**Introduction au Big Data**

Table des matières

[Introduction : 2](#_Toc413881437)

[Chapitre 1 : **Titre1** 2](#_Toc413881438)

[Chapitre 2 : **Titre1** 2](#_Toc413881439)

[Chapitre 3: **Titre1** 2](#_Toc413881440)

[Chapitre 4: **Titre1** 2](#_Toc413881441)

[Chapitre 5: **Titre1** 2](#_Toc413881442)

## Introduction :

Dans un époque ou l’internet deviens indispensable pour le shopping, les réseaux sociaux, les vidéos ,les transaction financière ,chaque minute deviens importante et on peux imaginer la voluminosité des information exemple : Amazon chaque minute il vend 222.283$

Donc il fallais trouver un moyen pour gérer ces informations enormes et en temps réel d’où viens le Big Data

## Chapitre 1 : **la définition de Big Data**

# Les contraintes et Outils  :

On a deux contraintes majeures :

* Stocker des volumes important.
* Analyser des gros volumes

Pour cela on a les outils suivants :

* Un Système de fichier distribués.
* Un principe de Map/Reduce distribue le stokage et les calcules

# Le déclancheur de Big Data( les 5 V)

Volume : un volume important qui sera difficile à traiter par les outils anciens

Variabilité : les différences data

Vélocité : capacité de système à traiter des données le temps nécessaire

Véracité : la source de données on va même façon.

Valeur ; gain d’argent.

# Les motivations :

Pour avoir plus de CPU qui excute le programme rapidement ,et aussi plus de disque qui permettra de stocker plus

# Quelques défis :

* Scabalité : la capacité de système d’adapter
* Tolérance aux pannes : stocker l’information capable de reprendre le traitement la données et le traitement.
* Facilité d’usage : facile a utilisé
* Traitement de donnée en streaming :capable de faire traitement en temps réel
* Capacité de traiter les données variées :image,son ….ect
* Précision et perfermance :capacité a éliminer les erreurs et aller plus vite.

# Scalabilité :

1. Scal up : il est cher plus gros,plus fiable
2. Scal out : moins cher,plus petit ,plus puissant moins fiable

La particularité de scal out c’est qu’il est plus tolérent qux pannes

## Chapitre 2 : **la tolérance aux pannes**

Dans le système massivement paralléliser les pannes sont très fréquent il exsite deux types de pannes :

1. Matériel :

* Panne de disque
* Panne de mémoire
* Refroidissement inadéquat

1. Indispensabilité des ressources :

* Surcharge de système.

# Protocole :

* Découper les données
* Lancer des petits traitements en cas de panne lancer juste le traitement en panne

## Chapitre 3 : **Hadoop**

# Définition :

C’est un système ou plateforme orchestrateur pour gérer les différents traitements qui contient plusieurs composants :

* HDFS : système de fichier disribué d’hadoop
* Yarn : Gestionnaire des ressources
* Map /Reduces :distribiteur de taches

Hadoop est une implémentation de map/reduce

# Ecosysteme Hadoop :

HDFS : l’ensemble des clusters (machines) c’est brique de gestion de données

YARN : L’orchestrateur et le gestionnaire de ressources.

BACH : traitement des données massives avec :

* Map /reduce ,Speark …
* Accessebilité donnée.
* Traitement par lot :

1. SQL : Hive interpot de données
2. SQL :Hbase les données stoker en format différents base données documentaire
3. Spark :traitement par bach et streaming

Intégration :

KAFKA :Proker de messages le points d’entrer des données ,KAFKA fonction avec Zookeeper

Zookeeper :c’est un coordinateur les traitement distribuer

Il ya d’autre outils pour accèder aux données et les indexé comme :

* Elastic search :Support stocker les données et les indexés
* Kibana :brqnche sur Elasticsearch pour la visualisation et reporting
* Logstash :l’équivalent de KAFKA intégrer

## Chapitre 4 : **HDFS**

# Les objectifs :

* Panne matériel .
* Acces aux données de taille importantes (une fois la donnée enregistrer on ne peux pas la modifié)
* Portabilité sur des système hétrogenes .

**Chaque cluster HDFS et composer de :**

Namenode : qui de son coté :

* Gere les métadonnées et sa distribution.
* Déclanche une réplication
* Il y’a qu’un seul Namenode qui dispose un backup .

# Datanode :

* Un ou plusieurs par cluster
* Stock la partie confié des données
* Responsable de la réplication de la donnée sur les autres nœuds
* Gere les opération sur le blocs
* Par défaut le nombre de réplication 3 :

## 