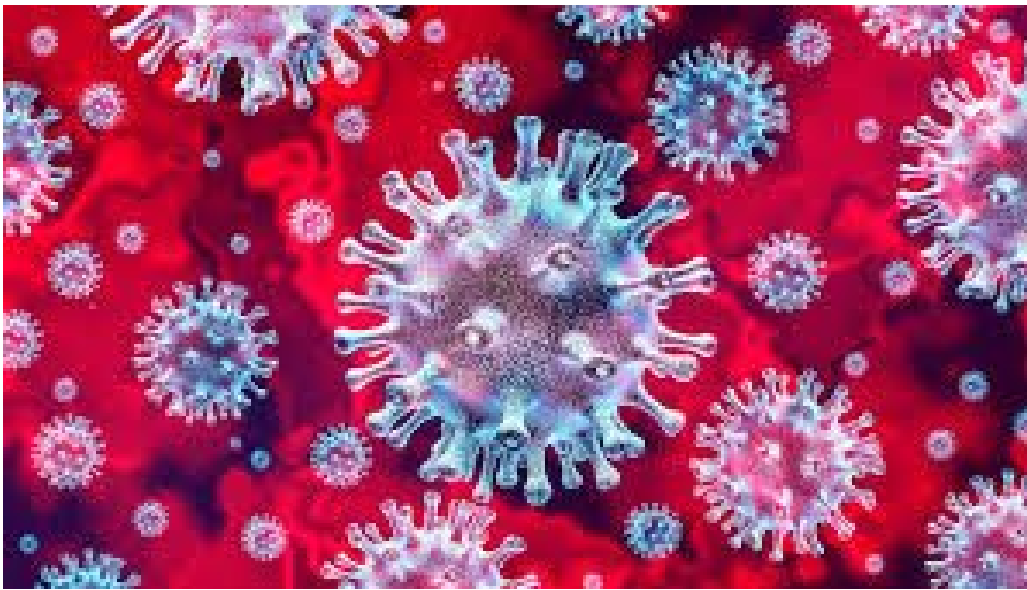


# Projet Big Data Ecosystem



BRUNO Charlène  
CHEONG Loïc

ECE PARIS 2020/2021  
ING5 BDA Gr 02

## Projet Technique : Ingestion quotidienne et automatisée de CSV

**objectif:** Analyser les données journalières liées au covid19 en France

**jeu de données:** données COVID19 journalières disponibles sur le site data.gouv (voir Annexes)

Voici les différentes étapes de notre démarche :

**1ère étape:** Récupérer les données à partir d'un site internet et les transformer

Dans un premier temps, nous avons programmé un script Python nous permettant de récupérer les fichiers CSV à partir de l'url du site data.gouv. Une fois les fichiers CSV récupérés, nous avons dû effectuer des transformations pour les rendre exploitables telles que le remplacement des séparateurs “;” par des “,” ou encore la suppression de guillemets superflus. Enfin, nous avons pu stocker nos données en local dans de nouveaux CSV nettoyés.

Ensuite, nous avons exécuté notre script sur le cluster Hadoop afin de récupérer le fichier CSV stocké dans un container de Yarn pour le mettre sur HDFS.

**2ème étape:** Stocker des données dans une BDD Hive (table externe +ORC et fichier parquet) à partir d'un CSV

Une fois nos données récoltées et nettoyées dans un fichier CSV, nous avons écrit un script HQL afin de pouvoir créer une table externe structurée pour effectuer nos analyses. On remplit cette table avec les données du fichier CSV préalablement nettoyées.

D'autre part, nous avons également créé une table ORC, de même schéma que la table externe, dans laquelle nous avons injecté les données de la table externe créée précédemment.

Nos 2 tables sont maintenant prêtes à être utilisées et nous pouvons faire des requêtes basiques dessus pour vérifier leur bon fonctionnement.

Nous avons programmé un dernier script pour stocker nos données dans un fichier parquet qui sera utilisé plus tard par spark dans Zeppelin.

```
jdbc:hive2://zoo-1.au.adaltas.cloud:2181,z> show tables in ece_2020_fall_bda_2;
```

tab_name	tab_name
bastien_nyc_drivers_ext	bastien_nyc_drivers_ext
charlene_nyc_drivers	charlene_nyc_drivers
charlene_nyc_drivers_ext	charlene_nyc_drivers_ext
charlenebruno_covid19	charlenebruno_covid19
charlenebruno_nyc_drivers_ext	charlenebruno_nyc_drivers_ext
henintsoa_nyc_drivers	henintsoa_nyc_drivers
henintsoa_nyc_drivers_ext	henintsoa_nyc_drivers_ext
imdb_name_basics	imdb_name_basics
imdb_title_basics	imdb_title_basics
imdb_title_crew	imdb_title_crew
imdb_title_ratings	imdb_title_ratings
lambert_nyc_drivers_ext	lambert_nyc_drivers_ext
loic_nyc_drivers	loic_covid19
loic_nyc_drivers_ext	loic_covid19_parquet
maxime_nyc_drivers_ext	loic_nyc_drivers
prossignol_nyc_drivers_ext	loic_nyc_drivers_ext
prossignol_nyc_drivers_man	maxime_nyc_drivers_ext
srahli_nyc_drivers	prossignol_nyc_drivers_ext
srahli_nyc_drivers_ext	prossignol_nyc_drivers_man
wangjames_nyc_drivers_ext	srahli_nyc_drivers
	srahli_nyc_drivers_ext
	wangjames_nyc_drivers_ext

**3ème étape:** Traitement automatisé pour mettre à jour les données avec oozie


En troisième lieu, nous avons utilisé Oozie pour ordonnancer l'exécution des scripts programmés précédemment (script python et scripts hql pour la création des tables Hive et script pour la création du fichier parquet). Oozie permet d'orchestrer les jobs Hadoop. Nous avons donc créé trois actions dans le workflow, chacune exécutant le script correspondant.

**4ème étape:** Planification de Oozie avec Cron

Enfin, nous avons utilisé Cron pour planifier et automatiser l'exécution des jobs sur Oozie tous les 20h chaque jour. Pour ce faire, nous avons ajouté un fichier coordinator.xml permettant de spécifier la fréquence d'exécution des jobs sur Oozie.

[Oozie Documentation](#)

Oozie Web Console

Workflow Jobs										Coordinator Jobs										Bundle Jobs										SLA										System Info										Instrumentation										Metrics										Settings																			
 All Jobs										Active Jobs										Done Jobs										Custom Filter▼																																																											
Job Id										Name										Status										User										Group										Frequency										Unit										Started										Next Materialization									
1 0000174-201011090406050-oozie-oozi-C										Coordinator_APP										RUNNING										I.cheong-ece																				0 20 ***										CRON										Wed, 30 Dec 2020 18:00:00 ...										Wed, 30 Dec 2020 20:00:00 ...									

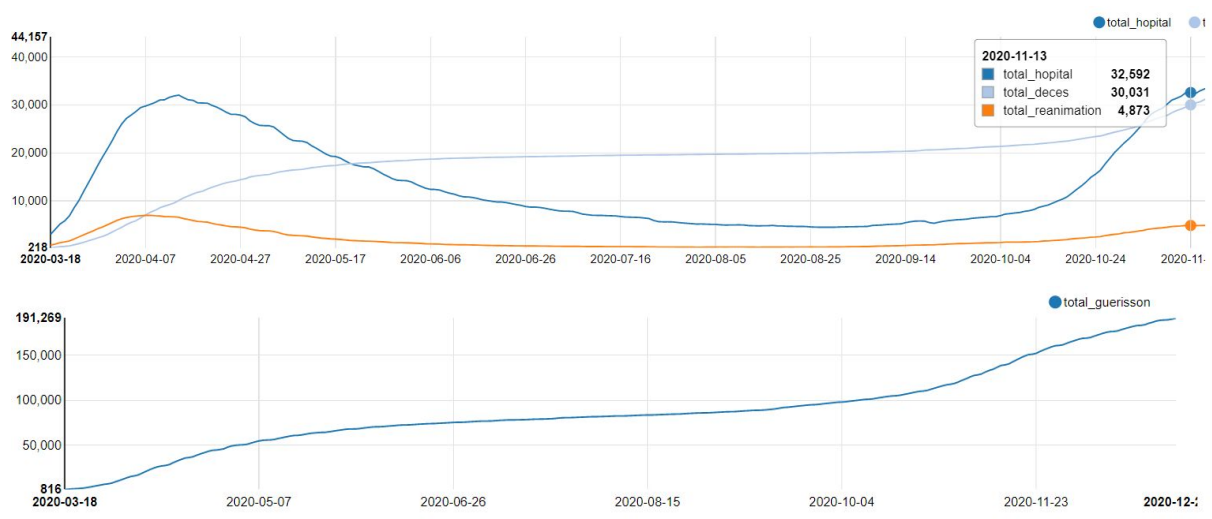
## 5ème étape: Afficher les données sur des graphiques à l'aide de Spark avec Zeppelin

Pour finir, nous avons créé un notebook Zeppelin permettant d'afficher les données récoltées sur la pandémie du covid19. Cela permet de surveiller son évolution grâce à la visualisation de statistiques récoltés à l'aide de requêtes SQL sur notre table au format parquet.

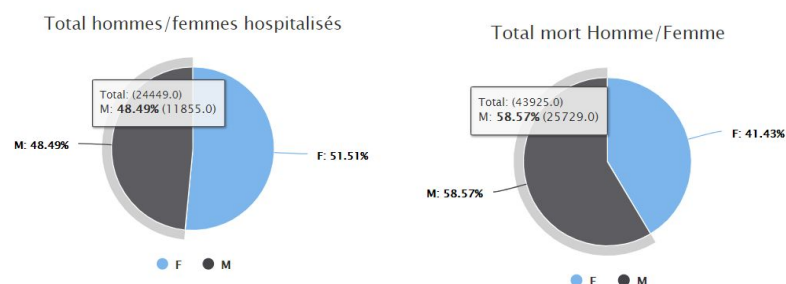
Les données de data.gouv sont des données cumulatives, on ne peut donc effectuer que des analyses globales.

Voici la liste des requêtes effectuées:

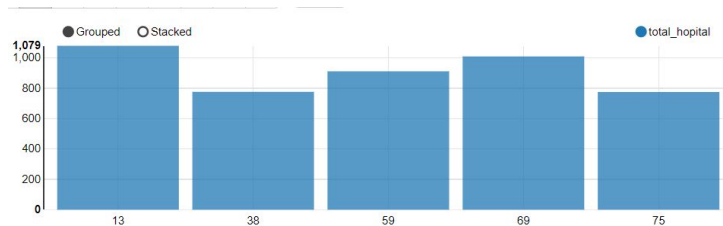
- Evolution du nombre d'hospitalisation, de réanimation, de guérison et de mort par jour sans différenciation de sexe: linechart



- Calcul du nombre d'hospitalisations et de décès total par sexe (homme ou femme): piechart



- Top 5 des départements les plus touchés en termes d'hospitalisation: histogramme



## Bibliographie

- <https://github.com/adaltas/ece-bigdata-2020-fall>
- [Oozie - \(apache.org\)](https://oozie.apache.org/)
- <https://www.adaltas.com/fr/2018/03/07/executer-du-python-dans-un-workflow-oozie/>
- [How to save a file in hadoop with python - Stack Overflow](https://stackoverflow.com/questions/48481421/how-to-save-a-file-in-hadoop-with-python)
- <https://www.data.gouv.fr/fr/datasets/donnees-hospitalieres-relatives-a-lepidemie-de-covid-19/>
- <http://zep-1.au.adaltas.cloud:9995/#/notebook/2FWHFDFTV>