Chap I: Le Big-Data

Tables des matières

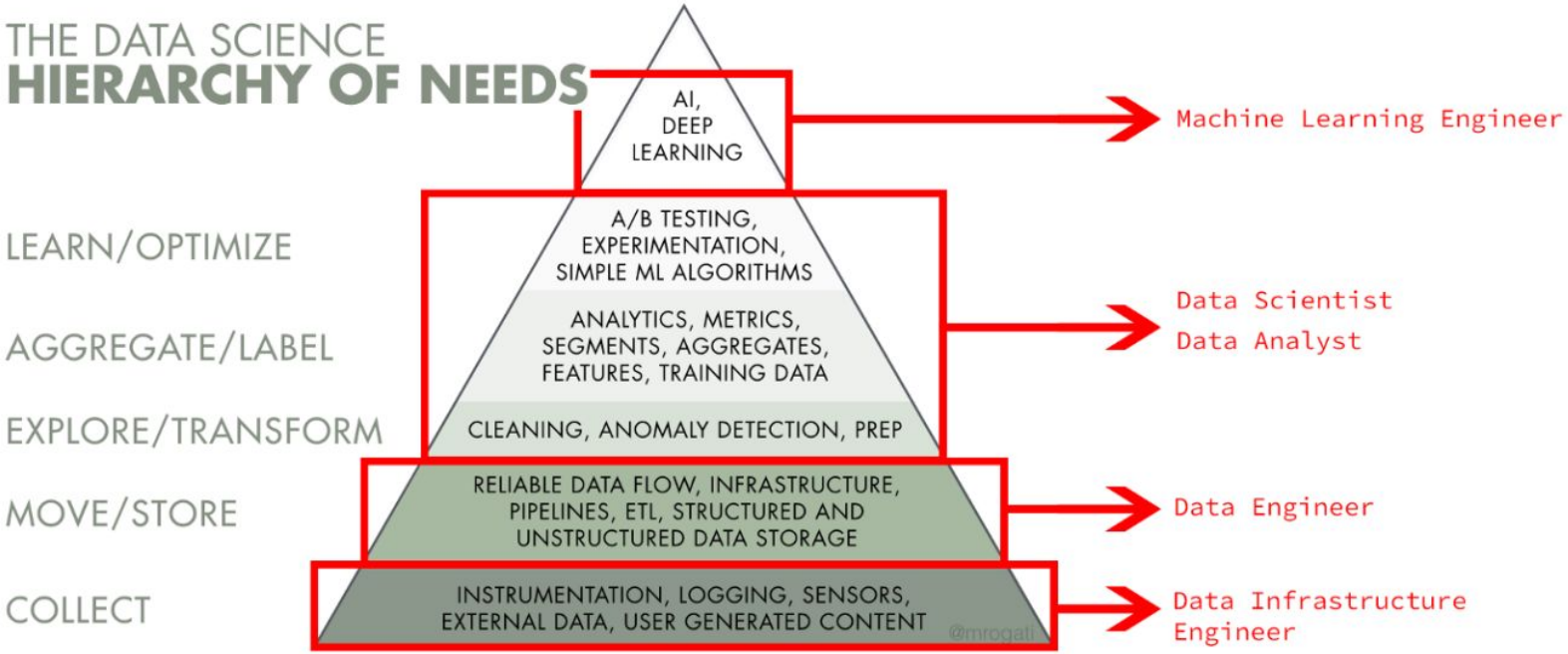
Tour de table

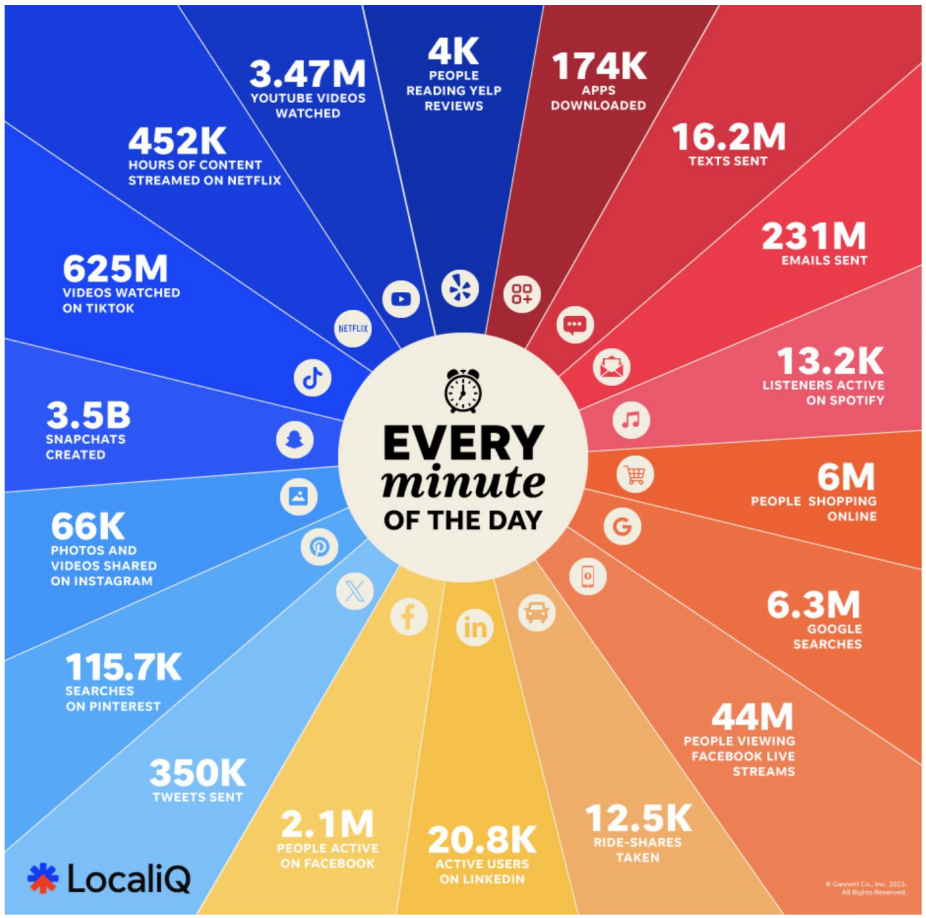
About Me

About Me

Data-Engineer, +3 ans d’expériences avec les techno Big-Data et Cloud.

Introduction




EVERY
minute
OF THE DAY
3.47M
YOUTUBE VIDEOS WATCHED
4K
PEOPLE READING YELP REVIEWS
174K
APPS DOWNLOADED
16.2M
TEXTS SENT
231M
EMAILS SENT
13.2K
LISTENERS ACTIVE ON SPOTIFY
6M
PEOPLE SHOPPING ONLINE
6.3M
GOOGLE SEARCHES
44M
PEOPLE VIEWING FACEBOOK LIVE STREAMS
12.5K
RIDE-SHARES TAKEN
20.8K
ACTIVE USERS ON LINKEDIN
2.1M
PEOPLE ACTIVE ON FACEBOOK
350K
TWEETS SENT
115.7K
SEARCHES ON PINTEREST
66K
PHOTOS AND VIDEOS SHARED ON INSTAGRAM
3.5B
SNAPCHATS CREATED
625M
VIDEOS WATCHED ON TIKTOK
452K
HOURS OF CONTENT STREAMED ON NETFLIX
NETFLIX
LocaliQ

Introduction

“ There was 5 exabytes of information created between the dawn of civilization through 2003, but that much informations now created every 2 days, and the pace is increasing."

Eric Schmidt, Former PDG Google, 2010

Définition

Le Big Data (ou mégadonnées) représente les collections de données caractérisées par un **volume**, une **vélocité** et une **variété** si grands que leur transformation en valeur utilisable requiert l'utilisation de **technologies et de méthodes analytiques spécifiques**.

Caractéristiques du big-data

Volume	Vélocité	Variété	Véracité	Variabilité	Valeur
Quantité importante de données issues de multiples sources.	La vitesse à laquelle la donnée est créée et consommée.	Différentes types de données (Structurées, non-structurées, semi-structurées)	A quelle mesure la donnée est fiable	Comportement de la donnée au cours du temps (anomalies,).	La valeur associée à l'information extraite de la donnée

Pourquoi le Big-Data

- Augmentation exponentielle de la quantité de données (non structurées notamment: chat, blog, web, musique, photo, vidéo, etc)
- L'augmentation de la capacité de stockage à un coût relativement bas
- Nouvelles techniques d'analyse/d'exploitation de la donnée (Visualisation, Machine Learning, etc)
- De nouvelles technologies plus adaptées (Hadoop)
- L'avènement du Cloud

Les sources de données

Comment la donnée est créée ?

- De façon analogue:
  - Ecritures (parchemins, …)
  - Langages
  - etc
- Par voie digital
  - messages via messagerie
  - transactions en lignes
  - trafic sur les réseaux
  - etc

Les différentes sources des données (digitales)

- **Les APIs** (Application Programming Interfaces): Une interface permettant l'accès à des services ou des données ou encore une interface permettant à un demandeur de faire une requête et de recevoir une réponse à sa requête

- **Les bases de données**:
  - OLTP (Online Transaction Processing): sont des systèmes de bases de données souvent associées à des applications (api), afin de stocker l'état de ces dernières. Souvent caractérisées par une fréquence élevée de lectures et d'écriture
  - OLAP: (Online Analytical Processing): sont des systèmes pour de l'analyse de données à grande échelles. Ils sont souvent utilisé comme datawarehouse.

Les différentes sources des données (digitales)

- **CDC** (Change Data Capture): permet de tracker les changements qui surviennent dans un systèmes

- **Les Logs:** Ce sont des données issues de systèmes informant de l'état de ces derniers.

- **Les systèmes de messaging/queues**: permettent de transiter de la données d'un ou plusieurs systèmes à un ou d'autres.

Les Systèmes de stockage

Les différentes types de systèmes de stockage de données

- Les bases de données relationnelles (Sql)
- Les bases de données Nosql
- Le Stockage Objet

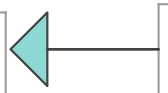
Les bases de données relationnelles

Définition

Une base de données relationnelle est un ensemble d'éléments de données dotés de relations prédéfinies entre eux. Ces éléments sont organisés en un jeu de tables composées de colonnes et de lignes.

students

id_stud	name	dept
1	Abdou	2
2	Jeanne	1
3	Ibou	3



id_dept	name
1	Maths
2	Infos
3	Bio

departements

Quelques notions clés

Attribut: Un attribut est un identificateur (un nom) décrivant une information stockée dans une base.
Schéma de relation: Un schéma de relation précise le nom de la relation ainsi que la liste des attributs avec leurs domaines.
Clé primaire: La clé primaire d'une relation est un attribut unique qui identifie de manière unique un élément d'une table.
Clé étrangère: Une clé étrangère dans une relation est formée d'un ou plusieurs attributs qui constituent une clé primaire dans une autre relation.
Transaction: Une transaction de base de données désigne une ou plusieurs instructions SQL exécutées en tant que séquence d'opérations (requête).

id_stud	name	dept
1	Abdou	2
2	Jeanne	1
3	Ibou	3

Propriété ACID

- **Atomicité:** La transaction est exécutée dans son intégralité ou invalidée dans son intégralité.
- **Cohérence:** requiert que les données écrites dans la base de données dans le cadre d'une transaction soient conformes à toutes les règles et restrictions définies.
- **Isolation:** L'indépendance des différentes transactions en cas de concurrence.
- **Durabilité:** requiert que toutes les modifications réalisées sur la base de données soient permanentes une fois la transaction exécutée avec succès

Avantages et Inconvénients

- **Les ++**
  - Un modèle de données simple
  - Préservation de l’intégrité des données
  - Un système de requêtage assez mature et facile à prendre en main

- **Les - -**
  - Un schéma parfois très rigide moins pratique pour des données qui varient constamment
  - Scalabilité horizontale parfois difficile à mettre en place
  - Souvent moins performant que le modèle non-relationnel

Cas d’usage

- Données structurées avec de fortes contraintes sur le modèle de données
- Pour des transactions complexes et/ou fréquentes sur les données
- Quand on a des relations entre les différentes entités présentes.

Exemples de systèmes de bases de données relationnelles

Le modèle non relationnel, le NoSql

Le NoSQL est un ensemble de technologies de base de données qui **repose sur un modèle différent de celui des BDD relationnelles**. Il se passe de l'exigence d'avoir un schéma prédéfini offrant ainsi plus de flexibilité avec la possibilité de stocker des données non-structurées, semi-structurées et structurées.

Il existe 4 grandes familles de base de données NoSql: Les bases de données orientées **clé-valeur,** les bases de données **document**, les bases de données orientées **colonne** et les bases de données type **graphe**.

Le modèle clé-valeur

Clé	Valeur
1	https://adresseweb.com
2	356
3	**mail**: monmail@gmail.com **date**: 25/10/2020 13:42:12

Exemples:

-  DynamoDB
- 

Le modèle document (basé sur le json et le xml)

Document (id: 5baf47)

```
{
 "nom": "Liquide vaisselle",
 "images": [
   "https://...",
   "https://..."
 ],
 "specs": {
   "parfum": "Orange"
 }
}
```

Document (id: ea53aa)

```
{
 "nom": "Shampooing",
 "images": [
   "https://...",
   "https://..."
 ],
 "specs": {
   "parfum": "Vanille"
 }
}
```

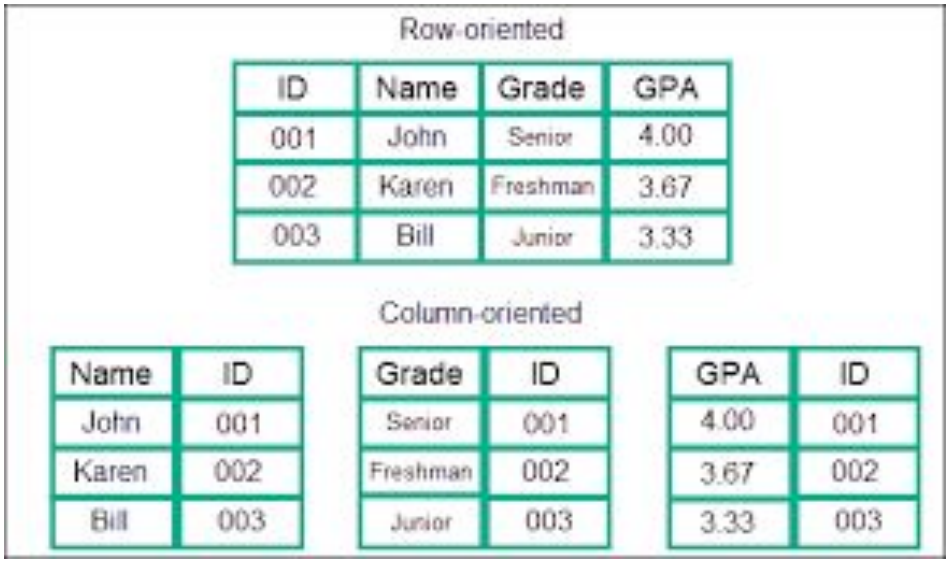
Document (id: d710bb)

```
{
 "nom": "Fromage blanc",
 "images": [
   "https://...",
   "https://..."
 ],
 "specs": {
   "mat_grasses": "0%"
 }
}
```

Exemples:

- mongoDB
- CouchDB

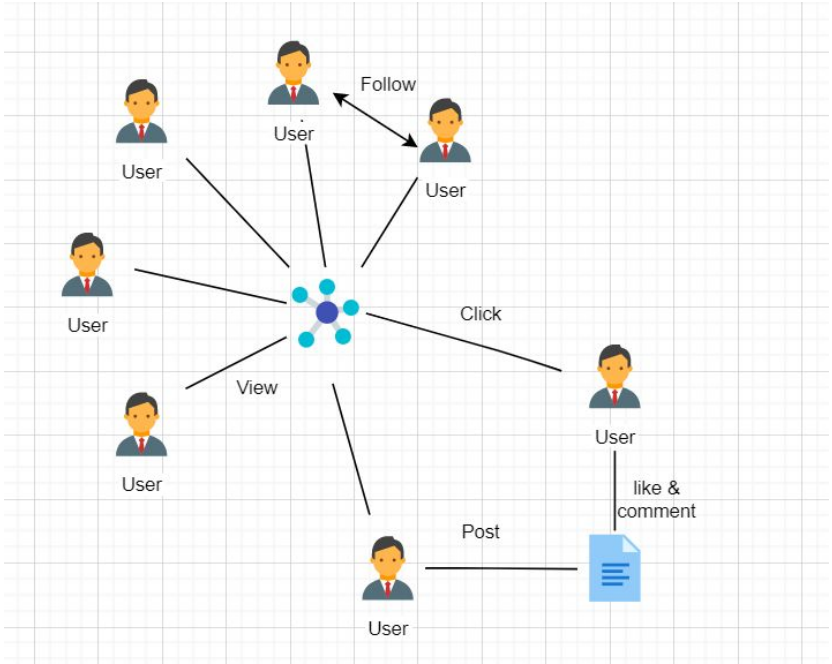
Le modèle orienté colonne



Exemples:

- APACHE HBASE
- cassandra

Le modèle Graphe



Exemple:

- neo4j

Le Stockage Objet (un modèle souvent basé sur le cloud)

Le **stockage Objet** est un format de stockage dans lequel les données sont stockées sous forme d'unités discrètes appelées objets. Chaque unité a un identifiant unique ou une clé qui permet de la retrouver où qu'elle soit stockée dans un système distribué.

Les avantages

- **Évolutivité :** une architecture simple

- **Données à la demande :** avec le stockage d'objets, il est plus facile de payer seulement pour la capacité de stockage effectivement utilisée.

- **Supporte multiples Api :** vous pouvez accéder et gérer les données dans des systèmes de stockage d'objets par différentes manières

- **Meilleure intégrité des données:** grâce au codage à effacement, les systèmes de stockage d'objets peuvent protéger l'intégrité des données en reconstruisant des morceaux de vos données et en effectuant des contrôles d'intégrité pour prévenir la corruption.
- **Disponibilité des données:** Profitant du cloud, ces systèmes offrent une haute disponibilité de la donnée.
- **Scalabilité** (jusqu'à plusieurs Teras de données)

cas d’usage

- En tant que **Datalake**
- Pour les gros volumes de données non-structurées (Photos, vidéos, fichiers etc)
- Machine-Learning, Data-Science
- Backup and archivage

Exemples

amazon
S3

Azure Data Lake Storage Gen2

Google Cloud Storage

Les formats de stockage de données

Les différents Format de stockage (données structurées ou semi-structurées)

- Csv

Comma separated values: “row-based”, en plus des lignes on trouve généralement une entête (header) représentant les noms des différentes colonnes. Ce format permet de représenter des données tabulaire sous forme de texte brut. On peut également utiliser d’autres séparateurs autre que la , comme ; ou \t etc.

- **Avantages:**
  - human readable
  - facile à manier/modifier/analyser
  - S'intègre facilement avec la plupart des systèmes
- **Inconvénients:**
  - Ne prend pas en charge le type des colonnes
  - Les données peuvent facilement être corrompues (séparateurs, caractères spéciaux etc)
  - Ne permet pas la représentation hiérarchique
  - Peu adapté au big data, gourmand en taille de stockage

Les différents Format de stockage (données structurées ou semi-structurées)

- **Csv**

```
name,grade,age
"Ibou","Master",20
"Alice","Bachlor",25
```

Les différents Format de stockage (données structurées ou semi-structurées)

- Json

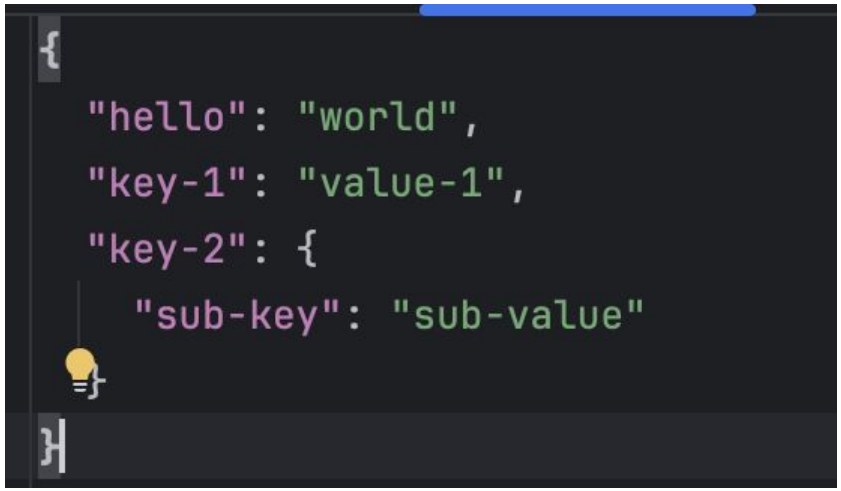
JavaScript object notation: “key/value pair”, permet de représenter des données semi-structurées sous formes de paires clé/valeur, supporte la représentation hiérarchique des données.

Les documents JSON sont couramment utilisés dans la communication réseau, en particulier avec des services web basés sur REST.

- **Avantages:**
  - human readable
  - facile à manier/modifier/analyser
  - S'intègre facilement avec la plupart des systèmes
  - Représentation hiérarchique
  - Très utilisé par beaucoup de base de données
- **Inconvénients:**
  - Peut facilement être complexe à parser
  - Peu adapté au big data, gourmand en taille de stockage

Les différents Format de stockage (données structurées ou semi-structurées)

- **Json**



Les différents Format de stockage (données structurées ou semi-structurées)

- Parquet

Pour le parquet on a un stockage en colonnes (columnar storage format). Ce format est très bien adapté au big data car une compression est automatiquement appliquée sur les données. Prend en compte les types des colonnes et plus généralement le schéma des données.

- **Avantages:**
  - Très bien adapté au big data: compression, schéma
  - Représentation hiérarchique
  - S'intègre à la plupart des systèmes big-data
- **Inconvénients:**
  - Not human readable
  - Ne prend pas en charge l’évolution de schéma

Les différents Format de stockage (données structurées ou semi-structurées)

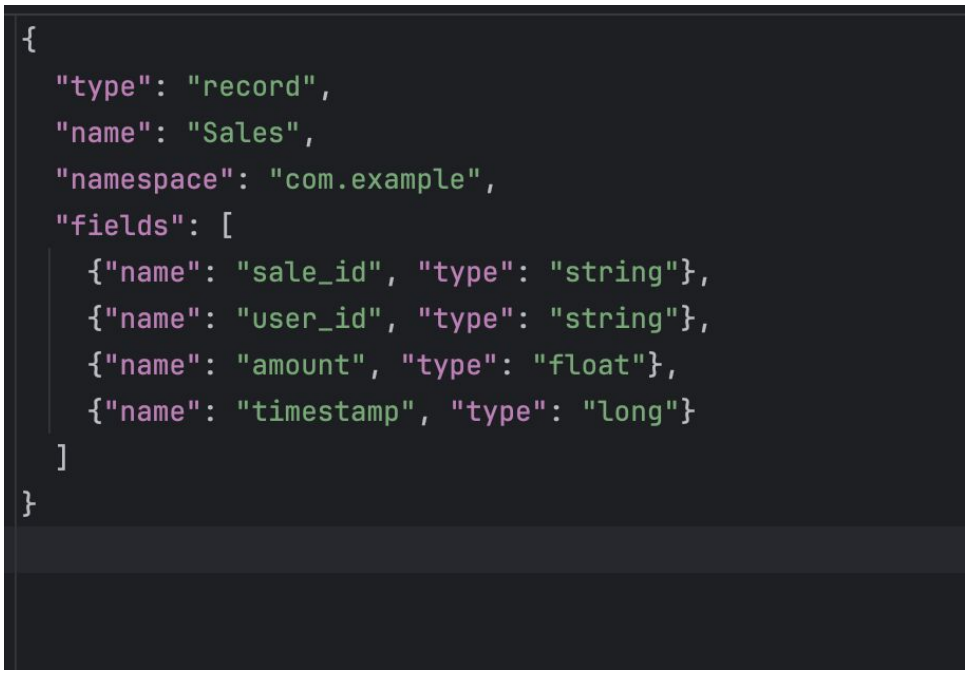
- Avro

Row-based, le avro stocke efficacement les données au format binaire et le schéma au format JSON. Il bénéficie d'un support fiable pour l'évolution du schéma, ce qui facilite la gestion des changements dans les données au fil du temps.

- **Avantages:**
  - Très bien adapté au big data: compression, schéma
  - Représentation hiérarchique
  - Evolution de schéma supportée
  - Utilisation dans le domaine du streaming afin de valider les schéma notamment
- **Inconvénients:**
  - Not human readable
  - Moins adopté, pas encore intégré à certains systèmes/langage de prog

Les différents Format de stockage (données structurées ou semi-structurées)

- **Avro**



```
{
  "type": "record",
  "name": "Sales",
  "namespace": "com.example",
  "fields": [
    {"name": "sale_id", "type": "string"},
    {"name": "user_id", "type": "string"},
    {"name": "amount", "type": "float"},
    {"name": "timestamp", "type": "long"}
  ]
}
```

Les différents Format de stockage (données structurées ou semi-structurées)

- Orc

Orc (Optimized Row Columnar) est un format de stockage de données orienté colonnes créé et rendu open source par Hortonworks en collaboration avec Facebook.

Il contient des groupes de données par ligne appelés "stripes" et est fortement compressible, réduisant la taille des données d'origine jusqu'à 75%.

- **Avantages:**
  - Adapté au big data: compression, schéma
  - Efficace pour les applications OLAP
- **Inconvénients:**
  - Not human readable
  - Ne Prend pas en charge l’évolution de schéma
  - Moins adopté, pas encore intégré à certains systèmes/langage de prog

extrait d’un article sur le blog de Linkedin

Les enseignements tirés de l'expérience concernant les formats de stockage de données sont les suivants :

- JSON est la norme pour la communication sur Internet.
- Parquet et Avro sont plus optimisés pour le Big Data, mais la lisibilité et la vitesse d'écriture sont assez faibles.
- Parquet est le meilleur choix en termes de performances lors du choix d'un format de stockage de données dans Hadoop, en tenant compte de facteurs tels que l'intégration avec des applications tierces, l'évolution du schéma et le support de types de données spécifiques.
- Les algorithmes de compression jouent un rôle significatif dans la réduction de la quantité de données et l'amélioration des performances.
- Apache Avro est un encodeur universel rapide pour les données structurées, garantissant de bonnes performances lors de l'accès à tous les attributs d'un enregistrement en même temps.
- Le format CSV est généralement le plus rapide à écrire, JSON est le plus facile à comprendre pour les humains, et Parquet est le plus rapide à lire pour un sous-ensemble de colonnes, tandis qu'Avro est le plus rapide à lire toutes les colonnes en une fois.
- Les formats colonnaires tels que Parquet conviennent à une ingestion rapide de données, à une récupération rapide de données aléatoires et à des analyses de données évolutives.
- Avro est généralement utilisé pour stocker des données brutes, et Parquet est utilisé pour des analyses ultérieures après prétraitement, lorsque toutes les champs ne sont pas requis.

Quelques bonnes pratiques

- Choisir le bon format, privilégier le parquet, sauf cas simple où l'on peut choisir le csv ou le json
- Compresser les données pour réduir la taille des fichiers donc le coût de stockage
- Partitionner les données en fonction de la manière dont ces dernières seront lues/accédées
- Mettre du versionning (si besoin)
- Mettre en place un lifecycle afin de déplacer automatiquement les données dans la bonne classe de stockage mais également de supprimer les données non utilisées

Data Ingestion

Data Ingestion

Le processus de collection de la donnée depuis différentes sources à un système de stockage. C’est la première étape de ce qu’on appelle le cycle de vie des données.
C’est une étape très importante et avant de l'entamer il est toujours important de prendre en compte les éléments suivant:

Data Ingestion

- Nature des sources: bases de données, api, stockage objet, systèmes de fichiers ….
- Types et formats des données: structurées (parquet, csv,...), semi-structurés(json, xml, …), audio, vidéo, images
- La fréquence de maj des données sources
- Les schémas (si (semi) structurées) des données sources: évolution, …
- La tailles des données
- Les besoins métiers
- Qualité et fraîcheur attendu

Data Ingestion: types d'ingestion

- **Pull based:**

Ici les consommateurs des données vont chercher la données au niveau des sources.

Exemple: connection sur une db pour récupérer des données, le système de message kafka

- **Push based:**

Ici la source pousse les données vers un système cible.

Exemple: Export de fichiers vers un stockage object

- **Poll based**: similaire au pull mais se lance uniquement quand il y a des changement au niveau de(s) la source(s)

Data Ingestion: Comment

Le comment dépend toujours des types de sources:

- databases (connection jdbc/odbc)
- stockage objet (accès via des rôles)
- api (token)
- système de messaging (producers + queue + consumers)
- cdc
- etc

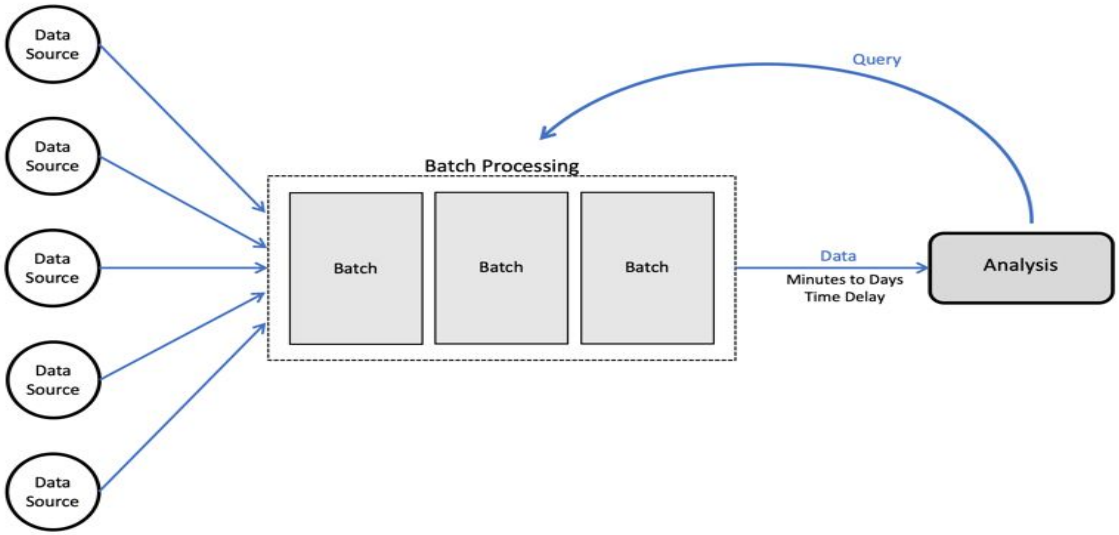
Data Processing

Data Processing

Étape de transformation, d’exploitation de la données. Il existe 3 types de data processing processing: Le Batch processing, le Streaming processing et l’Event processing.

Batch Processing (Traitement par lots)

Le *traitement en batch* consiste à exécuter des jobs de traitements répétitifs contenant des volumes importants. Le traitement se fait généralement sur des données cumulées sur une période bien précise.



Batch Processing (Traitement par lots)

Cas d’usage

- Reporting
- Data warehousing
- Retail (Inventaire)

Avantages et Inconvénient

Avantages

- Simplicité
- Optimisation de l’utilisation des ressources
- Gouvernance et cohérence des données

Inconvénients:

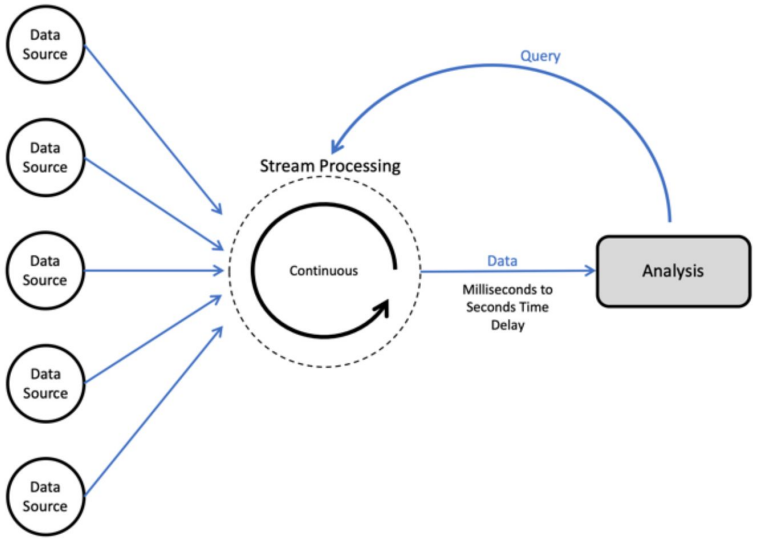
- Manque de fraîcheur des données
- Debugging/Error Recovery (implique généralement de gros volume de données)

Quelques Outils pour faire du batch processing

Orchestrateurs

Streaming Processing (Traitement en continue)

A l'inverse du batch processing, ici le traitement se fait en quasi-instantané (temps réel) sur un flux de données continue. On parle de stream pour désigner ces flux de données continues



Streaming Processing

Cas d’usage

- Détection de fraud (traitements de transactions)
- Recommandations de films sur Netflix/Youtube
- Capteurs/Iot

Avantages et Inconvénient

Avantages

- Qualité des données
- Meilleure expérience utilisateurs
- Pas ou peu de latence
- Scalabilité

Inconvénients:

- Complexe à mettre en place et aussi à maintenir
- Les coûts

Quelques outils pour faire du streaming

Event Processing

Souvent associé au streaming processing de part le event processing est également basé sur du temps réel mais ici le flux n’est pas contenu on réagit plutôt à certains événements.

Event Processing - Cas d'usage

- Trading: acheter ou vendre des actions si le prix atteint un certain seuil
- monitoring de systèmes

Event Processing

Avantages:

- Optimization des ressources
- faible ou pas de latence
- Scalabilité

Inconvénients:

- Complexe à mettre en place