Projet BigData

GDELT: les évènements marquants de l'année

Yujia Fan, Thomas Loiseau,
Liliane Manitchoko, Kovila Pauvaday

15/02/2023

Sommaire

GDELT

- *The Global Database of Events, Language, and Tone (GDELT)*, est une base de données d'événements géopolitiques, automatiquement obtenues et traduites en temps réel à partir de centaines de sources d'information en 65 langues.
- Plus qu'une simple base de données historique, GDELT est une base de données vivante, mise à jour toutes les 15 minutes.

Objectif

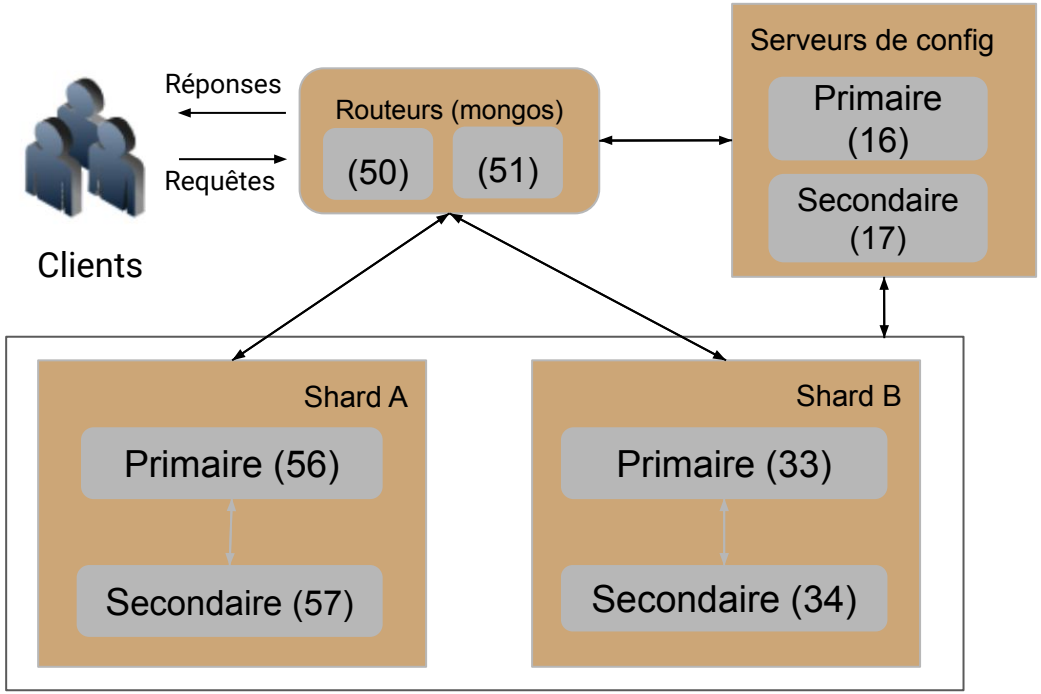
Proposer un système de stockage distribué, résilient et performant pour les données de GDELT permettant d’étudier l’année 2022 via son impact média

Technologie utilisée

MongoDB:

- **Scalabilité** pour le stockage et la gestion des grandes bases de données
- **Flexibilité** pour l'évolution de la structure ou du modèle de données
- **Performance** pour les requêtes
- **Tolérance** aux pannes
- **Fonctionnalités: agrégation de données et indexation rapide** pour l'analyse de données en temps réel
- **Simplicité d'utilisation et intégration facile avec de nombreux langages de programmation** pour le développement d'application web et mobile

Architecture: Configuration du cluster MongoDB



Toutes le machines fonctionnent sur le port 27017
Problème : pas assez de machines par shard pour être résistant à la panne

Installation du Cluster

Nous avons suivi le tutoriel d'installation de Mongodb :

https://www.mongodb.com/docs/manual/tutorial/deploy-shard-cluster/

Attention par défaut, mongodb n’a pas accès en lecture et écriture aux fichiers des logs et de stockage des bases de données

```
sudo chown -R mongodb:mongodb /var/lib/mongodb
sudo chown -R mongodb:mongodb /var/log/mongodb
sudo chown -R mongodb:mongodb /tmp
```

Shard Key

- cardinalité élevée
- fréquence basse
- pas de changement monotone
- facilité des requêtes/agrégations

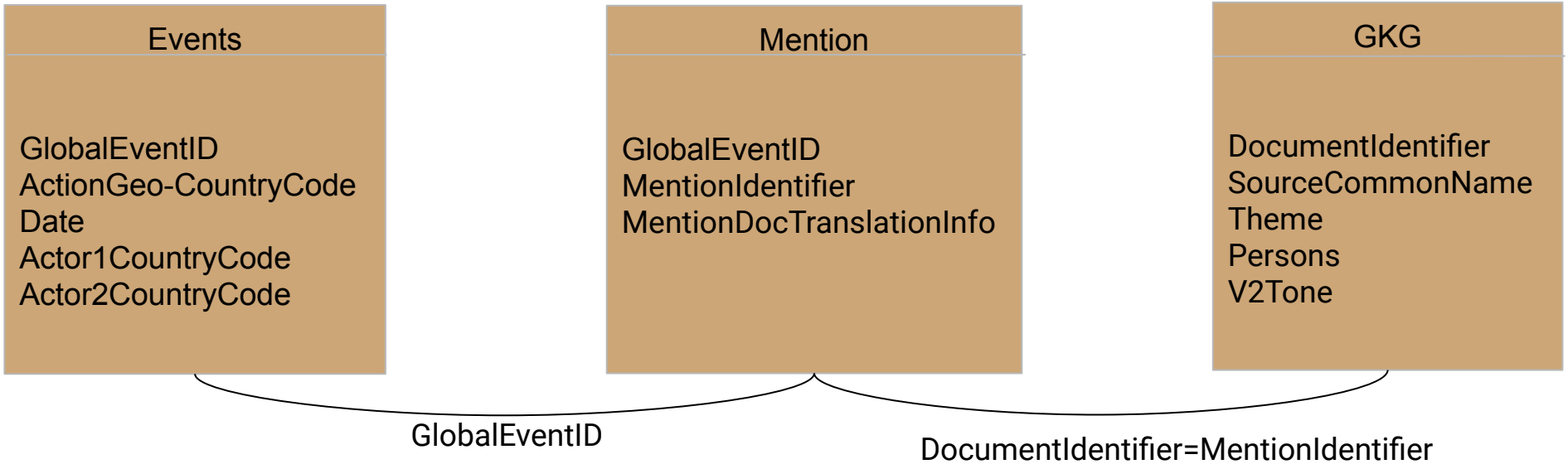
```
sh.shardCollection("gdelt.mention", { MentionIdentifier: "hashed", GLOBALEVENTID: 1 } )

sh.enableSharding("gdelt")
sh.startBalancer()
```

```
Totals
{
  data: '66.17GiB',
  docs: 39412291,
  chunks: 4,
  'Shard shard_B': [
    '49.96 % data',
    '49.97 % docs in cluster',
    '1KiB avg obj size on shard'
  ],
  'Shard shard_A': [
    '50.03 % data',
    '50.02 % docs in cluster',
    '1KiB avg obj size on shard'
  ]
}
```

Données nécessaires pour nos requêtes

1. afficher le nombre d'articles/évènements qu'il y a eu pour chaque triplet (jour, pays de l'évènement, langue de l'article).
2. pour un pays donné en paramètre, affichez les évènements qui y ont eu place triées par le nombre de mentions (tri décroissant); permettez une agrégation par jour/mois/année
3. pour une source de donnés passée en paramètre (gkg.SourceCommonName) affichez les thèmes, personnes, lieux dont les articles de cette sources parlent ainsi que le nombre d'articles et le ton moyen des articles (pour chaque thème/personne/lieu); permettez une agrégation par jour/mois/année.
4. étudiez l'évolution des relations entre deux pays (specifies en paramètre) au cours de l'année. Vous pouvez vous baser sur la langue de l'article, le ton moyen des articles, les themes plus souvent citées, les personalités ou tout element qui vous semble pertinent.

Modélisation: Relation des données de GDELT

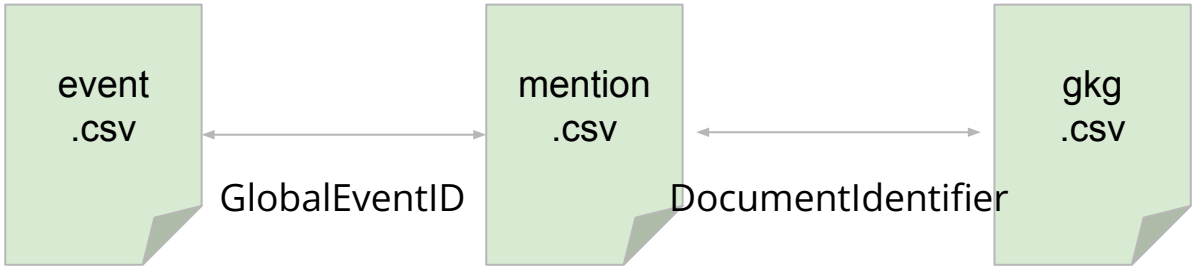


Modélisation: Modèle de données utilisé

```
db.mention.findOne()

{
  _id: ObjectId("63e2a7521630ca6796f3a6d0"),
  GLOBALEVENTID: 1021421880,
  MentionTimeDate: Long("20220101000000"),
  MentionIdentifier: 'http://www.pasienis.lt/lit/Ccfec91be70bad25853d5e30aedf1eb8',
  event: {
    GLOBALEVENTID: 1021421880,
    SQLDATE: '20211225',
    Actor1Geo_CountryCode: 'LH',
    Actor2Geo_CountryCode: NaN,
    ActionGeo_CountryCode: 'LH'
  },
  article: {
    GKGRECORDID: '20220101000000-T784',
    DATE: Long("20220101000000"),
    SourceCommonName: 'pasienis.lt',
    DocumentIdentifier:
'http://www.pasienis.lt/lit/Ccfec91be70bad25853d5e30aedf1eb8',
    Themes: [
      'SMUGGLING',
      'BORDER',
      'ARREST',
      'EPU_POLICY_TAX',
      'EPU_CATS_TAXES',
      'WB_696_PUBLIC_SECTOR_MANAGEMENT',
      'WB_840_JUSTICE',
      'TAX_ETHNICITY_BELARUSIAN',
      ''
    ],
    Persons: NaN,
    V2Tone: -2.8735632183908,
    TranslationInfo: 'lit'
  }
}
```

Chargement des Données - stratégie 1

- Outils: **pyspark, yarn**
- Stratégie chargement: jointure



- **Ressources csv:**
  - hdfs: fichiers trop petits
  - sparkContext (driver, executors)
- 1 jour de données (1 machine) → 50 mins
- 1 an de données (6 machines) → **50 h**
- **arrêt imprévu du chargement:** recommencer

Chargement des Données - stratégie 2

- Outils: **pymongo**
- Stratégie chargement: (par mois)
  - chargement **mentions**,
  - update des documents mentions avec **events**
  - update des documents mentions avec **articles**

- 6 machines en parallèle

- Temps (1 mois):
  - mention: 20 mins
  - events: 60 mins
  - articles: 90 mins

Temps (1 ans) : **6 h**

- Volume: 66 Gb
- Arrêt imprévu: Problématique Doublons pour mentions

Démonstration

Nous avons eu un problème la veille de la présentation, nous avons **perdu toute la collection de données.**

Pour faire la démo nous avons rechargé ce matin l'ensemble des documents de **janvier à mars**

Totals

```
{
  data: '66.17GiB',
  docs: 39412291,
  chunks: 4,
  'Shard shard_B': [
    '49.96 % data',
    '49.97 % docs in cluster',
    '1KiB avg obj size on shard'
  ],
  'Shard shard_A': [
    '50.03 % data',
    '50.02 % docs in cluster',
    '1KiB avg obj size on shard'
  ]
}
```

'19.55GiB'
11453940

Question 1 :

Afficher le nombre d'articles/évènements qu'il y a eu pour chaque triplet (jour, pays de l'évènement, langue de l'article).

```
mention_collection.create_index([("event.SQLDATE", 1),("event.ActionGeo_CountryCode", 1),
                                 ("article.TranslationInfo", 1) ], unique=False )
```

```
%%time

def get_nb_article_event(collection):
    result = collection.aggregate([
        { "$group" : { "_id" : { "date" : "$event.SQLDATE",
                                 "pays" : "$event.ActionGeo_CountryCode",
                                 "langue":"$article.TranslationInfo" },
                       "unique_article": {"$addToSet": "$MentionIdentifier"},
                       "unique_event": {"$addToSet": "$GLOBALEVENTID"}
                      }
        },
        { "$project" : { "date" : "$_id.date",
                         "pays" : "$_id.pays",
                         "langue":"$_id.langue",
                         "nb_distinct_article" : {"$size":"$unique_article"},
                         "nb_distinct_event" : {"$size":"$unique_event"}
                        }
        }
    ])
    return pd.DataFrame(list(result))

get_nb_article_event(mention_collection)
```

Question 2 :

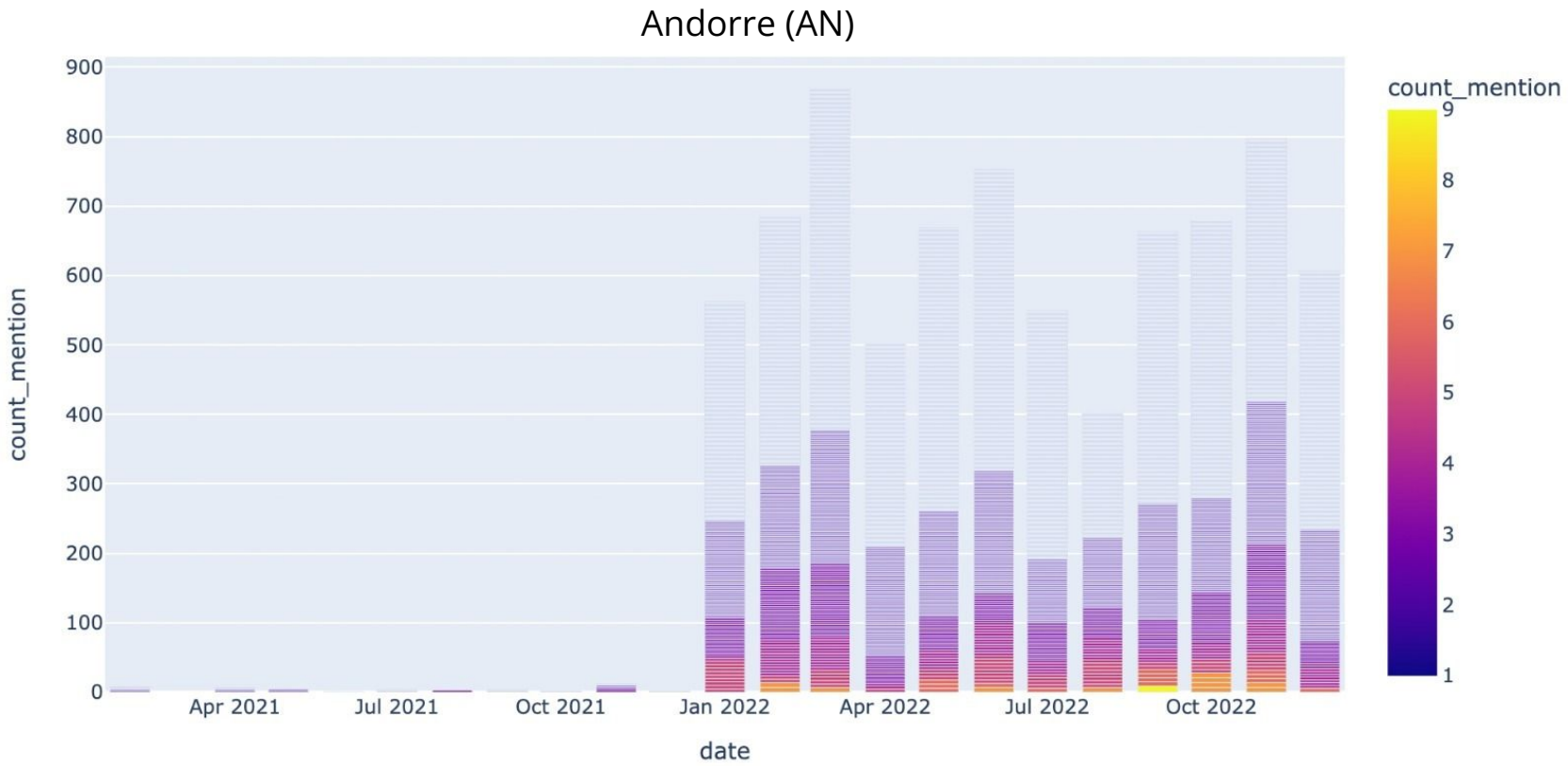
Pour un pays donné en paramètre, affichez les évènements qui y ont eu place triés par le nombre de mentions (tri décroissant); permettez une agrégation par jour/mois/année

```
mention_collection.create_index([("event.ActionGeo_CountryCode", 1)], unique=False )

mention_collection.create_index([("event.GLOBALEVENTID", 1),("event.SQLDATE", 1) ], unique=False )
```

```
%%time

def find_events_from_country(collection, country="FR", time_granularity="day"):
    date_format="%Y-%m-%d"
    if time_granularity == "month":
        date_format="%Y-%m"
    if time_granularity == "year":
        date_format="%Y"

    result = collection.aggregate([
        { "$match" : { "event.ActionGeo_CountryCode" : country } },
        { "$group" : { "_id" : {"eventID": "$event.GLOBALEVENTID",
                                "date":{ "$dateToString": { "format": date_format, "date":{"$dateFromString":{
                                    "dateString": "$event.SQLDATE", "format": "%Y%m%d"}}}}
                               },
                       "count_mention":{ "$sum": 1}
                      }
        },
        { "$project" : {"_id" : 0,
                        "eventID":"$_id.eventID",
                        "date": "$_id.date",
                        "count_mention": { "$sum": "$count_mention"}
                       }
        },
        {"$sort": {"count_mention": -1}}
    ])
    return pd.DataFrame(list(result))

df_events_from_country = find_events_from_country(mention_collection, country="FR", time_granularity="month")
df_events_from_country
```

Question 2 :

Sur toute l'année



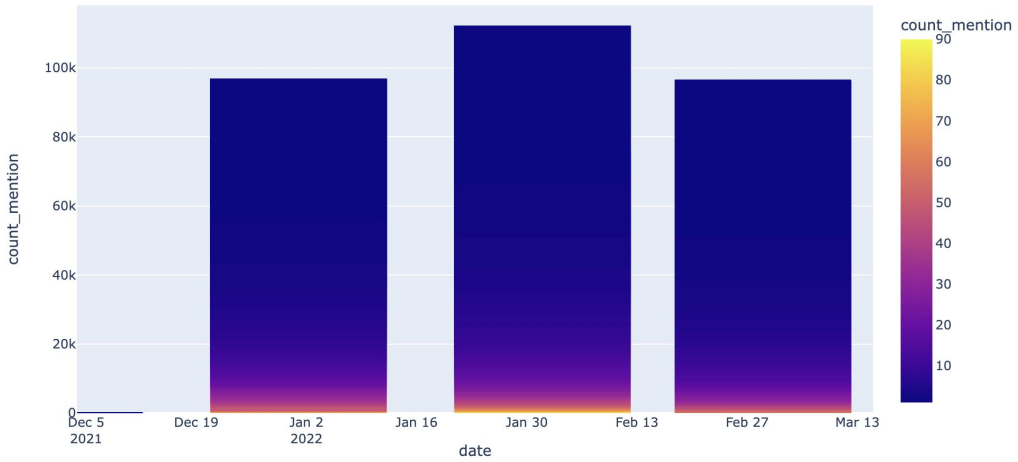
Question 2 :

Sur Janvier à Mars

France (FR)

```
CPU times: user 1.48 s, sys: 655 ms, total: 2.13 s
Wall time: 34.4 s
```

	eventID	date	count_mention
0	1029352307	2022-02	90
1	1024069096	2022-01	86
2	1030039456	2022-02	85
3	1029996204	2022-02	83
4	1026941723	2022-02	81
...	...	...	...
173169	1032448462	2022-03	1
173170	1022545298	2022-01	1
173171	1036899992	2022-03	1
173172	1027599449	2022-02	1
173173	1027023738	2022-02	1



Question 3 :

Pour une source de donnés passée en paramètre (gkg.SourceCommonName) affichez les thèmes, personnes, lieux dont les articles de cette sources parlent ainsi que le nombre d'articles et le ton moyen des articles (pour chaque thème/personne/lieu); permettez une agrégation par jour/mois/année.

```
mention_collection.create_index([("article.SourceCommonName", 1)], unique=False )
```

```
%%time

def article_from_source(collection, source_name= "pardubickenovinky.cz" , time_granularity="day"):
    date_format="%Y-%m-%d"
    if time_granularity == "month":
        date_format="%Y-%m"
    if time_granularity == "year":
        date_format="%Y"

    result = collection.aggregate([
        { "$match" : { "article.SourceCommonName" : source_name } },
        { "$unwind": { "path": "$article.Themes", "preserveNullAndEmptyArrays": False } },
        { "$unwind": { "path": "$article.Persons", "preserveNullAndEmptyArrays": False } },
        { "$group" : { "_id" : {"date":{ "$dateToString": { "format": date_format, "date": {"$dateFromString":{
            "dateString": {"$toString": {"$toLong" : "$article.DATE" }}, "format": "%Y%m%d%H%M%S"}}}},
                                "theme":"$article.Themes",
                                "person":"$article.Persons",
                                "lieu": "$event.ActionGeo_CountryCode"
                               },
                       "unique_article": {"$addToSet": "$article.GKGRECORDID"},
                       "V2Tone_list": { "$push" : "$article.V2Tone"}
                      }
        },
        { "$project" : {"_id" : 0,
                        "date": "$_id.date",
                        "theme":"$_id.theme",
                        "person":"$_id.person",
                        "lieu": "$_id.lieu",
                        "nb_distinct_article" : {"$size":"$unique_article"},
                        "mean_V2Tone" : { "$avg" : "$V2Tone_list"}
                       }
        }
    ])
    return pd.DataFrame(list(result))

df_article_from_source = article_from_source(mention_collection, source_name="lefigaro.fr" , time_granularity="month")
df_article_from_source
```

Question 3 :

Sur toute l'année

pardubickenovinky.cz avec une agrégation au mois

```
CPU times: user 8.54 s, sys: 1.4 s, total: 9.94 s
Wall time: 31.5 s
```

	_id	month	theme	person	nb_distinct_article	mean_V2Tone	lieu
0	{'date': None, 'theme': nan, 'person': nan}	None	NaN	NaN	1	1.626016	NaN
1	: nan, 'person': 'david ...	None	NaN	david bachman	1	-5.937684	NaN
2	{'date': None, 'theme': nan, 'person': 'dmitry...	None	NaN	dmitry kulikov	1	-6.382979	NaN
3	{'date': None, 'theme': nan, 'person': 'jakov ...	None	NaN	jakov kedmi	1	-6.382979	NaN
4	{'date': None, 'theme': nan, 'person': 'muslim...	None	NaN	muslim uighurs	1	-5.937684	NaN
...	...	...	...	...	...	...	...
461187	{'date': 2022-12-31 00:00:00, 'theme': 'WB_698...	2022-12	WB_698_TRADE	southfront lucas	1	-3.351955	CH
461188	{'date': 2022-12-31 00:00:00, 'theme': 'WB_738...	2022-12	WB_738_SOCIAL_COHESION	kylie jenner	1	-3.351955	CH
461189	{'date': 2022-12-31 00:00:00, 'theme': 'WB_738...	2022-12	WB_738_SOCIAL_COHESION	southfront lucas	1	-3.351955	CH
461190	{'date': 2022-12-31 00:00:00, 'theme': 'WB_739...	2022-12	WB_739_POLITICAL_VIOLENCE_AND_CIVIL_WAR	kylie jenner	1	-3.351955	CH
461191	{'date': 2022-12-31 00:00:00, 'theme': 'WB_739...	2022-12	WB_739_POLITICAL_VIOLENCE_AND_CIVIL_WAR	southfront lucas	1	-3.351955	CH

pour faire défiler la sortie ; double-
pour masquer

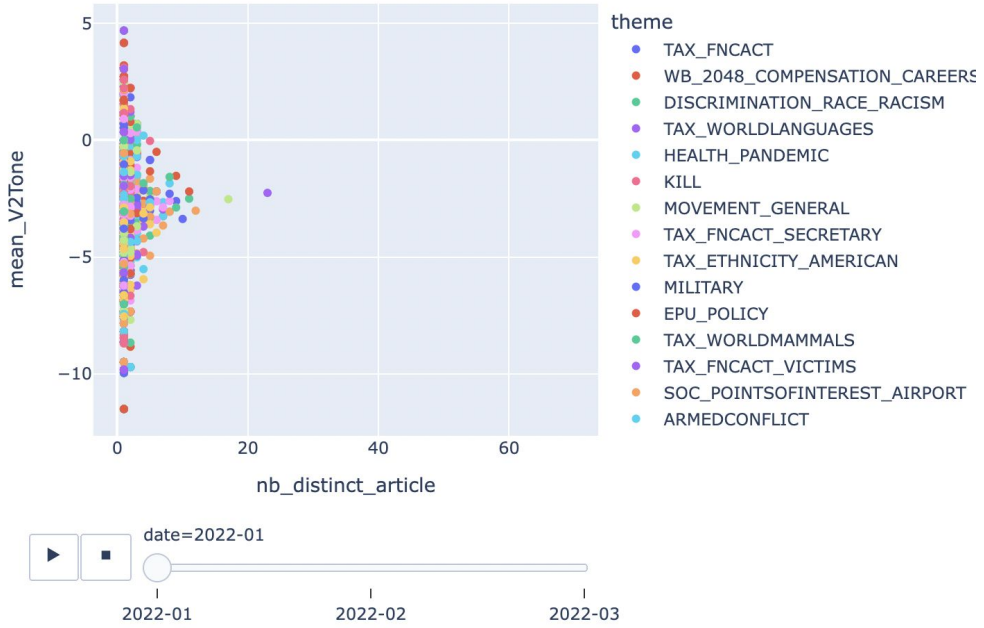
Question 3 :

Sur Janvier à Mars

lefigaro.fr avec une agrégation au mois

```
CPU times: user 1.27 s, sys: 472 ms, total: 1.74 s
Wall time: 6.34 s
```

	date	theme	person	lieu	nb_distinct_article	mean_V2Tone
0	2022-01	TAX_FNCACT	bachir omar	DJ	1	-7.384615
1	2022-01	WB_2048_COMPENSATION_CAREERS_AND_INCENTIVES	marcos jr	RP	1	-1.690821
2	2022-01	DISCRIMINATION_RACE_RACISM	NaN	UV	1	-7.444169
3	2022-01	TAX_WORLDLANGUAGES	anita anand	FR	1	-1.488095
4	2022-01		eddie murphy	US	1	-0.674764
...	...	...	...	...	...	...
123391	2022-03	MANMADE_DISASTER_IMPLIED	NaN	RS	12	-3.280739
123392	2022-03	TAX_WORLDLANGUAGES_FRENCH	joe biden	NaN	3	-4.265816
123393	2022-03	WB_621_HEALTH_NUTRITION_AND_POPULATION	vladimir putin	TU	1	-2.631579
123394	2022-03	MEDICAL	vladimir putin	SY	1	-2.631579
123395	2022-03	TAX_FNCACT	emily horne	RS	1	-3.389831



Question 4 :

Etudiez l'évolution des relations entre deux pays (specifiés en paramètre) au cours de l'année. Vous pouvez vous baser sur la langue de l'article, le ton moyen des articles, les themes plus souvent citées, les personnalités ou tout élément qui vous semble pertinent.

```
mention_collection.create_index([("event.Actor1Geo_CountryCode", 1),
                                 ("event.Actor2Geo_CountryCode", 1) ], unique=False )
```

```
%%time

def evolution_relation(collection, pays_A="RS", pays_B="UP"):
    result = collection.aggregate([
        { "$match": {"$or" : [{ 'event.Actor1Geo_CountryCode': pays_A, 'event.Actor2Geo_CountryCode':pays_B },
                              { 'event.Actor1Geo_CountryCode': pays_B, 'event.Actor2Geo_CountryCode':pays_A }
                             ]
                    }
        },
        { "$group" : { "_id" : {"actor1": "$event.Actor1Geo_CountryCode",
                                "date": { "$dateToString": { "format": "%Y-%m-%d", "date": {"$dateFromString":{
                                    "dateString": {"$toString": {"$toLong" : "$article.DATE" }}, "format": "%Y%m%d%H%M
                                },
                       "V2Tone_list": { "$push" : "$article.V2Tone"}
                      }
        },
        { "$project" : {"actor1":"$_id.actor1",
                        "date": "$_id.date",
                        "mean_V2Tone": {"$avg" : "$V2Tone_list"},
                        "_id" : 0
                       }
        },
        {"$sort": {"date": 1}}
    ])
    return pd.DataFrame(list(result))

df_evolution_relation = evolution_relation(mention_collection, pays_A="UP", pays_B="RS")
df_evolution_relation
```

Question 4 :

Sur toute l'année

Evolution des relations entre la Russie et l'Ukraine

```
CPU times: user 664 ms, sys: 301 ms, total: 966 ms
Wall time: 2min 47s
```

	_id	actor1	date	mean_V2Tone
0	{'actor1': 'UP', 'date': '20221111'}	UP	20221111	-2.822157
1	{'actor1': 'UP', 'date': '20211218'}	UP	20211218	-4.772775
2	{'actor1': 'UP', 'date': '20220811'}	UP	20220811	-3.945061
3	{'actor1': 'RS', 'date': '20220625'}	RS	20220625	-3.644938
4	{'actor1': 'RS', 'date': '20220318'}	RS	20220318	-3.347416
...	...	...	...	...
1102	{'actor1': 'RS', 'date': '20211012'}	RS	20211012	4.899135
1103	{'actor1': 'RS', 'date': '20211211'}	RS	20211211	-3.042613
1104	{'actor1': 'RS', 'date': '20220922'}	RS	20220922	-3.531144
1105	{'actor1': 'UP', 'date': '20220918'}	UP	20220918	-4.350933
1106	{'actor1': 'RS', 'date': '20221019'}	RS	20221019	-3.532091

1107 rows × 4 columns

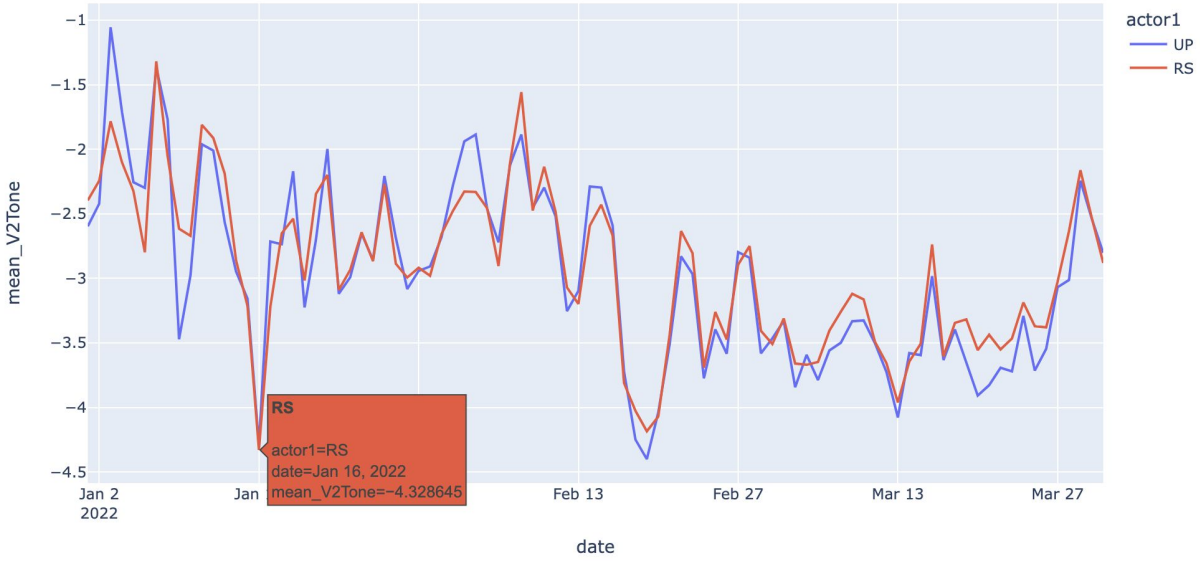
Question 4 :

Sur Janvier à Mars

Evolution des relations entre la Russie et l'Ukraine

```
CPU times: user 80.3 ms, sys: 27.8 ms, total: 108 ms
Wall time: 37.5 s
```

	actor1	date	mean_V2Tone
0	UP	None	NaN
1	RS	None	NaN
2	UP	2022-01-01	-2.596710
3	RS	2022-01-01	-2.395629
4	RS	2022-01-02	-2.244259
...	...	...	...
177	RS	2022-03-29	-2.161984
178	RS	2022-03-30	-2.532367
179	UP	2022-03-30	-2.538688
180	UP	2022-03-31	-2.801545
181	RS	2022-03-31	-2.880651



Démonstration

Des questions?

github

- load_data
  - ssh_scripts
  - pyspark
  - pymongo
- queries
- https://github.com/Kovi11Day/telecom_paris_nosql.git