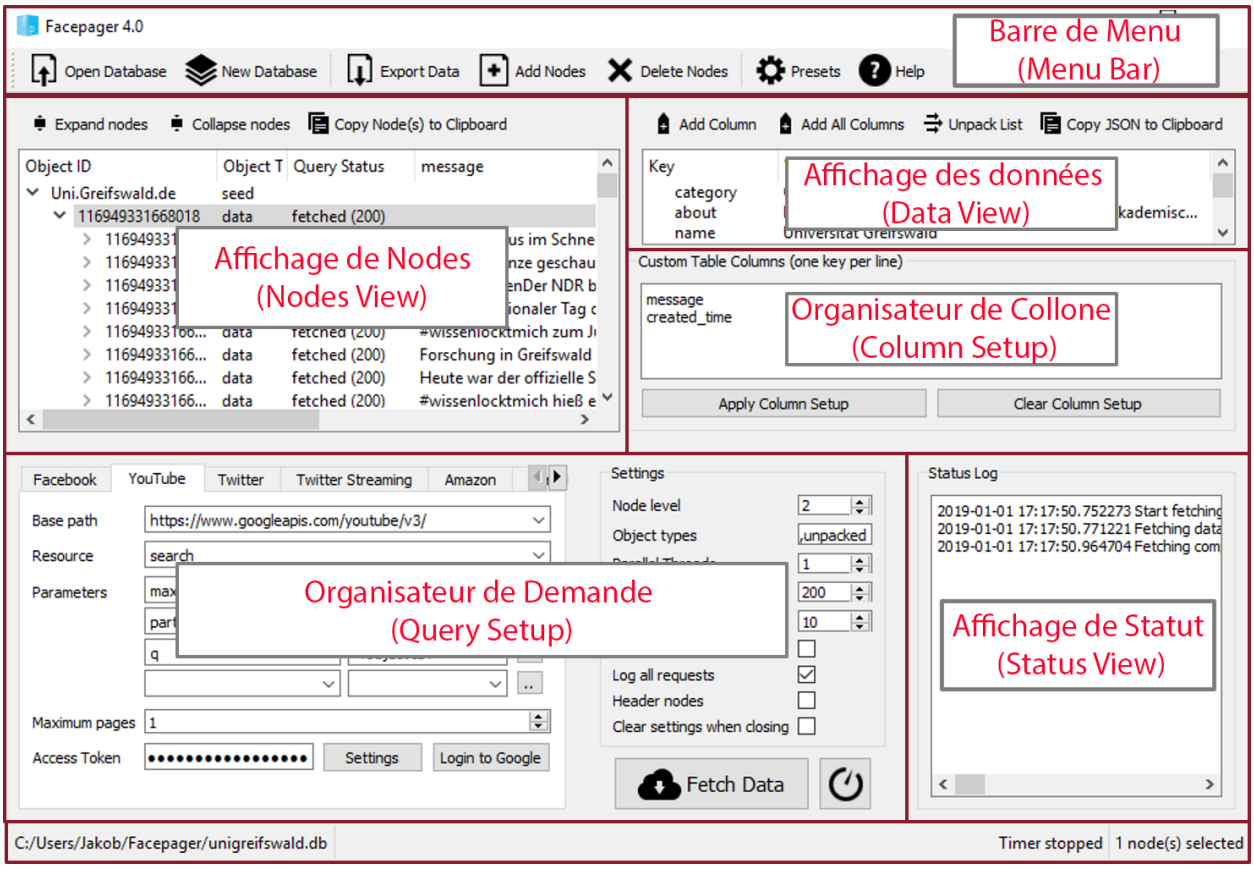
mais de nouvelles informations insérées pour spécifier une recherche. Par exemple, les *placeholders* "*since*" et "*until*" servent à déterminer une période temporelle. Chercher les publications de novembre à décembre par exemple. Cette information sert à spécifier la recherche et est donc indépendante des nodes.

Key (clef)

Un autre concept important de *Facepager* est la notion des *Keys* ou clef en français. Une *key* est une partit de l'information comprise dans une *node*. Par exemple, le message d'un commentaire serait une *key*, le nombre de "*like*" sur le commentaire serait une autre *key* et l'*Object ID* en serait une autre. Collectivement ces *keys*, ou informations, forment la *node*. Les *keys* peuvent être utilisées pour remplir un *placeholder* et sont principalement les résultats de notre recherche. Dans *Facepager* les *keys* correspondent aux colonnes de notre base de données. Il faut comprendre que ce n'est pas toutes les *keys* qui sont affichées par défaut dans l'afficheur de *nodes*. De plus, lors de l'exportation en CSV, *Facepager* exporte uniquement l'information présente dans l'afficheur de *nodes*. Il faut donc sélectionner les *keys* que nous désirons avant d'exporter notre base de données. Il sera possible de revenir plus en détail sur l'importance des *keys* lorsque nous expliquerons l'exportation de données.

L'interface

Pour mieux comprendre *Facepager*, il faut comprendre comment l'interface fonctionne. Il sera possible de comprendre par la suite certains concepts de base propre à *Facepager*.


Facepager 4.0
Open Database
New Database
Export Data
Add Nodes
Delete Nodes
Presets
Help
Barre de Menu (Menu Bar)
Expand nodes
Collapse nodes
Copy Node(s) to Clipboard
Object ID
Object T
Query Status
message
Uni.Greifswald.de
seed
116949331668018
data
fetched (200)
Affichage de Nodes (Nodes View)
Add Column
Add All Columns
Unpack List
Copy JSON to Clipboard
Key
category
about
name
Affichage des données (Data View)
Custom Table Columns (one key per line)
message
created_time
Organisateur de Collone (Column Setup)
Apply Column Setup
Clear Column Setup
Facebook
YouTube
Twitter
Twitter Streaming
Amazon
Base path
https://www.googleapis.com/youtube/v3/
Resource
search
Parameters
Organisateur de Demande (Query Setup)
Maximum pages
Access Token
Settings
Login to Google
Settings
Node level
Object types
,unpacked
Log all requests
Header nodes
Clear settings when closing
Fetch Data
Status Log
2019-01-01 17:17:50.752273 Start fetching
2019-01-01 17:17:50.771221 Fetching data
2019-01-01 17:17:50.964704 Fetching com
Affichage de Statut (Status View)
C:/Users/Jakob/Facepager/unigreifswald.db
Timer stopped
1 node(s) selected

Menu bar (Barre de Menu)

La *menu Bar* présente certaine des commandes les plus importantes dans *Facepager*. Les commandes `New Database` et `Open Database` permetent de commencer une base de données ou d'ouvrir une base de données existante.

La fonction `Add Nodes`, permet de rajouter des *nodes* à notre base de données. Ces *nodes* seront les *parent nodes*, qui seront le point de départ de nos requêtes. Pour effacer des *nodes*, il suffit de sélectionner les *nodes* désirés et d'utiliser l'option `Delete Nodes` dans la *menu Bar*. L'option `Presets` offre différentes demandes préprogrammées, qui facilite l'expérience d'extraction de données et devrait être le premier endroit où regarder pour faire une demande.

L'option `Export Data` permet d'exporter la base de donnée finale en fichier CSV. Les fichiers CSV sont plus pratiques qu'un format comme le JSON, puisqu'ils peuvent être lus dans la majorité des logiciels de statistique.

Nodes view (Affichage de Nodes)

La *nodes view* permet de visualiser les objects de notre collecte de donnée. Comme vue dans la section *nodes*, les *nodes* sont un ensemble de données organisé par *Facepager*. Par exemple, un commentaire avec sont ID, son message et le nombre de "*Like*" sur le commentaire. Chaque rangée représente une *node* différente et chaque colonne représente une *key* différente (voir section Key). Cette fenêtre correspond aussi à l'information qui sera exportée. Cette fenêtre sert donc à visualiser, gérer et utiliser les *nodes* comme *placeholder* pour faire des demandes.

Data view (Affichage des données)

Cette fenêtre affiche les données brutes d'une rangée (*node*) sélectionnée. Ces informations sont formatées dans le format de l'API. Principalement en JSON. Ainsi, cette fenêtre est utile pour comprendre quelle sorte d'information est comprise dans notre *node*. Ces données correspondent aux *keys*. De plus, si une *node* présente une erreur cette fenêtre donnera plus d'information sur le problème. Par la suite, regarder la documentation de l'API permettra de confirmer quel a été le problème.

Collum setup (Organisateur de Colonne)

La fenêtre *collum setup* permet de sélectionner quelles informations sont présente dans l'affichage de *nodes*. De plus, uniquement les colonnes sélectionnées dans le *collum setup* seront exportées. Chaque ligne représente une colonne de l'affichage de *nodes* et donc correspond à une *key* spécifique à une ou plusieurs *nodes* présentes dans la fenêtre *data view*. Ainsi, si une *key* dans l'affichage de donnée est nommée "*message*", il suffit d'inscrire le nom de la *key* dans le *collum setup* et d'appuyer sur l'option `Apply collum setup` de cette fenêtre. Contrairement, il est possible d'effacer une *key* du *collum setup*, d'appuyer sur `Apply collum setup` et celle-ci sera enlevée de l'affichage de *nodes*.

Au lieu d'ajouter manuellement nos colonnes, deux options existent. La première consiste à sélectionner une *key* dans le *data view* et d'appuyer sur l'option `Add Collumn`. Ceci ajoutera cette donnée dans le *collum setup*. La deuxième option consiste à utiliser l'option `Add All Collumns`, qui ajoutera toutes les *keys* présente dans les *nodes* sélectionnées. Ainsi, il suffit de sélectionner les *nodes* que nous désirons dans l'affichage de *nodes* et appuyer sur `Add All Collumns` pour avoir l'entièreté de nos *keys* dans l'organisateur de colonne.

Query Setup (Organisateur de demandes)

La fenêtre *query setup* est le point de départ de l'extraction de données sur un API. Cette région de l'interface est l'endroit où il sera possible de formater les demandes sur les divers API. Dans le haut de la section, il y a les différents modules préétablis. Il faut donc choisir le module Facebook pour extraire des données sur Facebook et le module Twitter pour communiquer avec l'API de Twitter et ainsi de suite.

Les options à droite, aussi appelé la section *settings* (options), de cette région servent à déterminer quel *nodes* sera affecté par la demande et la vitesse de la demande. La région centrale sert à formater notre demande à l'aide d'options préprogrammées qui agiront comme *placeholder*. Ces options servent donc à construire notre "URL" modulaire, qui donnera le chemin nécessaire pour extraire les données. Finalement la section du bas est réservée à la connexion avec l'API. Cette section sert à créer son *Access Token* permettant de se connecter à l'API.

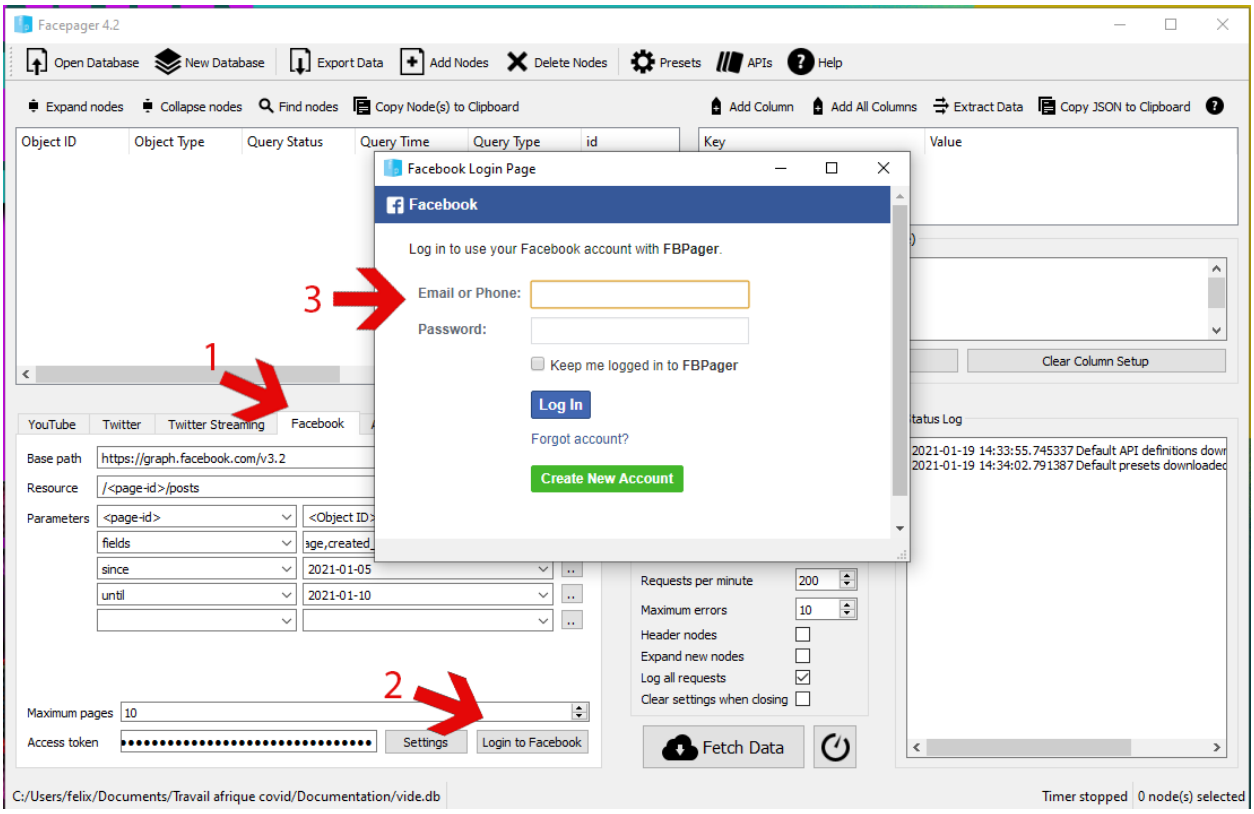
Status View (Affichage de Statut)

Cette section est principalement pertinente pour le débogage. Les messages apparaissant dans cette fenêtre informe sur l'état des requêtes et renvoient les codes d'erreurs si un problème est arrivé. Il y a donc la fenêtre *data view* et la fenêtre *status view*, qui permetent de comprendre les erreurs dans notre requête. Ensuite utiliser les codes d'erreurs et la documentation de l'API permet de comprendre le problème.

Procédure *Facepager* pour Facebook

Se connecter à Facebook

La première étape de notre collecte de données avec *Facepager* consiste à se connecter à l'API de Facebook. Cette procédure est relativement simple avec *Facepager*.



1. Sélectionner le module Facebook se trouvant en haut de la zone *query setup*.
2. Utiliser l'option `Login to Facebook` se trouvant en bas de la zone *query setup*.
3. Entrer les informations relatives à la connexion à son compte Facebook (Courriel et mot de passe).

Une fois connecté à notre compte Facebook, *Facepager* obtiendra automatiquement un *access Token*, qui nous permettra de faire des demandes sur l'API.

L'option `Settings`, à côté du bouton `Login to Facebook`, permets de demander des accès personnalisés. Pour Facebook ceci est principalement relié au page privé dont vous êtes un administrateur. Normalement l'API de Facebook ne permet pas des demandes sur les pages privées, mais si vous êtes l'administrateur de la page il est possible de faire des demandes sur celle-ci. L'option `Settings` permets cette procédure en insérant le *Page ID* dans les options.

L'une des erreurs qui peut souvent arriver lors de la collecte d'information est du au fait que l'utilisateur a été déconnecté de l'API. Il suffit de refaire la procédure de connexion. De plus, un autre problème qui peut survenir lors d'extraction d'un grand nombre d'informations, vient du fait que Facebook considère cette activité comme étant suspecte. Pour protéger votre compte, Facebook bloque celui-ci et demande une réinitialisation de votre mot de passe. Ceci est frustrant, mais il ne faut pas oublier que Facebook cherche à protéger nos informations.

Ajouter nos Parent Nodes/Seed (Nodule Parente/Graine)

Une fois connectée à l'API de Facebook, il est possible d'ouvrir une nouvelle base de donnée avec l'option `New Database` ou d'ouvrir une base de données existante avec l'option `Open Database` présentent dans la section *menu bar*. Pour commencer une recherche sur Facebook ou tout autre site internet, il faut donner à *Facepager* un point de départ à la recherche. Ses points de départ sont appelés les *parents nodes* ou *seeds* et sont toujours situé sur le premier *node level* (voir section Nodes). Pour ajouter de nouvelles *parents nodes*, il suffit d'appuyer sur l'option `Add Nodes` de la *menu bar*. Ceci fera apparaitre une fenêtre permettant d'écrire nos points de départ. Chaque ligne de cette fenêtre de texte représente une *parent node* et, tout en respectant le format une ligne égale une node, il est possible dans ajouté plusieurs en même temps.

Pour Facebook, la majorité des points de départ correspondront à l'identifiant(ID) d'une page. Aussi appelé *page ID*. Le *page ID*, comme tout autre ID (publication, commentaire, etc.) sur Facebook est composé d'une série de chiffres. Toutefois, pour faciliter la navigation sur son site internet, Facebook associe souvent à cette suite de chiffre un mot ckef. Celui-ci peut être utilisé comme ID pour nos *parents nodes*. Voilà pourquoi nous ferons la distinction entre un *Hard ID* ou *numerical ID*, correspondant à la suite de chiffre, et un *Soft ID*, qui sont les mots associés à cet ID pouvant le remplacer.

Par exemple, le *hard ID* de la page Facebook de Barack Obama est "6815841748" et le *Soft ID* est "barackobama". Ceci est pourquoi il y a deux techniques pour trouver l'ID d'une page. La première consiste à regarder l'URL de la page désirée. Ceci résulte principalement en des *soft ID* représentatif du nom de la page et la seconde consiste à passer par un tiers site nous renvoyant le *hard ID* de la page. Les deux techniques ont des avantages et des inconvénients, toutfois la deuxième technique semble la plus approprié, puisque les *hard ID* sont moins susceptible aux erreurs.

Soft ID

Cette technique est la plus facile à utiliser et permet d'avoir des *page ID* dont le nom représente la page Facebook. Ce qui facilite la lecture de nos informations dans *Facepager*. Cette technique demande de se rendre sur la page Facebook désiré, regarder l'URL de la page et de sélectionner le *Path to ressource* de celle-ci (voir section URL). Voici comment l'URL de la page sera constitué :
https://www.facebook.com/<Soft ID>

Prenons l'exemple de la page Facebook de Barrack Obama. Une fois sur la page de l'ancien président, il est possible d'observer que l'URL est composée ainsi:
https://www.facebook.com/barackobama

L'ID de la page sera donc "barackobama". Cet ID peut être entré comme nouvelle *parent node* dans notre base de données. Permettant d'extraire, par exemple, les publications sur sa page.

Cette technique est très avantageuse du à sa simplicité, mais peu, des fois, poser problème avec des URL plus complexe. Par exemple, il arrive que l'URL d'une page soit composée ainsi:
facebook.com/NomDeLaPage-Hard ID

Dans ce cas, sélectionner l'entièreté du *Path to ressource* mènera à un échec. Les différences dans les structures des URL sur Facebook sont dues à l'architecture du site. Dans cet exemple, il suffit de sélectionner uniquement le *hard ID* et le tour est joué.

Prenons l'exemple de la page Facebook du ministère de la Santé du Burkina Faso. Son URL est composée ainsi:
https://www.facebook.com/MinistC3A8re-de-la-SantC3A9-Burkina-Faso-1444809365833949

Dans cet exemple, l'ID de page n'est pas:
MinistC3A8re-de-la-SantC3A9-Burkina-Faso-1444809365833949
Mais bien:
1444809365833949
Ce qui correspond au *hard ID* de la page.

Quoi que cette technique est facile d'approche et permet d'avoir une base de données dans *Facepager* aillant des *seeds* rappelant les pages Facebook au lieu de suites de chiffres abstrait, cette technique est plus susceptible d'amener des problèmes et c'est pourquoi la technique utilisant les *hard ID* est recommandée.

Hard ID

Les *hard ID* sont nettement plus fiables, mais demande souvent un peut plus d'effort pour les obtenir, puisque la majorité des pages Facebook non pas cet ID dans leur URL. Ils sont plus fiables, puisqu'ils sont le "réel" ID de la page. Contrairement au *soft ID*, qui n'est qu'une image ou liens vers le *hard ID*. La meilleure façon de trouver ce type d'ID est en utilisant un site tiers.

Le site : https://findidfb.com/, permet à l'aide de l'URL d'une page de connaitre le *hard ID* de celle-ci. Il suffit d'aller sur la page Facebook désirée, de copier l'URL et de la coller dans la barre de recherche du site. Celui-ci affichera le *hard ID* de cette page. Cet ID sera la *parent-node* idéale pour nos recherches.

Gérer les ID de page

Lors d'une recherche, il devient important de géré les *page ID* pour se souvenir quel ID est associé à quel compte Facebook. C'est pourquoi il semble important de se garder un document regroupant les noms, les URL et les ID des pages étudié. Ceci est particulièrement important lorsque nous travaillons avec les *Hard ID*. Un simple document Excel suffit pour cette tâche. Voici un exemple de base de données pouvant conserver ces informations.

##

```
## -- Column specification --------------------------------------------------------
## cols(
##   `Nom de la page` = col_character(),
##   `Page ID` = col_character(),
##   URL = col_character()
## )
```

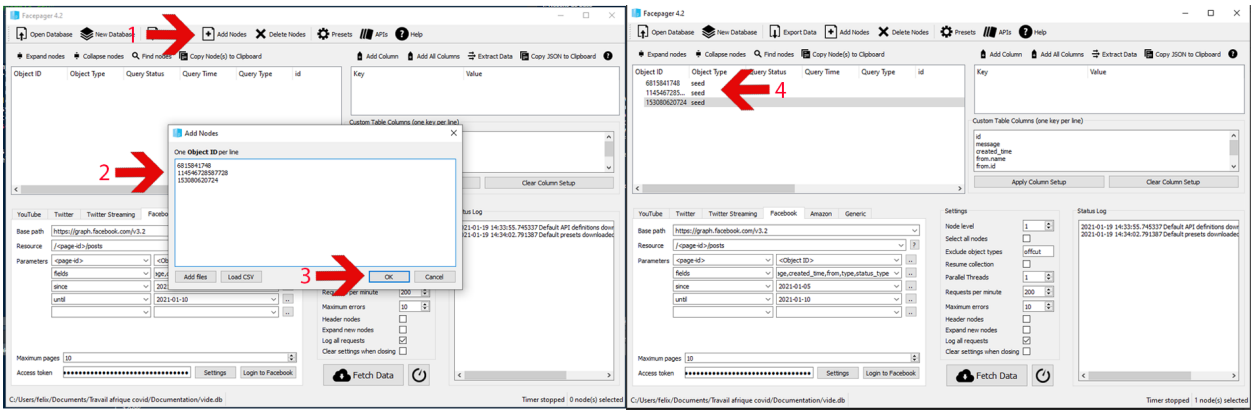
```
## # A tibble: 3 x 3
##   `Nom de la page` `Page ID`        URL
##   <chr>            <chr>            <chr>
## 1 Barack Obama     x6815841748      https://www.facebook.com/barackobama
## 2 George W. Bush   x114546728587728 https://www.facebook.com/georgewbush
## 3 Donald J. Trump  x153080620724    https://www.facebook.com/DonaldTrump
```

Il est possible d'observer dans cet exemple que chaque *page ID* commence par la lettre "X". Ceci est volontaire et a été rajouté. Puisque les ID sont principalement de longues suites de chiffres, la majorité des logiciels les interpréteront ainsi et utiliseront, par exemple, la notation scientifique pour noter ses informations dans nos tableaux. Le fait que cette variable du tableau est traitée par défaut comme une variable numérique ou intégrée amène donc des problèmes puisque ceux-ci sont plutôt un code agissant comme un nom. Il semble donc plus pertinent pour conserver l'authenticité de nos variables d'avoir celles-ci en numérique/string. Rajouter la lettre "x" au début de nos ID permet d'éviter cette confusion dans les logiciels de traitement de données. De plus, puisque les ID sont des noms, avoir un x systématique devant chacun ne pose pas trop de problèmes lors de l'analyse. Il faut simplement garder en tête que cette manipulation a été exécutée pour éviter les problèmes possibles. Pour utiliser notre liste d'ID dans *Facepager*, il suffira de retirer la première lettre de chaque ID avant d'ajouter nos *seeds*.

De plus, un des avantages de posséder une base de données contenant les ID de pages est qu'il devient possible de faire "communiquer" celle-ci avec la base de donnée extraite de Facepage. Par exemple, dans notre base d'ID de président américain, il serait possible de rajouter la variable "parti politique qu'il représentait". Ensuite à l'aide des pages ID, il serait possible de rajouter cette variable dans notre base de données sortant de Facepager. Permettant de ne pas faire uniquement des recherches sur les publications de tel ou tel président, mais aussi sur l'ensemble des présidents démocrates par exemple.

Mettre nos Pages ID dans Facepager

Une fois les que les identifiants de pages sont récoltés, il suffit d'utiliser l'option `Add Nodes` de la barre de menu et de rentrer un *page ID* par ligne et d'appuyer sur le bouton "Ok". Les nouvelles *seeds* apparaitront dans l'affichage de node.

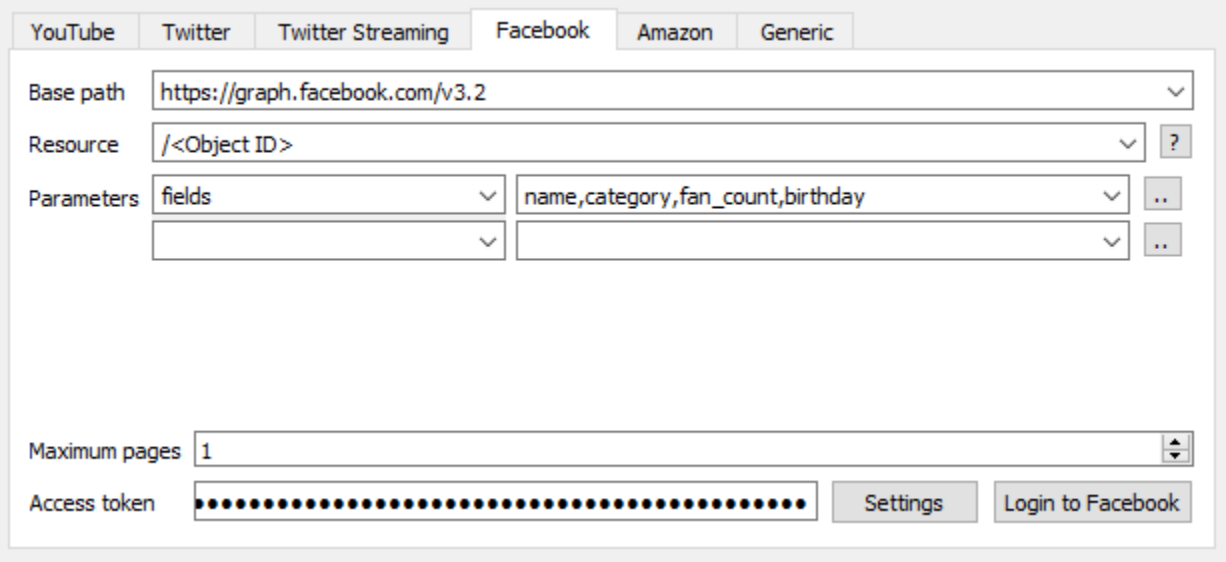


1- Appuyer sur `Add Nodes` 2- Insérer les *pages ID* des pages désirés. Une ligne = un ID 3- Confirmé en appuyant sur `OK` 4- Les nouvelles *nodes* apparaissent dans l'affichage de *nodes*

Faire des demandes sur l'API

Une fois les *Seeds* importés dans *Facepager*, la collecte de donnée peut commencer. Pour faire une demande sur l'API, il est possible à partir de l'organisateur de demande de créer des demandes personnalisées ou d'utiliser des *Presets*. Les *presets* sont des configurations préprogrammés par *Facepager* permettant de faciliter l'extraction de données. Pour mieux comprendre à la fois comment les *presets* et le processus de demande fonctionnent, il semble important de regarder comment les options de demande (*Query Settings*) fonctionnent pour Facebook. Il faut comprendre que Facebook fonctionne avec une structure hiérarchique : les pages ont des publications, qui elles ont des commentaires. Ainsi, pour obtenir les commentaires, il faut commencer par extraire la publication. Ceci est représenté par les *nodes levels* dans *Facepager*.

Organisateur des demandes



Query settings	Explication
Base path	Cette option devrait toujours être la version du *Graph API* de Facebook la plus récente. Plusieurs problèmes peuvent être associés au fait que le *Base path* est associé à une version plus vieille de l'API. En général, il est mieux de ne pas toucher à cette option, car elle permet de se connecter à l'API.
Resource	Cette option permet de sélection le type d'objet que nous désirons extraire de Facebook. Les options disponibles sont des suggestions. Il est possible de personnaliser nos recherches encore plus avec cette option, mais pour la majorité des besoins les suggestions sont suffisantes.
Parameters	Cette option permet de préciser l'information désirée sur l'objet sélectionné. Par exemple, les publications des trois derniers jours et quelles réactions que nous désirons sont des exemples d'option possible.

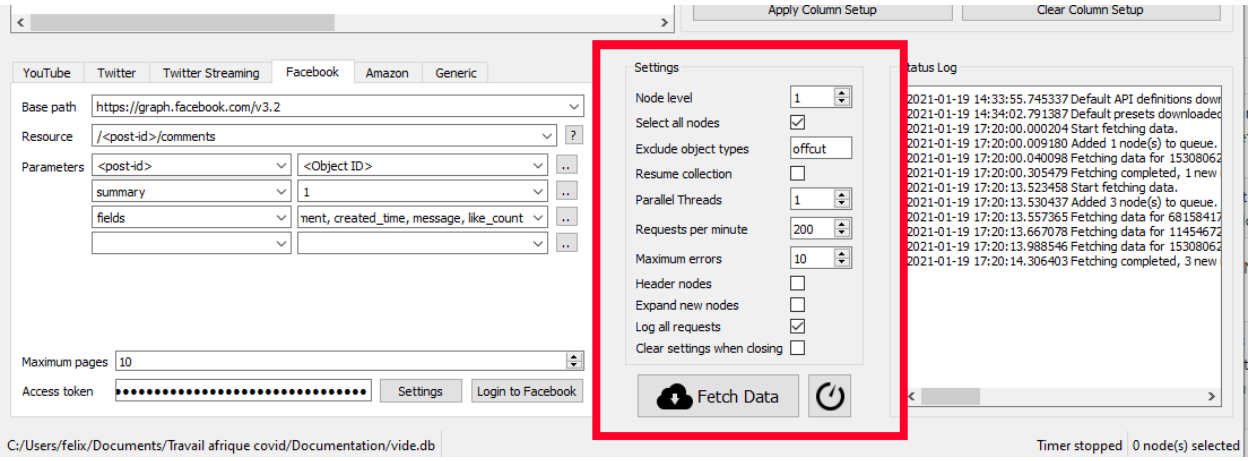
Query settings	Explication
Maximum pages	Cette option permet de déterminer le nombre de "pages" que *Facepager* tente de télécharger sur une *node*. En général si nous tentons de récolter 900 commentaires sur une publication avec une limite de 10 pages, nous ne recevrons pas toute l'information. Cette limite d'information par "page" est déterminée par l'API. Fonctionnellement parlant, plus il y a d'information à extraire, plus il faut un nombre de pages maximum élevé.
Access Token	Notre *Token Access* pour se connecter à l'API (voir section se connecter à Facebook)

Pour faire des demandes sur l'API, il faudra utiliser principalement les options Ressource et Paramètre. Ceux-ci permettront de parler à l'API et construisant des URL pour recevoir l'information désirer.

Option

Une fois que *Facepager* connait quelle sorte d'information nous tentons d'extraire, il faut lui indiquer sur quels *nodes* nous désirons faire la demande. La technique la plus facile consiste à sélectionner manuellement celles que nous désirons et d'appuyer sur l'option `Fetch Data`.

Toutefois, cette technique est plus limitée et comprendre les options générales de *Facepager* permet de faciliter à long terme notre expérience avec *Facepager*. Les options générales permettent de personnaliser notre demande au niveau de la procédure utilisée et les nodes utilisées. Ils se retrouvent à la droite de l'organisateur de demande et juste en haut de l'option `Fetch Data`, qui permet de faire la demande. Les choses sont bien faites, puisque configurer ses options est la dernière étape avant de faire une demande.



Option	Explication
Node level	Cette fonction fonctionne comme un filtre. Si l'on sélectionne la *parent node* et qu'on augmente le niveau par un, la demande se ferra sur toutes les *child nodes* celle-ci. Ceci représente le niveau sur lequel la demande est faite (voir la section Nodes). Ceci permet de faire des demandes sur toutes les *childs nodes* sans les sélectionner manuellement.
Select all nodes	Permets de faire la demande sur toutes les *nodes* du niveau sélectionné. Cette option est parfaite pour les larges bases de données.
Exclude object types	Cette option permet de sauter les *nodes* dont le type est inscrit dans la boite. Par défaut, le type *Offcut* y est inscrit et normalement on ne touchera pas à cette option.
Resume collection	Permets de recommencer la collecte à l'endroit où elle a été arrêtée ou annulée.
Parallel Threads	Permets de collecter de l'information en parallèle (faire plus d'une demande en même temps). Cela augmente la vitesse de collecte de donnée. Toutefois certains API peuvent suspendre votre compte si votre vitesse est trop rapide. Cette option est à utiliser avec précaution.

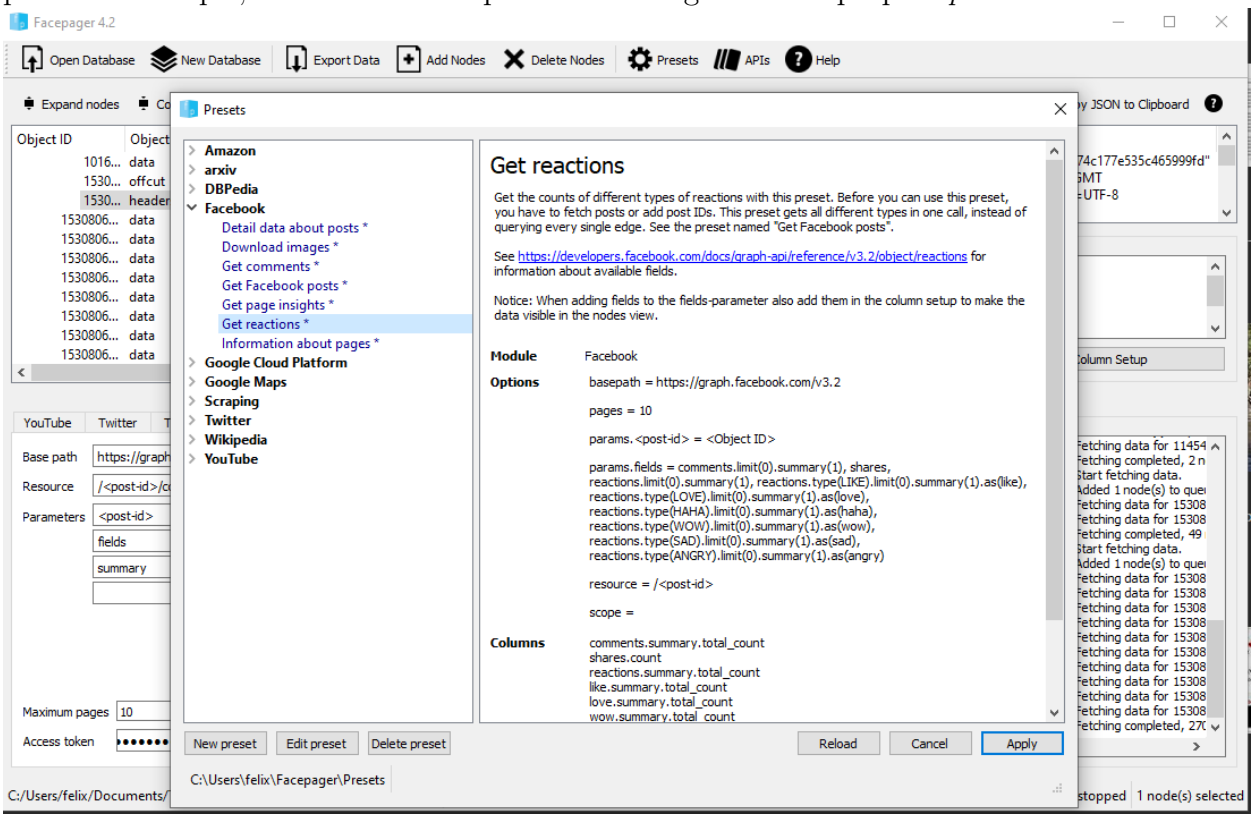
Option	Explication
Request per minute	Certains API limitent le nombre de demandes par minute (voir Rate limitée). Ralentir le nombre de demandes par minute peut permettre un flux plus constant de demande.
Maximum errors	Le nombre d'erreurs consécutives maximal que *Facepager* attend avant d'arrêter les demandes. Il est recommandé de garder ce nombre relativement bas pour votre base de données et les serveurs. Les demandes erronées mettent beaucoup de stress sur les serveurs et votre base de données sera remplit de *nodes* erreurs se qui embrouille votre base de données. Autour de 5 à 10 erreurs maximum semblent adéquates.
Header nodes	Sers à créer une *node* contenant les informations *headers* renvoyées par l'API. Ces informations comportent des données supplémentaires sur les *nodes* présentent dans notre liste. Souvent des informations plus générales sur notre demande. Cette option ne semble pas être utile pour l'objectif de ce document, puisque l'information supplémentaire ne semble pas être pertinente et cruciale.
Expand new nodes	Permets d'automatiquement expandre les nouvelles *nodes* rajoutées. Pour les demandes massives, il est recommandé de fermer cette option pour accélérer le processus.
Log all request	Permets de voir chacune des demandandes faite dans l'affichage de statut. Sans cette option activée. Uniquement les informations cruciales, comme les erreurs, seront affichées dans l'affichage de statut.
Clear settings when closing	Permets d'effacer toute l'information enregistrée lorsque le logiciel est fermé. Principalement utilisé lorsque nous travaillons sur des ordinateurs publics pour s'assurer que notre *access token* ne soit pas sauvegardé.

Juste avant de faire une demande, il faut donc regarder les options. Il y a beaucoup d'options disponibles, mais les deux plus importante sont `Node level` et `Select all nodes`. Normalement appuyer sur la *node* que nous désirons est suffisant, mais pour une plus grande base de données les options nous facilitent le travail. En utilisant l'option `Select all nodes` et en choisissant le `Node level` approprié à notre demande, il devient facile de faire une demande sur plusieurs pages, publications ou commentaires en même temps.

Presets

Quoiqu'il est important de connaitre comment faire des demandes personnalisées, il semble important de noter que *Facepager* possède des *Presets*. Les *presets* sont des demandes préprogrammées qui facilitent le processus. Les *presets* conservent de l'information sur le type de demande et les clefs affichées dans l'affichage de *nodes*. Ainsi, utiliser un *preset* modifie l'organisateur de demande et l'organisateur de colonne.

L'option `Presets` se trouve dans la barre de menu et ouvre une liste de demande préprogrammée contenant une brève description de celle-ci. Les *presets* préprogrammés sont indiqués par un astérisque, mais il est aussi possible d'enregistrer nos propres *presets*.



Boutons | Fonction Apply | Sélectionne le *preset* et le transfert dans l'organisateur de demande et l'organisateur de colonne. New preset | Permets de créer un nouveau *preset* basé sur l'organisateur de demande et de colonnes actuel. Edit preset | Permets de modifier un *preset* existant. Les *presets* d'origine, marquée d'un astérisque, ne peuvent être édités. Delete preset | Supprime le *preset* sélectionné. Reload | Permets de recharger les *presets*. Cancel | Permets

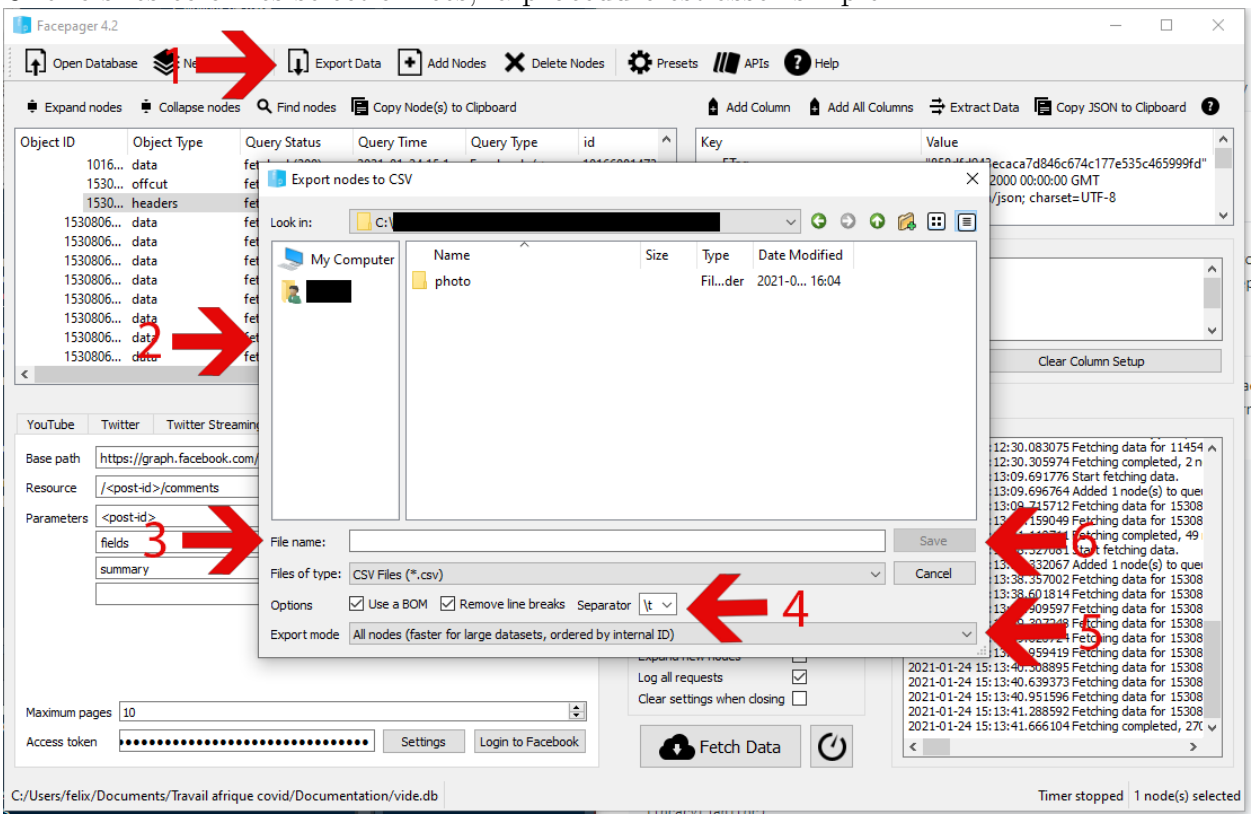
la fenêtre de *presets*.

Avant toute recherche, il est recommandé de regarder si un *preset* existe déjà pour le genre de demande désiré. De plus, enregistrez nos demandes les plus couramment utilisées en *presets*, facilite le travail. Surtout après une longue période sans utiliser *Facepager*.

Exporter

Une fois les données collectées, il faut exporter notre base de données en format CSV pour pouvoir l'utiliser avec R. avant d'exporter notre base de données, il faut commencer par sélectionner les clefs que nous désirons dans notre base de données. En utilisant l'organisateur de colonnes, il est possible de choisir quelles données apparait dans notre affichage de *nodes*. Ceci est important puisque seulement les clefs affichées comme colonne dans l'affichage de *nodes* seront exportées dans notre base de données. Il est possible d'utiliser l'option `Add all columns` pour s'assurer d'avoir toutes nos clefs, toutefois plus il y a de colonnes d'affiché, plus notre base de données sera dense. Ceci implique qu'elle prendra plus de place sur notre ordinateur, prendra plus de temps à ouvrir et à travailler avec sur Rstudio. Ainsi, sélectionner uniquement les informations pertinentes est une bonne pratique. Surtout lorsque nous avons une grande base de données.

Une fois les colonnes sélectionnées, la procédure est assez simple:



1- Appuyer sur l'option `Export data` de la barre de menu pour ouvrir la fenêtre d'exportation des données. 2- Sélectionner le fichier dans lequel vous désirez sauvegarder le document. 3-

Donner un nom à votre base de données. 4- Sélectionner le séparateur utilisé dans votre base de données. Un séparateur est un symbole utilisé dans notre base de données pour séparer les informations. C'est un peut un symbole de ponctuation qui informe nos logiciels (Excel, R, etc.) quand il faut changer de colonne et ranger dans notre base de données. Par défaut, *Facepager* sélectionne le séparateur le plus commun soi le ";" appelé point virgule. Toutefois, lorsque nos informations sont principalement du texte, comme des commentaires ou des publications, il arrive qu'une personne utilise le point virgule dans son texte. Ceci amène des problèmes dans notre base de données qui sépare le commentaire en deux. C'est pourquoi, que je favorise le séparateur "\t" appelé Tab pour les bases de données avec beaucoup de texte. Une fois dans *Rstudio*, il suffira de s'assurer d'indiquer le bon séparateur avant d'importer la base de données. 5- Décider si nous désirons exporter uniquement les *nodes* sélectionnés ou l'entièreté de notre base de données. 6- Appuyer sur `Save` pour sauvegarder notre base de données.

Une fois notre base de données exportée, il faut encore la traiter avec un logiciel comme Rstudio. Effectivement cette base de données n'est pas propre et regroupe plusieurs informations différentes. Par exemple, des publications avec des commentaires. Il semble préférable d'avoir une base de données de commentaires et une base de données de publications. Voilà pourquoi il reste du travail pour nettoyer notre base de données avant l'analyse.

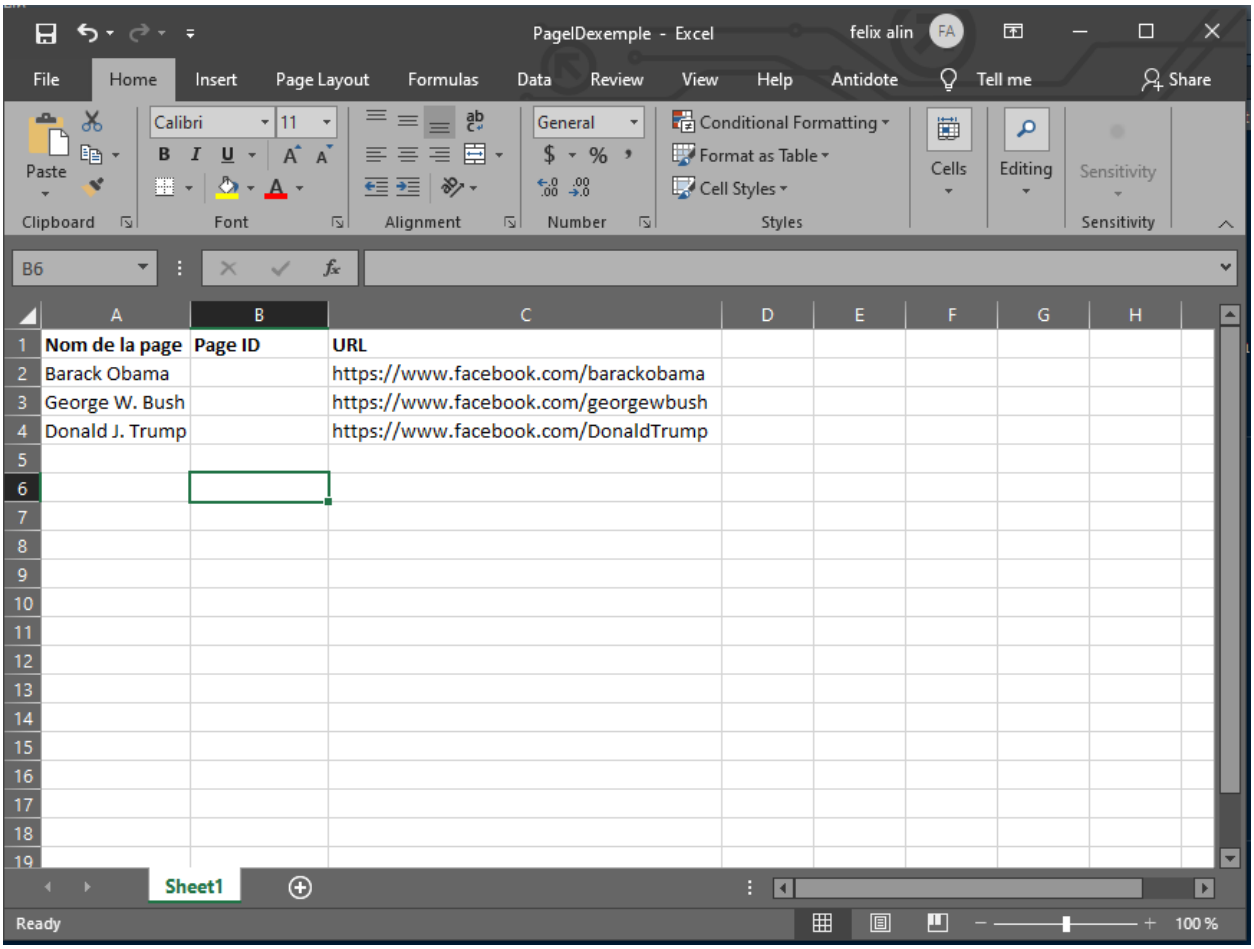
Exemple

Il semble important pour la compréhension de faire un exemple concret avec toute cette information. Imaginons que nous désirons faire une base de données sur les comptes Facebook des trois derniers présidents des États-Unis. Donals Trump, Barrack Obama et Joe Biden. Cette base de données cherchera à extraire les publications, les réactions et les commentaires entre 2021-01-01 et 2021-01-10. Voici les étapes qui permettront de faire cette extraction:

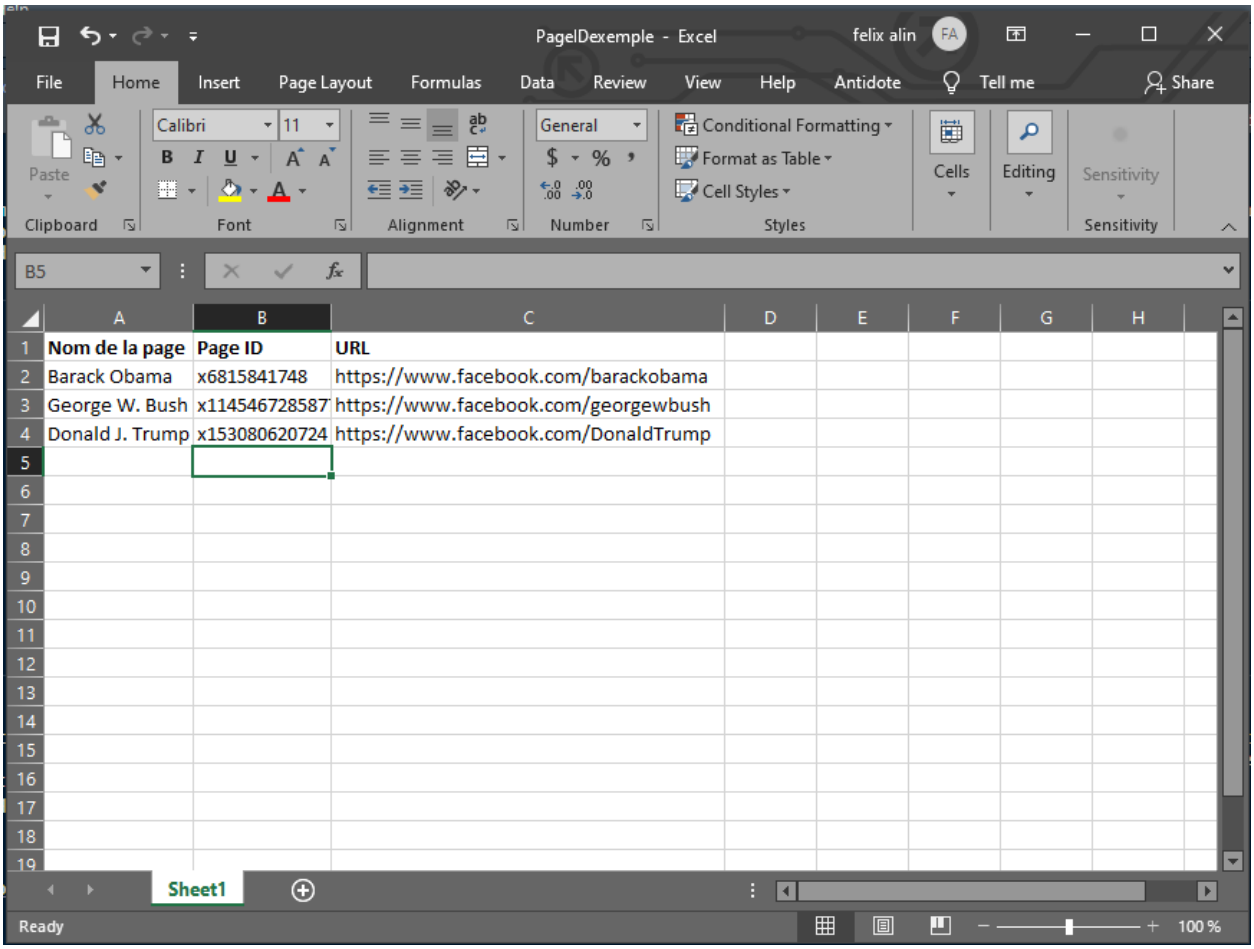
1- Trouver les identifiants (ID) des pages. 2- Se connecter à Facebook dans *Facepager*. 3- Créer nos *parents nodes*. 4- Extraire les publications. 5- Extraire les réactions. 6- Extraire les commentaires. 7- Exporter la base de données.

Trouver les identifiants (ID) des pages

Commençons par ouvrir un fichier Excel nous permettant de conserver les informations de nos pages Facebook. Dans ce document trois informations de base seront utiles pour toute recherche sur Facebook: le nom de la page, l'URL de la page et l'ID de la page. En aillant ses trois informations, nous sommes garanti de pouvoir retrouver quelle information correspond à quel page. Commençons par aller sur chaque page Facebook et prendre en note le nom de la page et l'URL.

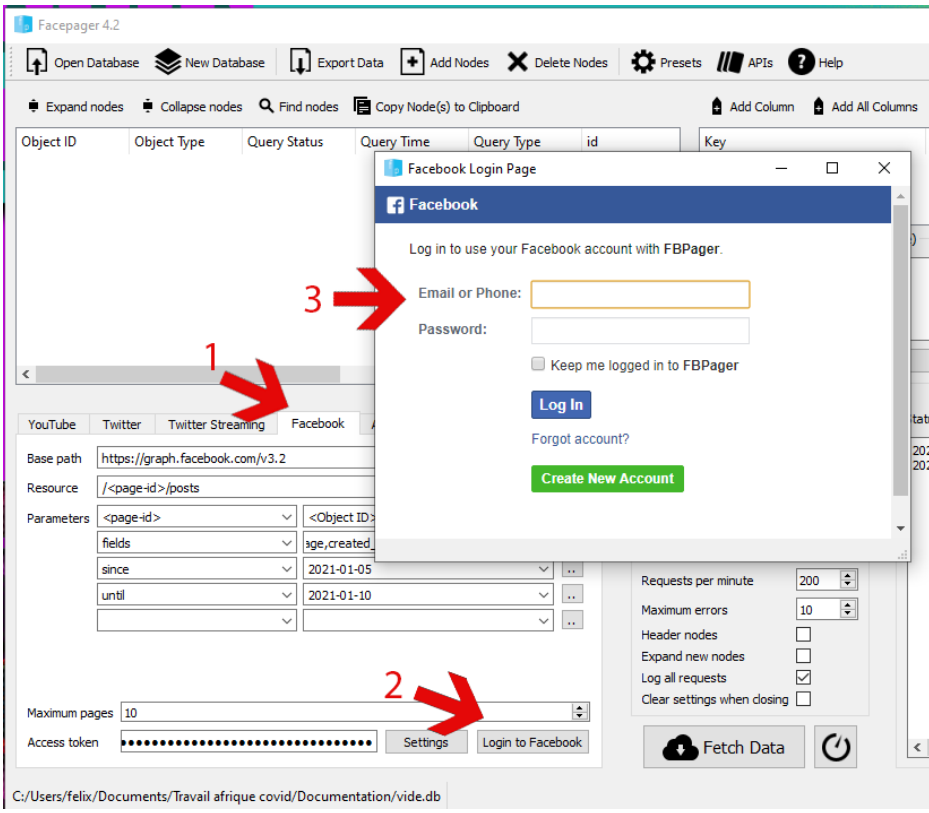


Ensuite, à l'aide du site https://findidfb.com/, il sera possible de trouver les *Hard ID* des pages Facebook. Souvenons-nous que les *Hard ID* sont les identifiants les plus fiables et ce seront donc ceux-ci que nous utiliserons. Sur le site, il suffit d'insérer l'URL d'une page dans la barre de recherche et d'appuyer sur le bouton `Find numeric ID`. Ceci nous donnera la série de chiffres correspondant à notre ID de page. Il suffit d'insérer ce nombre dans notre base Excel. Toutefois, il est recommandé d'ajouter un "x" au début de l'identifiant pour éviter les problèmes pouvant survenir lorsque les logiciels interprètent ce nombre comme une variable numérique et la simplifient comme étant un nombre scientifique. Voici le résultat final:



Se connecter à Facebook dans *Facepager*

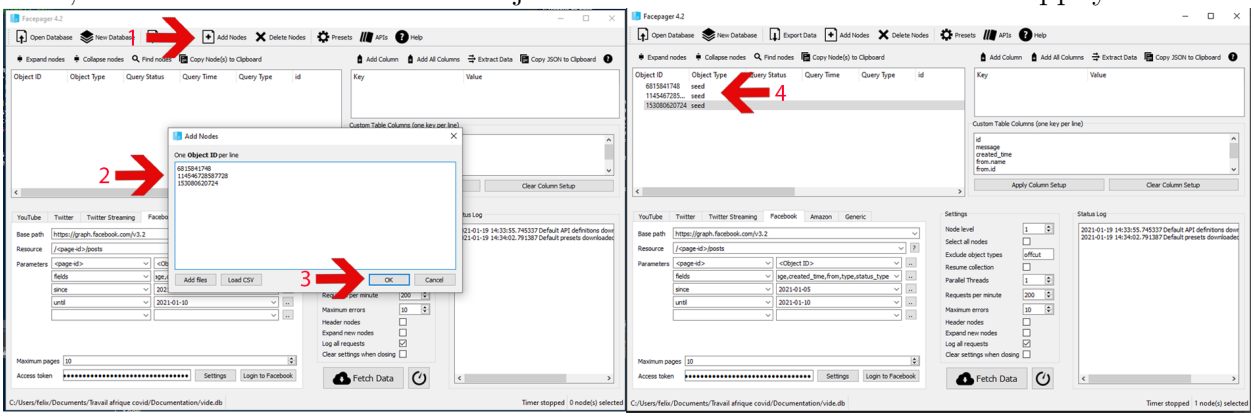
Ensuite, il suffit d'ouvrir *Facepager*, de choisir le module Facebook dans l'organisateur de colonnes et d'appuyer sur l'option `Login to Facebook` pour recevoir son *access Token* permet-



tant de se connecter à l'API Facebook.

Créer nos *parent node*

Il faut ensuite créer nos points de départ de la collecte de donnée. À l'aide de l'option `Add Nodes`. Une fois dans la fenêtre `Add Nodes`, il suffit d'inscrire les identifiants de pages dans la fenêtre en respectant la formule : une ligne = un ID. Soit un identifiant par ligne. Dans notre cas, nous utiliserons l'option copiée-collée pour transférer la colonne d'identifiant dans notre document Excel dans la fenêtre. Ensuite, il suffit d'enlever les "x" rajouté devant nos identifiants et d'appuyer sur `OK`.

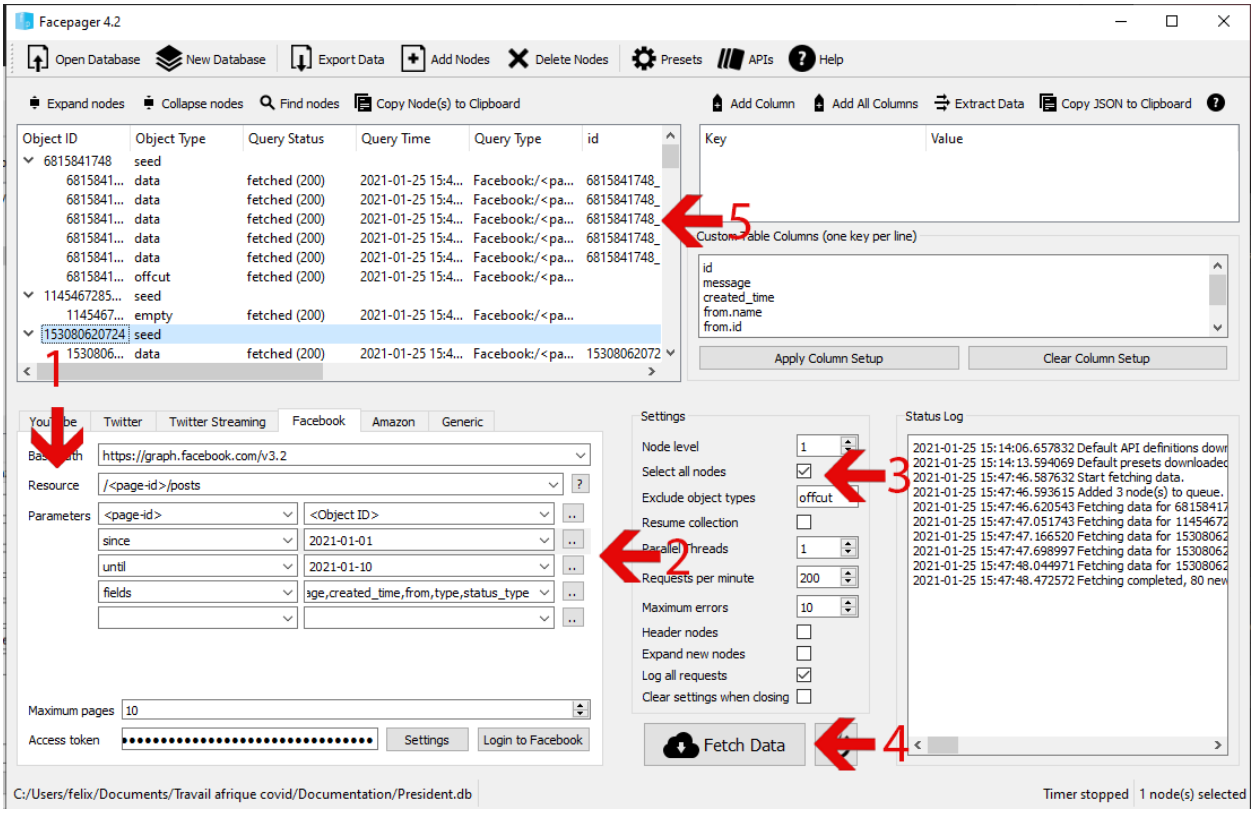


Extraire les publications.

Maintenant que nous avons accès aux pages Facebook, nous pouvons aller chercher l'information sur les publications publiée par nos pages Facebook. Pour ce faire, il faut regarder dans l'option ressource de FacePager situé dans l'organisateur de demande. Cette option nous permet de faire des demandes d'informations à Facebook sur les diverses Nodes que nous avons. Pour extraire les publications, l'option qui nous intéresse est l'option nommée: `/<page-id>/posts`. Ensuite, nous avons accès aux paramètres qui nous permettent de modifier notre demande sur les publications. Nous utiliserons trois paramètres, toutefois il y en a plusieurs autres que vous pouvez expérimenter.

- **Since**: La date des plus vieilles publications que l'on désire extraire.
- **Until**: La date des plus récentes publications que l'on désire extraire.
- **Fields**: Les informations sur les publications que l'on désire extraire. L'information de base est suffisante pour notre analyse.

Une fois les paramètres programmés, il suffit de sélectionner les pages sur lesquelles on désire faire la demande ou de cocher l'option `Select all nodes` (pour faire la demande sur toutes les nodes) en s'assurant que l'option `Node level` soi sur un et appuyer sur `Fetch data`.



1. Ressource = *//posts*
2. Régler les paramètres. Les dates des publications principalement, la section *fields* est utilisée pour chercher des informations plus spécifiques. Pour notre exemple: **Since** =

2021-01-01 **Until** = 2021-01-10 **Fields** = les options de base nous suffisent. Il suffit de ne rien toucher.

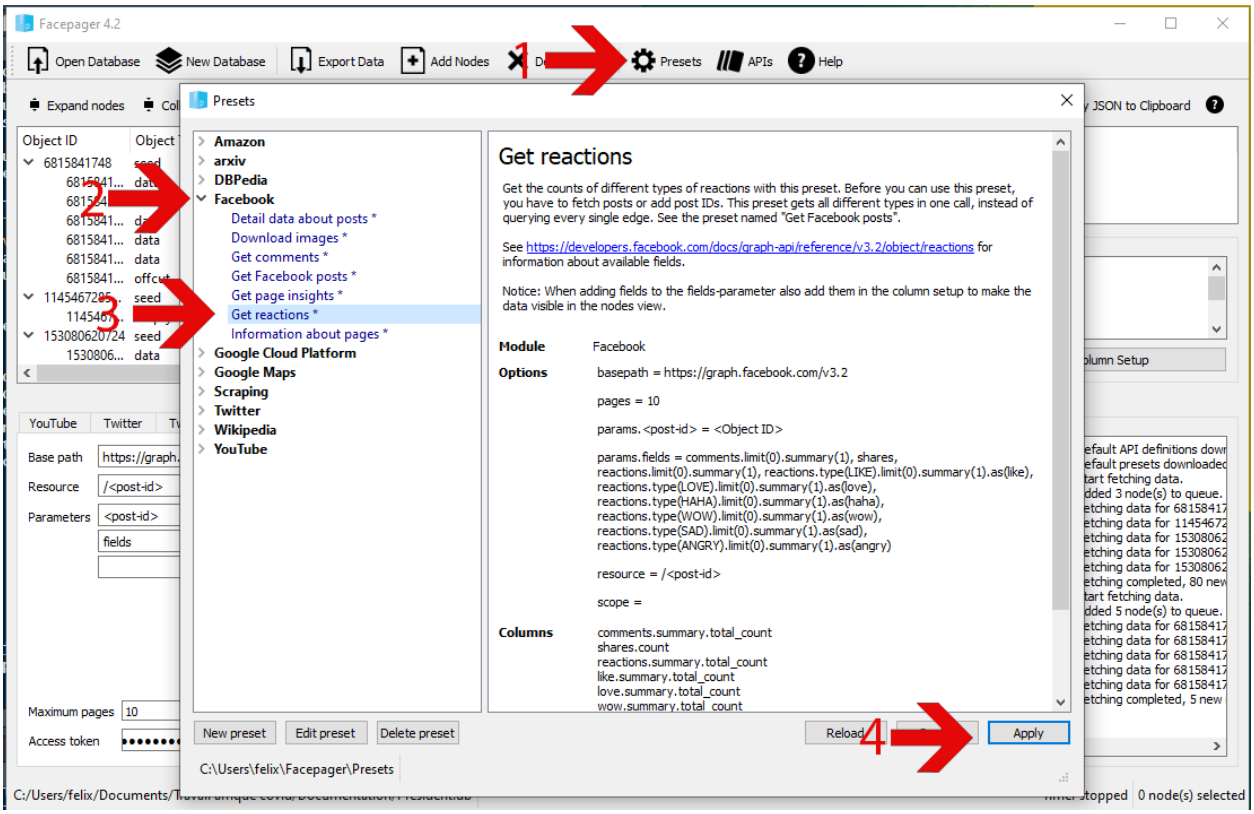
3. Sélectionner les pages sur lesquelles vous voulez faire la demande ou sélectionner la fonction `Select all nodes`. Ce que nous ferons pour cet exemple.
4. Appuyer sur `Fetch data`
5. Voici nos *child nodes* qui contiennent les publications sur nos pages.

Notez que les publications sont apparues en dessous des Nodes des pages. C'est parce que chaque publication est associée à une page et est donc mis sur un niveau plus bas de Nodes. Ceci est bien pour garder organisées nos informations. De plus, la deuxième page, celle de Donald Trump. Renvoie une *node* de type *empty*. Ceci est dû aux faits qu'il n'a pas publié sur sa page durant cette période.

Lors de la collecte d'une grande quantité de publications, il se peut qu'il manque des publications. FacePager par défaut, mets un maximum de pages d'informations qu'il télécharge. Toutefois, nous pouvons augmenter cette limite avec l'option `maximum pages` *se trouvant en bas de l'organisateur de demande, juste en haut du Access Token.*

Extraire les réactions.

Une fois nos publications extraites, nous allons chercher à télécharger le nombre de réactions sur chaque publication. La meilleure façon de faire est d'utiliser le preset *get reaction*, qui associera à chacune de nos publications le nombre total de réactions, le nombre individuel de chaque réaction (nombre de likes, nombre de HAHA, etc.), le nombre de partages et le nombre de commentaires. Toutefois, le nombre de commentaires ne semble pas toujours être juste et semble être souvent en dessous du nombre réel de commentaires. Nous utiliserons donc une autre technique pour les commentaires lors de l'extraction des dits commentaires. Pour charger le preset, il suffit de cliquer sur l'option `Presets` dans la barre de menu de Facepager, puis de cliquer sur l'onglet Facebook, *get_reaction* et finalement de cliquer sur load.



Une fois le preset appliqué, sélectionner vos publications ou utiliser l'option `Select all nodes` tout en s'assurant que l'option `Node level` est sur deux, puisque les publications sont subordonné au page Facebook et sont donc sur le deuxième niveau et cliquer sur *Fetch Data*. Remarquer que la *node* apparait sous la *node* de la publication et est donc considérée sur le troisième *node level*.

Il faut noter un problème qui peut arriver lorsque la demande est faite sur plusieurs pages en même temps, souvent il y a une duplication des nodes qui se produit. Il est possible de régler se problème une fois sur R toutefois cela augmente la grosseur de notre base de données artificiellement et lorsque l'on traite beaucoup de données gérées la grosseur de cette base devient un objectif important pour accélérer le traitement et réduire les erreurs. Si ce problème vous arrive, je conseille de faire le processus page par page.

Extraire les commentaires.

Finalement, nous allons extraire les commentaires sur les publications. Pour se faire, il faut sélectionner la ressource `/<post-id>/comment`. Dans les options de cette demande, il y aura trois options qui nous intéresse : `fields`, `filter` et `summary`.

L'option `fields` nous permet de sélectionné qu'elles informations nous intéresse. L'option de base est suffisante, mais nous rajouterons à la fin de la ligne l'option `comment_count`, qui nous donnera le nombre de commentaires sur le commentaire.

L'option `summary` nous permettra d'avoir un sommaire du nombre de commentaires surs la

publication. Toutefois pour avoir ce sommaire, il faut rentrer `total_count` à la place du un dans cette option.

L'option `filter` sert à déterminer si nous voulons les commentaires selon leur popularité ou selon leur date de publication. Cette option doit être mit sur `stream`, qui permet d'avoir les commentaires selon la date de publication. `filter` est important, car combiné avec l'option `Summary = comment_count` cela nous donne le nombre exact de commentaire sur la publication. Ce qui nous permettra de régler le problème du mauvais nombre de commentaires lors de l'extraction des réactions. Il est à noter que cette information sera dans une *node* de type *offcut* au lieu du type *data*.

Avant de faire notre demande, il semble intéressant de sauvegarder ses paramètres dans un *Preset* pour pouvoir utiliser dans le futur cette demande sans avoir à tout réécrire. Pour se faire, il suffit de cliquer sur l'option `Presets` dans la barre de menu et d'appuyer sur `New preset` Ensuite, il faut donner un nom à cette commande, par exemple: commentaire et nombre de commentaires, de sélectionner la catégorie, dans notre cas Facebook, et il est possible de rentrer une courte description. En appuyant sur `OK` *Facepager* sauvegardera notre configuration de demande et de colonnes permettant un accès facile à cette demande.

Une fois les paramètres rentrés, il suffit de sélectionner nos nodes de publication ou encore une fois sélectionner l'option `Select all nodes` avec un niveau de *node* mit à deux et d'appuyer sur *Fetch Data*.

Comme pour les réactions, s'il y a des problèmes de dédoublement, je conseille de faire les demandes pages par pages. Ceci peut arriver avec les grandes bases de données.

Exporter la base de données.

Une fois l'extraction terminée, il faut exporter notre base de données en format CSV pour permettre à Rstudio de lire celle-ci et faire notre analyse. Ceci se fera en deux étapes, la première consiste à s'assurer des colones sélectionnés dans l'interface, car Facepager créera une base de données avec les colones sélectionnés uniquement. La deuxième étape consiste simplement à l'exporter.

Sélectionner les colonnes

Sélectionner les colonnes est important pour deux-points. Premièrement, cette étape est importante pour s'assurer que toutes les données seront dans notre base de données. Par exemple, si la colonne du nombre de likes n'est pas incluse, elle ne sera pas présente lorsque nous l'ouvrirons dans Rstudio. De plus, cette étape est importante pour s'assurer qu'il n'y a pas d'information superflue. Effectivement chaque colonne superflue augmente artificiellement la grosseur de la base de données. Lorsque l'on analyse beaucoup d'information, beaucoup de *nodes*, même une colonne peut augmenter la taille de notre fichier de façons substantiel. Ainsi, pour faciliter le traitement de données dans Rstudio et aussi réduire la demande

sur l'ordinateur, s'assurer que nous avons uniquement l'information voulue au moment de l'exportation est important.

Je conseille de sélectionner une node de chaque type (réaction, publication et commentaire), puis d'appuyer sur l'option `Add all colums` situé en haut de la boite d'affichage des données sur la droite. Ceci garantira que toute l'information est présente et vous pourrez la voir dans la boite affichage de *nodes* sous forme de colonne. Ensuite, vous pourrez passer colonne par colonne et déterminer si celle-ci est importante pour vous. Pour enlever une colonne, simplement trouver sont nom dans la section *Costum Tables Columns* situé à droite du sélecteur de node, effacer son nom de la liste et appuyer sur *Apply Columns Setup*.

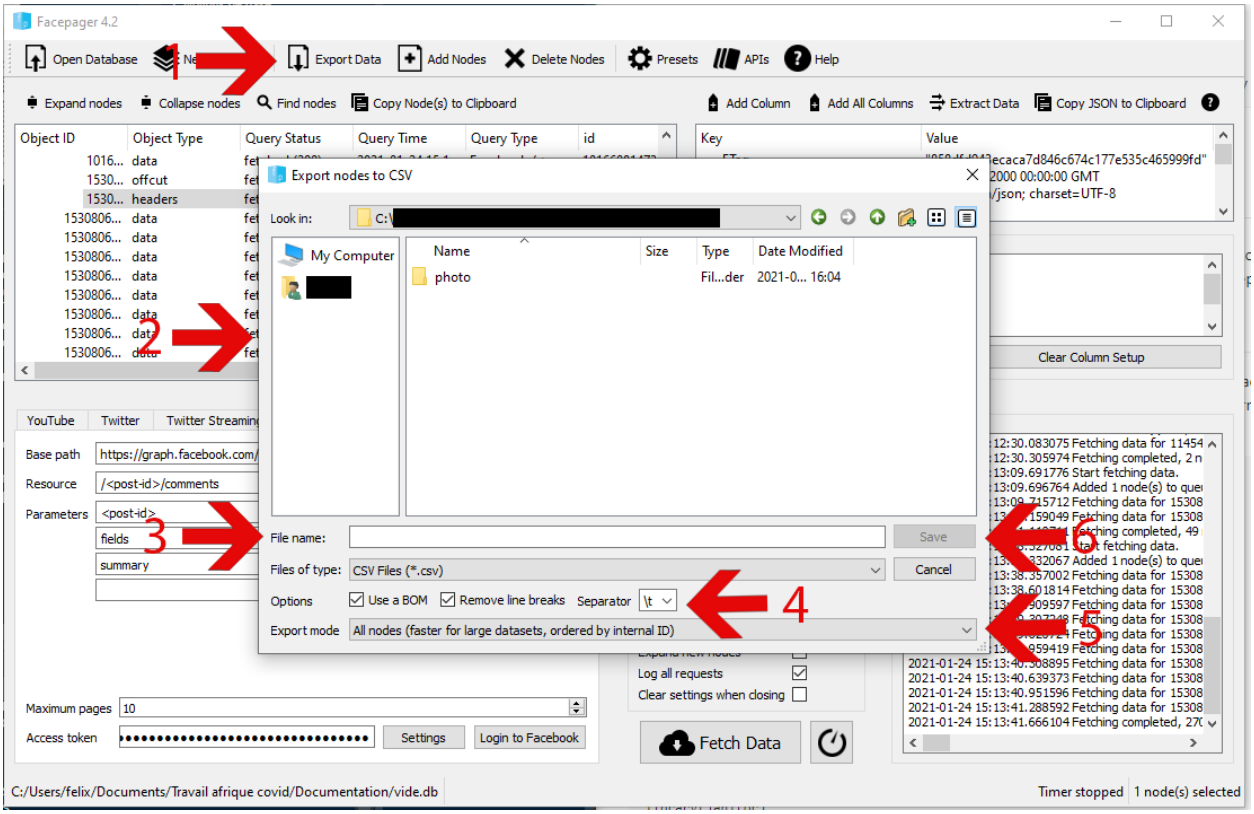
Pour notre exemple voici les colonnes utilisées: comments.summary.total_count shares.count reactions.summary.total_count like.summary.total_count love.summary.total_count wow.summary.total_count haha.summary.total_count sad.summary.total_count angry.summary.total_count idl attachment.type created_time message like_count comment_count summary.total_count from.name from.id type status_type

De plus, puisque les *presets* sauvegardent aussi les colonnes, nous irons modifier le *preset* que nous venons de créer, pour inclure ces colonnes. Il faut s'assurer que le *preset* est actif pour que les options de demande restent les mêmes. Une fois le *preset* actif, modifier les colonnes désirées. Ensuite, sélectionnez votre *preset* et utilisez l'option `Edit preset` dans la fenêtre de *Presets*. Il faut par la suite cocher la boite `Overwrite parameters with current settings` et appuyer sur `OK`. Ainsi, notre *preset* "Commentaire et nombre de commentaires" servira à la fois à télécharger les commentaires et à sélectionner nos colonnes.

Exporter

Pour exporter notre base de données, il suffit de cliquer sur le bouton *Export Data* de la barre de menu. Une fois dans la fenêtre, il est possible de sélectionner l'endroit où le sauvegarder, de donner un nom à la base de données et de formater la façon de séparer l'information.

Le séparateur est important, car il peut mener à des erreurs. Il y a trois séparateurs disponibles dans *Facepager*: la virgule, le point-virgule et le format Tab. Puisque nous prenons beaucoup de texte, nous ne pouvons pas utiliser un format de séparateur comme la virgule. La virgule ou *comma* étant très présent dans l'écriture notre base de donné aura de la difficulté à bien diviser les commentaires et publications. Par défaut le séparateur est le point-virgule ou *semicolon*, ceci marche la majorité du temps, mais dans mon expérience, des publications peuvent utiliser le point-virgule. Je considère que pour nos besoins le format Tab ou */t* est le plus adapté à nos besoins. Ainsi, il faut simplement sélectionner sous l'option séparateur */t*. Une fois dans Rstudio il faudra s'assurer d'indiquer que notre fichier CSV utilise les séparateurs Tab.



Une fois exporté notre base de données est loin d'être prêtre à faire de l'analyse. Par exemple, celle-ci comporte à la fois des commentaires et des publications, il sera important de les séparer. De plus les réactions sont considérées comme des *nodes* différentes des publications, il sera donc important d'associer chaque réaction avec sa publication d'origine. Ainsi, suite à ce processus, il faut prendre du temps dans Rstudio pour comprendre les variables de notre base de données et la nettoyer avant de pouvoir analyser de l'information.

Références (Vissého)

https://github.com/strohne/Facepager/wiki

https://developers.facebook.com/docs/graph-api/o