NoSql : Projet GEDELT

- Youssef ELBRINI
- Yassine CHAFAI
- Xin XU
- Aurelien RAULO

TELECOM Paris

IP PARIS

lesfurets

29/01/2024

PLAN

1. OBJECTIFS
2. LA PIPELINE & L'ARCHITECTURE
3. LES LIMITES & LES SOLUTIONS
4. LES REQUÊTES & LES RÉSULTATS
5. CONCLUSION ET PERSPECTIVES

OBJECTIFS

- Mettre en oeuvre une technologie BigData

- Mettre en place un système de stockage :
  - Distribué
  - Résilient (supporter la panne d'un noeud)
  - Performant (rapide et robuste)

- Répondre à une problématique spécifique :
  - Volume de données initiales (CDELT) > 100 Go
  - Volume de données importantes à stocker (Cassandra) > 5 Go
  - Répondre aux 4 requêtes spécifiques

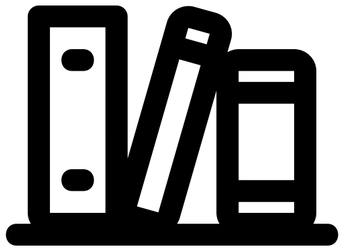
Moyens à disposition : 8 Machines Virtuelles (VM)

2	8 (2 masters)	6	3
Apache Zeppelin	APACHE Spark™	cassandra	
• Lancer des jobs Spark depuis une interface NoteBook • Afficher les résultats d'analyse • Limitation des langues de programmation supportées	• Distribution & parallélisation des tâches • Utilisation des RDD • Langues de programmation (Scala, Python, R) • Permet de faire des requettes (GroupBy, Aggrégation & Jointure)	• Stockage distribué • Réplication des données • Scalable • Configuration simple • Impossibilité de faire des requettes (groupby, aggrégation & jointure)	• Élection du Master • Gestion des disponibilitées

LA PIPELINE

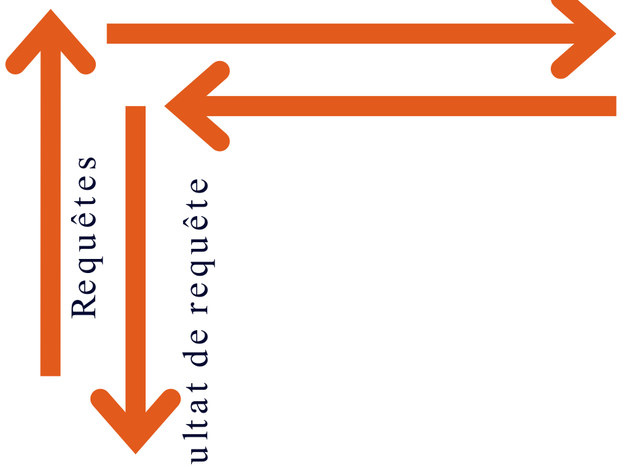
~15Go

Téléchargement des données



GDELT

Instructions de téléchargement



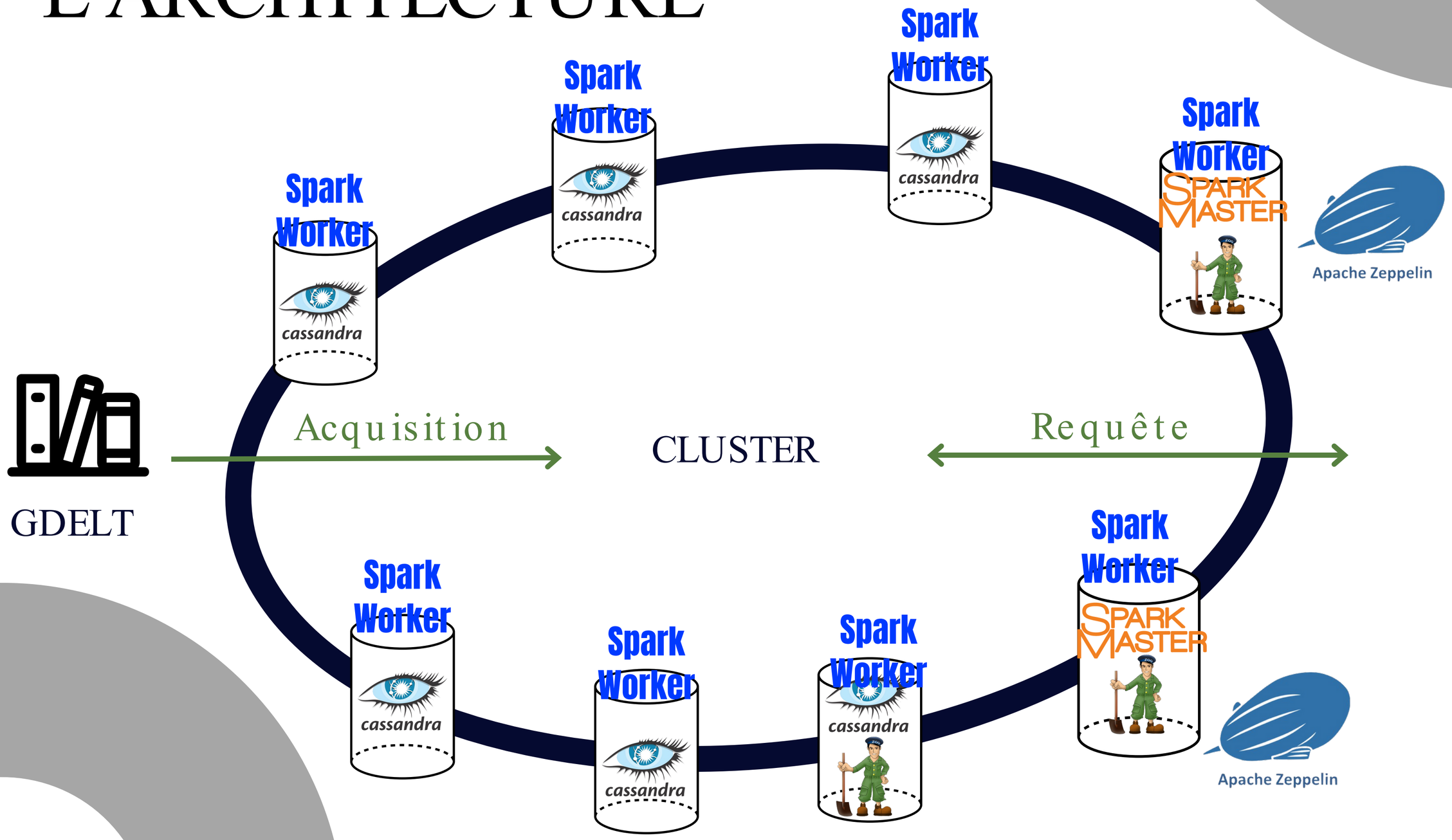
Requêtes

Résultat de requête

~1,6Go

- Filtrage & transformation de données
- Stockage de données
- Suppression des données brutes

L'ARCHITECTURE



LES LIMITES & LES SOLUTIONS

Connexion entre SPARK et CASSANDRA

- La version du connecteur dépend des versions de spark et cassandra
- Choix de la version du connecteur avant le choix des versions Spark et Cassandra

Gestion de la panne d'un noeud Spark Master:

- Difficulté de la gestion des élections de Spark Master
- Utilisation de ZooKeeper pour gérer plus facilement les elections

Nombre important de machines (8)

- Nécessité de configuration sur toutes les machines
- Utilisation de Script Bash pour automatisation

Limitation de l'espace de stockage:

- Impossibilité de télécharger toutes les données en une seule fois
- Téléchargement des données par tronçons

REQUÊTE N°1

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

Données sur Cassandra

Table 1 :

- GlobalEventid
- ActionGeo_countryCode
- EventDate
- MonthYear
- Year

Table 2 :

- id
- GlobalEventid
- MentionTranslationInfo
- MentionTimeDate

Résultat de la requête

globaleventid1	eventdate	actiongeo_countrycode	mentiondoctranslationinfo	count
962219712	20210101	CH	eng	1
962219775	20210101	US	eng	37
962219970	20210101	US	eng	1
962219997	20210101		eng	3
962220225	20210101	US	eng	2
962220312	20210101	EG	eng	5
962220747	20210101		eng	43
962221213	20210101	US	eng	1
962221260	20210101	SO	eng	7
962221565	20210101	US	eng	8
962221666	20210101	US	eng	1
962221691	20210101	US	eng	11
962222402	20210101	CO	eng	7
962222572	20210101	UK	eng	1

La requête

```
val joined = eventsDF_test.join(mentionsDF_test, eventsDF_test("globaleventid1") === mentionsDF_test("globaleventid2")).drop(mentionsDF_test("globaleventid2"))

val counts_req1 = joined.groupBy("globaleventid1", "eventdate", "actiongeo_countrycode", "mentiondoctranslationinfo")
  .count()
```

REQUÊTE N°2

Pour un pays donné en paramètre, affichez les évènements qui y ont eu place triées par le nombre de mentions (tri décroissant)

Données sur Cassandra

Table 1 :

- GlobalEventid
- ActionGeo_countryCode
- EventDate
- MonthYear
- Year

Table 2 :

- id
- GlobalEventid
- MentionTranslationInfo
- MentionTimeDate

Résultat

```
+---------+--------------------+------+
|eventdate|actiongeo_countrycode| count|
+---------+--------------------+------+
| 20210107|                  US|384904|
| 20210120|                  US|288639|
| 20210106|                  US|268485|
| 20210108|                  US|267301|
| 20210113|                  US|254129|
| 20210421|                  US|222110|
| 20210112|                  US|212599|
| 20210114|                  US|211265|
| 20210111|                  US|197618|
| 20210115|                  US|195391|
| 20210121|                  US|194747|
| 20210105|                  US|191603|
| 20210414|                  US|188942|
```

La requête

```
// filtrage par pays
val filteredjoined = joined.filter($"actiongeo_countrycode" === "US")

// aggrégation par jour, mois ou année
val aggregationLevel = "day" // Choisissez parmi "day", "month", "year"
```

```
val aggregatedDF = aggregationLevel match {
  case "day" => filteredjoined.groupBy("eventdate", "actiongeo_countrycode").count().orderBy($"count".desc)
  case "month" => filteredjoined.groupBy("monthyear", "actiongeo_countrycode").count().orderBy($"count".desc)
  case "year" => filteredjoined.groupBy("year", "actiongeo_countrycode").count().orderBy($"count".desc)
}
```

REQUÊTE N°3

Pour une source de donnés passée en paramètre affichez les thèmes, personnes, lieux dont les articles de cette source parlent ainsi que le nombre d'articles et le ton moyen des articles

Données sur Cassandra

Table 2 :

- GkgRecordid
- SourceCommonName
- Themes
- Person
- Location
- Date
- MonthYear
- Year
- AverageTone

La requête

```
val chosenSource = "vnews.com" // Remplacez par la source de votre choix

val filtered_gkg_data = gkg_data.filter($"sourcecommonname" === chosenSource)


// Définir le niveau d'agrégation : "day", "month", "year"
val aggregationLevel = "day" // Remplacer par "month" ou "year" selon le besoin

val aggregated_gkg_data = filtered_gkg_data
  .groupBy(
    col("sourcecommonname"),
    aggregationLevel match {
      case "day" => col("date2")
      case "month" => col("monthyear")
      case "year" => col("year")
    }
  )
  .agg(
    count("gkgrecordid").as("article_count"),
    collect_list("themes").as("themes"),
    collect_list("persons2").as("persons"),
    collect_list("locations").as("locations"),
    avg("v15tone3").as("average_tone")
  )
```

REQUÊTE N°3

Pour une source de donnés passée en paramètre affichez les thèmes, personnes, lieux dont les articles de cette source parlent ainsi que le nombre d'articles et le ton moyen des articles

Résultat de la requête

sourcecommonname	date2	article_count	themes	persons	locations	average_tone
vnews.com	20210107044500	6	[[EDUCATION, SOC_...	[[jackie sevigny,...	[[], [2#Vermont, ...	-2.045355757077535
vnews.com	20210116024500	1	[[UNGP_PHONE_INTE...	[[janet watton, t...	[[2#Vermont, Unit...	1.8731988668441772
vnews.com	20210123053000	1	[[EDUCATION, MANM...	[[donald trump, r...	[[1#Lebanon#LE#LE...	1.6233766078948975
vnews.com	20210114193000	1	[[ELECTION, EDUCA...	[[danielle corti]]	[[3#Bradford Elem...	0.4454343020915985
vnews.com	20210108230000	1	[[EDUCATION, SOC_...	[[michael wojtech...	[[1#Lebanon#LE#LE...	-1.894736886024475
vnews.com	20210129231500	1	[[KILL, CRISISLEX...	[[peter fennelly,...	[[2#Florida, Unit...	1.0416666269302368
vnews.com	20210111043000	6	[[EPU_POLICY, EPU...	[[], [julia griff...	[[1#Hungarian#HU#...	-0.1900726060072581
vnews.com	20210128023000	1	[[TAX_FNCACT, TAX...	[[julia griffin, ...	[[]]	-0.1420454531908035
vnews.com	20210101120000	1	[[EDUCATION, SOC_...	[[connor wahl, ka...	[[1#Lebanon#LE#LE...	3.4482758045196533
vnews.com	20210121100000	1	[[TAX_FNCACT, TAX...	[[james jackson]]	[[1#Lebanon#LE#LE...	-6.382978916168213
vnews.com	20210121040000	1	[[GENERAL_HEALTH,...	[[donald trump, g...	[[3#Waterville Va...	-3.4610631465911865
vnews.com	20210117044500	1	[[]]	[[greta schutz, j...	[[1#Lebanon#LE#LE...	3.1088082790374756
vnews.com	20210119014500	9	[[LEGISLATION, EP...	[[martin luther k...	[[], [3#Springfie...	-0.41438989341259
vnews.com	20210104063000	3	[[RETIREMENT, WB_...	[[stephen marchew...	[[1#Polish#PL#PL#...	0.7644799947738647

REQUÊTE N°4

Étudiez l'évolution des relations entre deux pays (specifies en paramètre) au cours de l'année (FRANCE - CHINE)

Données sur Cassandra

Table 3 :

- GlobalEventId
- EventDate
- Actor1CountryCode
- Actor2CountryCode
- NumberArticle
- AverageTone

La requête

```
val filtered_events4_data = events4_data.filter(
  ($"actor1countrycode" === "CHN" && $"actor2countrycode" === "FRA") ||
  ($"actor1countrycode" === "FRA" && $"actor2countrycode" === "CHN")
)
val aggregated_events4_data = filtered_events4_data
  .groupBy($"eventdate")
  .agg(
    count($"numarticles").as("total_articles"),
    avg($"avgtone").as("average_tone")
  )
  .orderBy($"eventdate")
```

Résultat de la requête

eventdate	total_articles	average_tone
20200102	12	2.2346367835998535
20200107	2	3.3057851791381836
20200117	6	3.4120733737945557
20200121	1	-1.7587939500808716
20200122	6	-1.4806469678878784
20210101	25	-0.01832471847534...
20210102	18	-1.5042487813366785
20210103	24	-1.4990868121385574
20210104	21	-1.567230597847984
20210105	41	-1.017481696314928
20210106	26	0.1807699753687932
20210107	48	-0.12231996438155572
20210108	52	0.11606472214827171
20210109	21	-1.7765141101110549

CONCLUSION

- Architecture résiliente et robuste
- Architecture scalable
- Utilisation de technologies Big Data Open Source (BDOS)

PERSPECTIVES

- Ajout du logiciel AirFlow pour la planification et l’automatisation des
- Ajout du logiciel Yarn pour monitorer les applications lancées

Merci

- Youssef ELBRINI
- Yassine CHAFAI
- Xin XU
- Aurelien RAULO

29/01/2024