



# Cours 4 : De la BI au Big Data

Un cours de Yann Fornier

# Historique et évolution : Origines de la BI

La **Business Intelligence (BI)**, ou **informatique décisionnelle** en français, a des origines remontant à plusieurs décennies, bien avant que le terme ne devienne courant dans les années 1990.

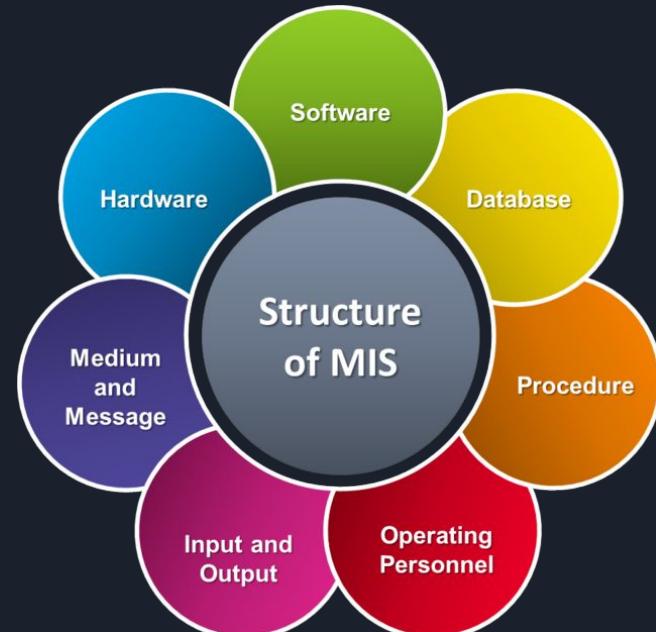


# Historique et évolution : Informatique et gestion des données (années 1950-1970)

L'idée de la Business Intelligence trouve ses racines dans les premières utilisations de l'informatique pour la gestion des données et la prise de décision.

Dans les années 1950 et 1960, les entreprises ont commencé à adopter des systèmes informatiques pour automatiser le traitement des données.

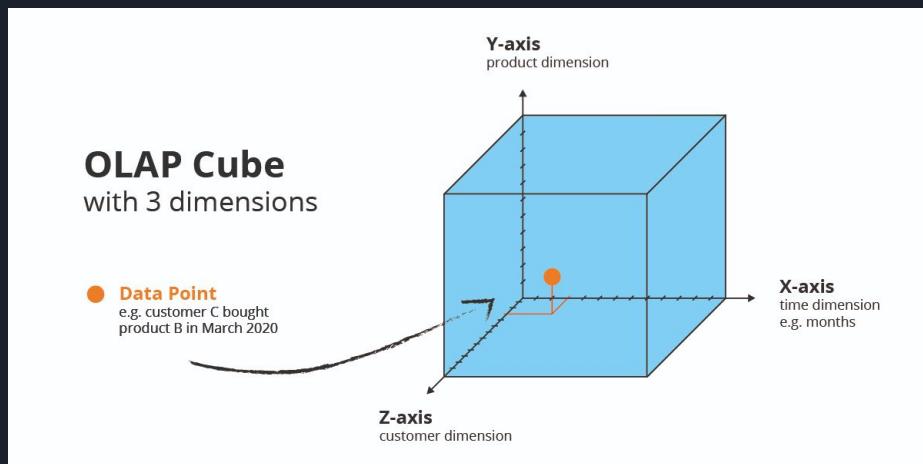
C'était l'époque des premiers systèmes d'information de gestion (MIS) qui permettaient de centraliser et de traiter les données pour produire des rapports de gestion.



# Historique et évolution : Décision informatique et OLAP (années 1970-1980)

Dans les années 1970, le concept de décision informatique a émergé, se concentrant sur l'utilisation de systèmes informatiques pour aider à la prise de décision.

Les bases de données relationnelles sont apparues dans les années 1970, suivies dans les années 1980 par le développement de la technologie **OLAP** (Online Analytical Processing), qui permettait des analyses multidimensionnelles de données.

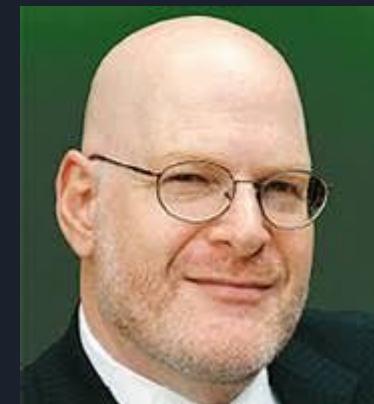


# Historique et évolution : Émergence du terme et évolution des outils (années 1990)

Le terme "Business Intelligence" a été popularisé dans les années 1990, notamment grâce à un rapport de l'analyste **Howard Dresner** de Gartner en 1989.

Il l'a défini comme un ensemble de concepts et de méthodes pour améliorer la prise de décision en utilisant des systèmes de support de décision basés sur des faits.

Les outils de BI ont commencé à se développer, notamment les solutions d'entrepôt de données (data warehouse), qui permettent de centraliser les données provenant de diverses sources pour une analyse approfondie.



*Howard Dresner*

# Historique et évolution : L'explosion de la BI dans les années 2000

Avec la généralisation d'internet et des technologies numériques, les années 2000 ont vu une explosion des outils de BI.

Ces outils sont devenus plus sophistiqués, avec l'introduction de solutions de reporting, de visualisation de données, et d'analyses prédictives.

Les entreprises ont commencé à intégrer la BI dans leurs processus quotidiens pour améliorer l'efficacité opérationnelle et la prise de décision stratégique.



*Power BI  
Microsoft*



*Tableau BI  
Tableau  
Software*

# Historique et évolution : Big Data et BI moderne (années 2010-2020)

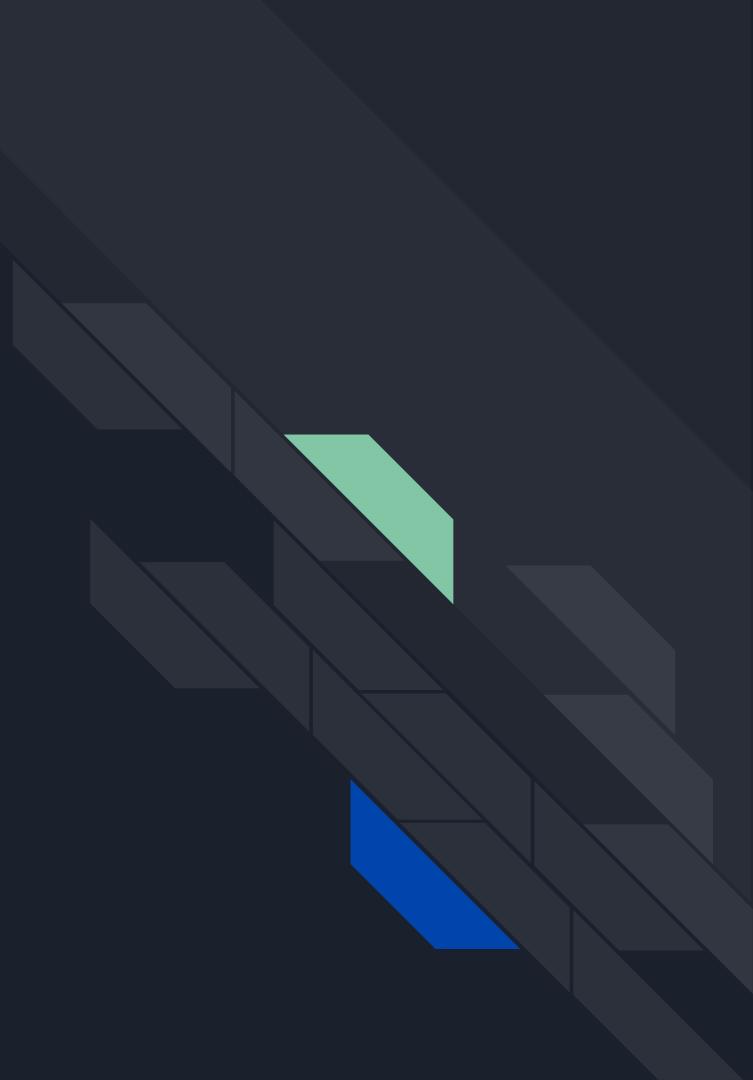
Les années 2010 ont marqué une transformation majeure avec l'émergence du Big Data.

Les entreprises se sont retrouvées avec des quantités massives de données provenant de sources variées (réseaux sociaux, capteurs IoT, etc.).

Cela a conduit à l'émergence de nouvelles technologies comme les plateformes de Big Data (**Hadoop**, **Spark**) et les outils de BI en temps réel, permettant d'analyser ces vastes ensembles de données pour en tirer des informations exploitables rapidement.



# Transition vers le Big Data : Motivations et enjeux



# Internet en 1 minute



# Le “déluge de données”

“La quantité de données massive rend son traitement obsolète par les méthodes scientifiques traditionnelles.”

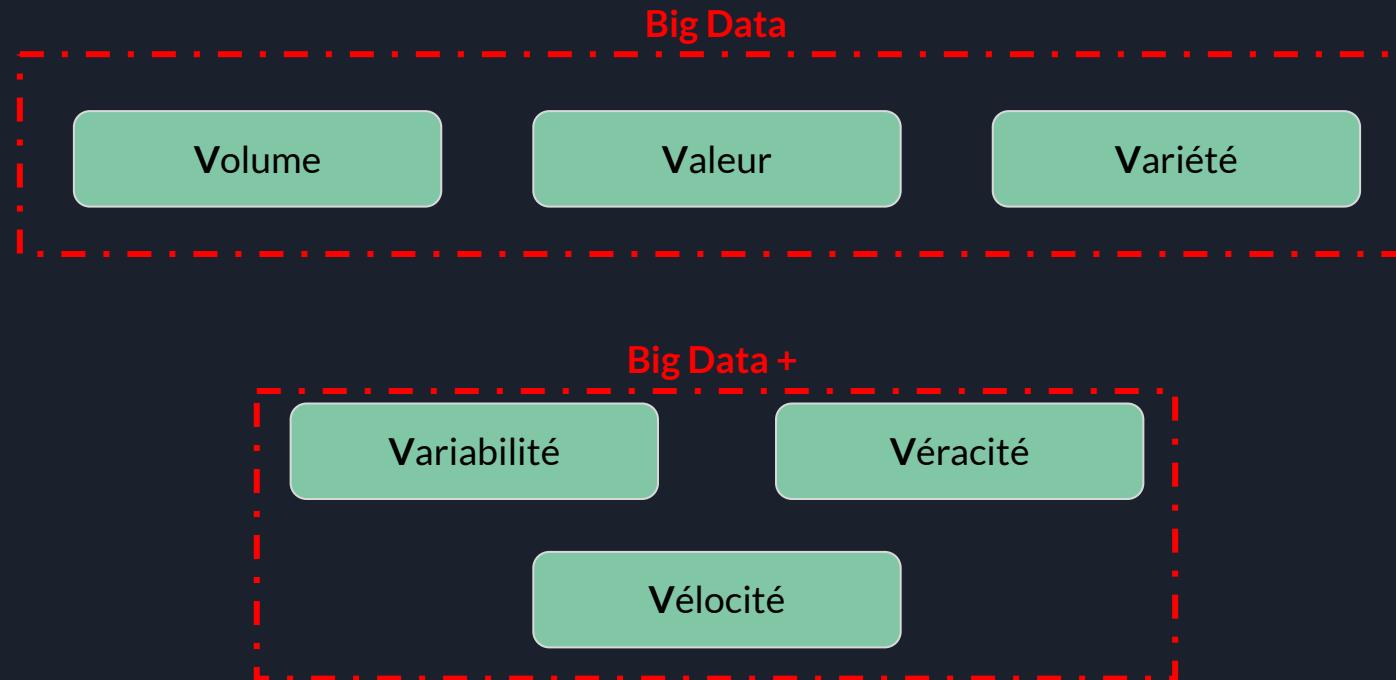


*Chris Anderson - Ex editor in chief of Wired*

<https://www.wired.com/2008/06/pb-theory/>



# Les “6 V” du Big Data



# Introduction

Data Lake

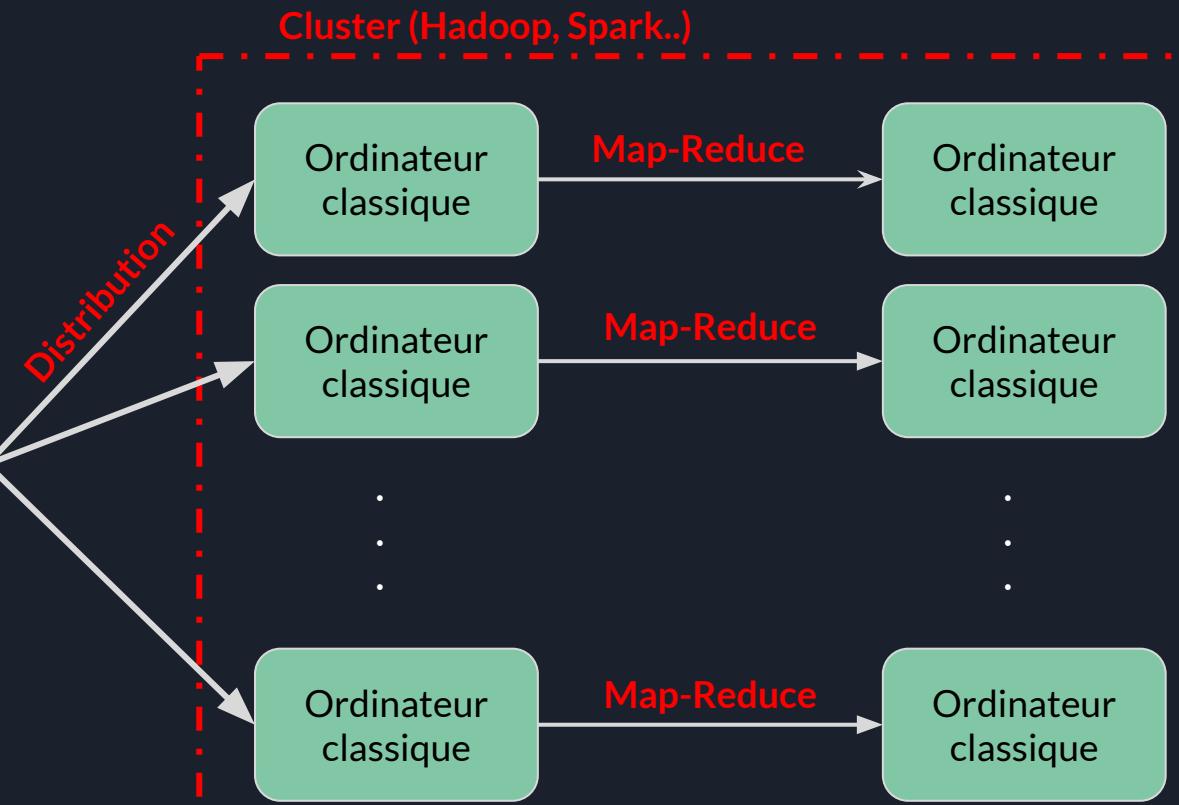
Volume  
Vélocité  
Variété  
Véracité  
Valeur  
Variabilité

**Surcharge**

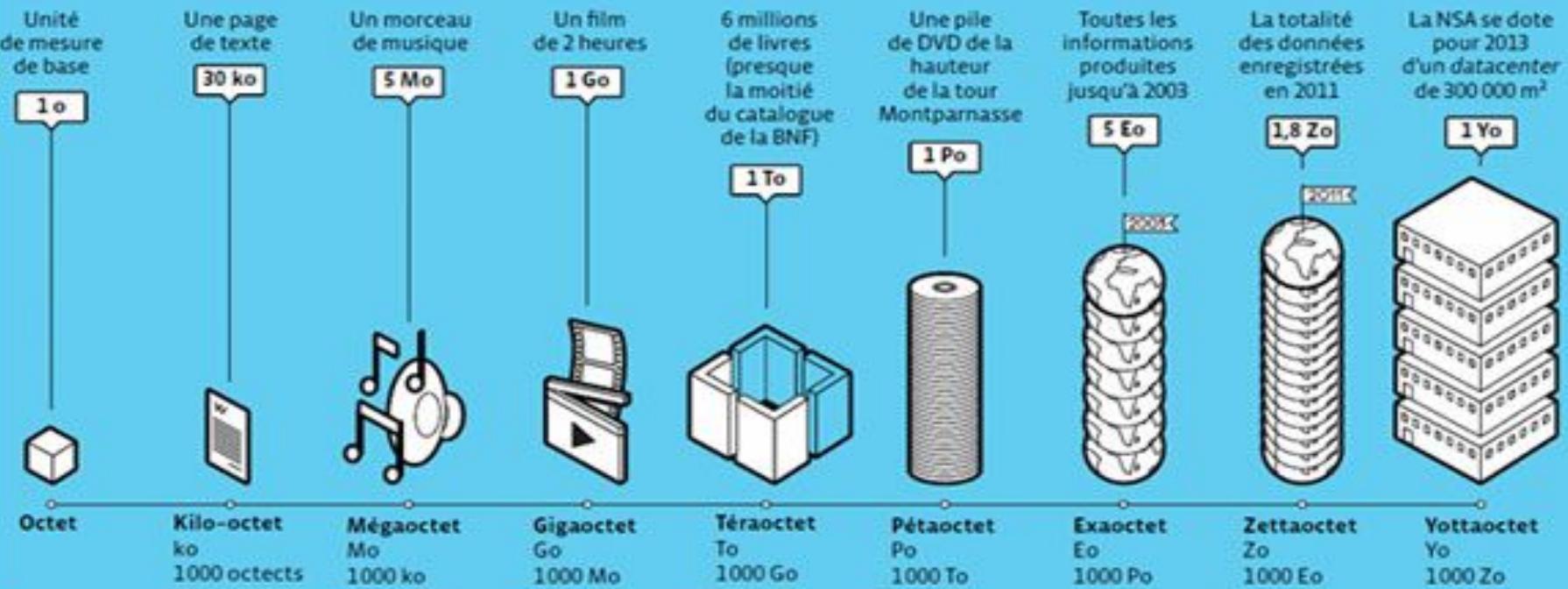
Ordinateur  
classique

# Introduction

Data Lake  
Volume  
Vélocité  
Variété  
Véracité  
Valeur



## L'ÉCHELLE DES OCTETS

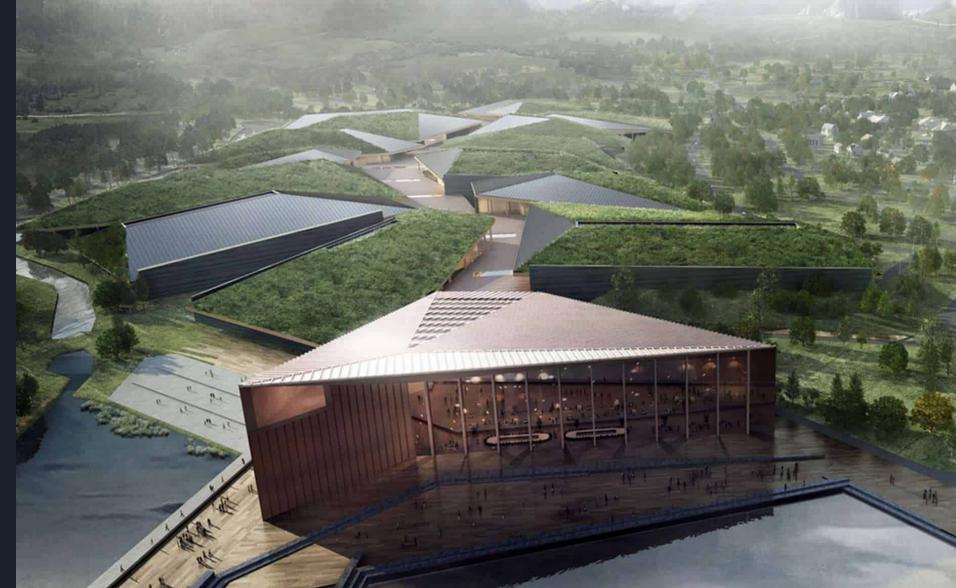


# NSA Data Center - Utah, USA

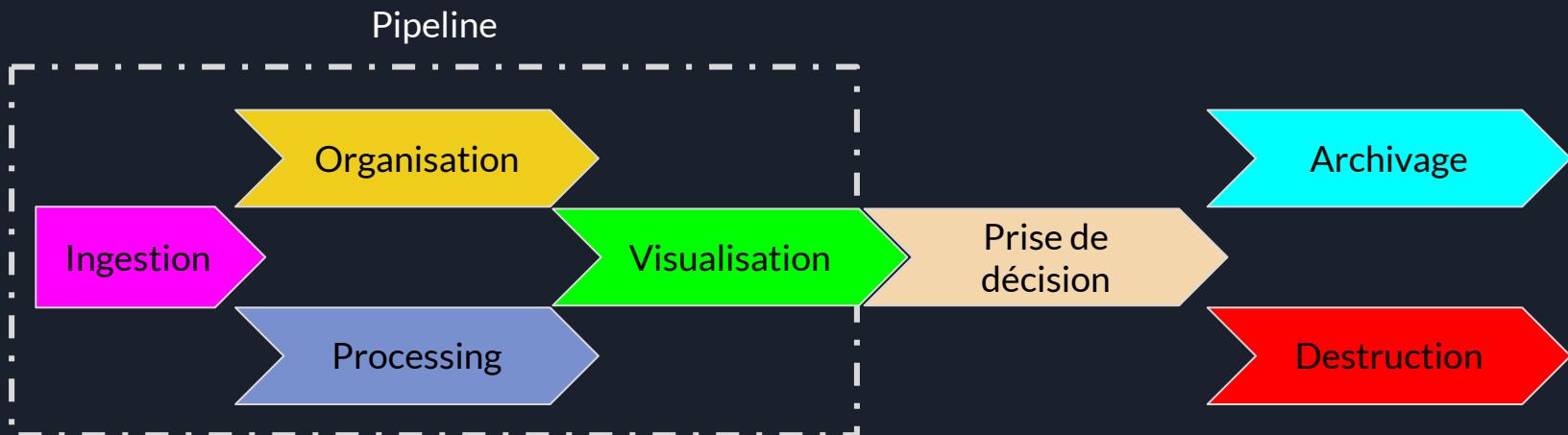




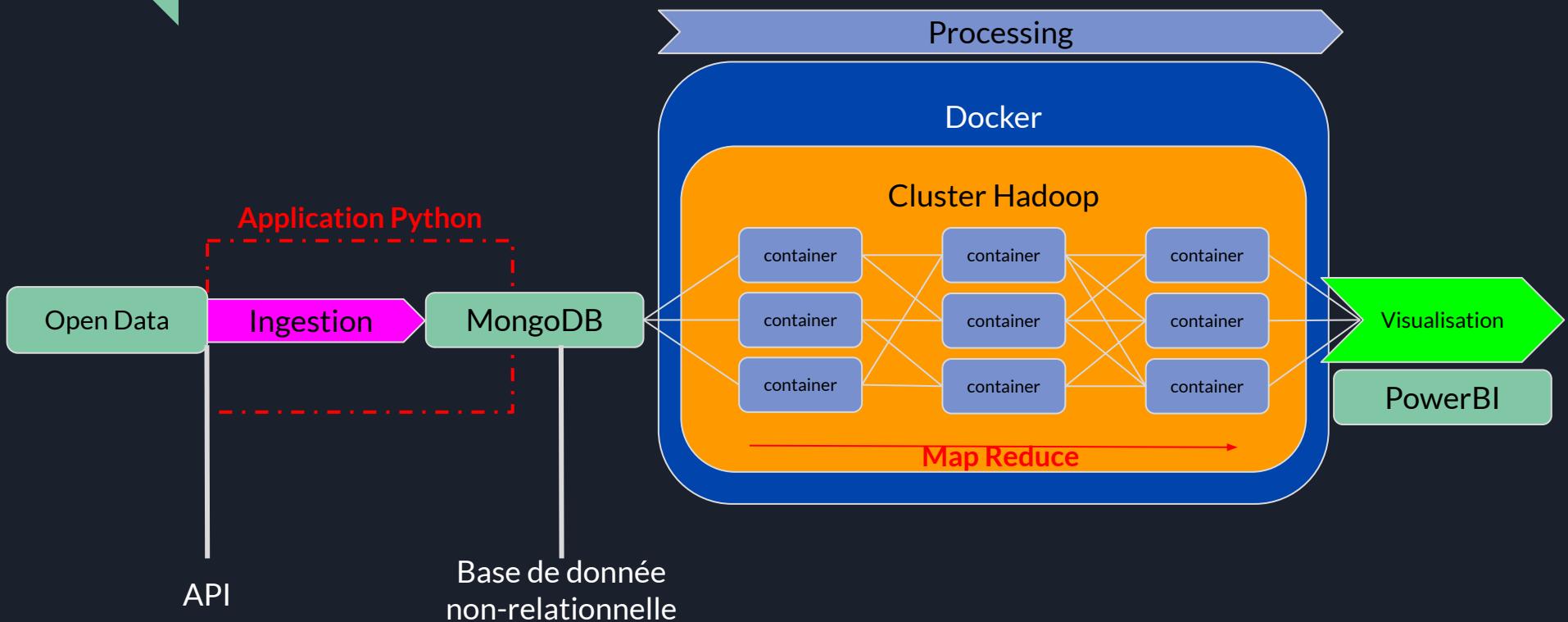
# Kolos (?) - Cercle Arctique - Norvège



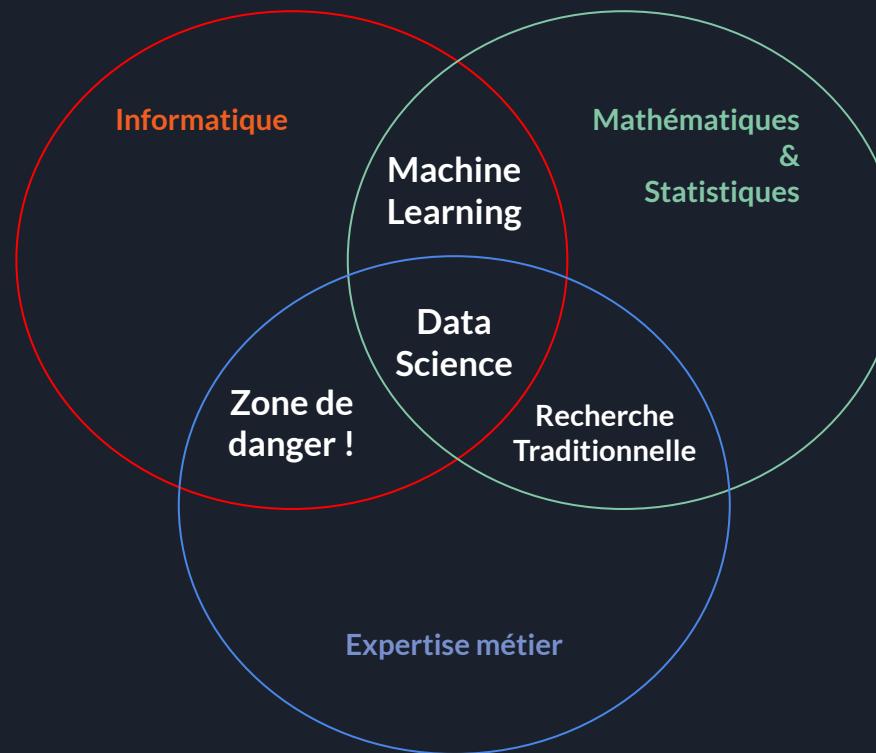
# Le Pipeline du Big Data



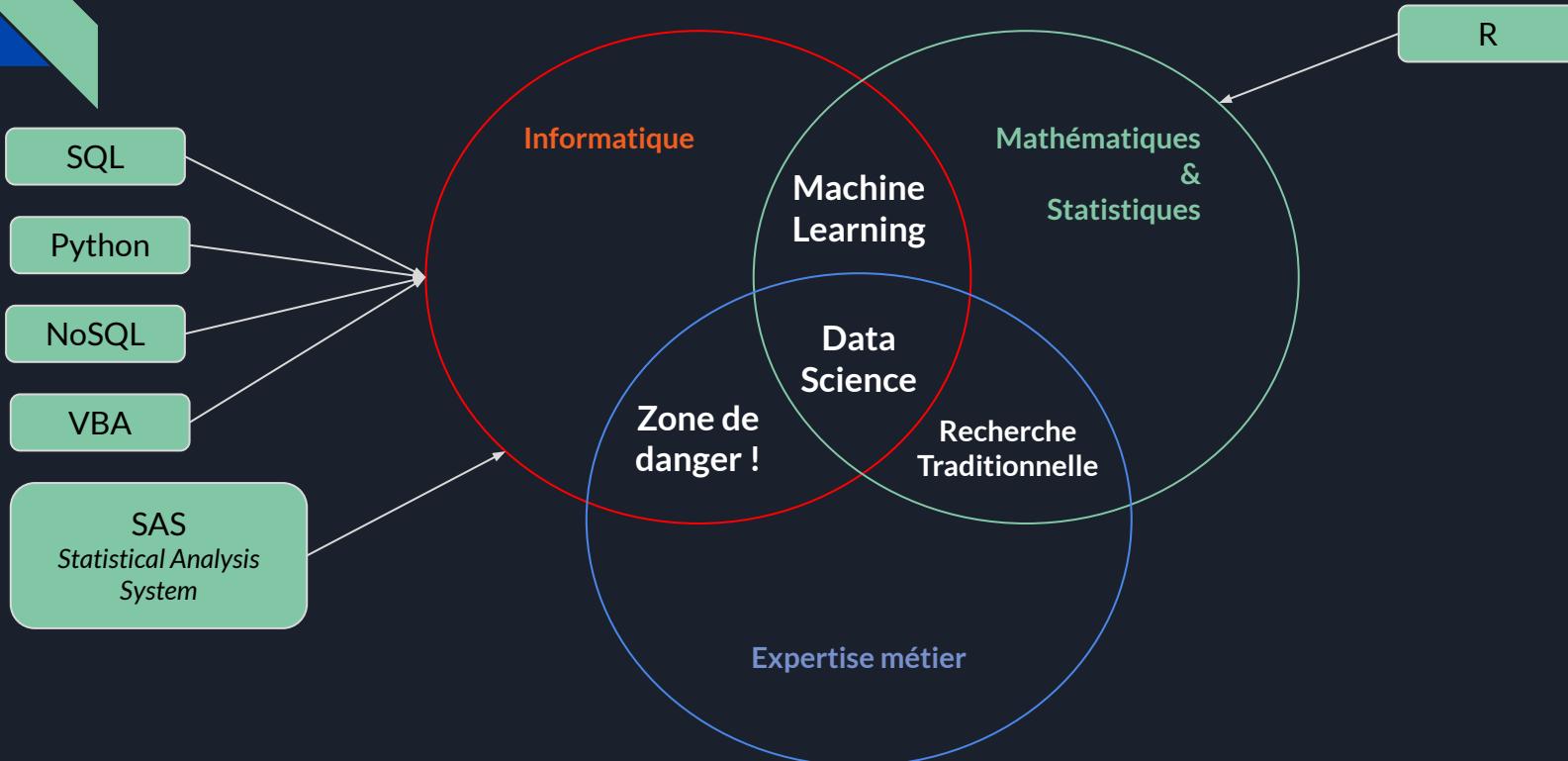
# Architecture d'un projet Big Data (Ingénieur)



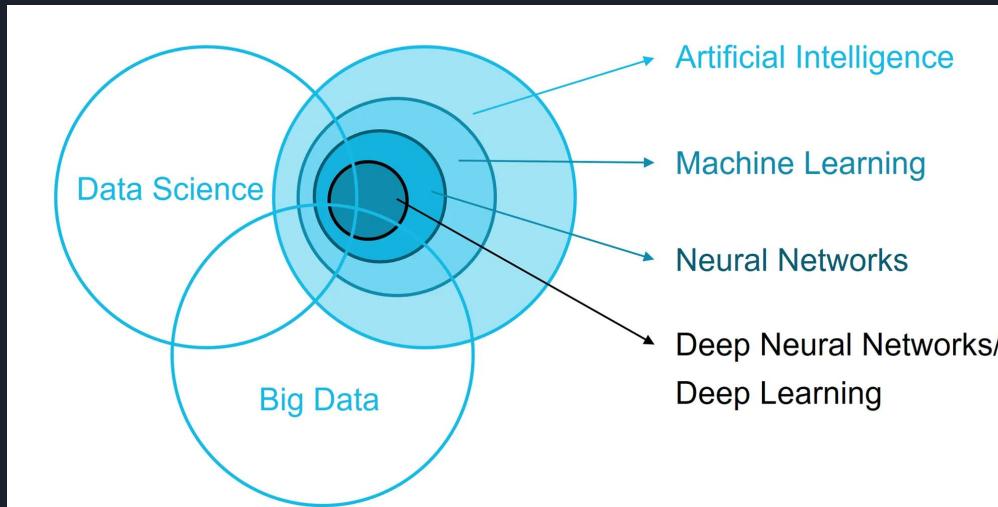
# Les domaines du Big Data



# Les domaines du Big Data

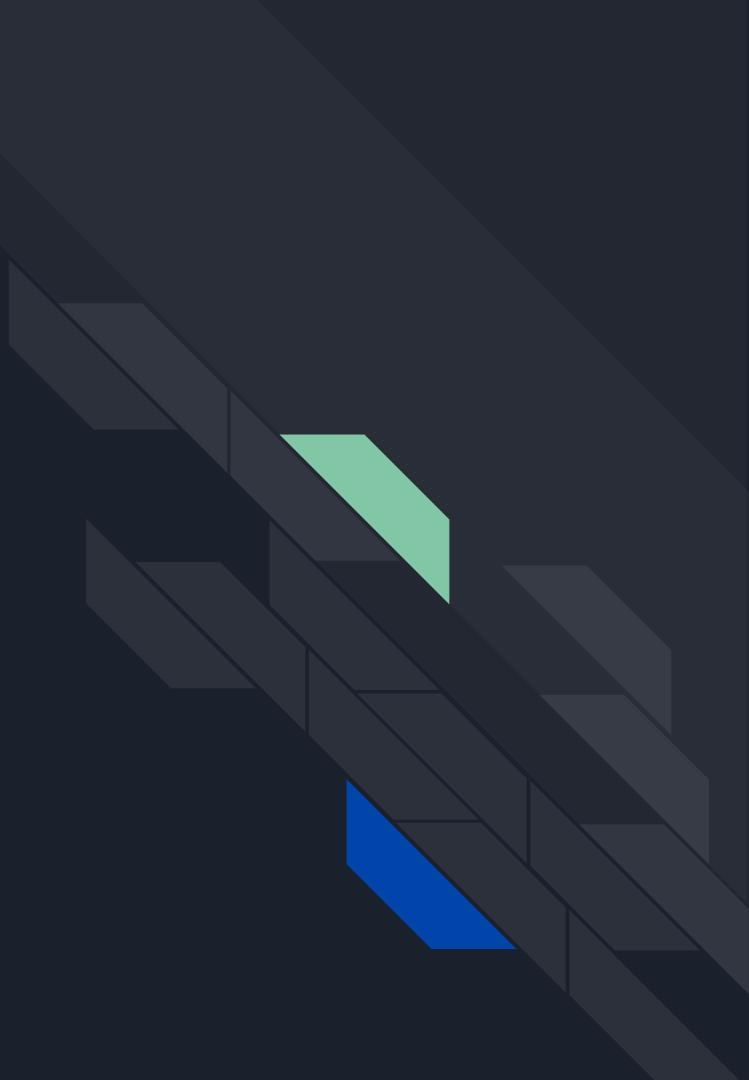


# Big Data & Data Science



<https://towardsdatascience.com/role-of-data-science-in-artificial-intelligence-950efedd2579>

# Les métiers du Big Data



# Les métiers du Big Data

DATA/IA	RÉMUNÉRATION ANNUELLE BRUTE EN K€			
	0-2 ans	2-5 ans	5-10 ans	10 ans et +
Développeur BI	35 - 45	45 - 60	60 - 70	70 - 80
Data engineer/Data scientist	38 - 50	50 - 60	60 - 70	70 - 85
DataOps engineer	38 - 50	50 - 65	65 - 75	75 - 85+
Analyst BI/Data analyst/Data stewart	38 - 48	45 - 60	60 - 70	70 - 80
Chef de projet BI/Big data	40 - 45	45 - 60	60 - 70	70 - 80
Architecte data	80 - 90	90 - 110	110 - 130	130 - 150+
DBA	40 - 45	45 - 55	50 - 60	60 - 70
Ingénieur IA/Développeur IA	40 - 45	45 - 60	60 - 70	70 - 80+
Data protection officer	38 - 45	40 - 60	55 - 75	75 - 85+
Machine learning engineer	40 - 50	50 - 60	60 - 75	75 - 90+
Data visualisation consultant	38 - 48	50 - 60	60 - 70	70 - 90+

*"Etude des salaires en 2023 des métiers du Big Data" - Michael Page*

<https://www.lepont-learning.com/fr/cartographie-metiers-data-demandes/>



# Activité : Cas d'usage

En groupe, vous allez présenter des scénarios où le Big Data complète ou remplace la BI traditionnelle. Vous allez réaliser une analyse de cas d'usage réels avec une mise en contexte des évolutions technologiques.

# Module 2 : Outils et Technologies de la BI et du Big Data





# Outils de BI traditionnelle

- Présentation des principaux outils : Tableau, Power BI,
- Fonctionnalités principales : ETL, reporting, visualisation

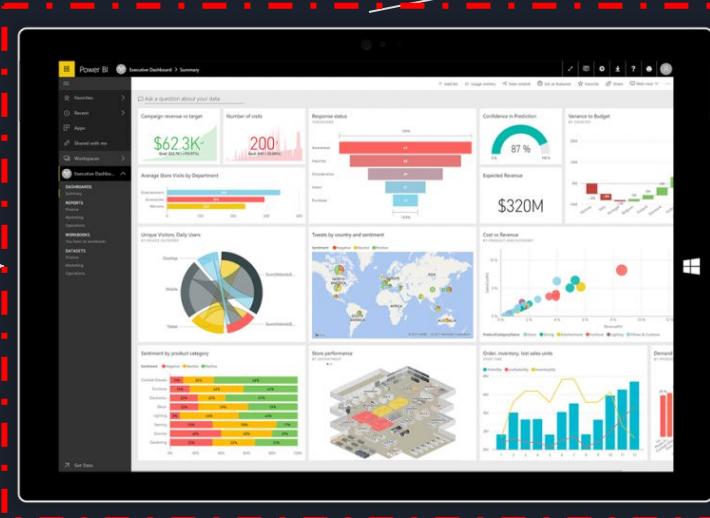
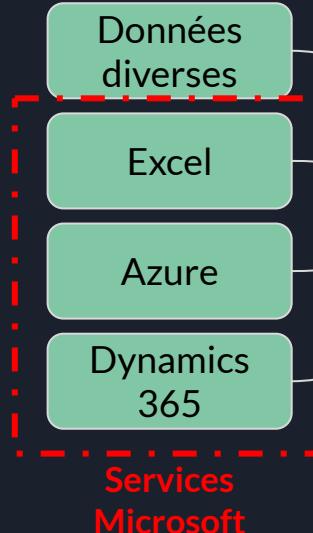
# Power BI : L'outil phare de Microsoft

Power BI est un ensemble de services d'analyse de données développé par Microsoft, conçu pour visualiser et partager des informations de manière interactive.

C'est un outil qui permet aux entreprises et aux utilisateurs individuels de transformer des données brutes en informations exploitables, via des rapports visuels, des tableaux de bord, et des visualisations.



# PowerBI



Modélisation de données

PowerBI

Création de visualisations

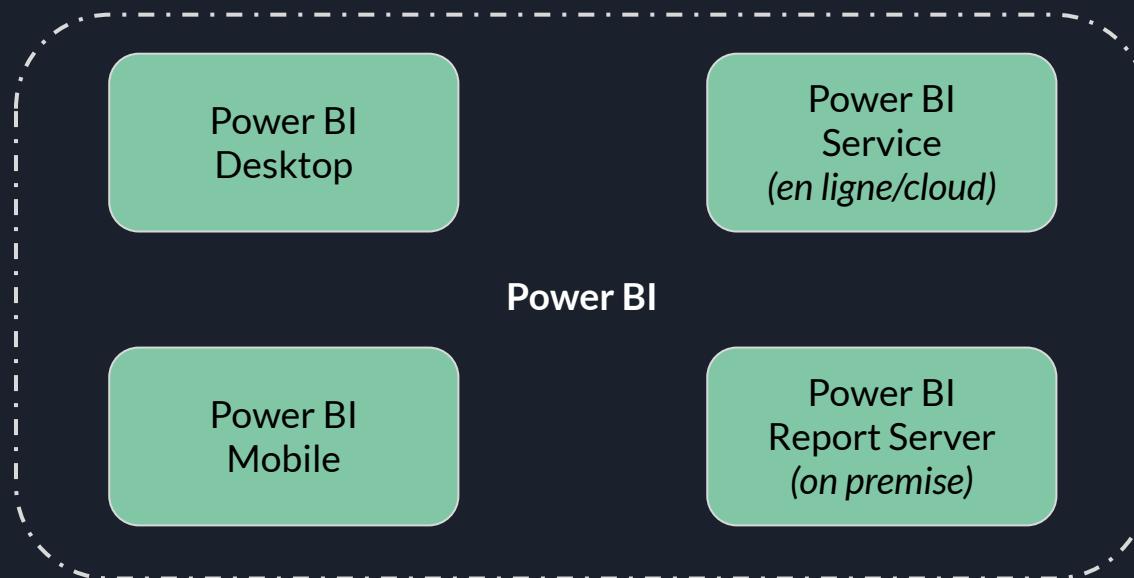
Analyse avancée

Génération de rapports

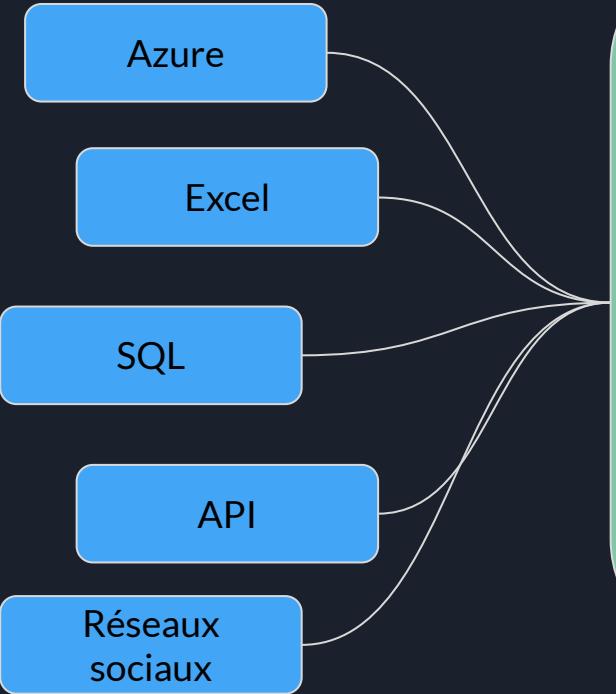
Modèles statistiques



# Power BI : L'outil phare de Microsoft



# Power BI : L'outil phare de Microsoft



Power BI

# Tableau BI

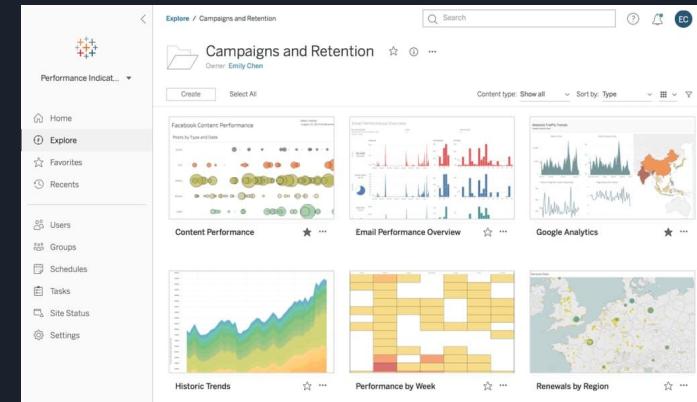
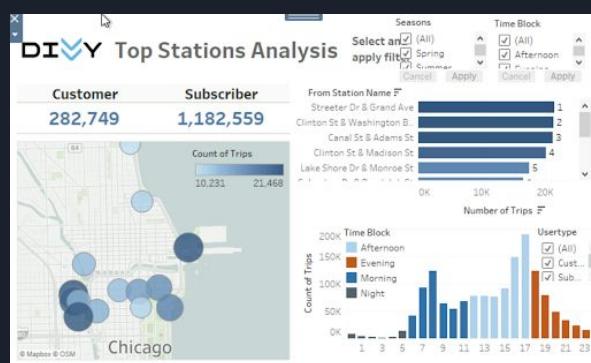
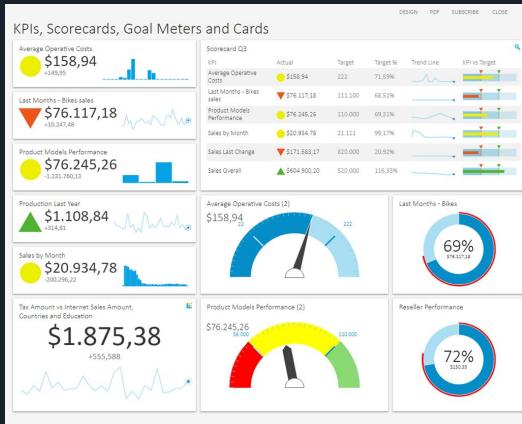
Tableau BI (Business Intelligence) est une suite d'outils analytiques et de visualisation de données développée par Tableau Software, une société fondée en 2003 et rachetée par Salesforce en 2019.

Tableau BI est conçu pour aider les organisations à transformer de grandes quantités de données en informations exploitables grâce à des visualisations.

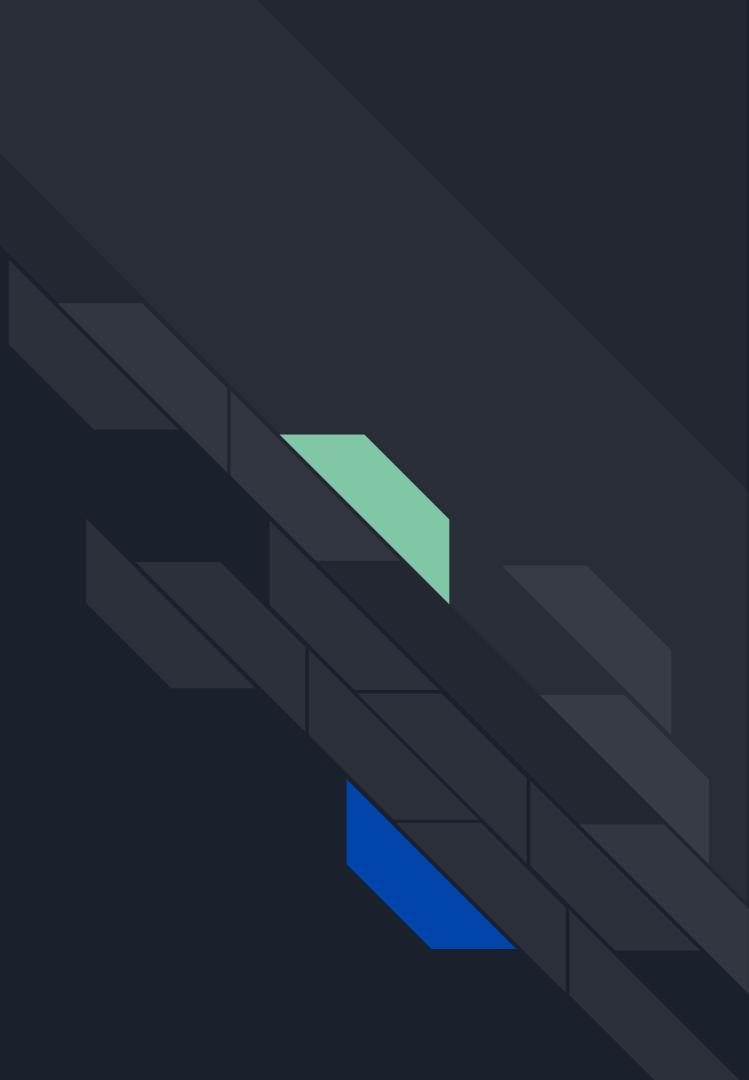
Son objectif principal est de faciliter la prise de décision basée sur des données en rendant l'analyse accessible à tous les niveaux d'une organisation, sans nécessiter de compétences techniques avancées.



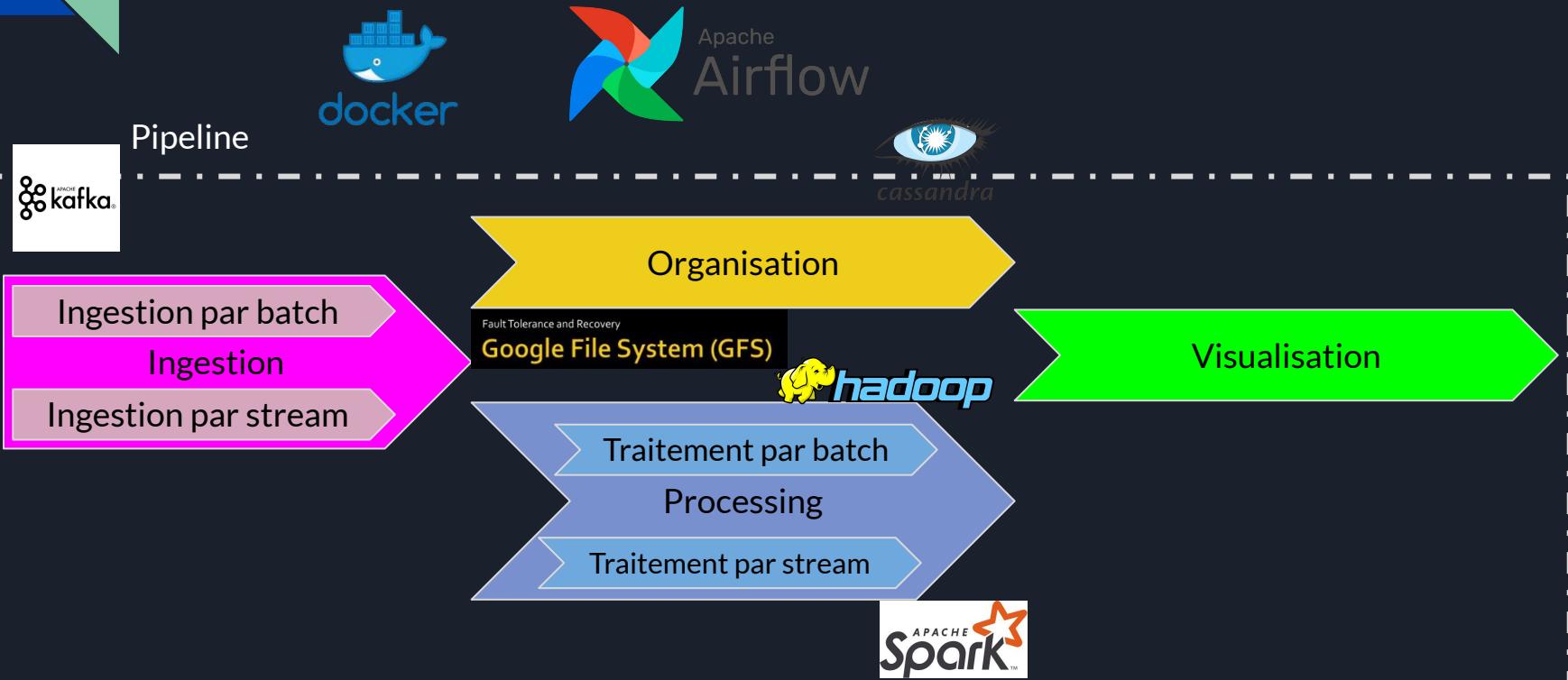
# Tableau BI



# Logging et Monitoring



# Pipeline outil



# Introduction

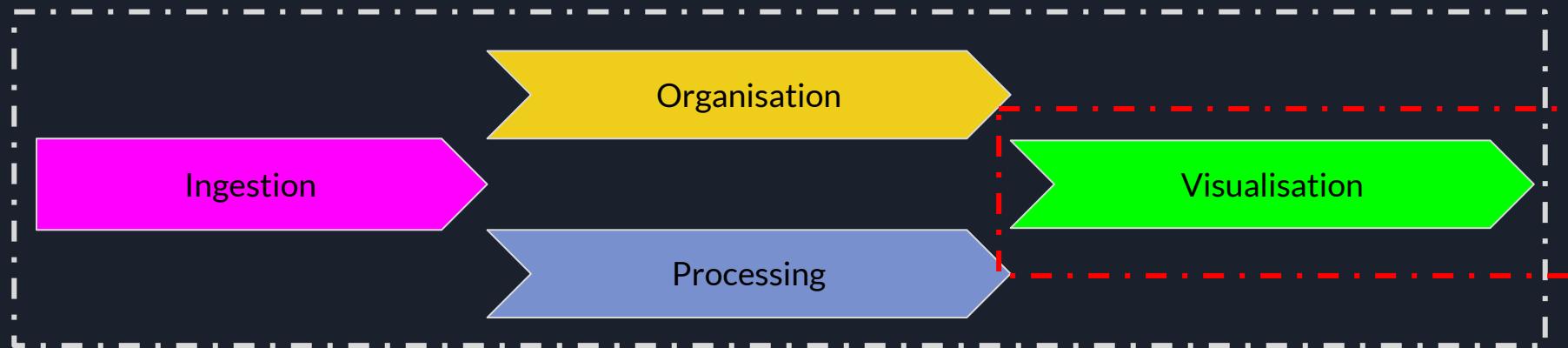
Pipeline

Ingestion

Organisation

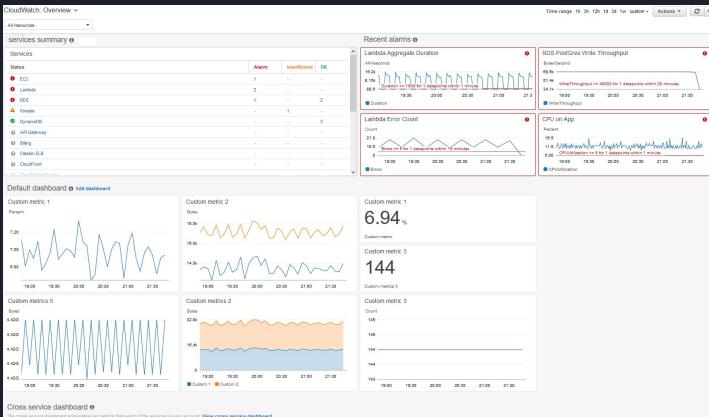
Processing

Visualisation



# Introduction

Le monitoring et le logging sont des pratiques clés pour assurer la disponibilité, la performance et la sécurité des services web.

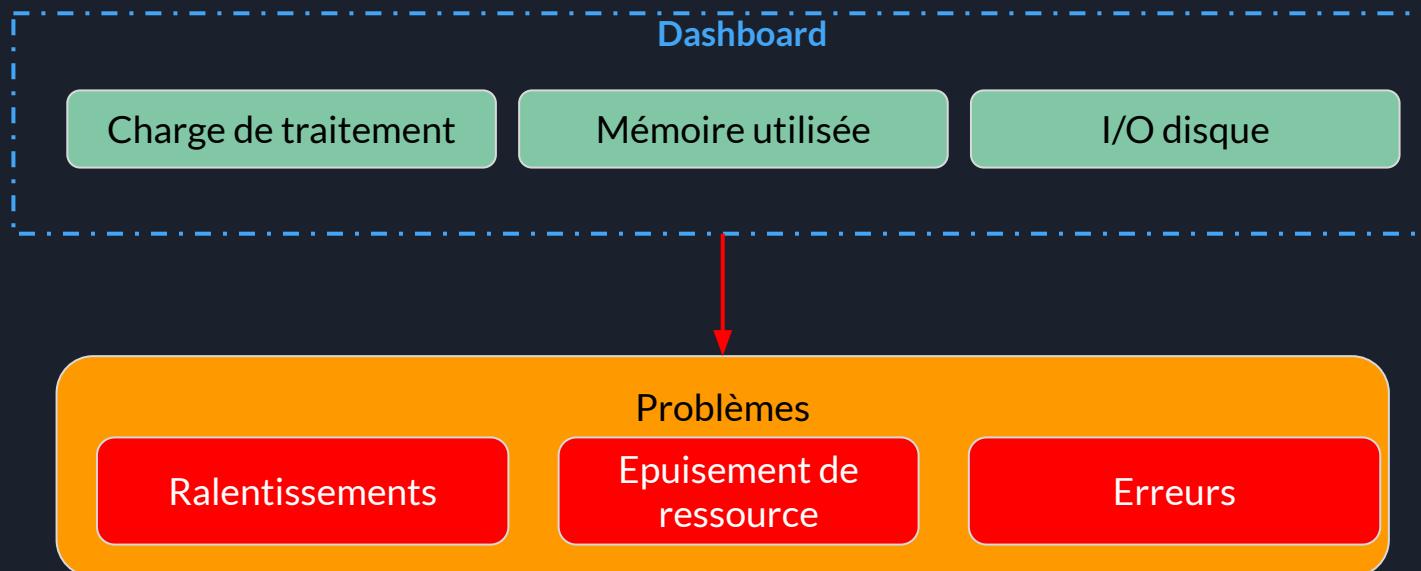


## Monitoring

## Logging

# Monitoring

Le monitoring est la surveillance continue d'un système ou d'une application pour détecter les anomalies et les problèmes potentiels.



# Logging

Le logging est l'enregistrement de données sur l'état et les activités d'un système.

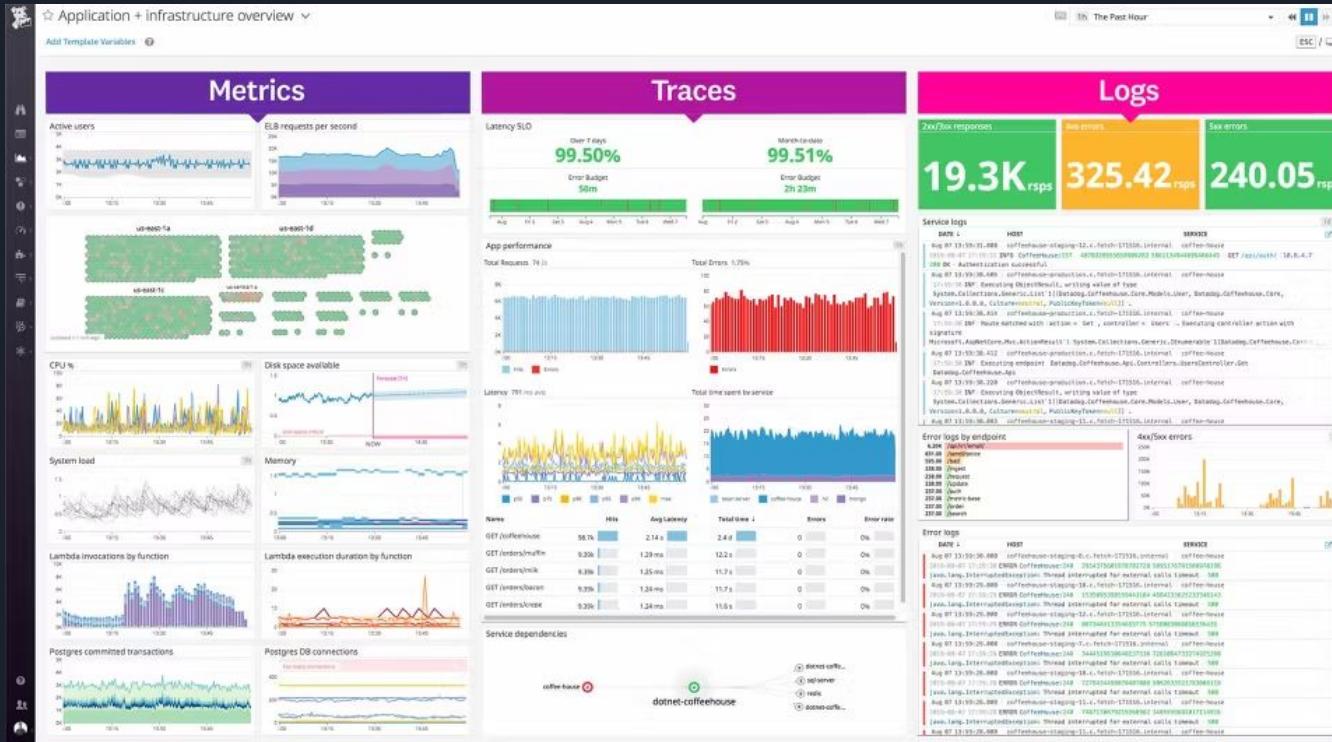
Il permet de récupérer des informations sur les erreurs et les anomalies qui se produisent, et de les utiliser pour résoudre les problèmes, comprendre les tendances et identifier les tendances à venir.

Les données de journalisation peuvent également être utilisées pour répondre aux exigences réglementaires ou pour la conformité.

```
Nov 29 10:10:02 lograzer filebeat[21206]: 2020-11-29T10:10:02.2782#011INFO#011[monitoring]#011log/lo
g.go1:4#011Non-zero metrics in the last 30s#011"monitoring": {"metrics": {"beat": {"cpu": {"system":
["ticks":167030], "total": ["ticks":360860, "time": {"ms":3}], "value":360860}, "user": {"ticks":193830, "time": {"ms":3}}}}, "handles": {"limit": {"hard":4096, "soft":1024}, "open":12}, "info": {"ephemeral_id": "ec16b3
47-4538-416b-a781-5892dd598d34", "uptime": {"ms":11829500721}}, "memstats": {"gc_next": 52521504, "memory_a
lloc": 27085744, "memory_total": 35717398272}, "runtime": {"goroutines":493}, "filebeat": {"open
_files":3, "running":2}, "libbeat": {"config": {"module": {"running":0}}, "pipeline": {"clients":4, "even
ts": [{"active":4118}], "registrar": {"states": {"current":4}}}, "system": {"load": {"1":0, "15":0, "5":0, "norm
": [{"1":0, "15":0, "5":0}]}}}}
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.1222#011ERROR#011pipeline/output.go:10
0#011Failed to connect to backoff(elasticsearch(http://192.168.1.114:9200)): Get http://192.168.1.11
4:9200: EOF
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.1222#011INFO#011pipeline/output.go:93#
011Attempting to reconnect to backoff(elasticsearch(http://192.168.1.114:9200)) with 1570 reconnect
attempts()
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.1232#011INFO#011[publisher]#011pipelin
e/retry.go:196#011retryer: send unsignal signal to consumer
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.1232#011INFO#011[publisher]#011pipelin
e/retry.go:198#011 done
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.1232#011INFO#011[publisher]#011pipelin
e/retry.go:173#011retryer: send wait signal to consumer
Nov 29 10:10:09 lograzer filebeat[21206]: 2020-11-29T10:10:09.1232#011INFO#011[publisher]#011pipelin
e/retry.go:175#011 done
Nov 29 10:10:32 lograzer filebeat[21206]: 2020-11-29T10:10:32.2792#011INFO#011[monitoring]#011log/lo
g.go1:4#011Non-zero metrics in the last 30s#011"monitoring": {"metrics": {"beat": {"cpu": {"system":
["ticks":167030, "time": {"ms":3}], "total": ["ticks":360870, "time": {"ms":6}], "value":360870}, "user": {"ticks":193840, "time": {"ms":3}}}}, "handles": {"limit": {"hard":4096, "soft":1024}, "open":12}, "info": {"eph
erical_id": "ec16b347-4538-416b-a781-5892dd598d34", "uptime": {"ms":11829500721}}, "memstats": {"gc_next": 52
521504, "memory_alloc": 27487040, "memory_total": 35717799568}, "runtime": {"goroutines":493}, "filebeat": {
"harvester": {"open_files":3, "running":2}, "libbeat": {"config": {"module": {"running":0}}, "output": {"re
ad": [{"errors":1}, {"write": {"bytes":125}}], "pipeline": {"clients":4, "events": [{"active":4118, "retry":3}]}}, "registrar": {"states": {"current":4}}}, "system": {"load": {"1":0.13, "15":0.01, "5":0.03, "norm": [{"1":0.13, "15":0.01, "5":0.03}]}}}
```

Logging

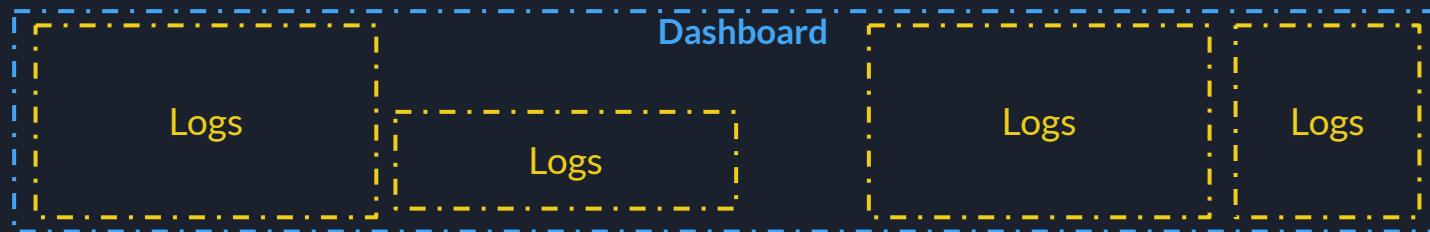
# Monitoring



Datadog

# Le monitoring

Les technologies de monitoring comme Datadog permettent aux administrateurs de systèmes de surveiller les performances de leurs services web en temps réel, collecter des métriques et des données de journalisation de leurs applications, et utiliser des alertes et des rapports pour identifier les problèmes potentiels avant qu'ils ne causent des perturbations de service.





# Les objets à moniter

Serveurs Web

Bases de  
données

Services  
d'authentification

Charge réseau

# Objectif du logging



Génère

## Journalisation

Log (Adresse IP, heure de connexion, lieu, activité...)

## Analyse Préventive

Déetecter un comportement inhabituel sur un SI et remonter le comportement pour voir si c'est un faux positif ou un évènement de cybersécurité

## Forensic

“Police scientifique” Étude des logs pour trouver des traces des pirates qui ont organisé une attaque

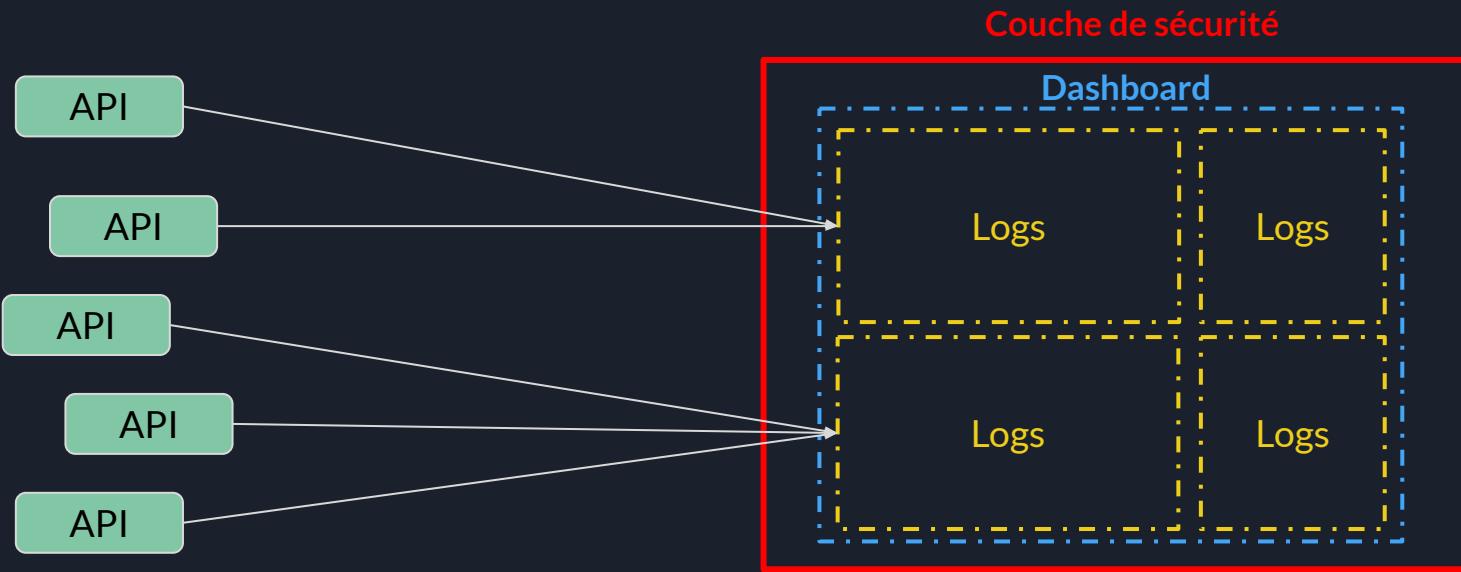


# Objectif du logging

En utilisant des outils de monitoring et de logging, les administrateurs de systèmes peuvent surveiller les performances des services web en temps réel, détecter les problèmes potentiels avant qu'ils ne causent des perturbations de service, et répondre aux exigences réglementaires et à la conformité.

# Sécurité des outils de monitoring

Il est important de noter que la sécurité de ces outils de monitoring et de logging doit être prise en considération pour éviter les fuites de données ou les accès non autorisés aux données collectées. Il est donc essentiel de mettre en place les protocoles de sécurité adéquats et de suivre régulièrement les politiques de sécurité.



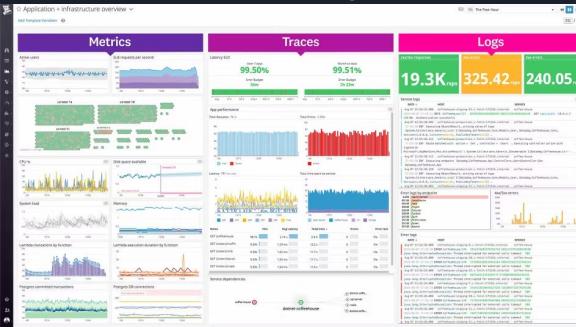
# Les outils pour du monitoring et log

Etude de cas (30 minutes)

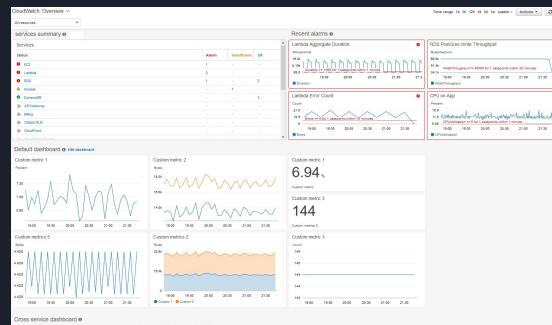
Groupe 1  
Grafana/Prometheus



Groupe 2  
DataDog



Groupe 3  
AWS CloudWatch



# Grafana

Tableaux de bord interactifs

Dashboard personnalisé Grafana

Graphiques

Prometheus

InfluxDB

Graphite

Elasticsearch

Format de données

Ingestion temps réel

Données de performance



Open source

Visualisation de données

Monitoring

Analyse de tendances

Alertes temps réel

Collaboration temps réel

# Grafana

## Dashboard personnalisé Grafana



Équipe DevOps

Surveillance de l'infrastructure temps réel

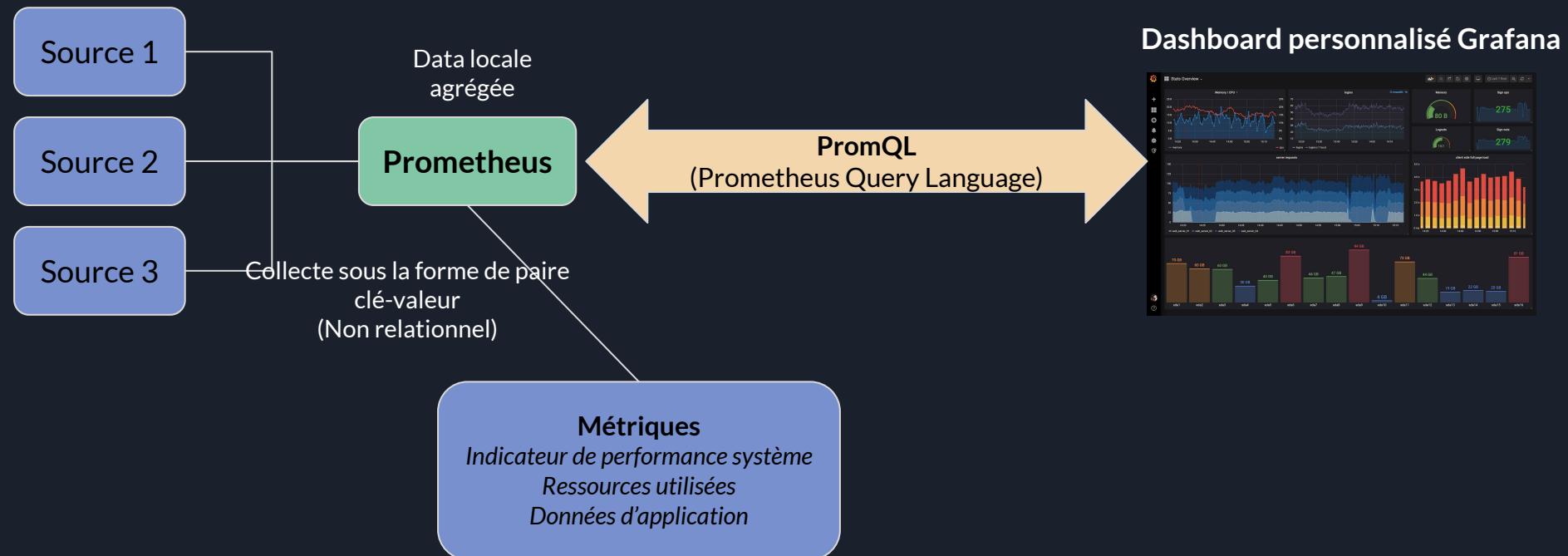
Équipe Cybersécurité

Identification des problèmes de sécurité temps réel

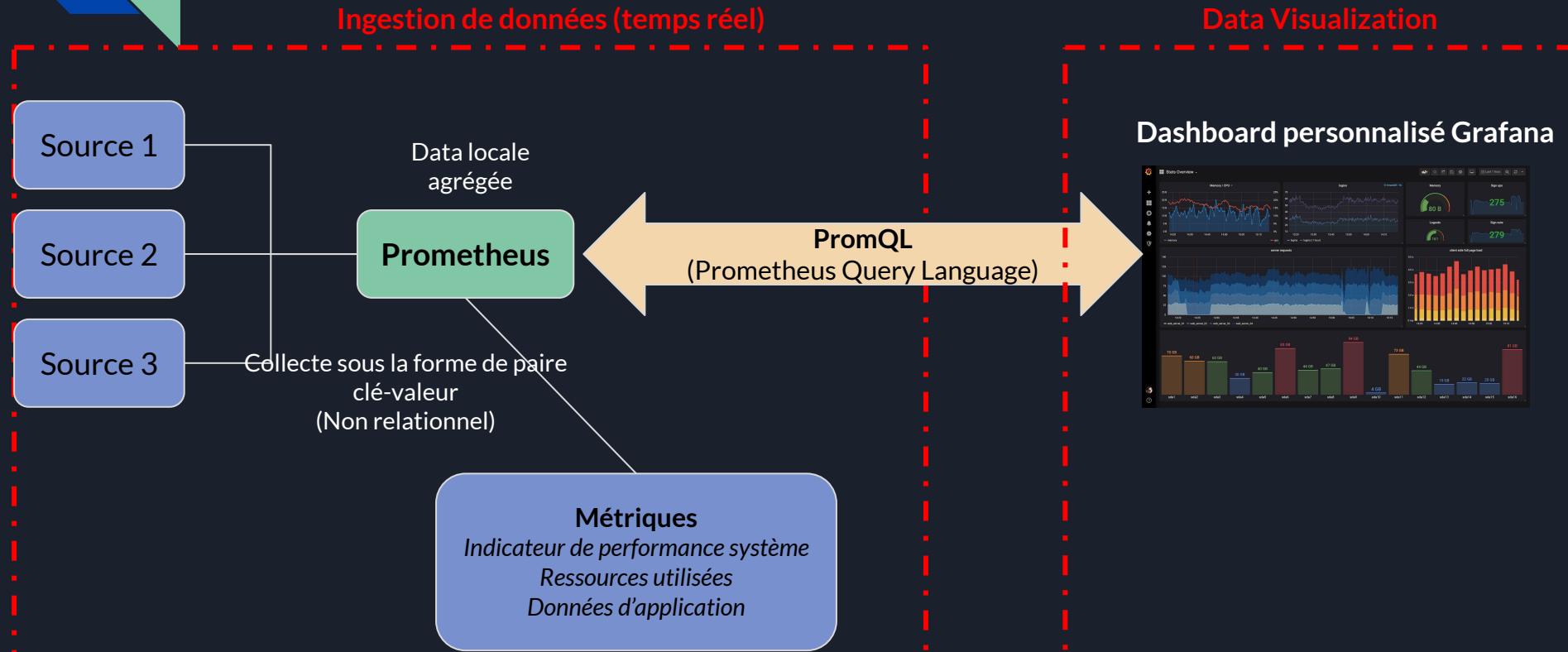
Équipe  
Gestion de performance

Gestion des performances temps réel

# Grafana et Prometheus

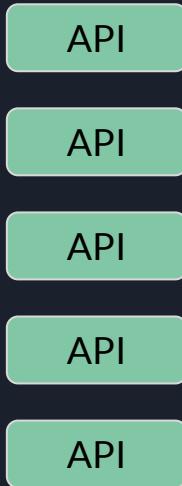


# Grafana et Prometheus

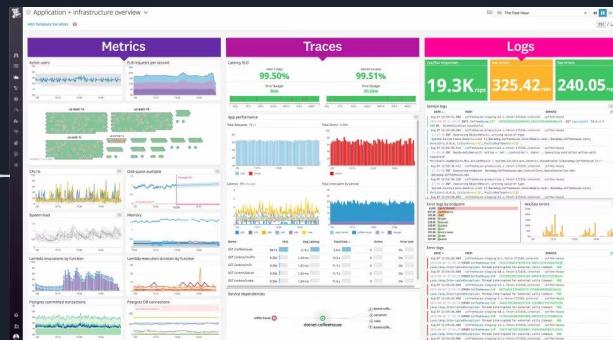


# Datadog

Services Web



Métriques stockées dans un système distribué  
Indicateur de performance système  
Ressources utilisées  
Données d'application



Cloud

AWS

Azure

Google Cloud Platform

Alertes pour des valeurs limites

Utilisation de ressources anormales

Erreurs

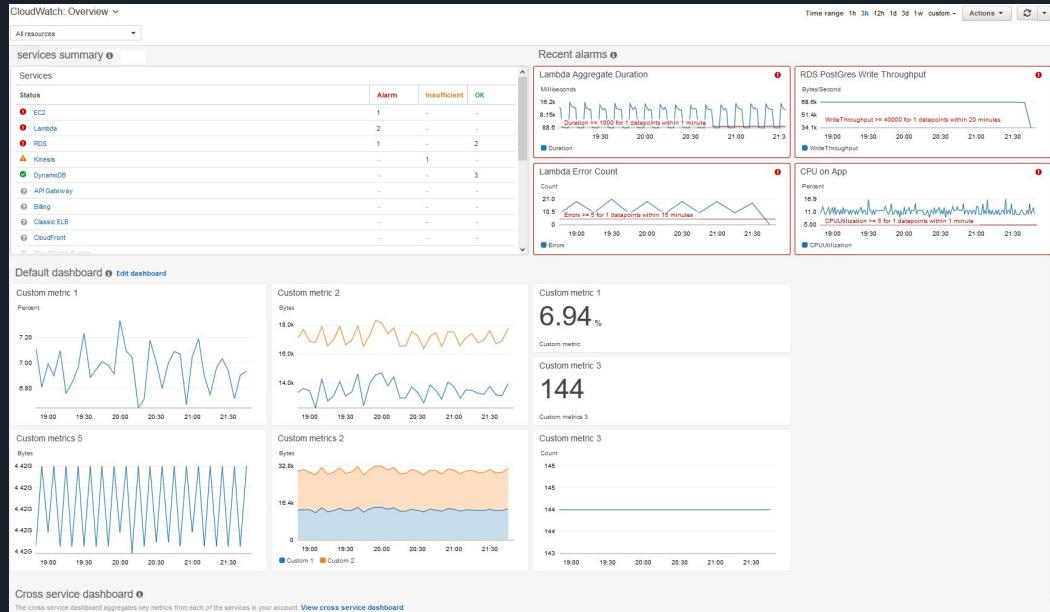
Analyse de journaux

Surveillance de l'infrastructure

Performance des requêtes

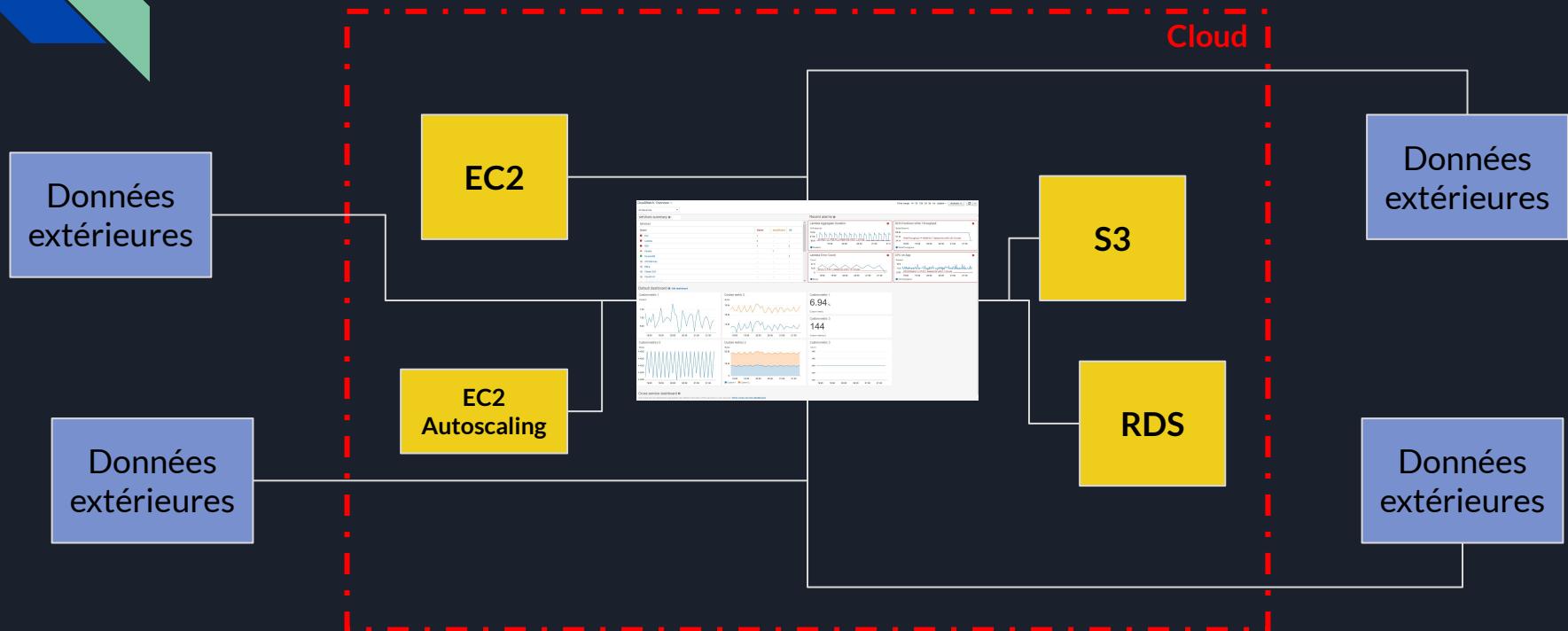
# AWS CloudWatch

AWS Cloudwatch est la plateforme de surveillance et de gestion des performances d'Amazon Web Services

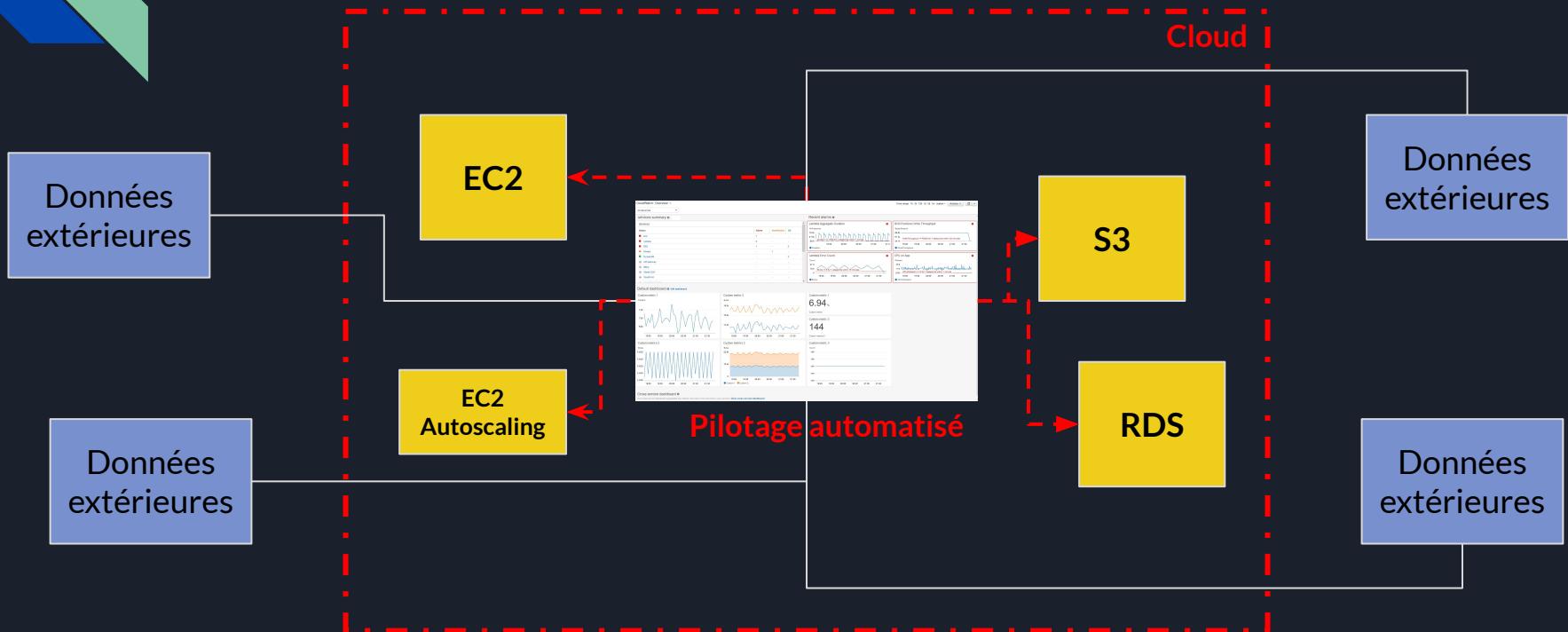


*Un dashboard CloudWatch*

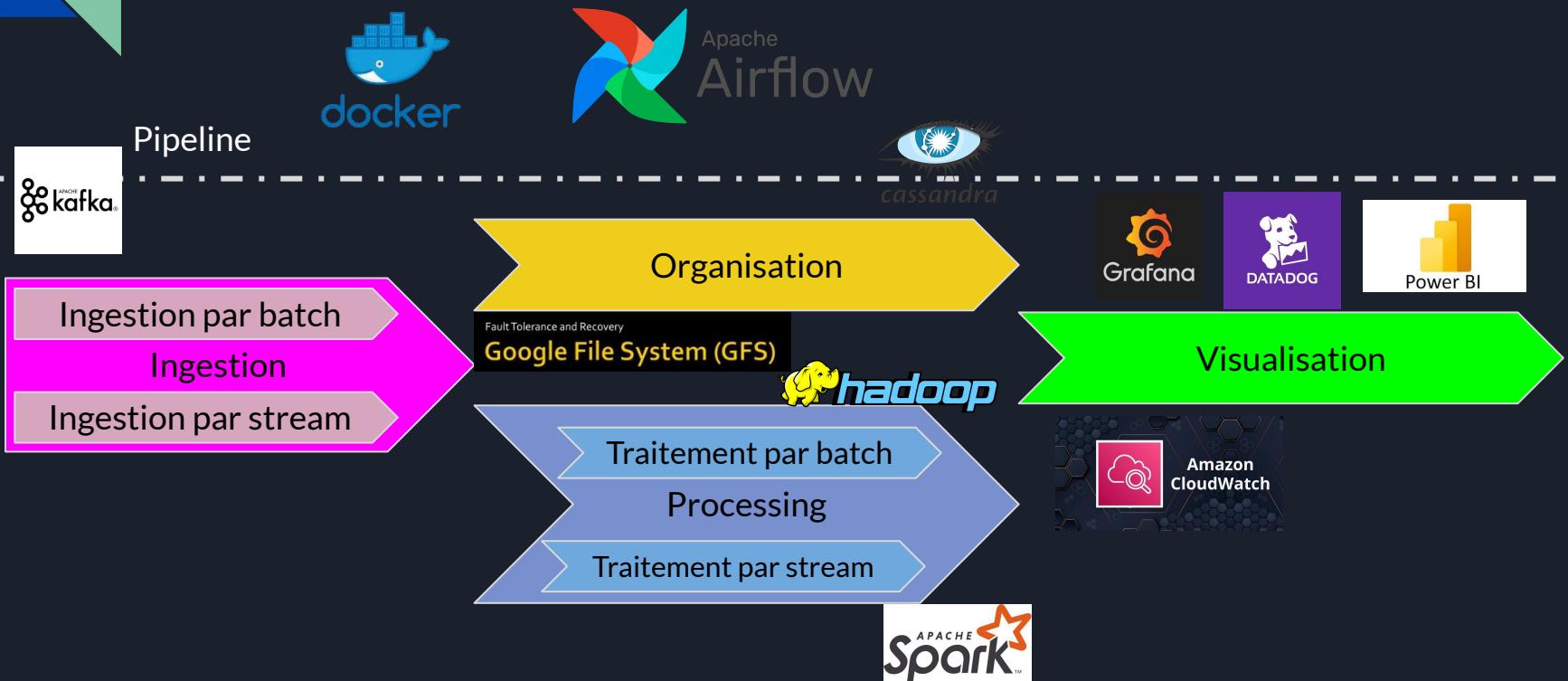
# AWS CloudWatch



# AWS CloudWatch



# Pipeline outil





# Intégration BI et Big Data

## Intégration BI et Big Data

- Architecture Lambda et Kappa
- Cas pratiques d'intégration des technologies BI avec des solutions Big Data

# Architecture Lambda

L'architecture Lambda est un modèle de traitement des données qui est largement utilisé dans le cadre du Big Data pour gérer et traiter de grandes quantités de données en temps réel et en mode batch.

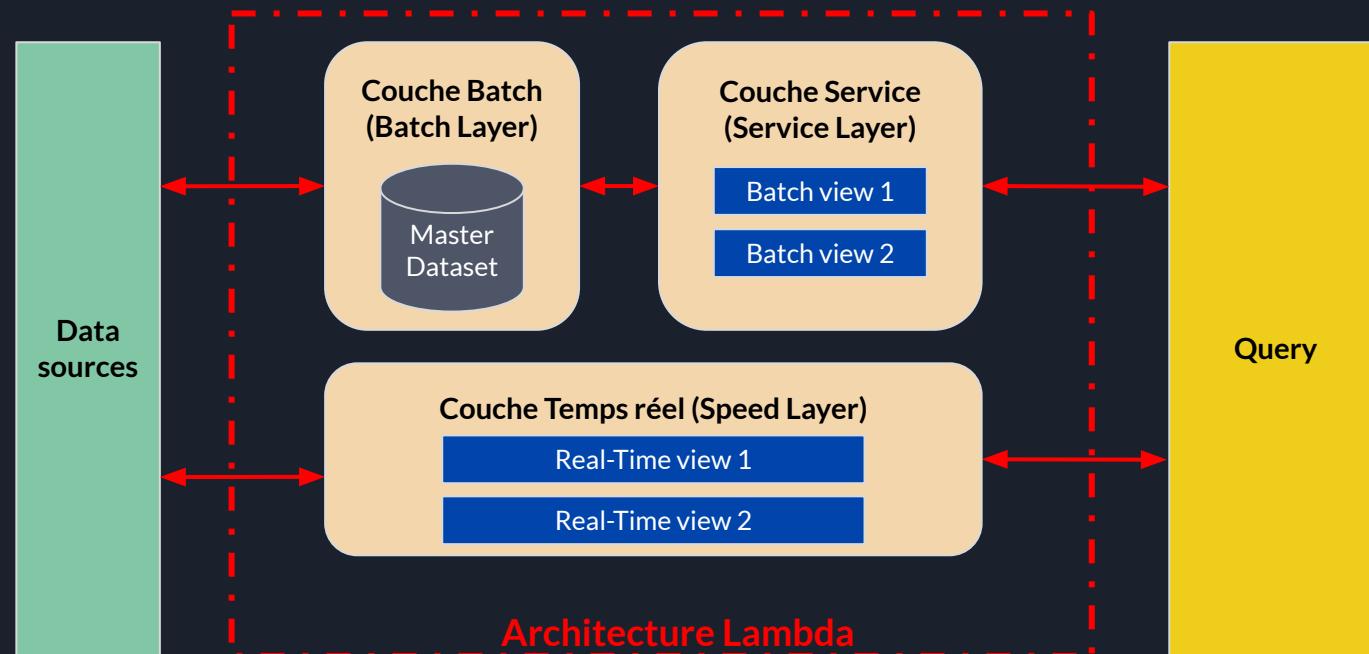
Proposée initialement par Nathan Marz, l'architecture Lambda répond à la nécessité de traiter des données à la fois en temps réel (ou quasi temps réel) et en mode différé (batch) de manière efficace et scalable.



*Nathan Marz*

# Composants de l'Architecture Lambda

L'architecture Lambda se compose de trois couches principales :

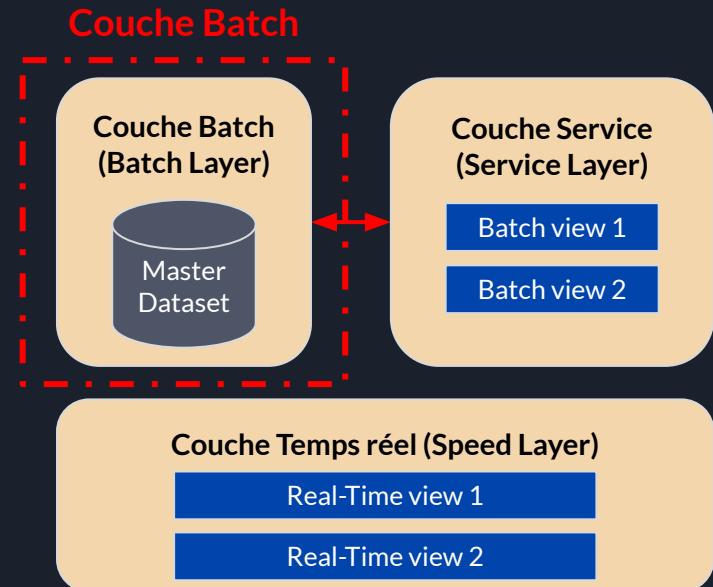


# Composants de l'Architecture Lambda

Objectif : Traiter de grandes quantités de données en mode différé pour fournir des résultats complets et précis.

Dans cette couche, les données sont accumulées sur une période donnée, puis traitées en bloc (batch). Les résultats de ces traitements sont souvent stockés dans un entrepôt de données ou un Data Lake.

Stockage de la vue : Les résultats du traitement batch sont stockés dans ce qu'on appelle une "vue batch" (batch view), qui peut être interrogée pour des analyses approfondies.



## Technologies couramment utilisées

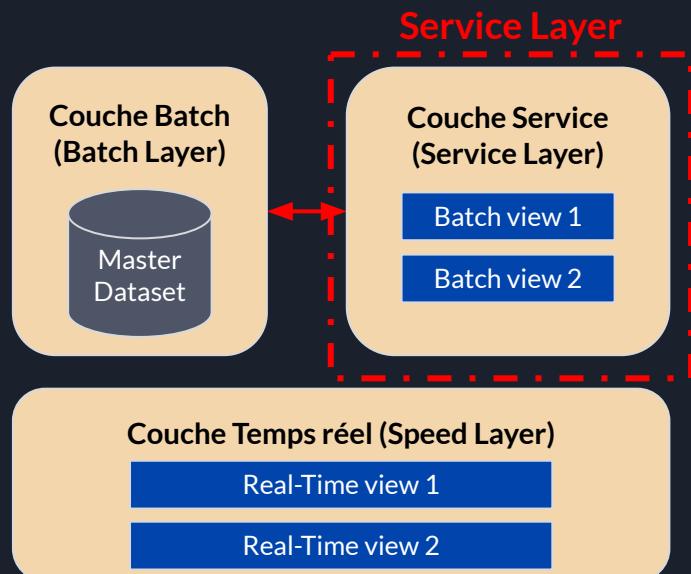


Google  
BigQuery



amazon  
EMR

# Composants de l'Architecture Lambda



Objectif : Traiter les données en temps réel pour fournir des résultats immédiats, mais potentiellement approximatifs.

Cette couche traite les données au fur et à mesure qu'elles arrivent. Elle est conçue pour répondre rapidement aux changements dans les données, offrant ainsi une faible latence dans la génération des résultats.

Les résultats en temps réel sont stockés dans une "vue temps réel" (real-time view), qui peut être interrogée parallèlement à la vue batch pour des analyses en temps réel.

## Technologies couramment utilisées



kafka



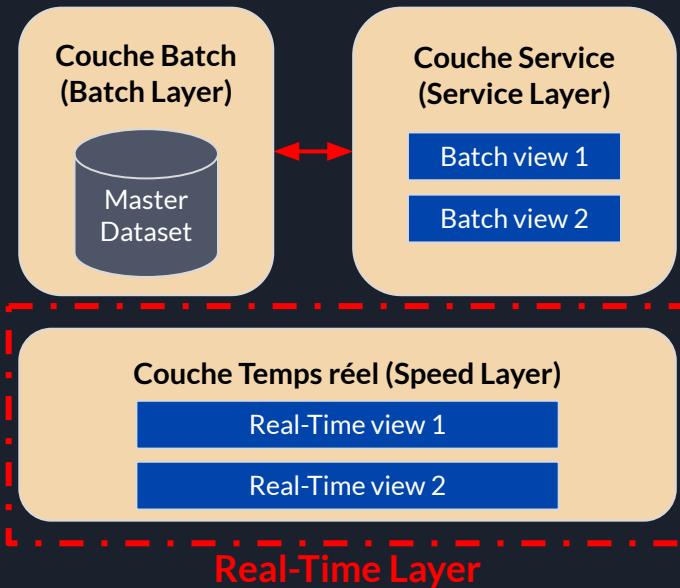
APACHE  
STORM™



Apache Flink



# Composants de l'Architecture Lambda



**Objectif :** Combiner les résultats des couches batch et temps réel et les rendre accessibles pour les requêtes.

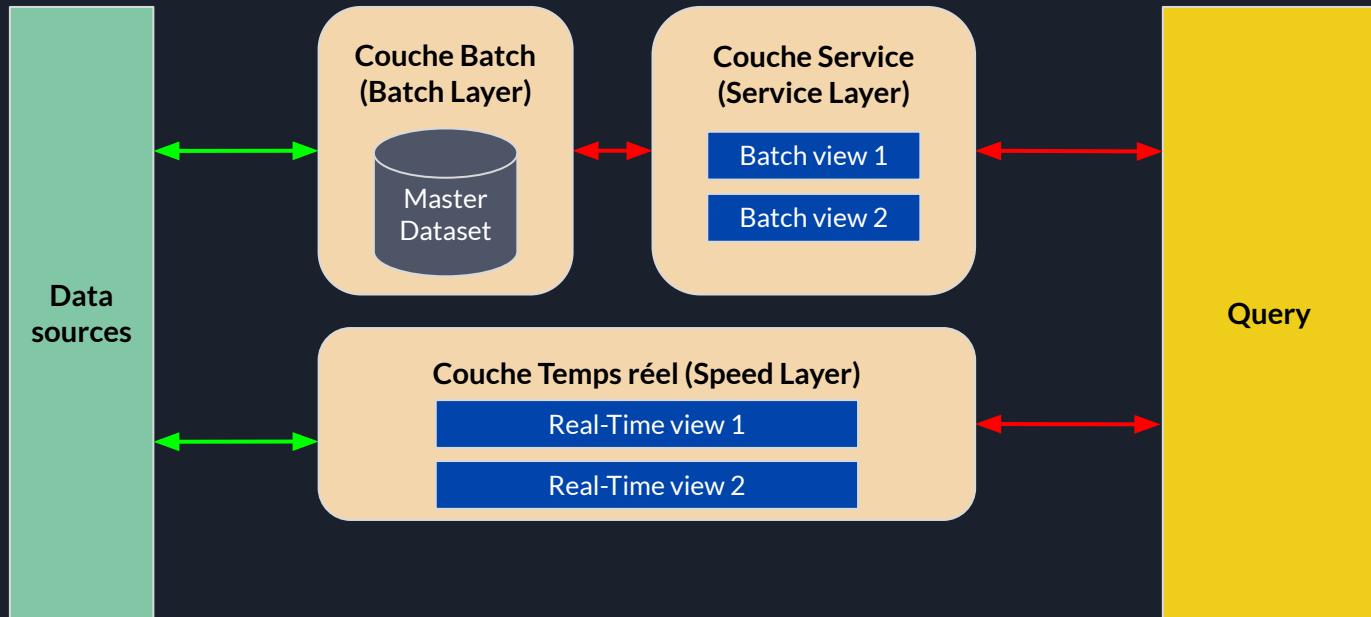
**Fonctionnement :** Cette couche unifie les données issues des deux couches précédentes. Lorsqu'une requête est effectuée, les données sont extraites à la fois de la vue batch (pour des résultats précis) et de la vue temps réel (pour des résultats récents). Cela permet d'obtenir une réponse rapide et complète.

## Technologies couramment utilisées



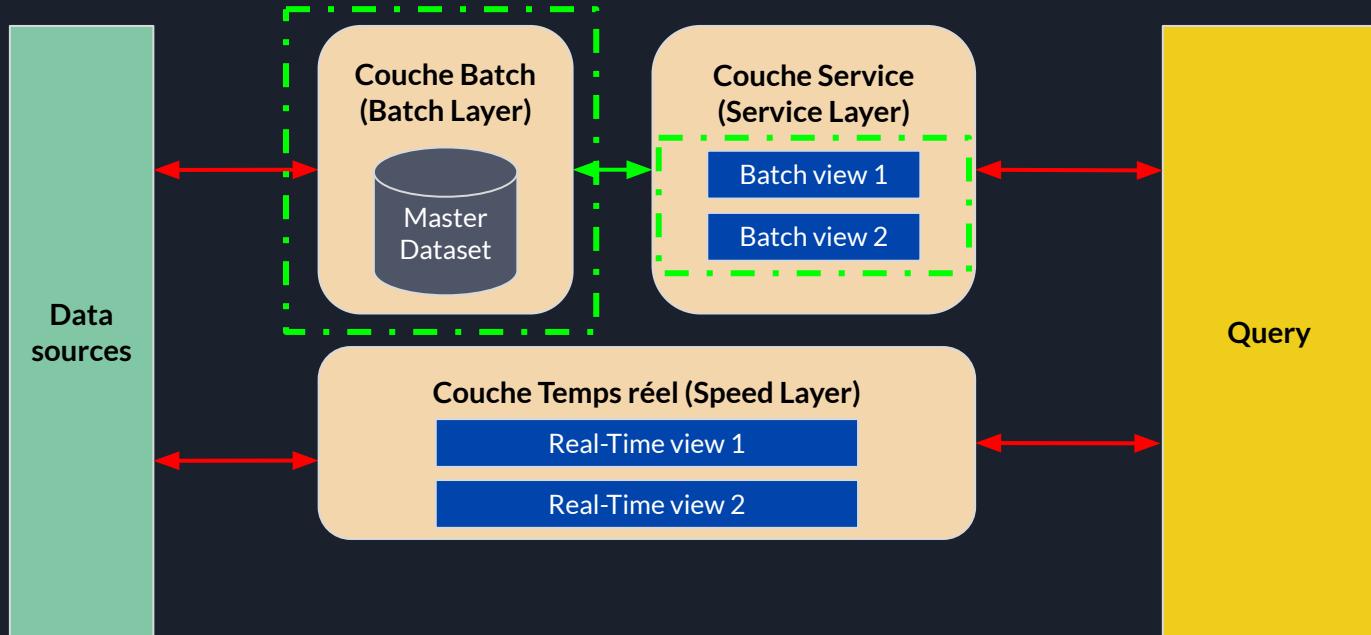
# Fonctionnement global de l'Architecture Lambda

Les données brutes entrent dans le système et sont simultanément envoyées à la couche Batch et à la couche Temps Réel.



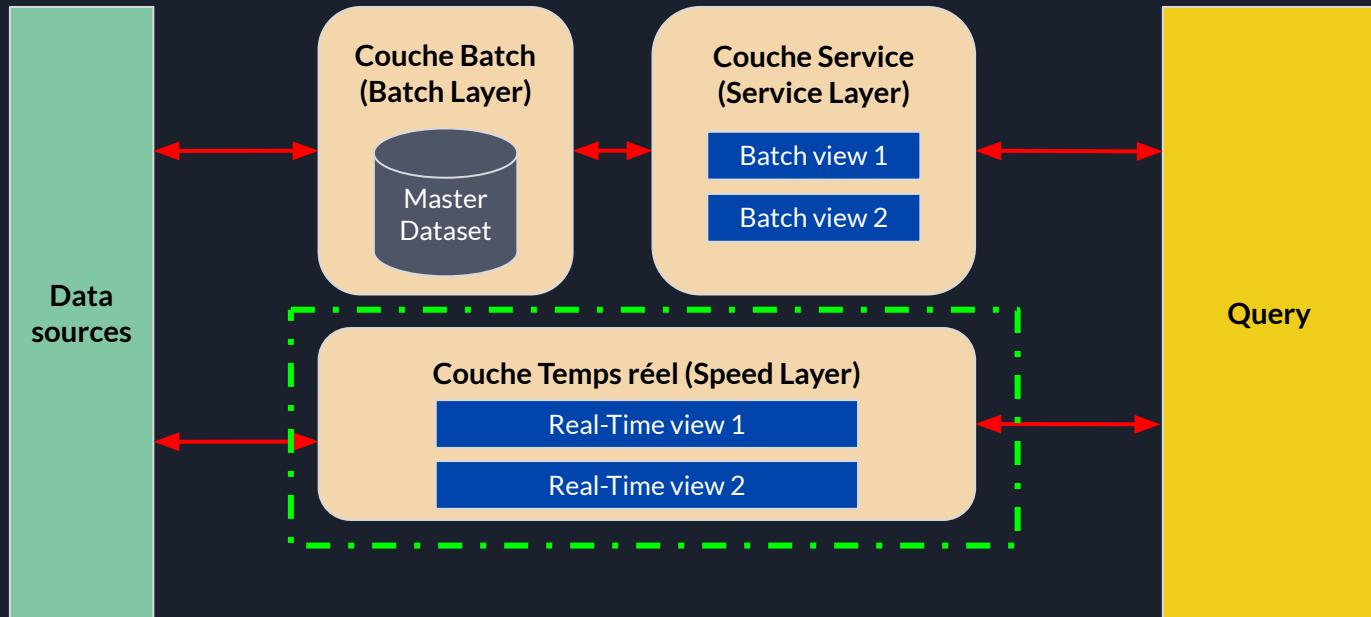
# Fonctionnement global de l'Architecture Lambda

La couche Batch traite les données en mode différé, exécutant des tâches d'analyse lourdes qui peuvent prendre du temps. Les résultats sont stockés pour une utilisation ultérieure.



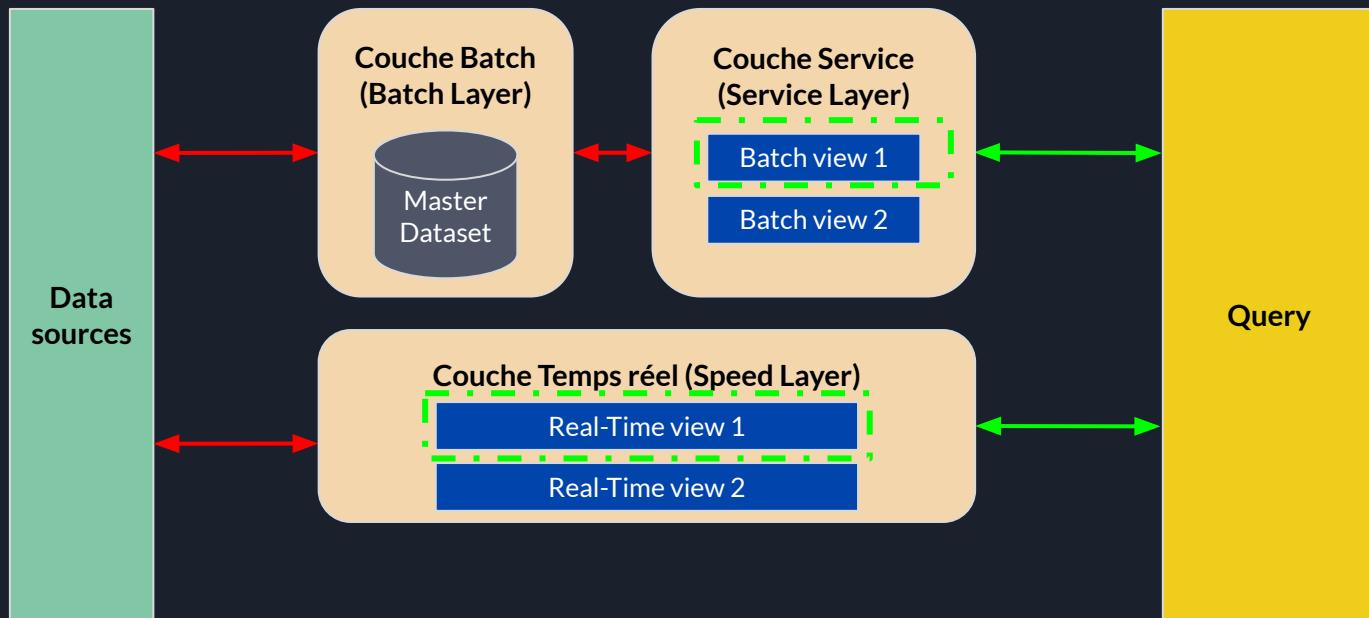
# Fonctionnement global de l'Architecture Lambda

En parallèle, la couche Temps Réel traite les données à la volée, offrant des mises à jour rapides, mais avec une précision moindre.



# Fonctionnement global de l'Architecture Lambda

Lors de la consultation des données, la couche de Service combine les résultats des deux couches pour fournir une vue d'ensemble à jour et précise.



# Avantages et inconvénients de cette Architecture

## Avantages

En **combinant** les approches batch et temps réel, elle est **résiliente face aux pannes** et garantit que les données sont toujours **disponibles**, qu'elles soient traitées immédiatement ou avec un léger retard.

Elle permet de gérer efficacement les flux de données **massifs**, tout en répondant aux besoins d'analyses en **temps réel**.

Elle est conçue pour **évoluer** facilement avec l'augmentation des volumes de données.

## Inconvénients

Mettre en œuvre et maintenir une architecture Lambda peut être **complexe**, car il nécessite la gestion de **deux pipelines de données distincts** (batch et temps réel).

Certaines opérations de traitement des données doivent être exécutées **à la fois** dans la couche **batch** et la couche **temps réel**, entraînant une **duplication** des efforts et des ressources.



# Une alternative : L'architecture Kappa

En réponse à la complexité de l'architecture Lambda, une alternative appelée architecture Kappa a été proposée.

L'architecture Kappa simplifie l'approche en traitant **toutes les données comme des flux**, éliminant ainsi la nécessité d'une couche batch distincte.

Dans ce modèle, tout le traitement est effectué en **temps réel**, et les traitements en batch sont simulés en rejouant les flux de données historiques.

# L'architecture Kappa

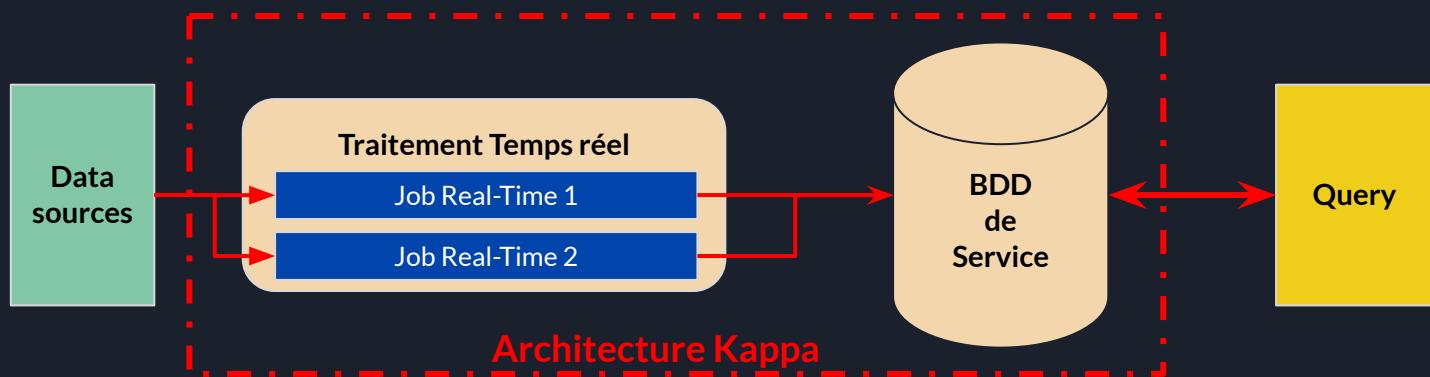
L'architecture Kappa est une approche simplifiée pour le traitement des données dans le cadre du Big Data, proposée par Jay Kreps, l'un des co-créateurs d'Apache Kafka. Elle a été développée comme une alternative à l'architecture Lambda, afin de résoudre certains des problèmes de complexité et de redondance associés à cette dernière.



*Jay Kreps*

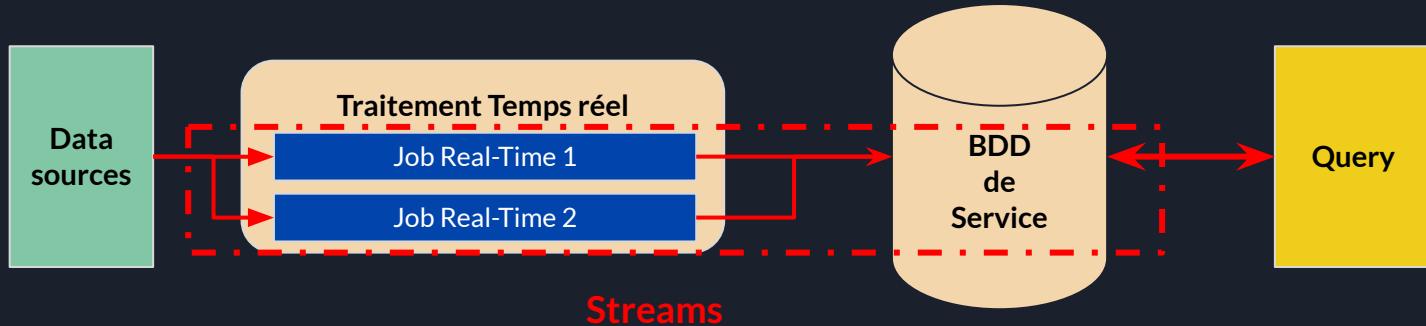
# Principe de l'Architecture Kappa

L'architecture Kappa repose sur un principe fondamental : traiter toutes les données comme des flux **continus**, éliminant ainsi la nécessité de gérer séparément des pipelines batch et temps réel. Dans une architecture Kappa, il n'y a qu'un seul pipeline de traitement des données, où toutes les données sont ingérées, transformées, et analysées en temps réel.



# Composants de l'Architecture Kappa

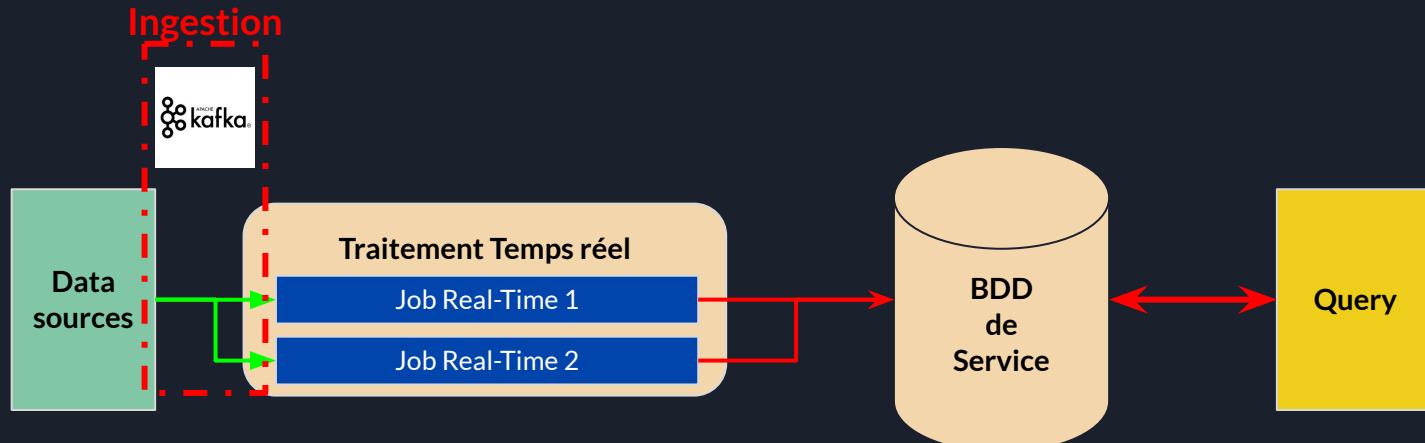
Toutes les données sont traitées sous forme de **flux** (streams), qu'il s'agisse de données en **temps réel ou de données historiques**. Les événements ou les messages sont produits par des sources de données (bases de données, capteurs, logs, etc.) et sont ensuite consommés par des applications ou des systèmes de traitement.



# Composants de l'Architecture Kappa

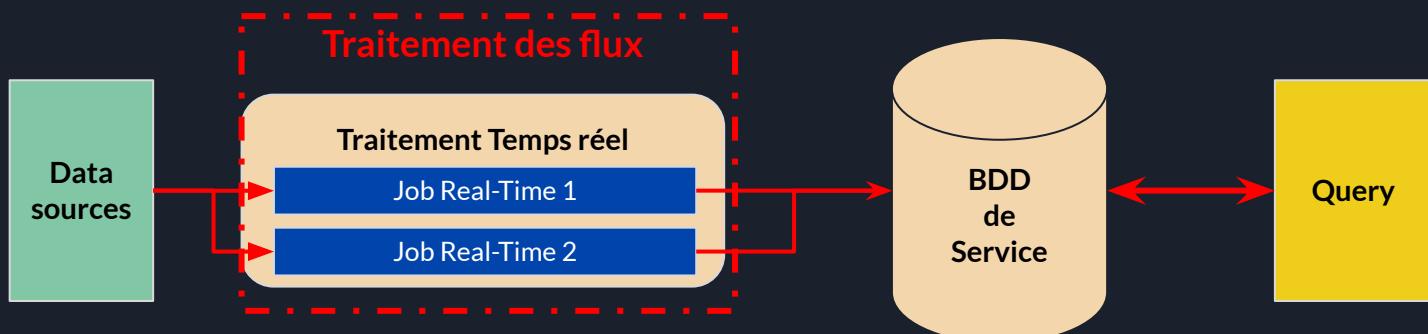
Un système de messagerie distribué, tel qu'Apache Kafka, est généralement utilisé pour ingérer les flux de données. Kafka capture les données en temps réel et les stocke de manière durable, permettant aux consommateurs de lire les flux en temps réel ou de rejouer les anciens événements pour des traitements de type batch.

Si des ajustements dans le traitement sont nécessaires (comme un changement dans la logique de transformation), les données historiques peuvent être rejouées depuis Kafka pour recalculer les résultats, simuler des traitements batch.



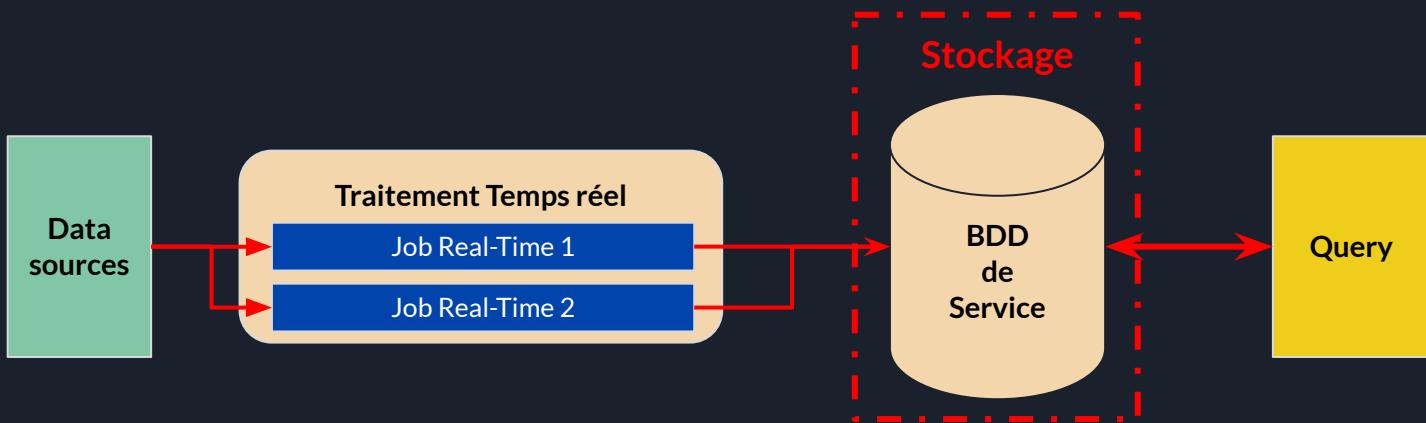
# Composants de l'Architecture Kappa

Le traitement des flux est effectué en temps réel en utilisant des moteurs de traitement des flux (stream processing) tels qu'Apache Flink, Apache Samza, ou Kafka Streams. Ces moteurs consomment les événements du flux, appliquent des transformations, des agrégations, des filtrages, et d'autres opérations de traitement.



# Composants de l'Architecture Kappa

Les résultats du traitement en temps réel sont stockés dans des systèmes de bases de données, des caches, ou des entrepôts de données (comme Elasticsearch, Cassandra, HBase) pour permettre des analyses et des requêtes rapides.



# Avantages et inconvénients de cette Architecture

## Avantages

Éliminer la distinction entre batch et temps réel réduit considérablement la complexité du système. Un **seul pipeline de traitement à gérer**.

**Pas de duplication** des efforts entre le traitement batch et temps réel.

La possibilité de rejouer les données permet de mettre à jour ou de corriger les traitements **sans interrompre le flux en temps réel**.

Le traitement en temps réel est **hautement scalable**, Kafka permet de gérer des volumes de données très importants.

## Inconvénients

L'architecture Kappa est **fortement dépendante** des capacités de traitement des flux en temps réel.

Si un traitement batch complexe est nécessaire, il peut être plus difficile à implémenter et à optimiser dans un pipeline purement en streaming.

Pour certaines analyses complexes qui bénéficient de la puissance d'un traitement batch traditionnel, l'architecture Kappa peut être moins adaptée.



# Cas d'utilisation de l'Architecture Kappa

Analyse en Temps Réel : Surveillance de l'état des systèmes, détection d'anomalies, traitement des données IoT.

Applications de Streaming : Applications où le traitement immédiat des données est crucial, comme les plateformes de trading en ligne, les réseaux sociaux en temps réel.

Rejet pour le Traitement Batch : Applications où la logique de traitement évolue fréquemment et où la capacité de rejouer des événements historiques est un avantage majeur.