Les enjeux des Big Data

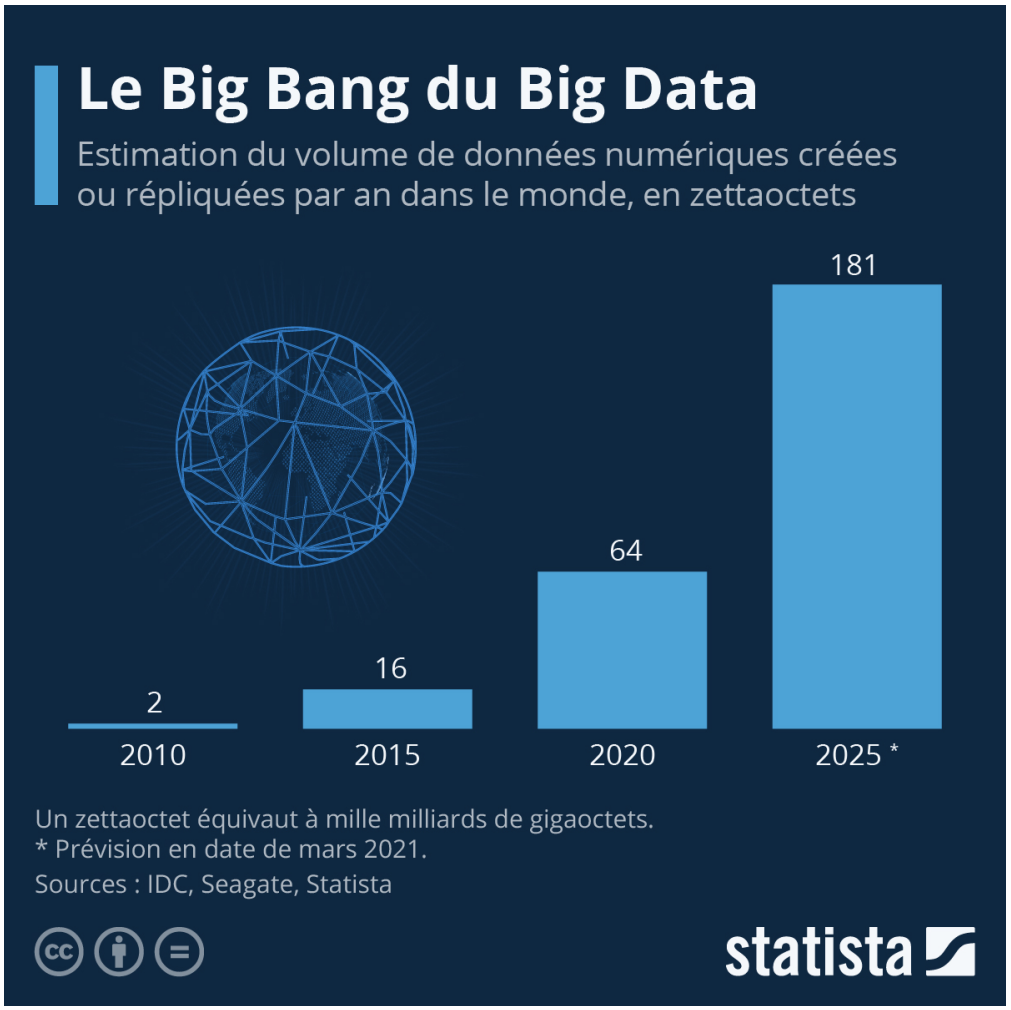
Evaluer la taille des données informatiques

- bit : la plus petite unité de données, prend les valeurs 0 ou 1
- octet (byte) : 8 bits, prend $2^8 = 256$ valeurs
- multiples de l'octet :

Nom	**Symbole**	**Valeur**
kilooctet	ko	10^3
mégaoctet	Mo	10^6
gigaoctet	Go	10^9
téraoctet	To	10^{12}
pétaoctet	Po	10^{15}
exaoctet	Eo	10^{18}
zettaoctet	Zo	10^{21}
yottaoctet	Yo	10^{24}

L'omniprésence d'internet et des réseaux, la multiplication des objets connectés (smartphones, GPS, ...), le développement de bases de données gigantesques, ... Depuis les années 80, le volume de données à traiter a subi une croissance exponentielle.

- Quelques ordres de grandeur
  - un livre : quelques dizaines de Mo
  - un film : 1 à 2 Go
  - le contenu de la Bibliothèque nationale de France (BnF) : 1800 To (au 31/12/2022)
  - données enregistrées chaque jour par Facebook : 4000 To
  - l'ensemble des données électroniques produites en 2021 : près de 80 Zo



(https://fr.statista.com/infographie/17800/big-data-evolution-volume-donnees-numeriques-genere-dans-le-monde/)

Au vue de ces quantités, la question du stockage et du traitement de ces données va se poser...

Quelques éléments sur l'architecture des ordinateurs

Les constituants d'un ordinateur

Le processeur (le "cerveau" de l'ordinateur)

Manipule les données binaires, exécute les instrutions, ...

- CPU (Central Processing Unit)
  - Unité de traitement du processeur
  - De 1 à quelques dizaines de coeurs (traitement paralèlle des instructions)
  - Fréquence de 1 à 5 GHz (1 GHz $\rightarrow 10^9$ opérations élémentaires par seconde)
- GPU (Graphics Processing Unit)
  - Jusqu'à quelques milliers de coeurs
  - Des opérations élémentaires effectuées en parallèle

La mémoire vive ou RAM (Random Access Memory)

Permet un accès rapide (mais temporaire) aux données

- De 4 Go à 2 To (windows 11 par exemple)

La mémoire de masse

	Capacité	Débit	Latence*
Disque dur HD	100 Mo - 10 To	50 - 120 Mo/s	2 ms
SSD	Jusqu'à 1 To	200 - 500 Mo/s	20 - 200 ns

*Délai entre la commande et le résultat attendu

Clusters d'ordinateurs

- De nombreux ordinateurs groupés dans des armoires (rack), eux-mêmes reliés entre eux.
- Connexion rapide entre les différents noeuds du cluster.

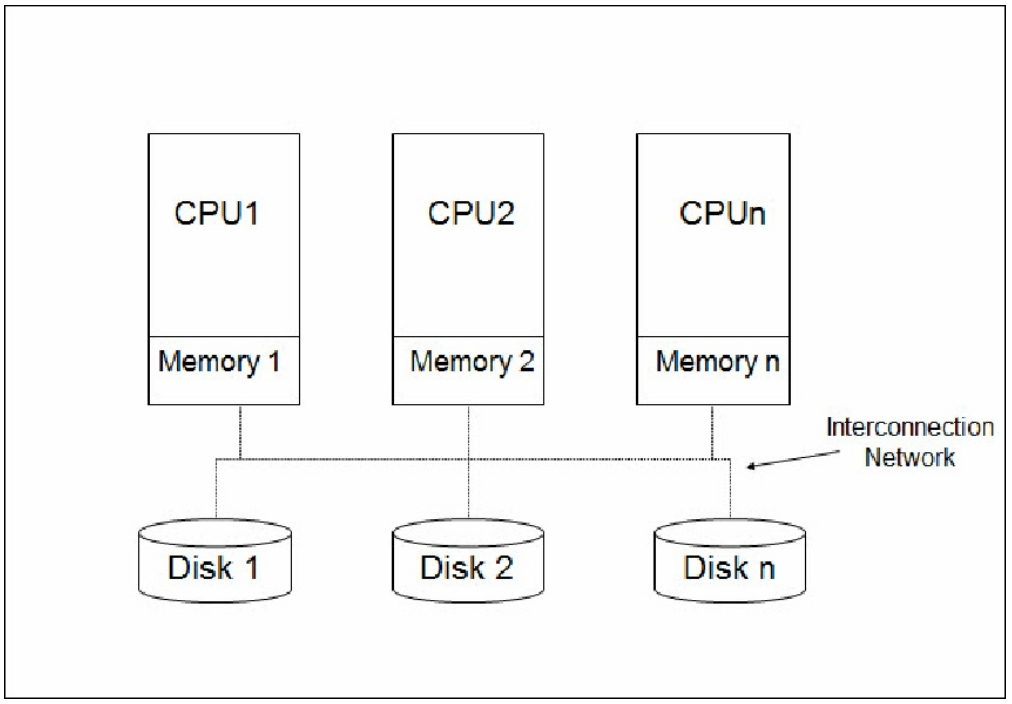
Facebook utilise ainsi un cluster de 4000 machines, avec un stockage de quelques centaines de millions de Go.

Coût énergétique

- Une machine : 400 W
- Un cluster de 4000 machines (utilisé par Facebook): 1.6 GW
- Indicateur d'efficacité énergétique (PUE , Power Usage Effectiveness) : rapport entre l'énergie totale consommée par l'ensemble du centre d'exploitation (avec la climatisation) et la partie qui est effectivement consommée par les systèmes informatiques que ce centre exploite. Facebook affiche un PUE de l'ordre de 1.1. D'où une consommation électrique de l'ordre de 1.8 GW pour un cluster de 4000 machines.

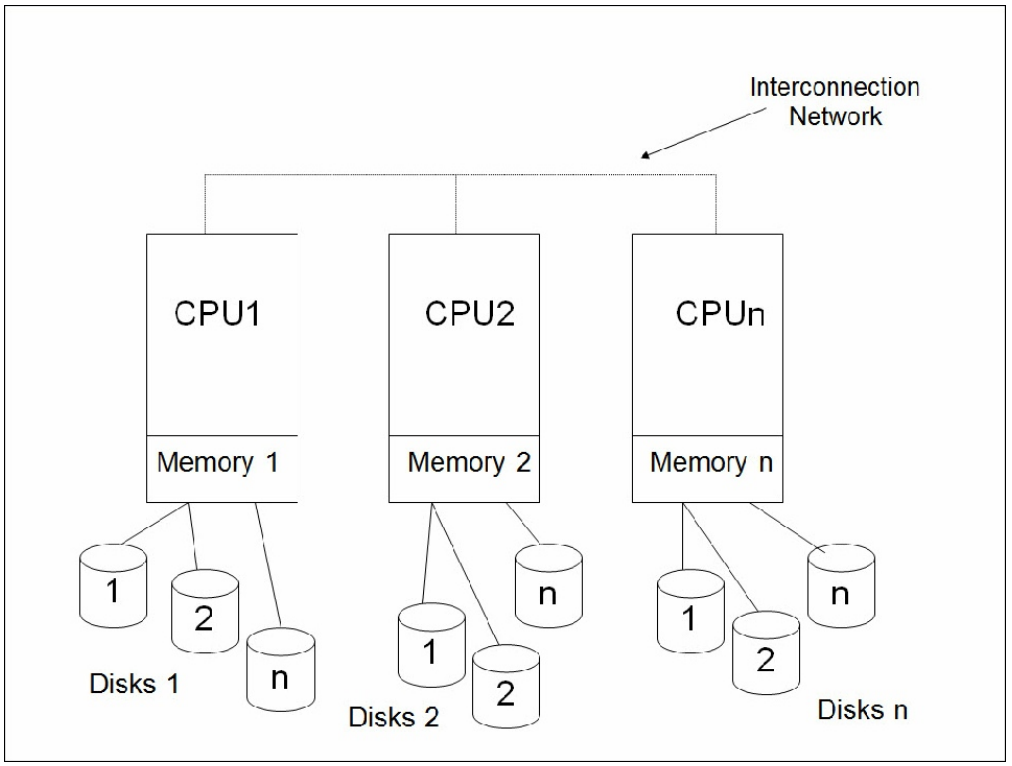
Architecture à disques partagés

Un serveur de fichiers abrite l'ensemble des fichiers et les distribue aux machines qui en ont besoin.



Architecture sans partage (Shared Nothing)

Chaque machine a son propre système de fichiers et travaille sur les fichiers qu'elle gère.



Echange de données entre les ordinateurs

L'unité est le bit/seconde et ses multiples (b/s, kb/s, Mb/s, Gb/s) ; attention à ne pas confondre avec des octets!

- Connexion ethernet : entre 100 Mb/s et 40 Gb/s
- Connexions spécialisées très haut de gamme : de 40 à 100 Gb/s

Big Data : les contraintes dues à la taille des données

- On parle de "Big Data" lorsque la taille des données traitées est plus grande que la taille de la mémoire vive (RAM) d'un ordinateur, voire de l'ensemble de la RAM du cluster.
- Étant donné le temps de transfert des données, chaque noeud du cluster doit traiter ses propres données (architecture "share nothing"), et on doit minimiser les échanges de données entre les noeuds.
- Extensiblité : le cluster doit pouvoir grandir lorsque la taille des données augmente
- Résilience : le cluster doit pouvoir résister à la défaillance d'une de ses composantes (noeud ou même rack)
- Répartition des tâches : chaque machine effectue des opérations sur les données qu'elle abrite sur ses disques propres

Big Data : des données souvent non structurées ou avec une structure riche

Les bases de données classique (type SQL) cherche à définir un **modèle conceptuel de données** avant leur stockage, afin de limiter l'espace disque utilisé et donc les coûts de stockage. La **3ème forme normale** garantit également la non-duplication des données.

La baisse des coûts de stockage ont permis l'émergence de nouveaux types de bases de données (JSON par exemple) où les informations sont dupliquées (résilience).

- Exemple : enregistrement des ventes d'un magasin
  - Une base de données classique consistera en une table des acheteurs et une table des produits (avec unicité de l'information par ligne) associé à une table de liens entre acheteurs et produits
  - Une base de données type JSON réalisera un enregistrement par couple "acheteur-produit" : il n'y a plus unicité de l'information.

Voici un exemple de données codées en json :

JSON

```
{
  "glossary": {
    "title": "example glossary",
     "GlossDiv": {
        "title": "S",
     "GlossList": {
            "GlossEntry": {
                "ID": "SGML",
            "SortAs": "SGML",
            "GlossTerm": "Standard Generalized Markup Language",
            "Acronym": "SGML",
            "Abbrev": "ISO 8879:1986",
            "GlossDef": {
                    "para": "A meta-markup language, used to create markup languages such as DocBook.",
                "GlossSeeAlso": ["GML", "XML"]
                },
            "GlossSee": "markup"
             }
           }
       }
   }
}
```

et un autre :

JSON

```
{"widget": {
    "debug": "on",
    "window": {
        "title": "Sample Konfabulator Widget",
        "name": "main_window",
        "width": 500,
        "height": 500
    },
    "image": {
        "src": "Images/Sun.png",
        "name": "sun1",
        "hOffset": 250,
        "vOffset": 250,
        "alignment": "center"
    },
    "text": {
        "data": "Click Here",
        "size": 36,
        "style": "bold",
        "name": "text1",
        "hOffset": 250,
        "vOffset": 100,
        "alignment": "center",
        "onMouseUp": "sun1.opacity = (sun1.opacity / 100) * 90;"
    }
}}
```

Exercice 1

On s'intéresse à un fichier extrait par l'API de recherche de la SNCF. On va essayer de comprendre sa structure en utilisant des outils python.

- Question 1

Télécharger le fichier suivant (par exemple en utilisant la commande unix `wget`) :

https://raw.githubusercontent.com/sdpython/ensae_teaching_cs/master/_doc/notebooks/td2a_eco/stop_areas.json

Regarder à quoi il ressemble.

- Question 2

Charger le fichier en utilisant la commande `load` du module `json`

- Question 3

Comprendre la strucure des données de ce fichier. On peut utiliser une version mieux formatée en utilisant la commande `pprint` du module `pprint` (pretty print).

- Question 4

Extraire du fichier les informations suivantes (par exemple en écrivant un fichier `csv`) :

nom de la gare	latitude	longitude

Solution

▾ Question 1

SHELL

```
wget https://raw.githubusercontent.com/sdpython/ensae_teaching_cs/master/_doc/notebooks/td2a_eco
/stop_areas.json
```

▼ Question 2

PYTHON

```
import json
with open('stop_areas.json','r') as f:
    data=json.load(f)
```

▼ Question 3

PYTHON

```
from pprint import pprint
pprint(data)

print(type(data))
print(data.keys())
print(type(data['stop_areas']))
print(data['stop_areas'][0].keys())
print(data['stop_areas'][0]['coord'])
```

▼ Question 4

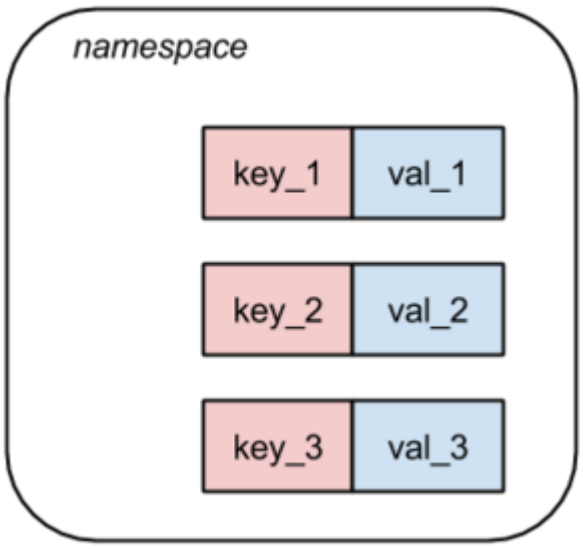
PYTHON

```
import pandas as pd
df = pd.DataFrame(columns=["nomGare","latitude","longitude"]) # création d'un dataframe vide à 3 colonnes

for gare in data['stop_areas'] :
    lat,lon = gare["coord"]["lat"],gare["coord"]["lon"]
    df.loc[len(df),] = [gare['name'],lat,lon]  # ajoute une ligne au dataframe

print("Résultats Question 4 :")
print(df, end="\n\n")
df.to_csv("ex1_out.csv", index=False)
print("Création du fichier : ex1_out.csv")
```

Les bases de données NoSQL (Not Only SQL)

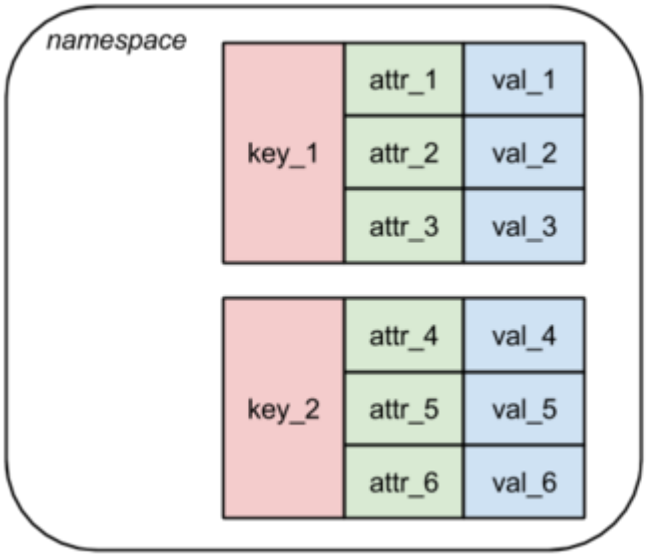
On connait le modèle tabulaire utilisé par les bases de données relationnelles. Pour des données ayant des structures plus complexes, ou plus susceptibles d'évolutions, on peut envisager d'autres types de stockage de données :

- Modèle clé-valeur



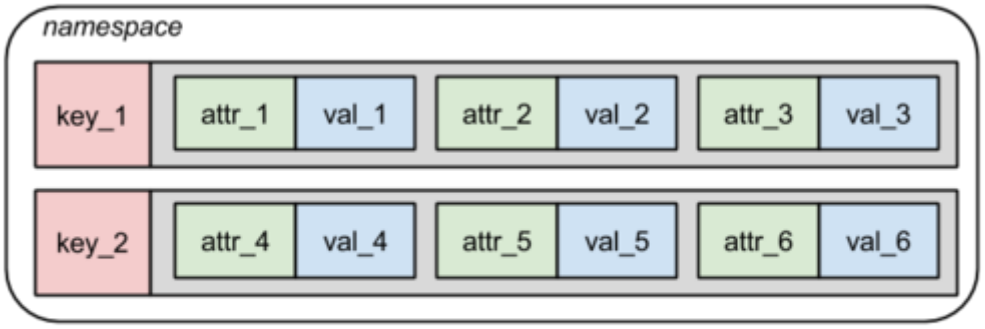
Modèle de données - base (clé, valeur)

- Modèle document



Modèle de données - base orientée document

- Modèle colonne



Modèle de données - base orientée colonnes

Types de calcul

- Calcul distribué
  - Tâche affectée à chaque noeud sur une portion des données. Les noeuds ne partagent pas de ressourcent entre eux et communiquent via un cluster.
  - Ajout simple de noeuds pour augmenter la vitesse de calculs
  - Résilience
- Calcul parallèle
  - Tâches exécutées simultanément sur une ressource commune
  - Augmentation de la vitesse de calculs par amélioration de la puissance du processeur
  - Mauvaise tolérance aux pannes

Les différentes composantes d'une architecture bigdata

- Gestion des données
  - Système de fichiers distribués et redondant : HDFS
  - Base de données NOSQL : MongoDB, HBase, Cassandra,...
- Exécution des tâches

- paradigme map-reduce
- ordonnanceur yarn ou mesos
- Apache Spark

- Sérialisation et formats de fichiers :
  - Parquet
  - Apache Arrow
- Interaction avec le moteur de calcul : ingestion de données et interfaces de haut niveau
  - PIG
  - HIVE

Index (https://math.univ-angers.fr/~badreau/)

Last updated 2023-10-11 13:11:17 +0200