# Projet Big-Data

Sommaire

[Projet Big-Data 1](#_Toc157762978)

[1. Préambule 2](#_Toc157762979)

[2. Thème choisi 2](#_Toc157762980)

[3. Source des données 3](#_Toc157762981)

[4. Outils 3](#_Toc157762982)

[5. Travail réalisé 4](#_Toc157762983)

[6. Pipeline 5](#_Toc157762984)

[7. Conclusion 6](#_Toc157762985)

## Préambule

L'objectif du projet "Big Data" est de créer une architecture de bout en bout pour la gestion des données, comprenant l'ingestion des données, leur transformation et leur exposition. Dans ce projet, vous êtes libre de décider quelles données vous souhaitez récupérer et de déterminer la sortie finale qui créera de la valeur à partir de vos données d'entrée.

La seule exigence à respecter est la mise en place d'une architecture de Data Lake (lac de données), qui nous permettra de structurer correctement les données, d'avoir un pipeline de données propre et des données partageables.

Un Data Lake est une infrastructure de stockage qui permet de stocker des données brutes de manière économique et de les traiter ultérieurement. Il peut contenir divers types de données, y compris des données structurées, semi-structurées et non structurées. Dans le cadre de ce projet, vous devrez mettre en place cette architecture pour gérer et organiser vos données de manière efficace.

Votre projet devrait inclure les étapes suivantes :

1. Ingénierie d'ingestion des données : Vous devrez déterminer quels types de données vous souhaitez récupérer, qu'il s'agisse de données provenant de sources externes, de bases de données, de flux de données en temps réel, etc. Vous devrez ensuite mettre en place des processus d'ingestion pour collecter ces données et les stocker dans votre Data Lake.
2. Transformation des données : Une fois les données collectées, vous devrez les transformer pour les préparer à l'analyse et à l'utilisation. Cela peut inclure la nettoyage des données, la fusion de sources de données différentes, l'enrichissement des données, etc.
3. Stockage des données : Vous devrez organiser les données dans votre Data Lake de manière à ce qu'elles soient facilement accessibles et gérables. Cela peut impliquer la mise en place de répertoires ou de structures de stockage appropriés.
4. Exposition des données : Enfin, vous devrez créer des interfaces ou des mécanismes pour permettre aux utilisateurs ou aux applications d'accéder aux données de manière sécurisée et efficace. Cela peut inclure la création de tableaux de bord, de rapports, d'API, etc.

L'objectif ultime est de créer une infrastructure de données robuste qui permettra d'exploiter efficacement les données pour en tirer des informations précieuses et prendre des décisions éclairées.

## Thème choisi

J'ai sélectionné comme sujet de projet l'analyse des parties classées (ranked) jouées dans le jeu "League of Legends". Mon choix consiste à mener des analyses à la fois globales, en se penchant sur l'ensemble des joueurs et des champions, et individuelles, en se concentrant sur l'examen des performances de chaque joueur de manière spécifique. Cette approche permettra d'explorer différents aspects des parties classées, tout en mettant en lumière des tendances et des insights tant au niveau général qu'au niveau individuel.

## Source des données

Sources de données pour le projet :

* Fichier JSON manuellement créé :
  + Contient les noms d'invocateurs des joueurs et leurs alias associés.
* API League of Legends (LoL) :
  + Utilisation du fichier JSON de données sur certains joueurs professionnels.
  + Extraction de l'identifiant de l'invocateur (Summoner ID).
  + Accès aux identifiants de matchs (Match ID) auxquels le joueur a participé.
  + Récupération de l'historique complet des parties pour chaque match individuel.

Il convient de noter que toutes les sources de données utilisées sont au format JSON, ce qui simplifie le processus d'acquisition, de transformation et d'analyse des données tout au long du projet. Cette approche permet une gestion flexible et efficace des données, tout en garantissant leur cohérence et leur intégrité pour l'analyse ultérieure.

## Outils

Dans le cadre de ce projet, plusieurs outils et services ont été utilisés pour mettre en place l'infrastructure de données. Voici une liste des outils et de leurs versions correspondantes :

1. **Hadoop (HDFS)** : La version 3.3.6 de Hadoop a été utilisée comme système de fichiers distribué pour le stockage et la gestion des données. Les commandes telles que **start-dfs.sh** et **start-yarn.sh** ont été exécutées pour démarrer les services HDFS et YARN respectivement.
2. **Spark** : La version 3.3.4 de PySpark a été installée à l'aide de la commande **pip install pyspark==3.3.4**. PySpark a été utilisé pour effectuer le traitement des données et les analyses.
3. **Airflow** : La version 2.8.0 d'Apache Airflow a été installée pour la gestion des tâches de flux de données. Les étapes d'initialisation d'Airflow ont été effectuées, notamment la création d'un utilisateur administrateur et le démarrage du serveur web avec la commande **airflow webserver** ainsi que le scheduler avec la commande **airflow scheduler**.
4. **Elasticsearch et Kibana** : Elasticsearch et Kibana sont utilisés pour le stockage et la visualisation des données. La version 7.10 d'Elasticsearch a été utilisée. Les commandes **sudo systemctl start elasticsearch** et **sudo systemctl start kibana** ont été exécutées pour démarrer ces services.
5. **GitHub** : GitHub a été utilisé comme plateforme de gestion de code source et de collaboration. Une clé d'authentification a été ajoutée pour faciliter l'accès aux dépôts GitHub.

Ces outils et versions ont été soigneusement sélectionnés et configurés pour mettre en œuvre l'architecture de données du projet. Ils ont permis de collecter, de stocker, de transformer et d'analyser les données de manière efficace et cohérente tout au long du processus.

## Travail réalisé

[Code source du projet](https://github.com/knekruz/projectBD)

Dans le cadre de ce projet, un ensemble de scripts et d'outils a été développé pour mettre en place une architecture de données de bout en bout. Voici un aperçu des principales tâches réalisées :

1. **Collecte de Données**
   * Un fichier JSON manuellement créé a été utilisé pour stocker les noms d'invocateurs des joueurs ainsi que leurs alias.
   * L'API de League of Legends a été exploitée pour obtenir les identifiants de joueurs professionnels en utilisant les noms d'invocateurs fournis.
2. **Gestion du Système de Fichiers HDFS (Hadoop Distributed File System)**
   * Les commandes HDFS ont été utilisées pour créer des répertoires, vérifier si HDFS était en cours d'exécution et téléverser des fichiers depuis le système de fichiers local vers HDFS.
3. **Initialisation et Configuration d'Airflow**
   * Apache Airflow a été installé pour automatiser et gérer les tâches de flux de données.
   * L'initialisation d'Airflow a été effectuée, y compris la création d'un utilisateur administrateur et le démarrage du serveur web Airflow.
4. **Utilisation d'Elasticsearch et Kibana**
   * Elasticsearch et Kibana ont été utilisés pour le stockage et la visualisation des données.
5. **Développement des Scripts Python**
   * Plusieurs scripts Python ont été développés pour collecter, traiter et charger les données :
     + **fetch\_summoner\_ids.py** : Collecte les identifiants de joueurs professionnels et met à jour les détails des invocateurs.
     + **fetch\_match\_ids.py** : Collecte les identifiants de matchs classés pour les invocateurs.
     + **fetch\_match\_histories.py** : Récupère les détails des matchs classés pour les invocateurs.
     + **history\_format.py** : Formate et sauvegarde les données des matchs dans HDFS.
     + **elastic\_load.py** : Charge les données formatées des matchs dans Elasticsearch avec une gestion des alias.
6. **Structuration des Données**
   * Une structure de dossiers a été mise en place pour organiser les fichiers et les données, garantissant une gestion propre des données et des scripts.
7. **Contrôle de Version avec GitHub**
   * GitHub a été utilisé comme plateforme de gestion de code source pour le suivi des modifications et la collaboration.
8. **Documentation et Gestion des Dépendances**
   * Un fichier **requirements.txt** a été utilisé pour spécifier les dépendances du projet.
   * Des fichiers de documentation ont été prévus pour détailler le projet et son architecture.

Dans l'ensemble, ce projet a permis de mettre en place une architecture de données solide pour collecter, stocker, traiter et exposer des données provenant du jeu League of Legends. Les scripts et les outils développés ont automatisé de nombreuses étapes du processus, facilitant ainsi la gestion des données et la création de rapports d'analyse.

## Pipeline

Dans le modèle ci-dessous, chaque DAG est exécuté séquentiellement après le précédent. Une fois que **fetch\_summoner\_ids\_dag** est terminé, **fetch\_match\_ids\_dag** est déclenché, et ainsi de suite. Cela garantit que chaque étape est effectuée dans l'ordre.  
  
**fetch\_summoner\_ids\_dag -> fetch\_match\_ids\_dag -> fetch\_match\_histories\_dag -> history\_format\_dag -> elastic\_load\_dag**

Voici un schéma expliquant la pipeline mis en place :

Une image contenant capture d’écran, diagramme, texte, Rectangle

Description générée automatiquement

## Conclusion

Ce projet a été une expérience incroyablement instructive et enrichissante. À travers chaque étape, de la collecte des données à leur exposition finale, j'ai acquis des compétences précieuses en gestion de données, en automatisation des flux de travail et en utilisation d'outils tels qu'Apache Airflow, Elasticsearch et Hadoop.

J'ai également eu l'opportunité de plonger profondément dans le domaine de l'analyse de données, en explorant les parties classées du jeu League of Legends. Les défis rencontrés tout au long du projet m'ont permis d'apprendre de nouvelles techniques de manipulation de données, de gestion des erreurs et d'optimisation des performances.

Plus que tout, ce projet a allumé ma curiosité et m'a inspiré pour entreprendre de futurs projets de données. Les possibilités sont infinies, et je suis enthousiaste à l'idée d'explorer d'autres jeux, d'autres sources de données ou même d'autres domaines d'application pour mes compétences en gestion de données.

## 