

# **BIG DATA :**

## **ENJEUX, STOCKAGE ET EXTRACTION**

**MODULE R6.01 - ANNÉE 2025-2026**

# A PROPOS DE CE MODULE

Ce cours est une introduction aux problématiques d'**exploitation des données** sur les plateformes modernes, dans le contexte aujourd'hui standard de "**Big Data**".

Il aborde des sujets qui vont concerner plusieurs profils au sein de l'entreprise : data engineer, data scientist, architecte, mais aussi data manager...



# A PROPOS DE MOI

**Consultant chez OCTO Technology (groupe Accenture)**  
depuis 2007



Thomas VIAL  
[thomas.vial@octo.com](mailto:thomas.vial@octo.com)  
+33 6 28 50 07 64

Au sein de l'équipe **SmartIndustry**, qui apporte des solutions innovantes de data et d'IA aux clients industriels d'OCTO

**Plein de choses m'intéressent :** architecture de SI, de données, data engineering, data science, modélisation mathématique, ...

**Certifications Hadoop et Google Cloud**

# PROGRAMME

- Introduction
- Plateforme de données
- Anatomie des données
- Les traitements en Big Data
- Le cloud
- Pour un usage responsable des données

# EVALUATION

- Un évaluation écrite (fin du 2<sup>ème</sup> CM)
- Un TP/TD noté sur 4
- Des questions de compréhension pour les autres TP/TD
- Rattrapages organisés en cas d'absence

# INTRODUCTION

# BIG DATA EN 2026

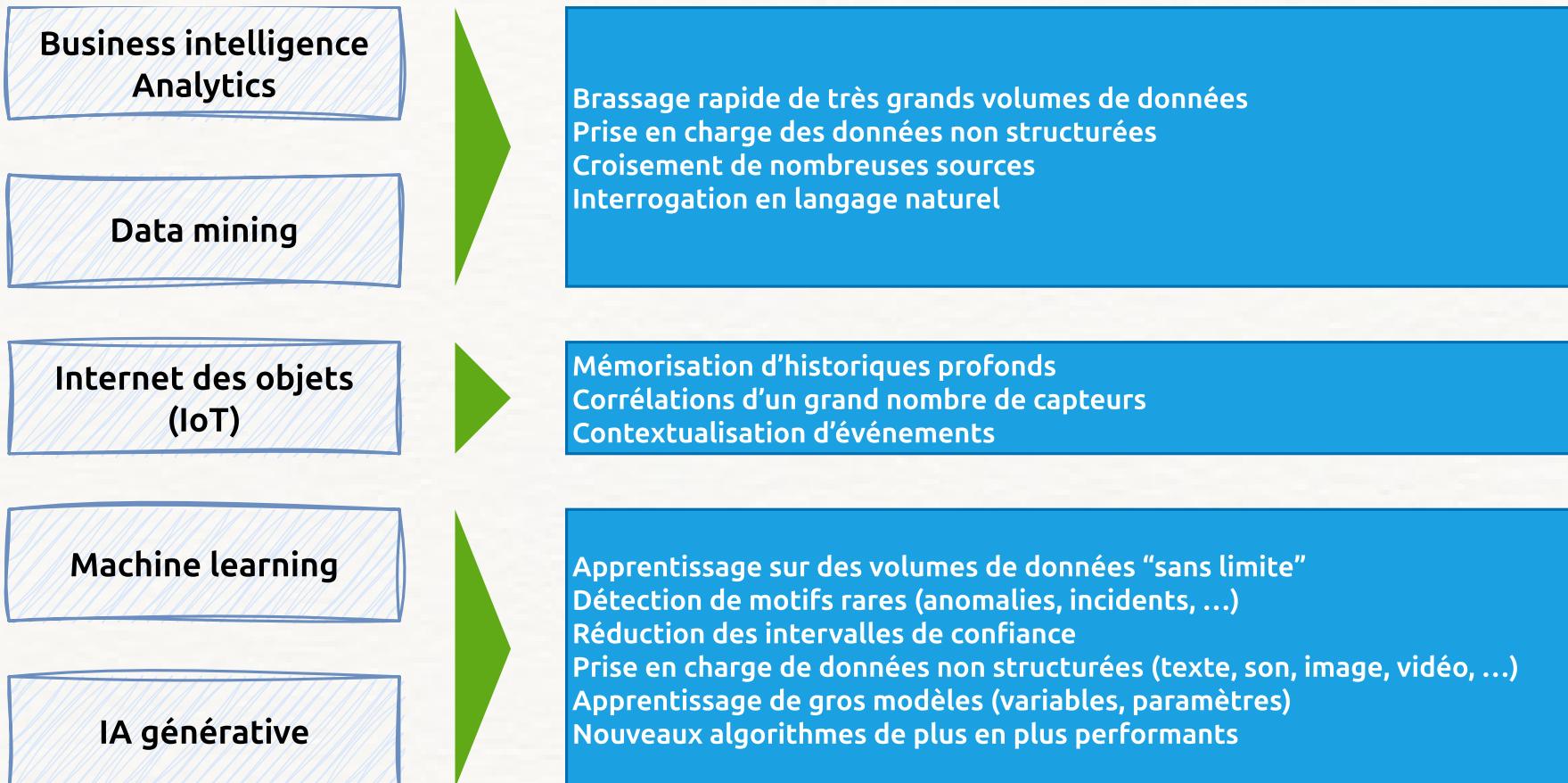
Big Data a longtemps été une affaire de spécialistes, dont les besoins très particuliers (les fameux "3 V") exigeaient des techniques de pointe.

Aujourd'hui, avec le cloud, "Big Data" = "Data" tout court.

L'héritage de Big Data se retrouve à plusieurs niveaux :

- outillage : la mise à disposition de **plateformes de données**
- modélisation : des pratiques de **mise en forme et de stockage**
- traitements : des **algorithmes** adaptés à la manipulation efficace
- gouvernance : des **règles de bon usage**, surtout pour les données sensibles

# AVANTAGES PAR RAPPORT À L'INFORMATIQUE "TRADITIONNELLE"

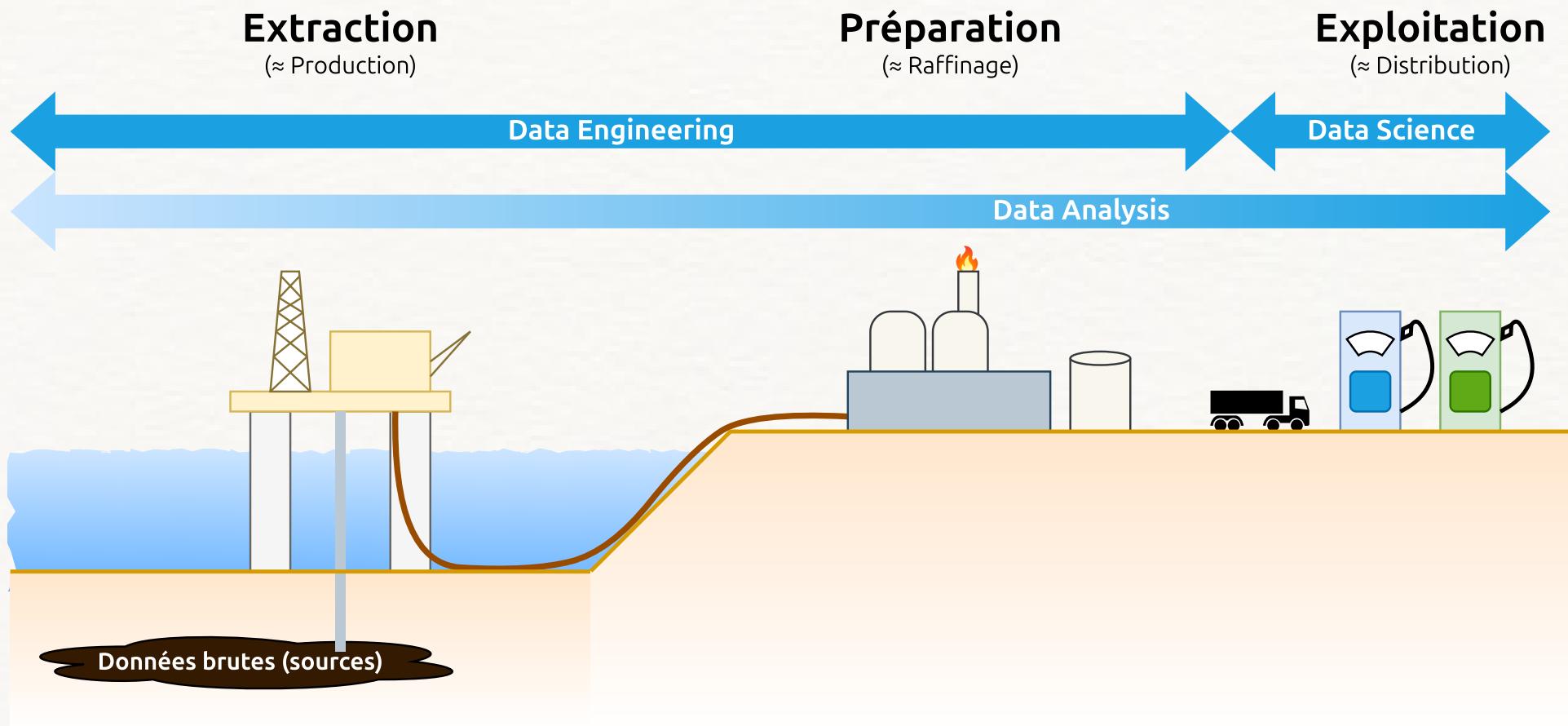


# ACTIVITÉS & MÉTIERS DE LA DATA



**Quels sont les activités et métiers de la data ?**

# ACTIVITÉS & MÉTIERS DE LA DATA

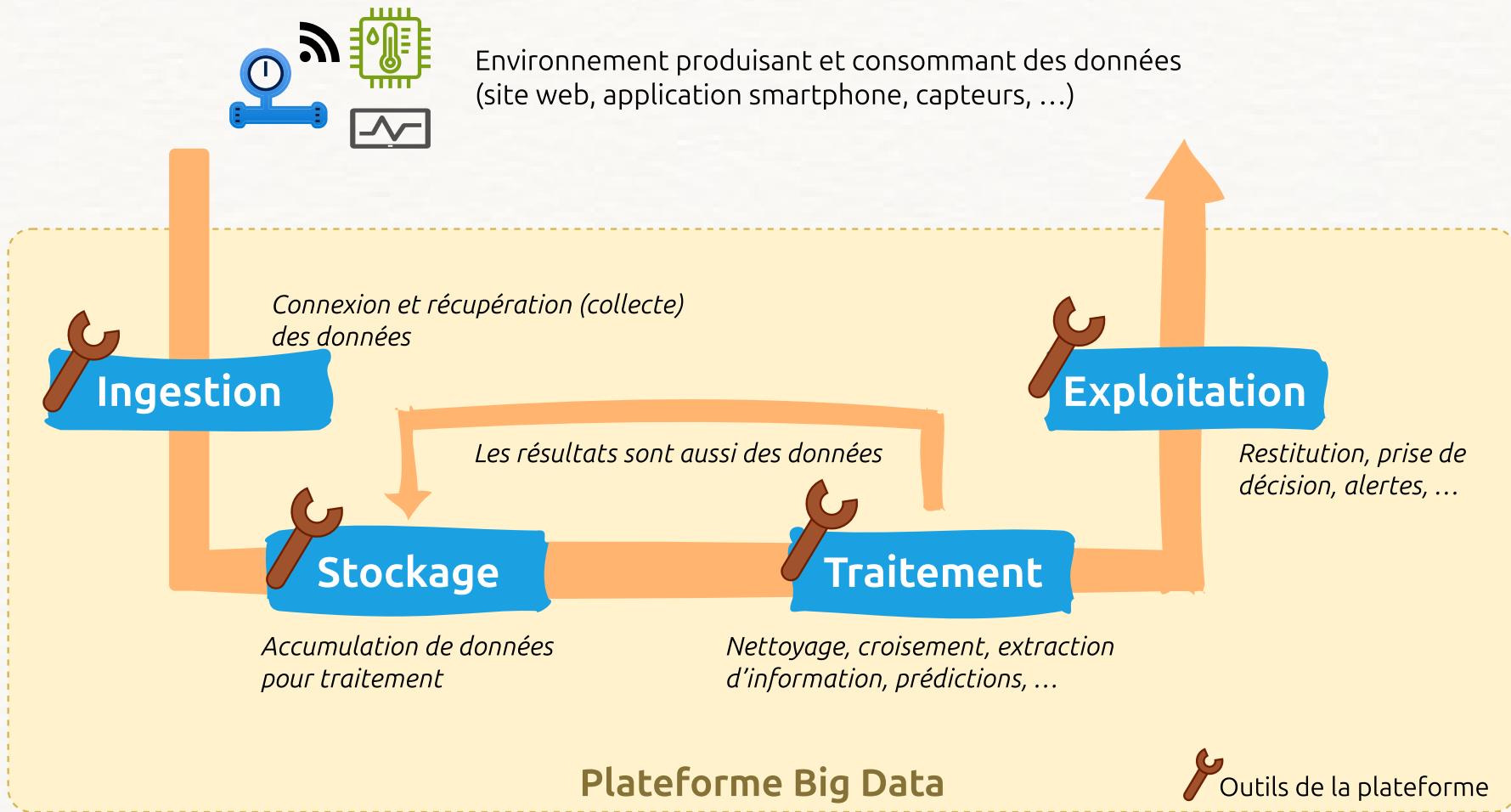


# DONNÉES, INFORMATIONS, ACTIONS

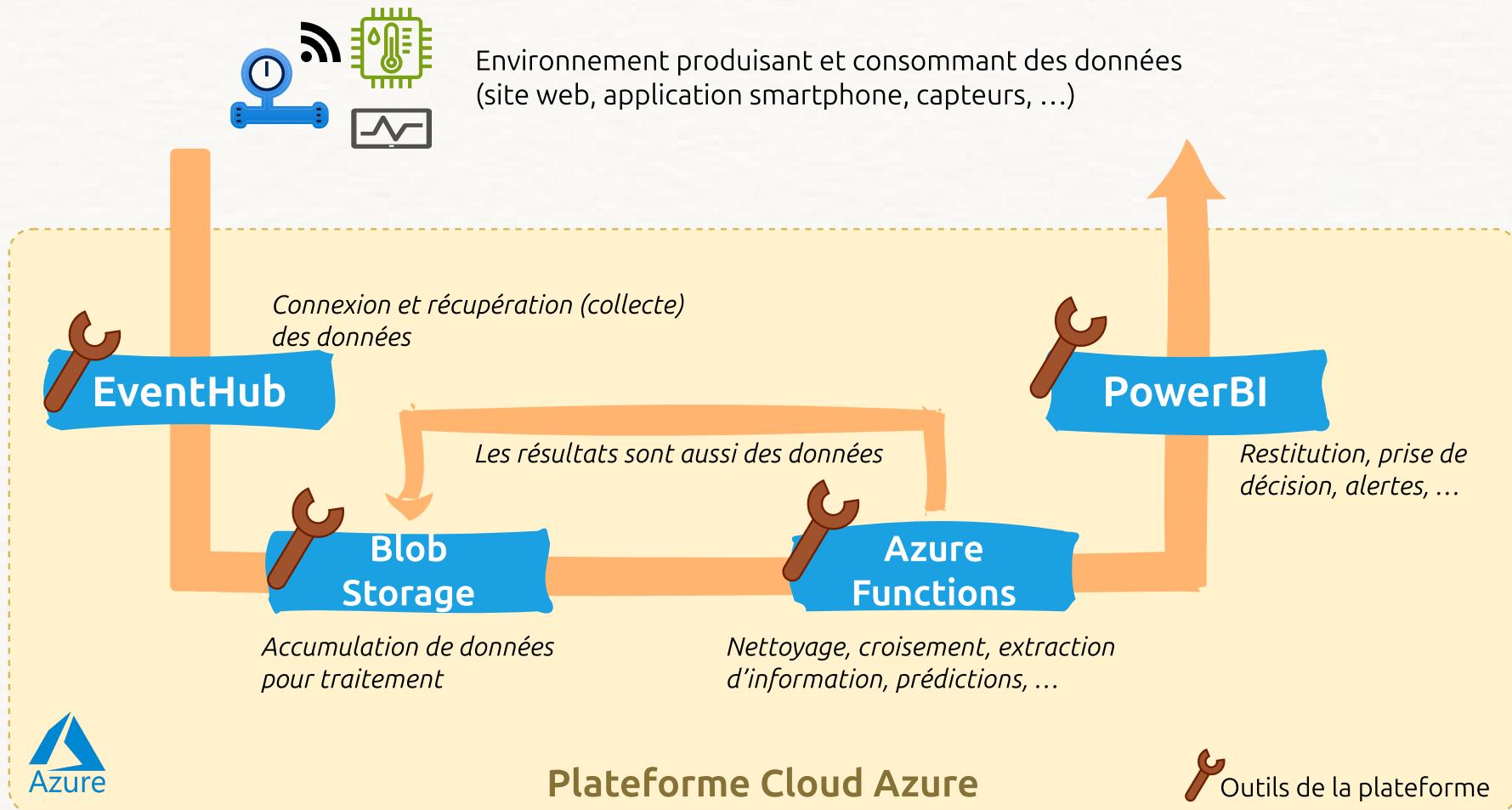
# PLATEFORME DE DONNÉES

Matériel et logiciels dédiés au big data

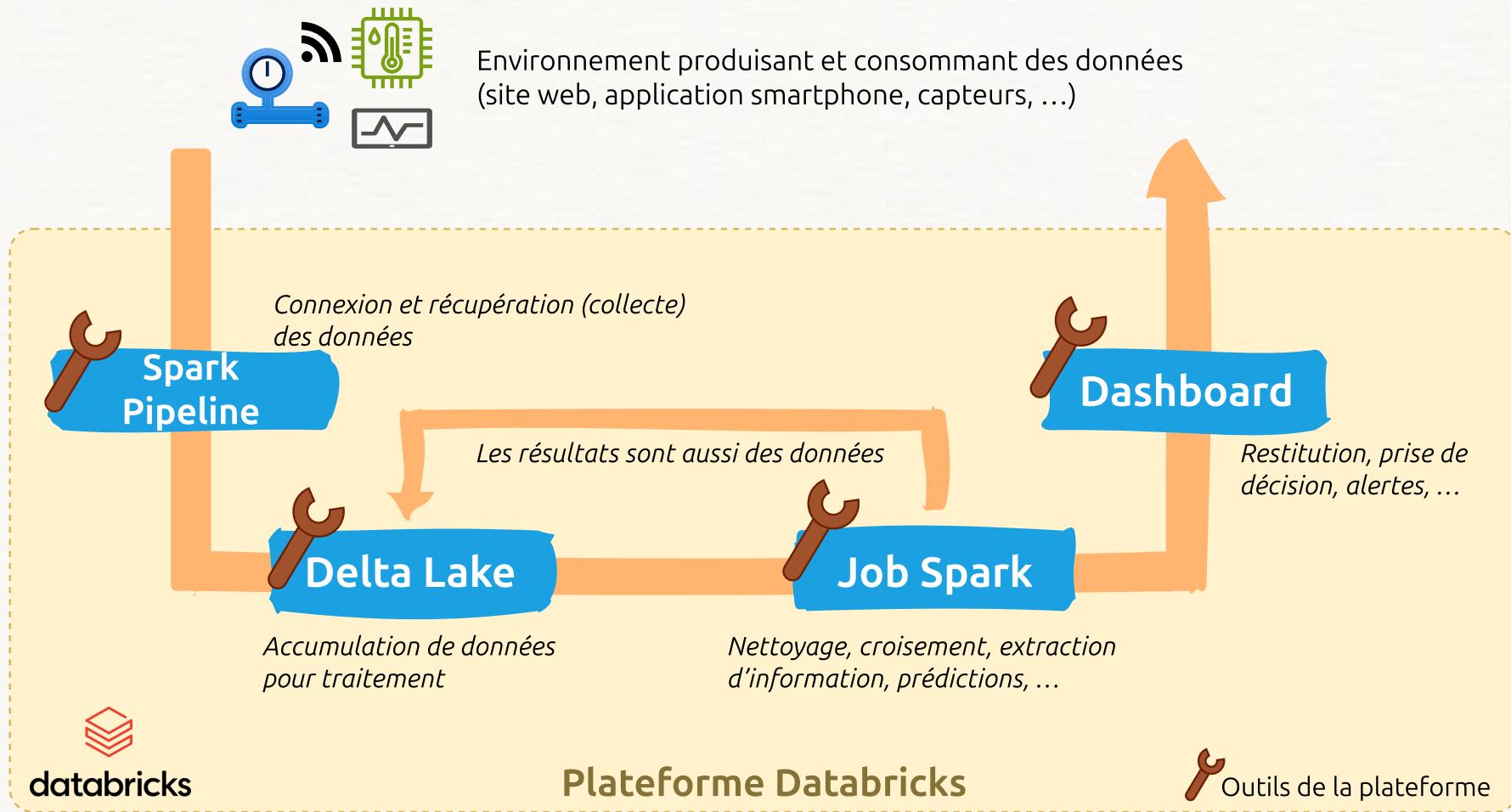
# RÔLES D'UNE PLATEFORME



# EXEMPLE 1



# EXEMPLE 2



# INGESTION

Amène la donnée vers le stockage de la plateforme. En règle générale, c'est une copie à l'identique des données brutes récupérées.

On rencontre souvent :

- des **espaces et API** de dépôt
- des **ETL** (*Extract - Transform - Load*) pour copier fichiers et bases de données
- des **files de messages (brokers)** pour l'ingestion en temps réel



# STOCKAGE



Offre de vastes espaces où les données d'entrée, produites, etc. peuvent s'accumuler de manière fiable.

Gère aussi le cycle de vie des données (ex. archivage, purge), leur chiffrement et les droits d'accès.

Quelques types de stockage courants :

- le **stockage "objet"** pour des gros fichiers et les **data lakes**
- les **bases de données relationnelles ou non** (NoSQL)

# TRAITEMENT

Assure la bonne exécution du code des traitements demandés à la plateforme. Le code doit être adapté à la plateforme en question.

On va y trouver deux grands types de composants :

- les **moteurs de calcul** exécutent des traitements unitaires (*jobs*) de manière performante
- les **ordonnanceurs** ou **orchestrateurs** planifient et exécutent des traitements composés de plusieurs jobs



# EXPLOITATION



Outils, langages et bibliothèques permettant d'exploiter les données de la plateforme.

Ils peuvent déclencher des traitements pour calculer des résultats temporaires (ex. requête SQL d'analyse).

On va y trouver en particulier :

- des **SDK** (*Software Development Kits*) à utiliser dans les programmes R / Python / ...
- des **requêteurs** type SQL
- des logiciels d'**analyse interactive** (PowerBI, Tableau, Dataiku, Excel...)
- les **ETL**, pour l'export de données en masse

# NOTION D'OFFRE DE SERVICES

Les outils vus ci-dessus font partie du **catalogue d'offre de services** de la plateforme.

Ils sont assortis :

- de la documentation de la plateforme
- de règles de bonne utilisation (gouvernance, sécurité, ...)
- de tutoriels pour démarrer rapidement
- d'un support technique pour dépanner

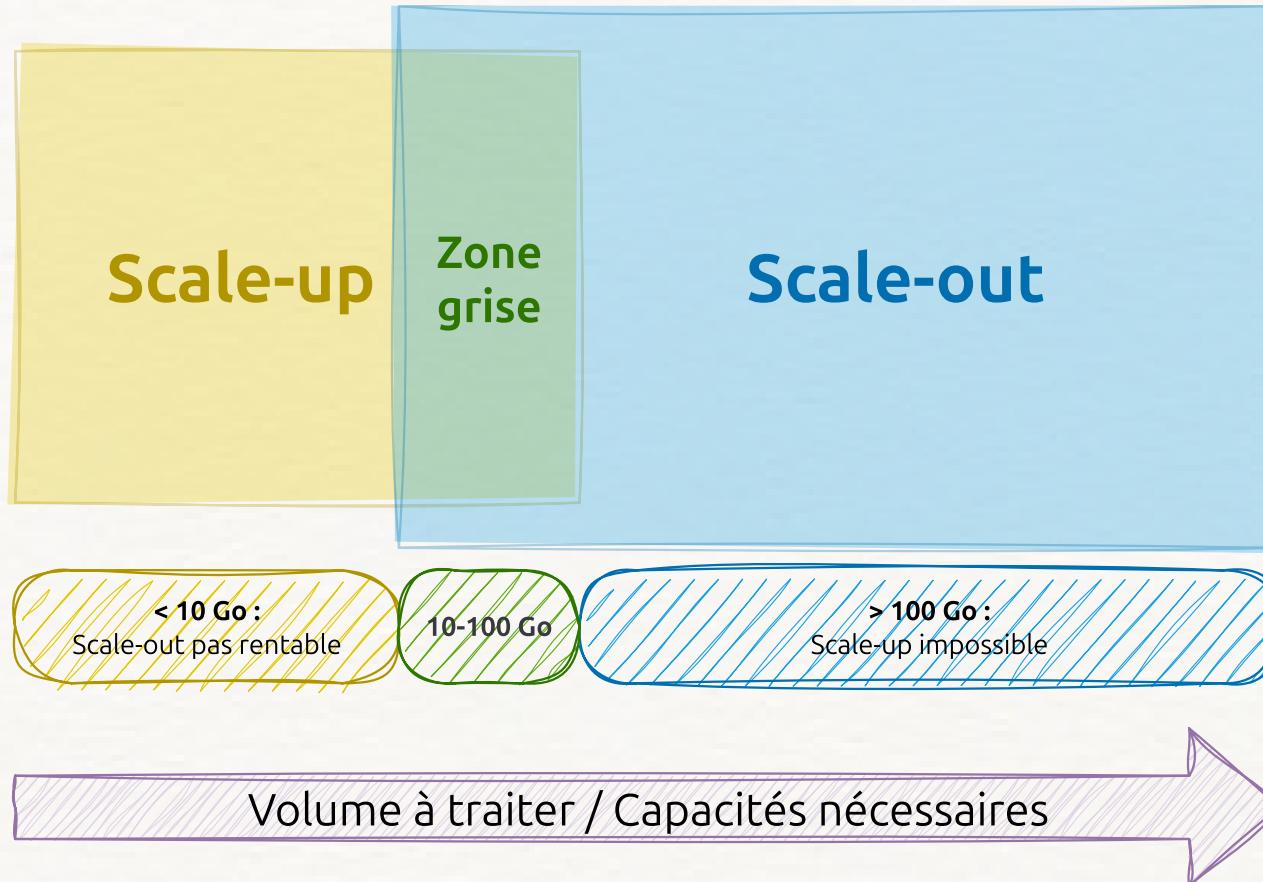


# QUAND LE VOLUME AUGMENTE

# QUAND LE VOLUME AUGMENTE

# METTRE LE CURSEUR

## ORDRES DE GRANDEUR



# ROBUSTESSE ET FIABILITÉ



Fondamentalement, une plateforme stocke et manipule des données.

**Que peut-il se passer  
d'imprévu, et comment y  
remédier ?**

# ROBUSTESSE ET FIABILITÉ

Nous avons vu que les plateformes de données sont distribuées. Leur taille augmente la probabilité qu'un incident affecte un noeud.

Par exemple :

- **Stockage** : crash d'un disque, ou d'un serveur qui en contient un
- **Calcul** : défaillance d'un processeur, crash d'un noeud de calcul, mémoire pleine
- **Réseau** : panne d'un équipement qui connecte les noeuds entre eux ou à l'extérieur
- **Code des traitements** : bug, mauvaise gestion de données imprévues



# **ROBUSTESSE ET FIABILITÉ**

# SYNTHÈSE

- Une plateforme de données offre les services nécessaires au big data
- Ces services peuvent être mis en correspondance sur le cycle de vie des données
- 4 couches se distinguent en particulier : ingestion, stockage, traitement, exploitation
- La plateforme est conçue pour gérer d'importants volumes de manière fiable

# ANATOMIE DES DONNÉES

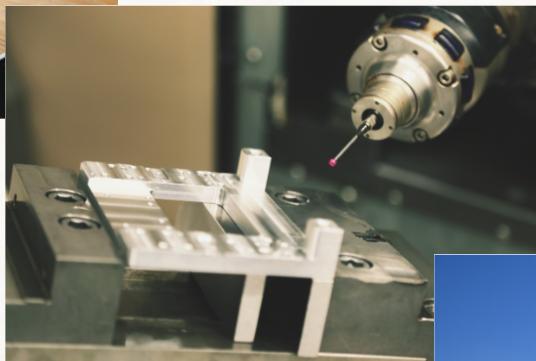
Comment décrire et organiser les données

# UNE GRANDE VARIÉTÉ



## Des humains

- Ouvrages, sites, discussions sur les réseaux, ...
- Archives documentaires
- Achats et préférences
- Données comportementales



## Des machines

- Résultats de traitements
- Traces d'exécution
- Diagnostics
- Rapports d'erreurs



## Des capteurs

- Environnement (température, ...)
- Mouvement
- Caméras
- Puces RFID

# MODÉLISATION & FORMAT

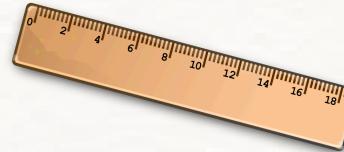
## Concept

## Modélisation

Comment la donnée est-elle décrite ?

## Format

Comment la donnée est-elle physiquement représentée ?



### Mesure ::=

- *date* (timestamp AAAAMMJJ)
- *id\_capteur* (entier)
- *valeur* (flottant)
- *état* (chaîne)

## Champs & types

```
date;id_capteur;valeur;état  
20240112;22;157;OK  
20240113;22;159;ALERTE  
20240114;22;158;OK  
...
```

CSV

```
01B6D7AA89E7C710BF611...
```

ZIP

# MODÉLISATION



Quels types de  
modélisation connaissez-  
vous ?

# **MODÉLISATION**

## **ZOOM SUR LE MODÈLE RELATIONNEL**

# MODÉLISATION

## ZOOM SUR LES SÉRIES TEMPORELLES

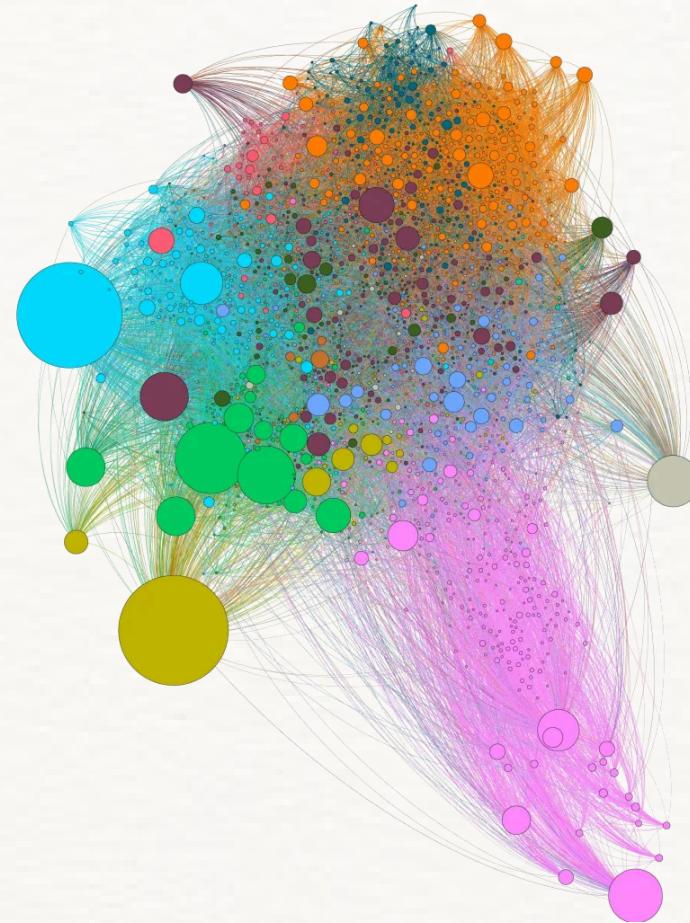
# MODÉLISATION

## ZOOM SUR LES GRAPHES

Les **graphes** (ou réseaux) représentent des objets ou des concepts (*noeuds*), liés entre eux par des relations (*arcs*)

Opérations intéressantes :

- Parcours en largeur (BFS) ou en profondeur (DFS)
- Recherche de chemins
- Calculs d'influence de noeuds (ex. PageRank)
- Recherche de sous-graphes
- ...
- Voir des modèles de deep learning (GNN) !



# MODÉLISATION

## ZOOM SUR LES VECTEURS

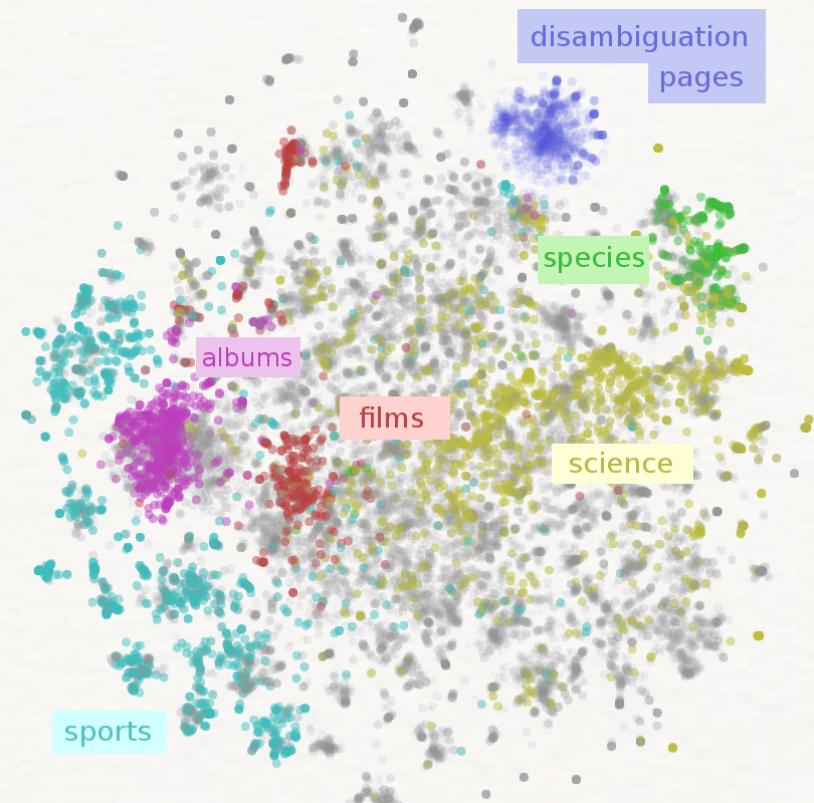
Les vecteurs ou *embeddings* encodent des données non structurées (mots d'un texte, documents, images, ...) au sein d'un corpus

- Exemple : [0.1, 2.32, -1.76, 0.8]

Ils sont au cœur des modèles d'IA type ChatGPT

Opérations intéressantes :

- Calcul de proximité via la distance entre 2 vecteurs
- Recherche des voisins proches
- Classification de documents ou concepts



<http://colah.github.io/posts/2015-01-Visualizing-Representations/>

# MODÉLISATION

## NOTION DE STRUCTURATION

Au niveau de la modélisation, on distingue en général 3 niveaux de structuration :

### Donnée **non structurée**

La donnée ne peut pas être interprétée par une machine sans traitement d'extraction préalable.

Formats associés :  
Documents (.txt, .docx...)  
Images, son, vidéo  
Flux d'octets bruts

### Donnée **semi-structurée**

La donnée est un agrégat d'informations structurées et non structurées.

Formats associés :  
XML/JSON  
HTML  
Logs machines

### Donnée **structurée**

La donnée possède une structure documentée (schéma) permettant d'interpréter ses constituants.

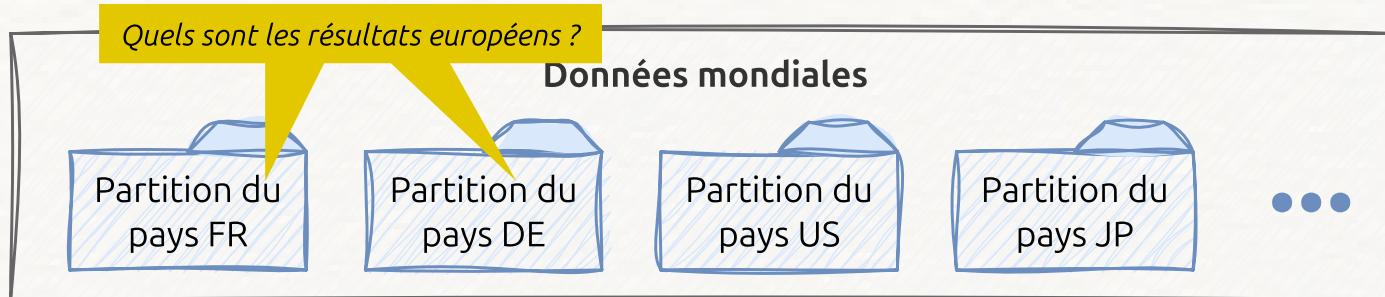
Formats associés :  
Bases de données  
CSV, Excel  
XML/JSON  
Parquet, Avro, HDF5

# MODÉLISATION

## PARTITIONNEMENT

Principe : “découper” les données selon un ou plusieurs champs, la **clé de partitionnement**

- Exemples fréquents : partitionnement par année (temporel), par pays (géographique)



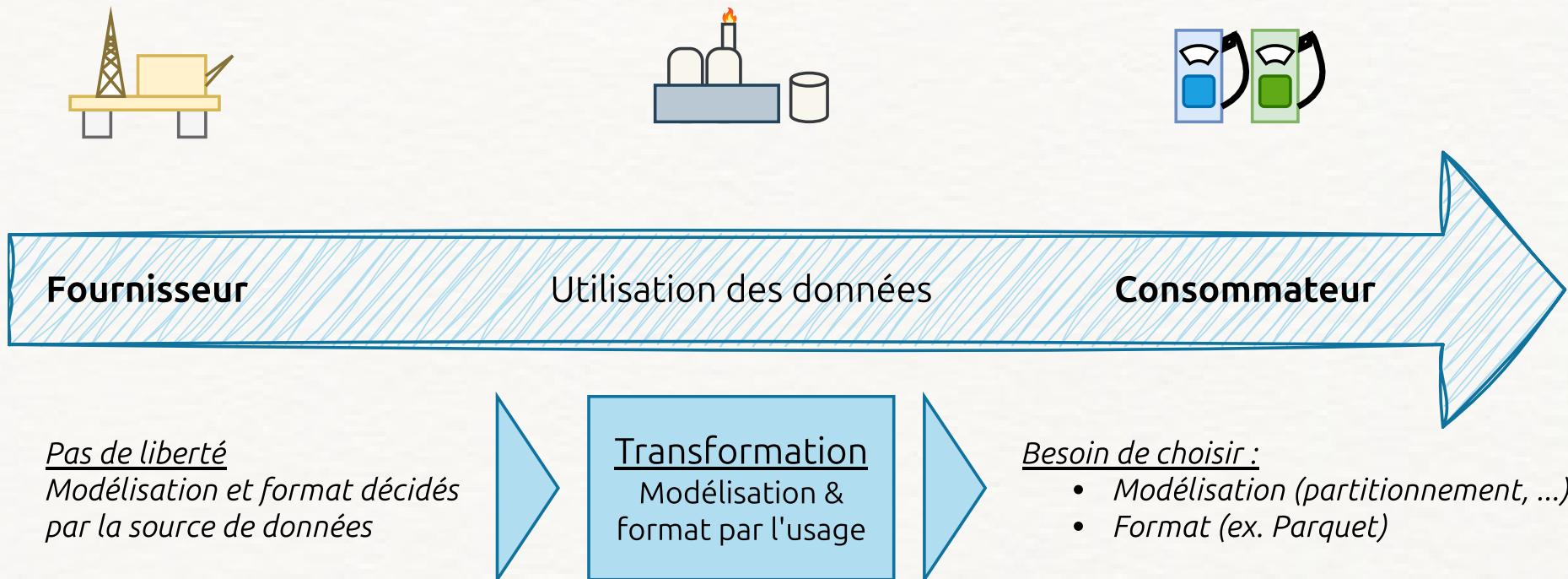
Cela permet aux traitements de cibler le sous-ensemble des données utiles en ignorant le reste : gain de performance

Les partitions peuvent être imbriquées à plusieurs niveaux (pays > année > mois > jour > ...)

La clé de partitionnement doit être adaptée aux traitements, et présenter un bon niveau de finesse (ni trop grossier, ni trop fin)

# MODÉLISATION ET FORMAT PAR L'USAGE

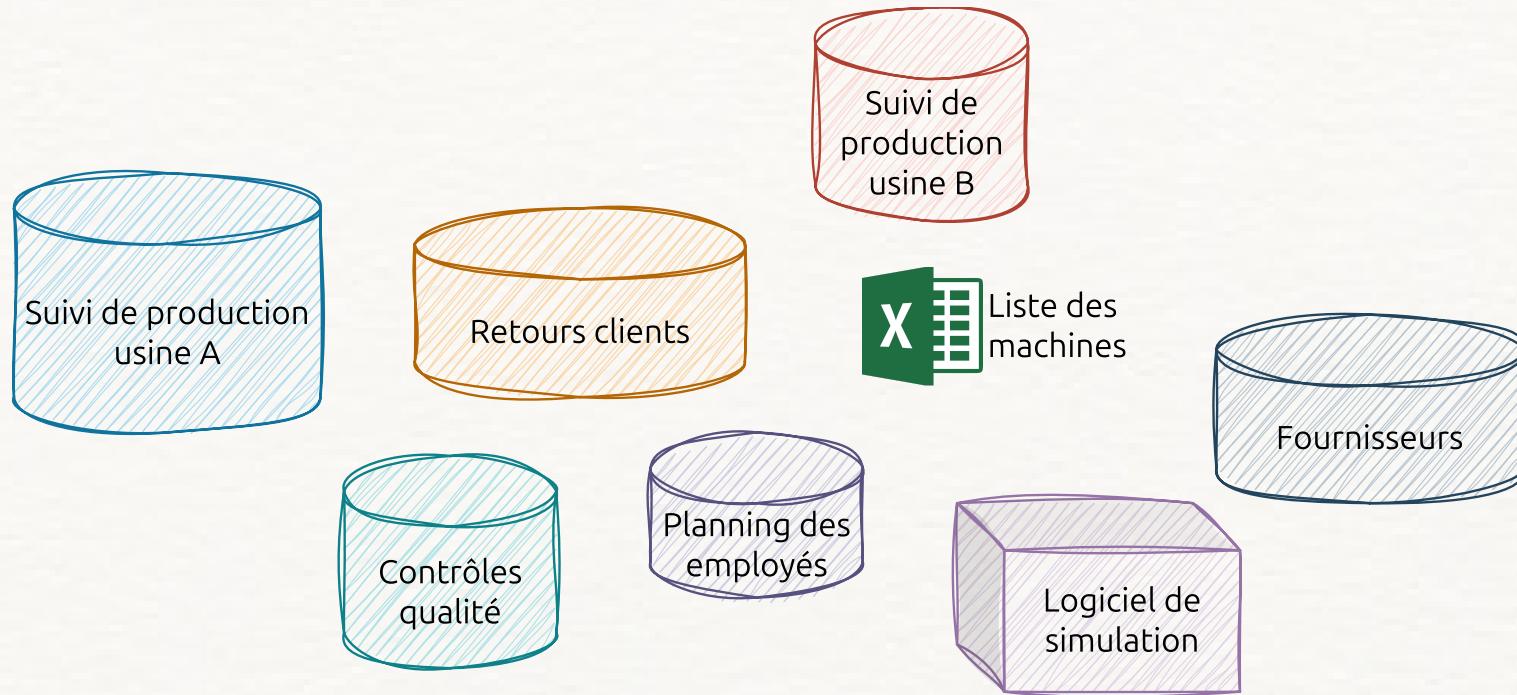
L'efficacité d'un traitement dépend en grande partie de l'**adéquation** entre la logique du traitement, la modélisation des données et le format dans lequel elle est encodée



# **FORMAT**

## **ZOOM SUR PARQUET**

# CENTRALISATION DES DONNÉES

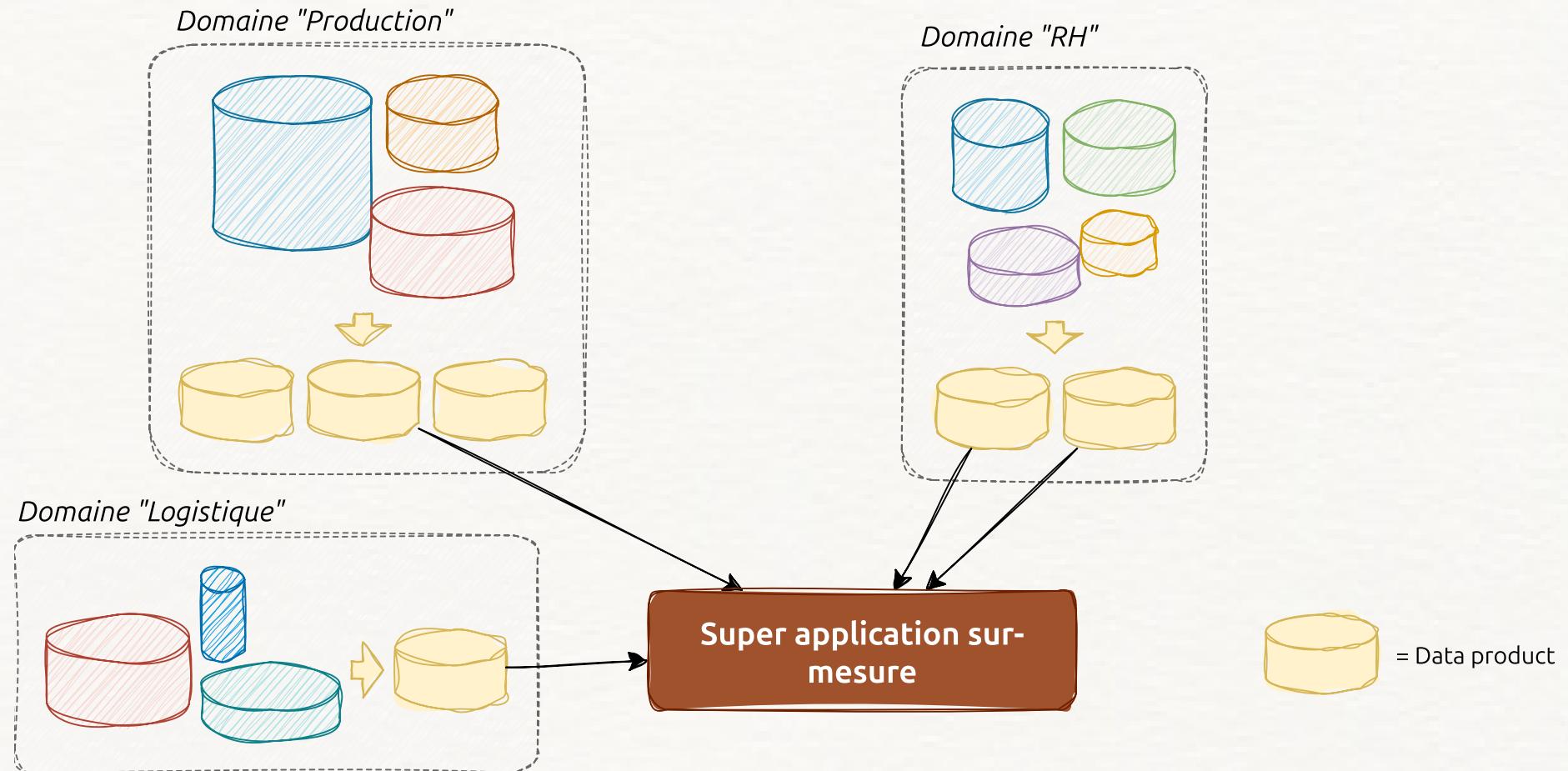


Comment répondre à la question :  
***“Quelles sont les causes des insatisfactions clients ?”***



# CENTRALISATION VS AUTONOMIE

## LE MODÈLE "DATA MESH"



# QUALITÉ DE DONNÉES

Les problèmes de qualité sont inévitables et biaisen les analyses. Quand on ne peut corriger à la source, on compense :

Problème	Suggestions
Valeurs manquantes	Interpoler, remplir (“imputer”) avec des moyennes ou du machine learning
Bruit ou valeurs aberrantes	Lisser (moyenne ou médiane), éliminer les anomalies (outliers statistiques), utiliser des modèles robustes
Doublons	Stratégie à choisir (choisir au hasard, garder la version la plus récente, fusionner les doublons, choisir au hasard...)
Liens (clefs étrangères) manquants	Ajouter une source de données qui possède les liens, rapprocher selon des critères heuristiques (ex. date)
Problèmes de texte (abus d'abréviations, orthographe, mélange de langues, ...)	Pré-traiter, utiliser des modèles pré-entraînés sur ces corpus problématiques

Quand les données sont suffisamment nombreuses, et les exceptions rares, on peut aussi les écarter.

# GOUVERNANCE DES DONNÉES

# GOUVERNANCE DES DONNÉES

## EXEMPLE



Supposons qu'un opérateur téléphonique vende à des tiers les cartes de fréquentation des rues d'une ville, en s'appuyant sur le bornage GPS ou cellulaire de ses abonnés

- Usage : optimiser la localisation des commerces dans les quartiers fréquentés

**Quelles questions de  
gouvernance cela soulève-t-il ?**

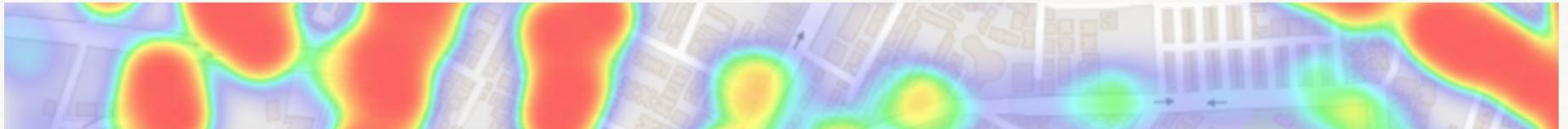


# GOUVERNANCE DES DONNÉES

## EXEMPLE

Quelques idées :

- Respect de la vie privée par anonymisation des terminaux et bornages
- Obligation de restituer ou supprimer les données d'un abonné sur demande (RGPD)
- Suivi et maintien d'une qualité de données en cohérence avec le prix du service
- Procédures de correction de données aberrantes
- Maîtrise des coûts en évitant l'accumulation *ad vitam aeternam* de données devenues caduques
- Garantie que d'autres applications internes “sauvages” ne détournent pas le jeu de données (RGPD, encore)
- Documentation à jour du jeu de données et des moyens techniques d'accès (API)



# SYNTHÈSE

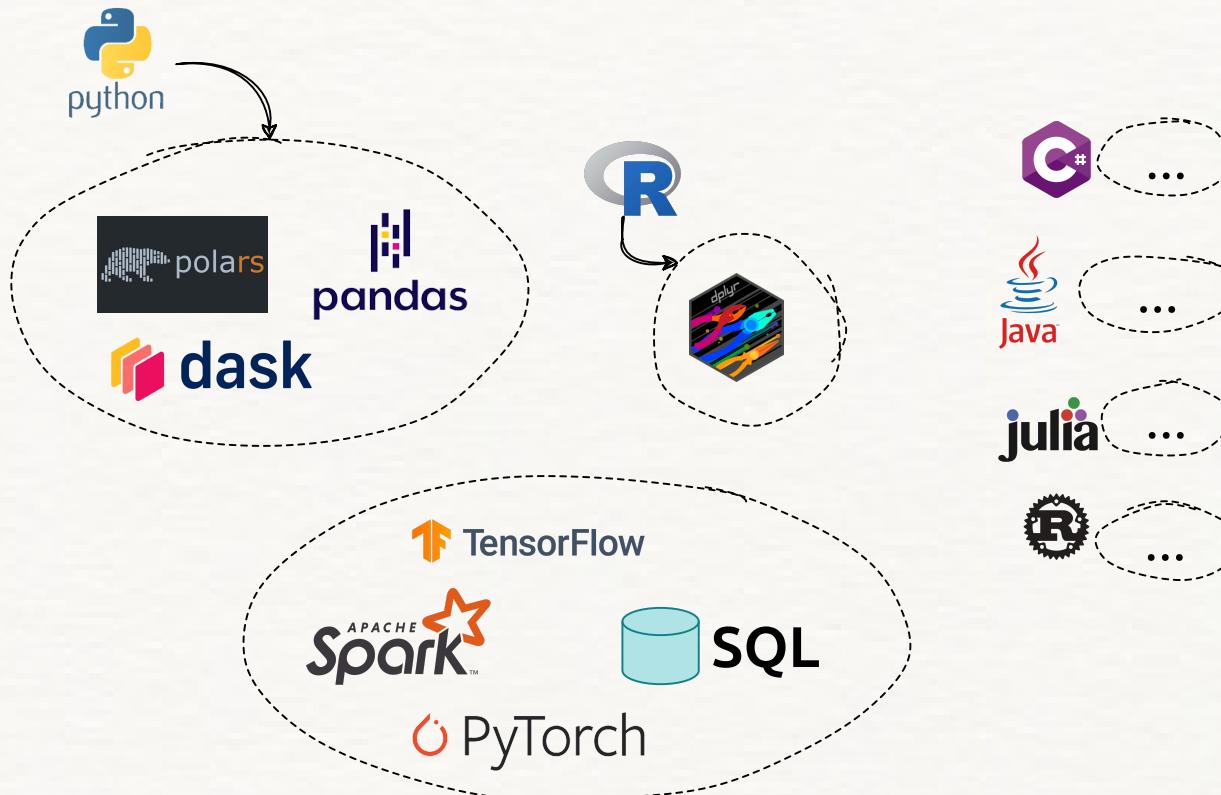
- Modélisation et format décrivent la représentation des données
- Pour la performance, ils doivent être adaptées à l'usage qui sera fait des données
- Le partitionnement et le format Parquet sont fréquemment utilisés en big data
- Outre les aspects techniques, il est important de respecter les règles de gouvernance des données

# LES TRAITEMENTS EN BIG DATA

Comprendre comment les plateformes arrivent à traiter efficacement de gros volumes de données

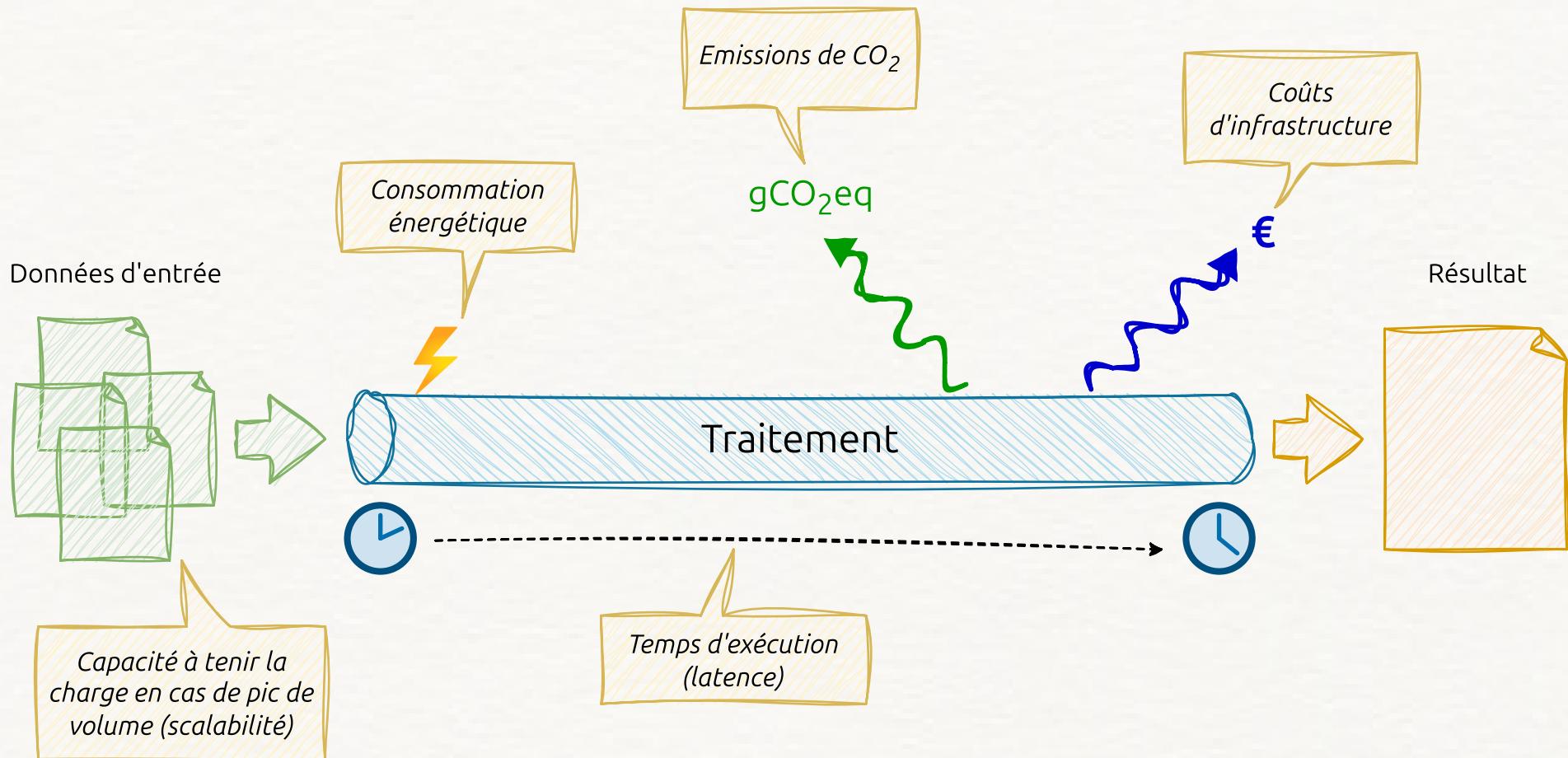
# DES VASTES ÉCOSYSTÈMES POUR DÉVELOPPER

(ce n'est absolument pas exhaustif)

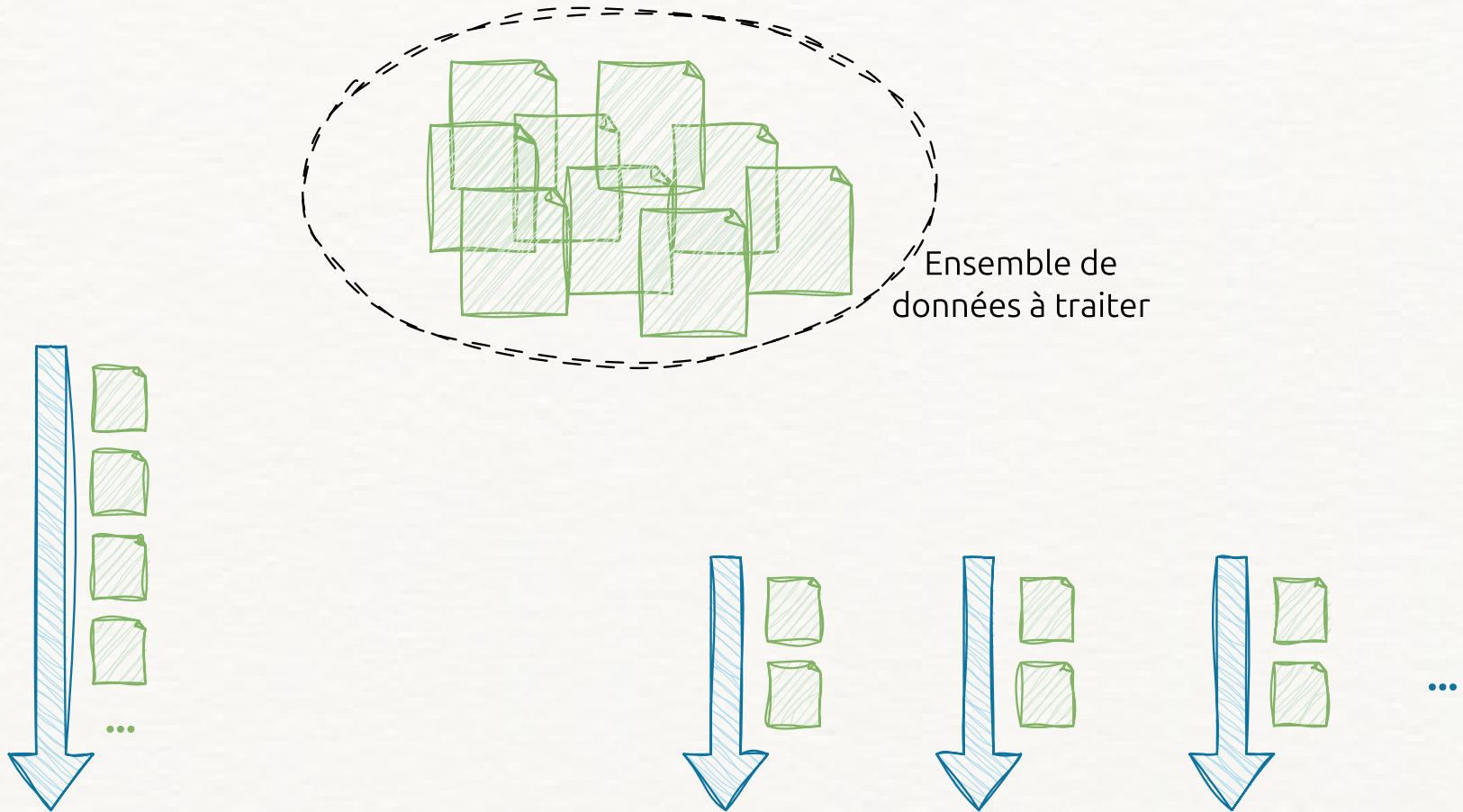


# PERFORMANCES D'UN TRAITEMENT ?

Qu'est-ce qui mesure les performances du traitement ?



# EN SÉRIE VS EN PARALLÈLE



## Traitement en série

Plus simple mais plus long

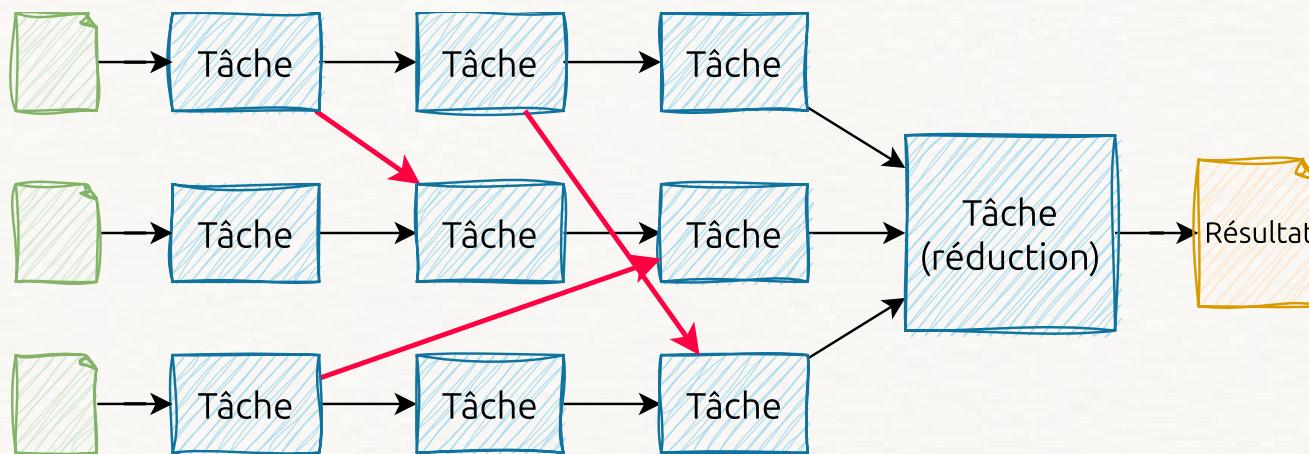
## Traitement en parallèle

Plus rapide mais pas toujours possible, et en général plus complexe

# PARALLÉLISATION

Principe : découper un gros travail en tâches plus petites et exécuter de manière indépendante les parties qui peuvent l'être.

Il y a souvent des tâches qui agrègent les résultats indépendants pour produire un résultat plus général. Ce sont des **goulets d'étranglement** à surveiller.



Si l'indépendance n'est pas vérifiée, on perd en efficacité.

# ESTIMATIONS DE VOLUME

## POURQUOI TANT D'IMPORTANCE ACCORDÉE AU VOLUME ?

Pour simplifier, plaçons-nous dans le cas d'un traitement en série sur un seul noeud de calcul.

Plus la donnée est "loin" (disque, réseau) plus son accès est lent. Ces lenteurs sont multipliées si l'algorithme doit faire des allers-retours à la source.

Un volume important signifie :

- plus de calculs à faire (of course)
- plus d'octets à transférer depuis le disque ou le réseau
- plus de données intermédiaires à stocker sur disque si la mémoire est pleine

*Rien ne sera jamais aussi rapide qu'un traitement dont les données tiennent entièrement en mémoire RAM !*

# **ESTIMATIONS DE VOLUME**

**COMMENT FAIRE LES 2 ESTIMATIONS SUIVANTES ?**

# ESTIMATIONS DE VOLUME

## VERSION CSV

CSV : tout est stocké en chaînes de caractères, même les nombres (on privilégie la lisibilité plutôt que l'efficacité).

Artiste	Album	Année	Pistes
Queen	Innuendo	1991	12
Queen	Hot Space	1982	11
R.E.M.	Out of Time	1991	11
Etienne Daho	Paris, ailleurs	1991	11
...	...	...	...

Diagramme illustrant les estimations de volume pour un CSV avec 10 000 enregistrements. Les colonnes sont : Artiste (17 caractères en moyenne, max 34), Album (24 caractères en moyenne, max 47), Année (4 caractères) et Pistes (1 ou 2 caractères). Un double flèche indique qu'il y a 10 000 enregistrements dans le fichier.

$$\Rightarrow \text{Moyenne} : 10\,000 \times (17 + 24 + 4 + 2) = 459 \text{ Ko}$$

$$\Rightarrow \text{Pessimiste} : 10\,000 \times (34 + 47 + 4 + 2) = 850 \text{ Ko}$$

⚠ Simplification : 1 caractère = 1 octet

# ESTIMATIONS DE VOLUME

## VERSION NUMPY

NumPy : les nombres sont stockés en représentation machine. On choisit le type de représentation en fonction des contraintes connues.

The diagram illustrates the memory footprint of a CSV file (10,000 records) when loaded into a NumPy array. It shows four columns with their corresponding data types and bit ranges:

- ID Artiste**: Entier sur 2 octets ( $< 2^{15}$ )
- ID Album**: Entier sur 4 octets ( $< 2^{31}$ )
- Année**: Entier sur 2 octets ( $< 2^{15}$ )
- Pistes**: 1 octet ( $< 2^8$ )

A vertical double-headed arrow on the right indicates a total of *10 000 enregistrements*.

ID Artiste	ID Album	Année	Pistes
181	7776	1991	12
181	7189	1982	11
217	9561	1991	11
94	453	1991	11
...	...	...	...

$$\Rightarrow \text{Estimation} : 10\,000 \times (2 + 4 + 2 + 1) = 88 \text{ Ko}$$

- *Rappel* : 459 Ko en moyenne pour le CSV

# ESTIMATIONS DE VOLUME

## UN EXEMPLE PLUS RÉALISTE

Des séries temporelles :

- 67 équipements
- 14 capteurs par équipement
- 1 mesure/capteur/seconde
- 10 ans d'historique

Enregistrements :

67 équipements  
x 14 capteurs  
x 10 ans  
x 365 jours  
x 24 heures  
x 60 minutes  
x 60 secondes  
= 295 807 680 000 mesures

ID Eqpt	ID Capteur	Timestamp	Valeur
84	4091	2024-12-01 12:32:17	7.33991
84	4091	2024-12-01 12:32:18	7.42819
84	4091	2024-12-01 12:32:19	7.35710
84	4091	2024-12-01 12:32:20	7.32399
...	...	...	...

Taille unitaire encodée (octets) :

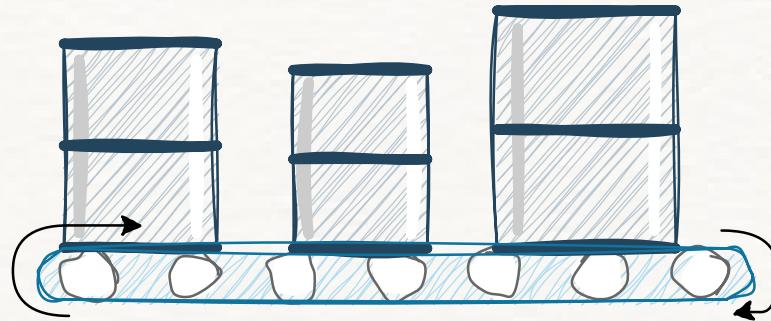
4 pour *ID Eqpt* ( $< 2^{32}$ )  
+ 4 pour *ID Capteur* ( $< 2^{32}$ )  
+ 4 pour *Timestamp* (entier "époque")  
+ 4 pour *Valeur* (flottant simple)

**Total = 4 408 Go**

# CINÉMATIQUE

## COMMENT LES DONNÉES ARRIVENT AU TRAITEMENT

Traitements par lots (ou *batch*) : chaque lot est traité comme un tout. Les lots peuvent être de tailles égales ou différentes.



Ordonnancement : les batches sont déclenchés sur événement (*trigger*), par exemple à fréquence fixe, à la réception d'un signal externe, à la fin d'un autre batch, ...

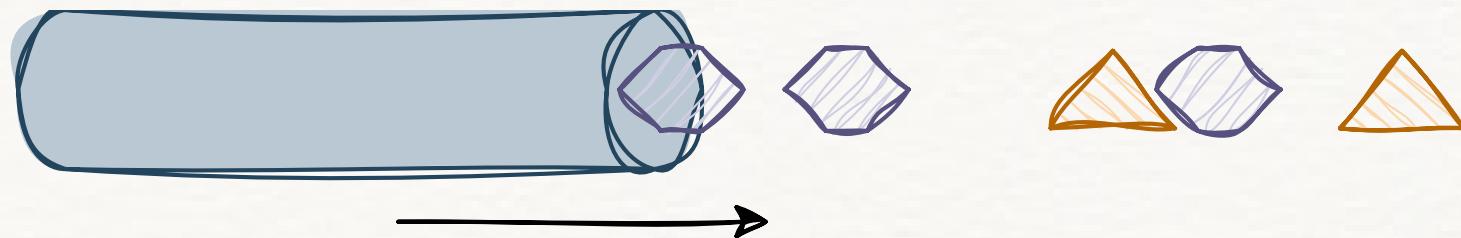
Exemples :

- Traitement quotidien des virements initiés pendant la journée
- Descente des prix centrale d'achat → magasins en grande distribution
- Analyse de la production après un changement d'équipe en usine
- Simulation physique sur demande, avec plusieurs jeux de paramètres candidats

# CINÉMATIQUE

## COMMENT LES DONNÉES ARRIVENT AU TRAITEMENT

Au fil de l'eau (ou *streaming*) : chaque donnée est un message traité immédiatement, dès sa réception.



Les messages sont le plus souvent indépendants entre eux et on ne contrôle pas leur ordre d'arrivée.

Exemples :

- Analyse d'une transaction suspecte, pour vérifier une fraude
- Enchère publicitaire sur un site web
- Remontée d'une mesure de capteur

NB : les deux cinématiques peuvent être combinées : stockage au fil de l'eau et traitement par lots après accumulation.

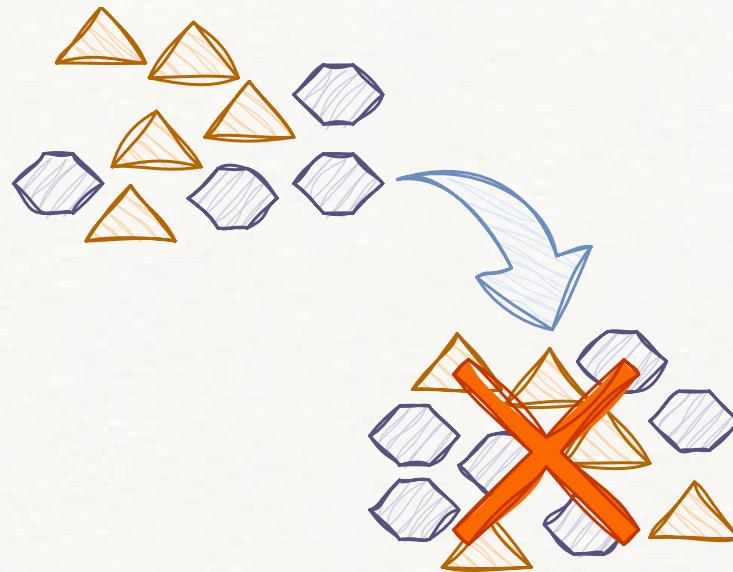
# NOTION D'ÉTAT (OU STATE)

# SÉMANTIQUE

## COMMENT LE TRAITEMENT INTERPRÈTE LES DONNÉES

Annule & remplace : les nouvelles données écrasent l'état de persistance.

On ne sait pas a priori ce qui a changé dans les données, en plus ou en moins.



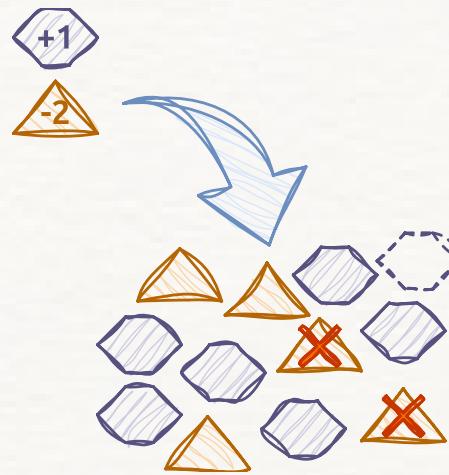
Exemples :

- Inventaire de stocks par recomptage
- Comptabilité : renvoi de l'état des comptes après corrections

# SÉMANTIQUE

## COMMENT LE TRAITEMENT INTERPRÈTE LES DONNÉES

Incrémental ou "delta" : les données entrantes sont des ordres de mises à jour à appliquer à l'état supposé connu (d'où l'importance de le garder intègre).



Exemples :

- Etat des stocks ajusté en fonction des ventes et des livraisons
- Arrivée et départ de personnel en entreprise
- Capteur renvoyant la variation d'une mesure et non sa valeur absolue

NB : en comparant 2 photos successives obtenues par "annule & remplace", on obtient... un "delta".

# CINÉMATIQUE, SÉMANTIQUE ET COMPLEXITÉ

Avantages et inconvénients à la louche, quand on peut choisir :

	Annule & remplace	Delta
Batch	Simple à développer Volume et latence plus élevés	Encore plus complexe Volume modéré
Streaming	Plus complexe Volume faible	Très complexe ! Volume et latence faibles

Qu'est-ce qui rend le streaming et le delta plus complexes ?

C'est la gestion de l'état de persistance (toujours nécessaire avec le delta, souvent nécessaire en streaming).

En delta, il faut de plus être très rigoureux sur la prise en compte des ordres de modifications, leur ordre d'application, etc.

# GESTION DES ERREURS

Il est important de distinguer **deux types d'erreurs** :

## Erreur **transitoire**

Causée par un incident passager, lié par exemple à l'infrastructure.

Sera résolue d'elle-même si on relance le traitement plus tard, après résolution de l'incident.

### Exemples :

Problème de plateforme (crash serveur...)

Manque de mémoire *dû à une surcharge*

Perte de connexion internet

## Erreur **permanente**

Causée par un défaut dans le programme du traitement ou dans les données.

Survient de nouveau si on relance le traitement dans les mêmes conditions (même code, même données).

### Exemples :

Bug du traitement ou de la plateforme

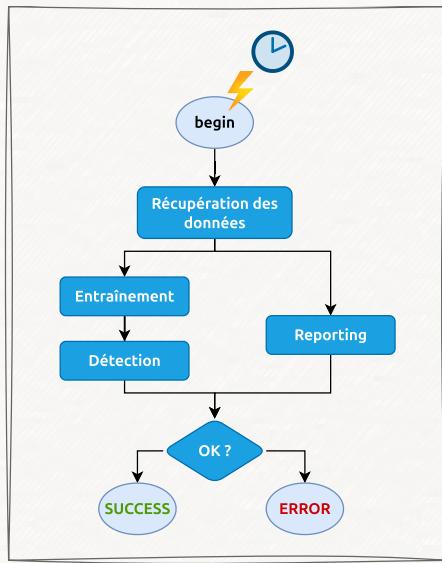
Données mal formées, corrompues

Manque de mémoire *dû au volume de données*

Les plateformes peuvent automatiser la relance (rejeu) des traitements en erreur, entiers ou en partie. Les logs sont importants pour comprendre a posteriori ce qu'il s'est passé.

# TRAITEMENTS COMPOSÉS

Un traitement peut être un enchaînement complexe d'étapes simples. On parle souvent de **workflow** ou de **DAG** (*Directed Acyclic Graph*).



```
@dag(
    schedule="@daily",
    start_date=pendulum.datetime(2026, 1, 1),
    catchup=False,
    tags=["example"],
)
def my_workflow():
    @task
    def get_data(): ...

    @task
    def train(): ...

    @task
    def detect(): ...

    @task
    def report(): ...

    @task
    def check(): ...

    data = get_data()
    detection_result = detect(train(data))
    reporting_result = report(data)
    check(detection_result, reporting_result)

my_workflow()
```

Code (Apache Airflow)

```
WITH summary AS (
    SELECT
        unit,
        SUM(prod) OVER (PARTITION BY eqpt)
    FROM `my_dataset.big_table`
    CROSS JOIN UNNEST(t) AS prod
    GROUP BY unit
)
SELECT *
FROM summary AS s
LEFT JOIN `my_dataset.units` AS u
ON s.unit = u.id
WHERE
    u.status = 'WORKING'
```

SQL compilé en plan d'exécution

Les workflows sont pris en charge par un ordonneur sur la plateforme, qui les déclenchent lorsque nécessaire et assure le bon enchaînement des étapes. Les étapes indépendantes peuvent être exécutées en parallèle.

# SYNTHÈSE

- La performance d'un traitement peut se mesurer de différentes manières
- Le volume de données joue un rôle central et on peut l'estimer
- Ecrire un traitement peut être compliqué : parallélisme, cinématique, sémantique et gestion des erreurs
- Le workflow est un moyen de découper un traitement complexe en étapes simples

# LE CLOUD

Super plateforme... et usine à plateformes