

# Webscrapping for Analytics

Vandy Berten

Smals Research

Webinaire 30/06/2020

# Smals Research



**Innovation with  
new technologies**



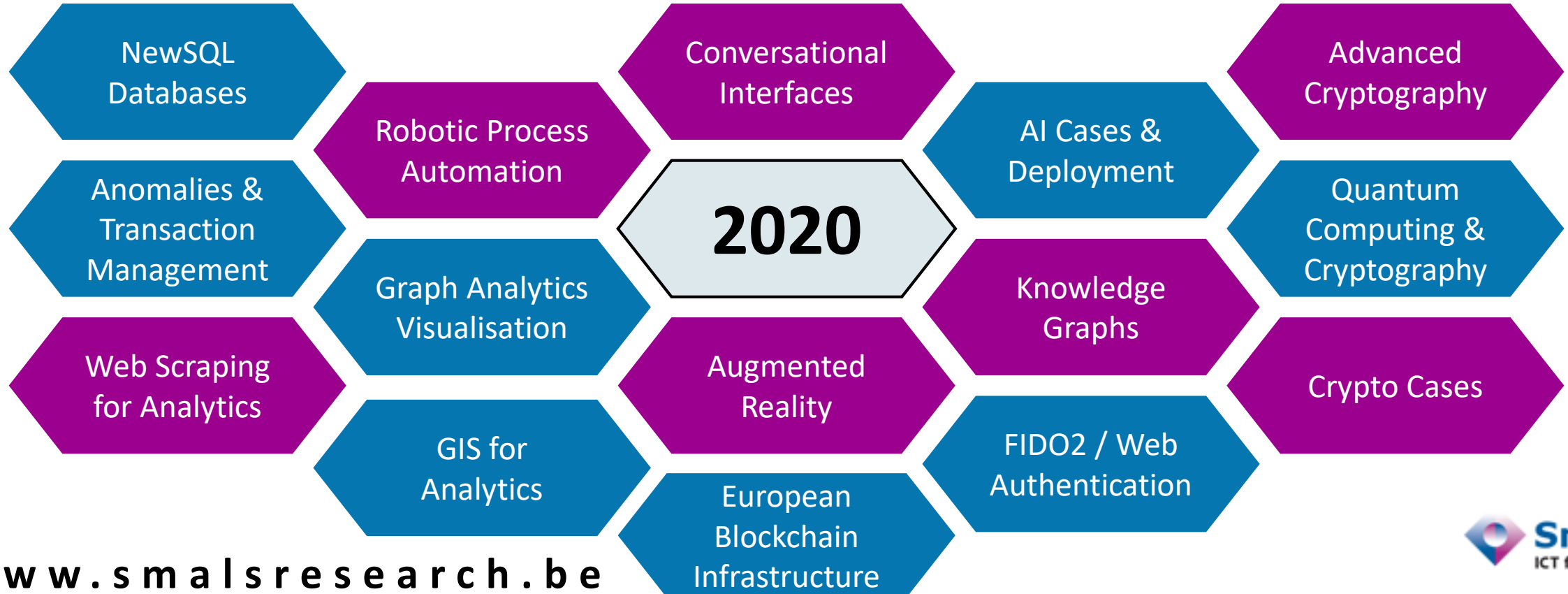
**Consultancy  
& expertise**



**Internal & external  
knowledge transfer**



**Support for  
going live**



- Définition
- Exemples eGov
- Approches
- Exemple Scrapy
- POC Horeca (Scraping + Entity Linking)
- Aspects juridiques
- Conclusions

- **Web scraping** : technique d'extraction de contenu sur des sites web, au moyen de scripts/programmes, dans le but de son utilisation dans un autre contexte
- Web crawling/scraping/harvesting
- Sans accès à des APIs dédiées (cf Twitter, Facebook, Google, flux RSS...)
- Le plus gros crawler : Google !



**LE WEBSCRAPING,**  
 la collecte et le traitement de données en ligne pour l'indice des prix à la consommation

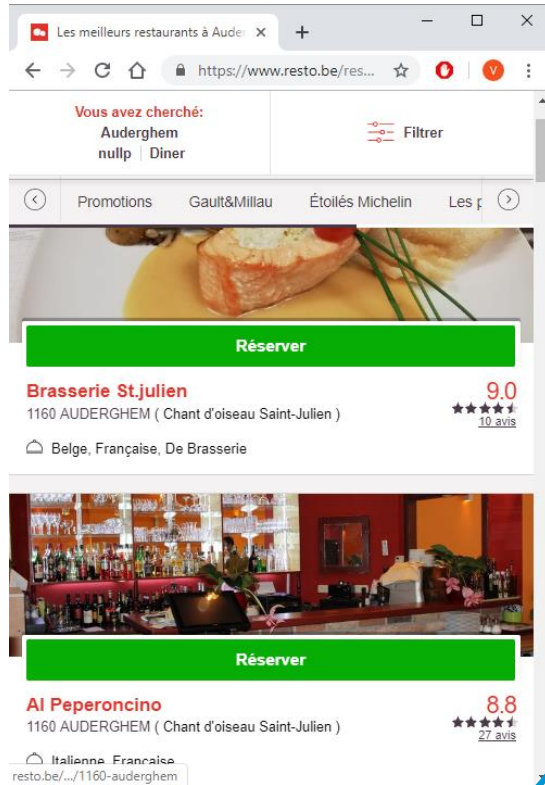
ANALYSE

[https://statbel.fgov.be/sites/default/files/files/documents/Analyse/FR/Webscraping\\_fr.pdf](https://statbel.fgov.be/sites/default/files/files/documents/Analyse/FR/Webscraping_fr.pdf)



- « Uber et Airbnb transmettent les données de leurs utilisateurs au fisc »
- Porte-parole du SPF Finance« [...] nous obtenions les données pertinentes concernant l'économie partagée au moyen de scripts, soit des petits programmes développés par le SPF Finances lui-même »

<https://moneytalk.levif.be/finance-et-bourse/fiscalite/uber-et-airbnb-transmettent-les-donnees-de-leurs-utilisateurs-au-fisc/article-normal-451629.html>

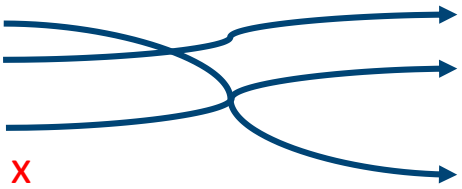


Web Scraping



Name	Adresse
Chez Léon	Rue des bouchers
Comme chez soi	Place Rouppe
La Villa Lorraine	Avenue Vivier d'Oie
Le Chalet de la Forêt	Drève de Lorraine
...	...

KBO	Name	Adresse
0421.614...	Au "Comme chez soi"	Place Rouppe
0414.370...	Villa-Lorraine	Avenue Vivier d'Oie
0455.779...	La sœur du Patron	Chaussée de Wavre
0412.387...	RESTAURANT CHEZ LEON	Rue des bouchers, 18
...	...	...



Entity Linking

- Sites “**html-oriented**” (resto.be & co) : contenu nécessaire dans le code HTML
- Sites “**javascript-oriented**” (Google, Facebook...) : contenu nécessaire généré “à la volée”
- **API** (OpenStreetMap, Google maps...) : contenu direct, structuré

```
<html> (...)
<link href='style.css'
      rel='stylesheet'>
<script src='scripts.js' />
  <span class="text">
    "My Text"
  </span>
  
</html>
```

Index.html



logo.png

```
function myFunction() {
  document.\
  getElementById("demo").\
  innerHTML="A text";
}
```

scripts.js

```
H1 {
  color: white ;
  border: 1px}
.text {
  color: black ;
  font-family: Arial}
```

style.css

- Chargement utile : page principale
- Permet d'être très efficace
- Moins lourd pour le serveur, mais ... plus détectable
- Exemple de Framework : Scrapy (Python)



Scrapy

The image shows a screenshot of the Resto.be website interface. The top navigation bar includes the Resto.be logo, 'Giftcards', and 'Add a restaurant'. The search bar is set to 'Auderghem' with filters for 'Kitchentype' and 'Date'. Below the search bar, it displays 'Restaurants | Brussels | 1160 AUDERGHEM' and 'Best Restaurants in Auderghem'. A list of restaurants is shown, with 'Bangalore' highlighted. The details for 'Bangalore' are shown in a modal window, including the address 'Chaussée de Wavre 1629, 1160 AUDERGHEM', a rating of 'Good 8.0', and contact information '02.662.27.63'. A code editor overlay on the right shows the HTML structure of the search results, with red circles highlighting the href attribute of the restaurant link and the restaurant name 'Bangalore'.

```
<div class="l-equal-height__item m-searchresult_generalinfo">  
  <div class="m-searchresult-info_top">  
    <div class="clearfix m-searchresult-info_items">  
      ::before  
      <a href="http://en.resto.be/restaurant/brussels/1160-auderghem/150081-bangalore/?availabilityHour=&availabilityPersons=&availabilityDate=" class="cw-restaurant-name-link">  
        <div id="cw-restaurant-name" class="m-searchresult-info_name">  
          Bangalore</div>  
        </a>  
      <div class="m-searchresult-info_adress">  
        <!-- <div  
          th:if="{restaurant.get('street') != null and restaurant.get('postbox') != null }"  
          th:text="{restaurant.t.street+' '+restaurant.postbox}">  
          Hoogstraat 5  
        </div--> == $0  
      </div>  
      <a href="/restaurant/auderghem/1160-auderghem" class="h-uppercas">1160 AUDERGHEM</a>  
    </div>  
    <div class="m-searchresult-info_rating">...</div>  
    <div class="h-clearfix m-searchresult_gaultmilhau-michelin">  
    </div>  
  </div>...</div>  
  ::after
```

```
<html> (...)  
<script>  
function loadContent(){  
  document.\  
    getElementById("body").\  
    innerHTML="MyContent";  
}  
</script>  
<body onload='loadContent()'/>  
</html>
```

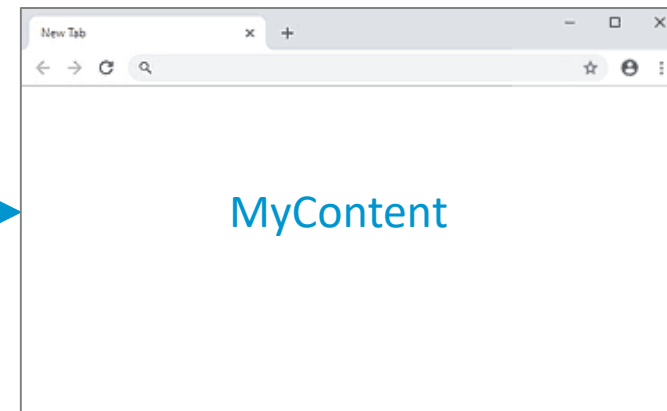
Index.html



```
<html> (...)  
<body onload='loadContent()>  
MyContent  
</body>  
</html>
```

Dynamic source

- On doit "simuler" le comportement d'un utilisateur (~RPA) avec un browser
- Possible sans affichage ("headless")
- Demande beaucoup plus de temps/ressources (chargement complet)
- Souvent difficile pour un script de savoir que la page est "chargée/prête"
- Exemple de Framework : Selenium



- Scrapy : framework en Python pour le webscraping
- À fournir :
  - Un “parseur” (fonction Python) par (type de) page
  - Une (liste d’) URL
- Scrapy :
  - Charge et schedule (en parallèle) les pages → pas de chargement inutile
  - Parse le HTML → On reçoit un objet “arborescent”
  - Est hautement paramétrable : débit maximal (délai, # threads...), respect ou non de “robots.txt”, rotation de proxies, ...



- Chaque “parseur” est un “générateur” (ou itérateur)
- Générateur : fonction qui **génère une liste**, mais en créant les éléments **au fur et à mesure** qu’on les consomme
- Un parseur peut générer :
  - Des **données** collectées :
    - Typiquement une ligne du tableau résultat
    - Plus généralement, un dictionnaire « {"key1" : "value1", "key2" : "value2",...} »
  - Une **URL** (+ le parseur associé) qui sera rajoutée au scheduler

## Quotes to Scrape

"The world as we have created it is a process changing our thinking."

by [Albert Einstein](#) (about)

Tags: [change](#) [deep-thoughts](#) [thinking](#) [world](#)

"It is our choices, Harry, that show what we truly are."

by [J.K. Rowling](#) (about)

Tags: [abilities](#) [choices](#)

"There are only two ways to live your life. One is as though everything is a miracle."

by [Albert Einstein](#) (about)

Tags: [inspirational](#) [life](#) [live](#) [miracle](#) [miracles](#)

"The person, be it gentleman or lady, who has no sense of humor is intolerably stupid."

by [Jane Austen](#) (about)

Tags: [aliteracy](#) [books](#) [classic](#) [humor](#)

```
<div class="quote" (...)>
  <span class="text" (...)>
    "The world as we have created it (...). "
  </span>
  <span>
    by <small class="author" (...)>Albert Einstein</small> (...)
  </span>
  <div class="tags"> (...) </div>
</div>
```

```
<div class="quote" (...)>
  <span class="text">"It is our choices, Harry (...)"</span>
  <span>
    by <small class="author" (...)>J.K. Rowling</small> (...)
  </span>
  (...)
</div>
(...)
<li class="next">
  <a href="/page/2/">Next (...)</a>
</li>
```

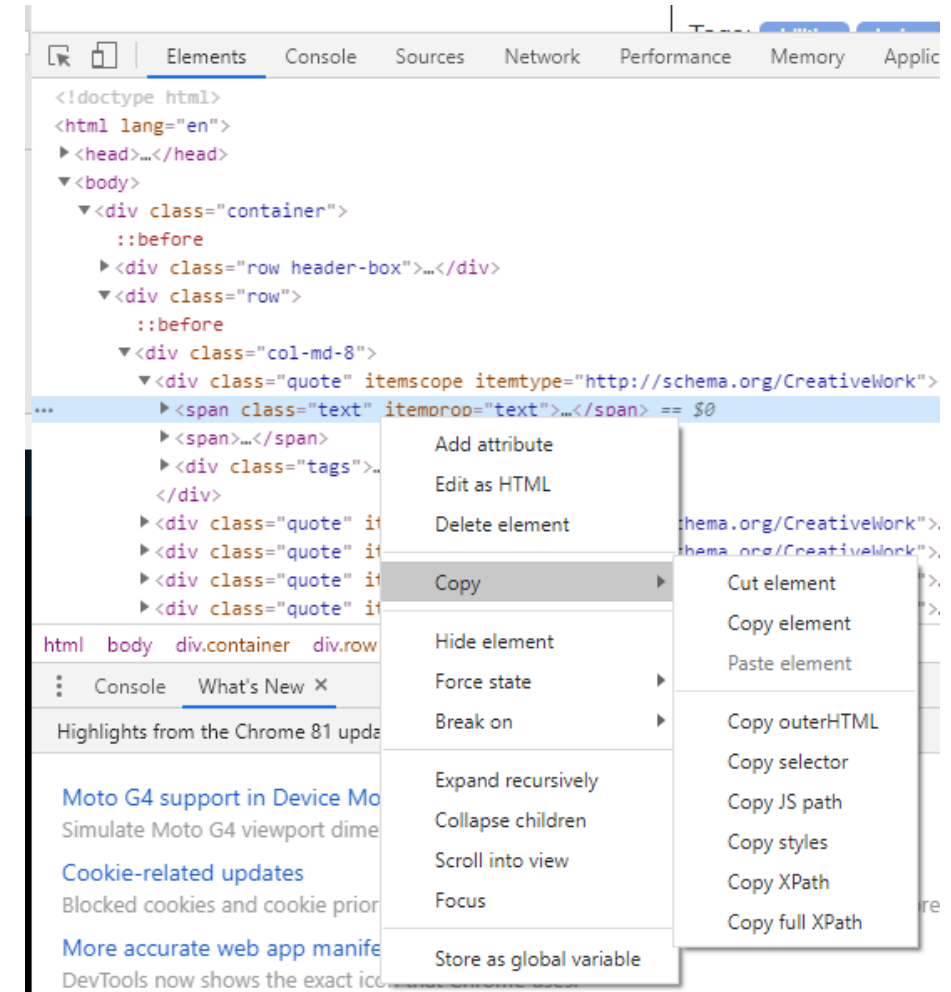
```
<div class="quote" (...)>
  <span class="text" (...)>
    "The world as we have created it (...). "
  </span>
  <span>
    by <small class="author" (...)>Albert Einstein</small> (...)
  </span>
  <div class="tags">(...)</div>
</div>

</div>
(...)
<li class="next">
  <a href="/page/2/">Next (...)</a>
</li>
```

```
def parse(self, response):
    for quote in response.css('div.quote'):
        yield {
            'text': quote.css('span.text::text').get(),
            'author': quote.xpath('span/small/text()').get(),
        }

    next_page = response.css('li.next a::attr("href)').get()
    if next_page is not None:
        yield response.follow(next_page, self.parse)
```

- Principale difficulté : identification des tags
  - Souvent bas dans l'arborescence
  - Difficiles à rendre "génériques"
- Deux pistes :
  - Chrome Developer tools
  - Extension Chrome "webscraper.io"



- POC :
  - Collecter une liste de restaurants
  - Les lier aux infos KBO
- Le POC a été fait en ciblant **Bruxelles**
- Les données suivantes ont été collectées :
  - Nom(s) et adresse de l'entreprise
  - Numéro de téléphone → quasi que des "02" (fixes)
  - Nombre de visites/commentaires
- Pas de données "personnelles" (?)
- Données collectées sur **Resto.be : 1.516 records**

```
import
```

```
class
```

```
    def
```

```
        for    in
            yield
```

```
    def
```

```
        for                in
```

```
            yield
```

```
    if
```

```
        yield
```

`def`

`yield`

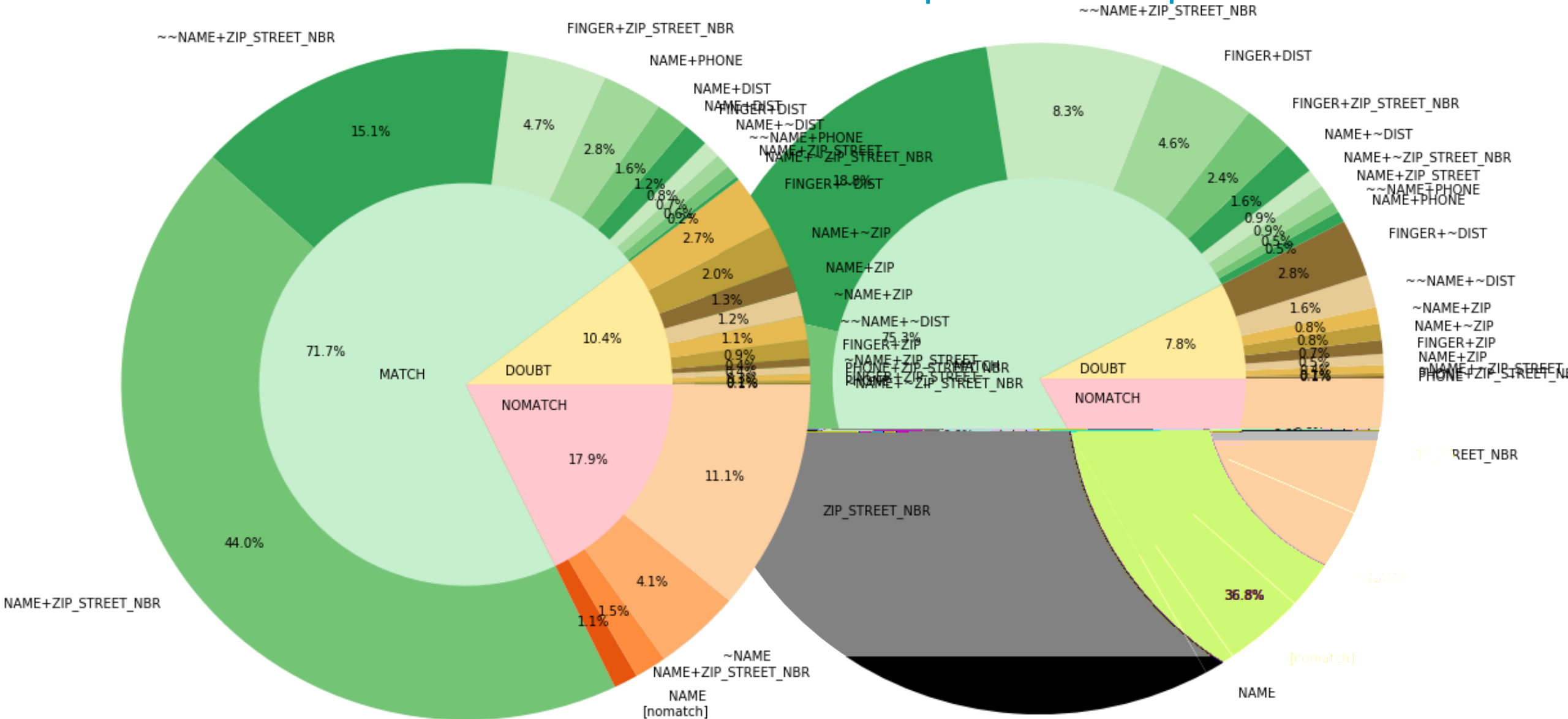
- On extrait de la KBO une liste “similaire” de
  - Noms (et variantes)
  - Téléphone
  - Adresses (toutes les UTE)
- Avec comme limitations :
  - NACE\_CODE “root” = 5
  - Adresse sur ~ Bruxelles (code postal commence par 1)
  - → 410,000 lignes pour 140.000 entreprises
- **Idéalement** : on compare chaque ligne des deux listes (site web vs KBO)
- Dans la **pratique** : impossible → technique de “keying”

	Flag	Définition	Exemple
Dénomination	NAME	“Exactement” égal	“Domino Pizza” vs “domino pizza”
	FINGER	“Fingerprint” égal	... vs “Pizza Domino”
	~NAME	Similarité (jaro) $\geq 90\%$	... vs “Domino Pizzas”
	~~NAME	$80\% < \text{Similarité} < 90\%$	... vs “Dominique Pizzeria”
Adresse	ZIP_STREET_NBR	Correspondance complète	
	ZIP_STREET	Code postal + rue	
	ZIP	Code postal	
	DIST	Coord. géo. proches	
Tél	PHONE	Correspondance	“02 787 57 11” vs “+3227875711”

- **MATCH :**
  - NAME + PHONE
  - NAME + ZIP\_STREET
  - ~~NAME + ZIP\_STREET\_NBR
  - ...
- **DOUBT :**
  - ~NAME + ZIP\_STREET
  - FINGER + ZIP\_STREET
  - ...
- **NOMATCH :**
  - ZIP\_STREET\_NBR
  - NAME
  - Le reste

# resto.be

# OpenStreetMap



	A	B	C	D	L	N	O	T	U			
1717					MATCH	NAME+ZIP_STREET_NBR	Atelier des Degres	Avenue du Prince de Ligne	14			
1718	L'atelier En Ville	Rue Haute	64	1000	MATCH	NAME+ZIP_STREET_NBR	Atelier En Ville	Rue Haute	64			
1719					MATCH	NAME+ZIP_STREET_NBR	L'ATELIER EN VILLE	Rue Haute	64			
1720	L'atelier Européen	Franklinstraat	28	1000	MATCH	NAME+ZIP_STREET_NBR	L'ATELIER EUROPEEN	Franklinstraat	28			
1721					MATCH	NAME+ZIP_STREET_NBR	L'Atelier europeen	Franklinstraat	28			
1722					MATCH	NAME+ZIP_STREET_NBR	L'ATELIER EUROPEEN	Franklinstraat	28			
1723	L'atlantide	Rue Franklin	73	1000	MATCH	NAME+ZIP_STREET_NBR	L'ATLANTIDE	Rue Franklin	73			
1724	L'auberge Des Maïeurs By La	Parvis Saint Pierre	1	1150	MATCH	NAME+ZIP_STREET_NBR	AUBERGE DES MAIEURS	Parvis Saint-Pierre	1			
1725	Finca Spl				MATCH	NAME+ZIP_STREET_NBR	L'Auberge des Maieurs by La Finca	Parvis Saint-Pierre	1			
1726					MATCH	NAME+ZIP_STREET_NBR	L'AUBERGE DES MAIEURS	Parvis Saint-Pierre	1			
1727	L'aurige	Avenue de Tervueren	24	1040	MATCH	NAME+ZIP_STREET_NBR	L'AURIGE	Avenue de Tervueren	24 A			
1728	L'autre Cantina	Chaussée de Vleurgat	117	1000	DOUBT	FINGER+ZIP	LA CANTINA	Rue du Jardin des Olives	13			
	LA CANTINA	Rue Antoine Dansaert	22						DOUBT FINGER+ZIP			
	LA CANTINA	Olivetenhof	13						DOUBT FINGER+ZIP			
	LA CANTINA	Rue Saint-Gery	31						DOUBT FINGER+ZIP			
	LA CANTINA	Rue Antoine Dansaert	22						DOUBT FINGER+ZIP			
	LA CANTINA	Antoine Dansaertstraat	22						DOUBT FINGER+ZIP			
	LA CANTINA	Antoine Dansaertstraat	22						DOUBT FINGER+ZIP			
	LA CANTINA	Sint-Goriksstraat	31						DOUBT FINGER+ZIP			
JBR	L'AUTRE ORFEO	Rue Edith Cavell	15			1736	L'autre Orfeo	Rue Edith Cavell	15	1180	MATCH	NAME+ZIP_STREET_N
JBR	L'Ecailler du Palais Royal	Rue Bodenbroek	18			1743	L'ecailler Du Palais Royal	Rue Bodenbroek	18	1000	MATCH	NAME+ZIP_STREET_N
_NBR	L'ECHALOTTE	Rue aux Fleurs	8			1744	L'echalote	Rue aux fleurs	8	1000	MATCH	NAME+ZIP_STREET
						1745	L'ecole D'a	Avenue Marie		1030	NOMATCH	[nomatch]
	Les Artistes	Avenue de la Brabanconne	40			1746	L'entree Des Artistes	Grote Zavel	42	1000	DOUBT	FINGER+ZIP
	Les Artistes	Brabanconnelaan	40			1747					DOUBT	FINGER+ZIP
	CAFE LES ARTISTES	Place du Jeu de Balle	19			1748					DOUBT	FINGER+ZIP
	CAFE LES ARTISTES	Vossenplein	19			1749					DOUBT	FINGER+ZIP
NBR	L'EPICERIE	Rue du Page	66			1750	L'epicerie	Rue du Page	66	1150	MATCH	NAME+ZIP_STREET
	EQUI	Langeweide				1751	L'equ	Avenue Henri		1140	NOMATCH	NAME
	EQUI	Parvis Saint-Pierre				1752					NOMATCH	NAME
	EQUI	Parvis Saint-Pierre				1753					NOMATCH	NAME
	DISP	Av. H.				1754					NOMATCH	ZIP_STREET_NBR
	EQUILIBRIUM	Sint-Pietersvoorlein	1			1755					NOMATCH	NAME

Disclaimer : je ne suis pas juriste !

- En général : va à l'encontre des **CGU**
- Exemple resto.be [1]: « Le contenu (...) est protégé par un droit sui generis qui (...) interdi[t] toute extraction et/ou réutilisation (...) de ce contenu. »
- Un **problème** ?
  - Pour certains : non tant qu'on ne clique pas sur « J'accepte les CGU » (jurisp. USA)
  - Pour d'autres : oui dès qu'on navigue sur le site web
- Selon [2], **utilisation** condamnée, pas le scraping
- Distinction à faire [3] :
  - **Droit pénal** : condamne l'introduction frauduleuse
  - Droit de la **propriété intellectuelle/de la concurrence** : condamne l'utilisation
- **GDPR** : Mon client en est-il exempt ? Qu'est-ce un donnée personnelle ?

[1] <https://www.resto.be/privacy-policy>

[2] <https://islean-consulting.fr/fr/transformation-digitale/scraping-pages-web-legal/>

[3] <https://www.actualitesdudroit.fr/browse/tech-droit/start-up/9404/le-web-scraping-une-technique-d-extraction-legale>

- Technique accessible, peu de moyens nécessaires
- Demande beaucoup de “bricolage” (surtout en “javascript-oriented”)
- Problème de pérennité : aucune garantie de “stabilité”
- Considérer les aspects juridiques/éthiques dès le début
- Collecter n’est pas finalité, il faut pouvoir exploiter les données
  
- POC :
  - Collecte : efficace pour l’horeca, secteur très “visible”
  - Reste à prouver pour des secteurs plus “underground” (nail shops, car washes...)
  - Linking : 70-75 % commerces trouvés dans la KBO avec haute fiabilité