MSE | MASTER OF SCIENCE IN ENGINEERING

TSM_Data_Management

Labo 4

12 December 2023

Daniel Ribeiro Cabral

MSE | Data Science

Hes·so
Haute Ecole Spécialisée de Suisse occidentale
Fachhochschule Westschweiz
University of Applied Sciences and Arts Western Switzerland

Exercice 1

Pour cet exercice, nous avions une marche a suivre afin de nous familiariser avec les outils d'Apache Superset. Pour cela, nous avions une commande SQL qui nous y étais donner. Cette commande nous donnait les données suivantes :

Afficher les nombres de clients français totaux

Voici la commande SQL effectuée :

```
SELECT * FROM dimcustomer AS c INNER JOIN dimgeography AS g
ON c.GeographyKey = g.GeographyKey
WHERE g. FrenchCountryRegionName = "France"
```

Voici le chart obtenu dans le Dashboard :

Exercice 2

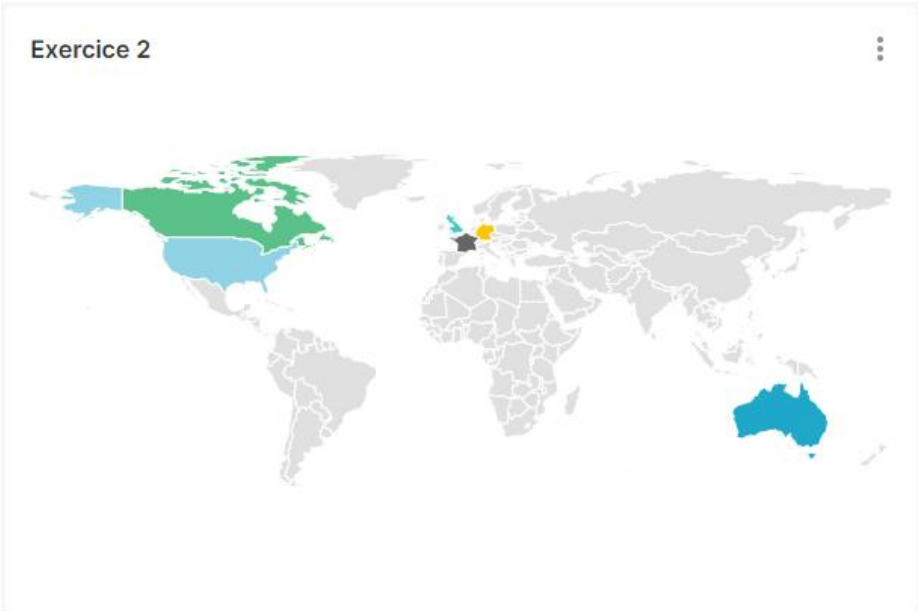
Pour cet exercice, nous avions comme objectif de :

Afficher le nombre de clients par pays (en anglais)

Requête SQL :

```
SELECT EnglishCountryRegionName, COUNT(CustomerKey) as NumberOfCustomers
FROM dimcustomer INNER JOIN dimgeography
ON dimcustomer.GeographyKey = dimgeography.GeographyKey
GROUP BY EnglishCountryRegionName;
```

Voici le chart obtenu dans le Dashboard :



Exercice 3

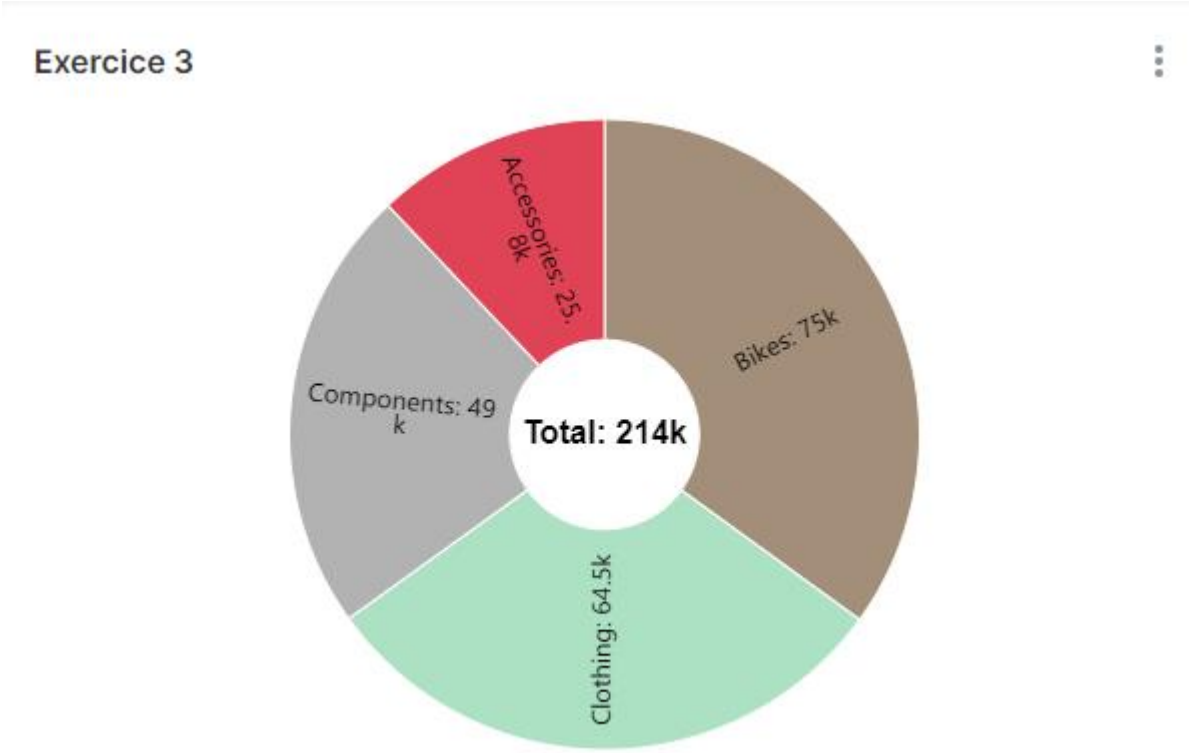
Pour cet exercice, nous avions comme objectif de :

Afficher nombre de ventes par catégories des produits anglais faites par les revendeurs

Requête SQL :

```
SELECT
    pc.EnglishProductCategoryName,
    SUM(fs.OrderQuantity) as NumberOfSales
FROM
    factresellersales fs
JOIN
    dimproduct p ON fs.ProductKey = p.ProductKey
JOIN
    dimproductsubcategory psc ON p.ProductSubcategoryKey = psc.ProductSubcategoryKey
JOIN
    dimproductcategory pc ON psc.ProductCategoryKey = pc.ProductCategoryKey
GROUP BY
    pc.EnglishProductCategoryName;
```

Voici le chart obtenu dans le Dashboard :



Exercice 4

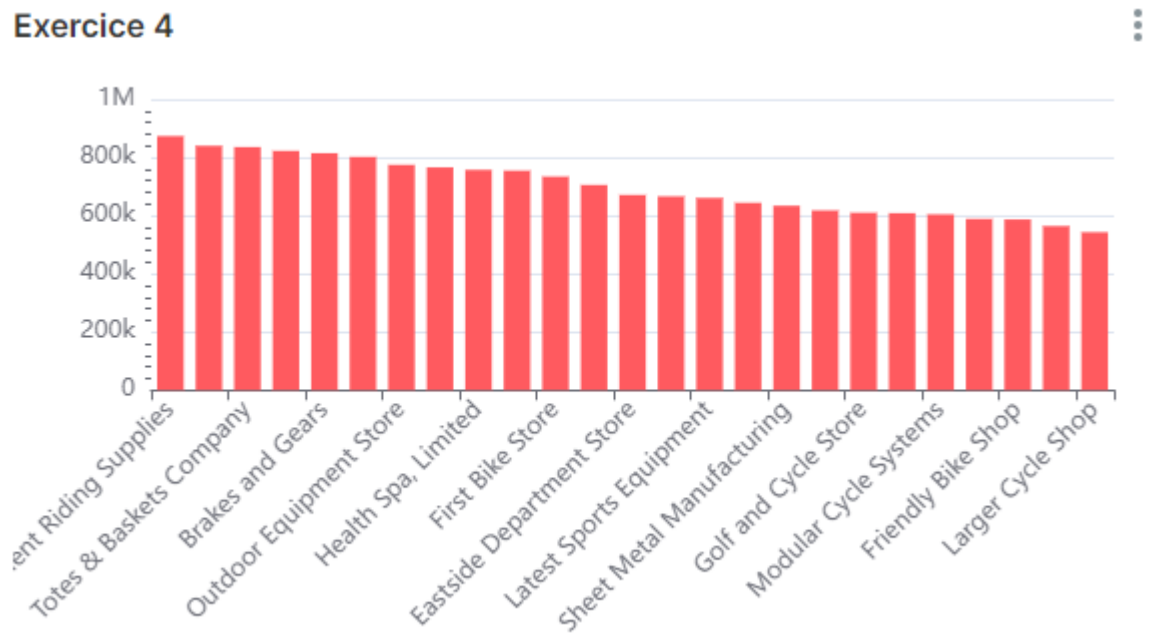
Pour cet exercice, nous avions comme objectif de :

Afficher le cout total des produits des revendeurs

Requête SQL :

```
SELECT
    r.ResellerName,
    SUM(fs.TotalProductCost) AS TotalProductCost
FROM
    factresellersales fs
JOIN
    dimreseller r ON fs.ResellerKey = r.ResellerKey
GROUP BY
    r.ResellerName
ORDER BY
    TotalProductCost DESC
LIMIT 25; -- pas obligatoires on peut le faire directement dans apache superset
```

Voici le chart obtenu dans le Dashboard :



Exercice 5

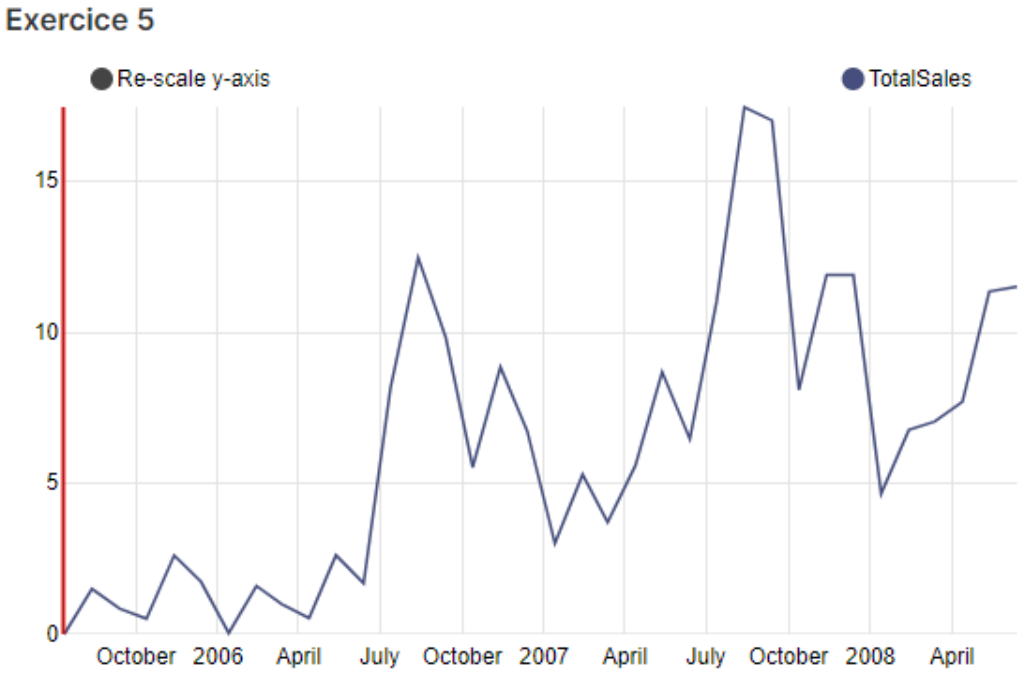
Pour cet exercice, nous avions comme objectif de :

Afficher la quantité des ventes des revendeurs sur le temps

Requête SQL :

```
SELECT
    dd.FullDateAlternateKey AS DateOfSale,
    SUM(fs.OrderQuantity) AS TotalSales
FROM
    factresellersales fs
JOIN
    dimdate dd ON fs.DueDateKey = dd.DateKey
GROUP BY
    dd.FullDateAlternateKey
ORDER BY
    dd.FullDateAlternateKey;
```

Voici le chart obtenu dans le Dashboard :



Le graphique démontre le la quantité (normalisé par Apache Superset) de ventes effectuées à chaque date durant la période donnée. Dans ce cas-là, de 2005 à 2008.

Exercice 6

Pour cet exercice, nous avions comme objectif de :

Afficher la quantité des ventes des revendeurs par catégorie sur le temps

Requête SQL :

```
SELECT
    d.FullDateAlternateKey AS SaleDate,
    pc.EnglishProductCategoryName AS ProductCategory,
    SUM(fs.OrderQuantity) AS QuantitySold
FROM
    factresellersales fs
JOIN
    dimdate d ON fs.DueDateKey = d.DateKey
JOIN
    dimproduct p ON fs.ProductKey = p.ProductKey
JOIN
    dimproductsubcategory psc ON p.ProductSubcategoryKey = psc.ProductSubcategoryKey
JOIN
    dimproductcategory  pc ON pc.ProductCategoryKey = psc.ProductCategoryKey
GROUP BY
    SaleDate, ProductCategory
ORDER BY
    SaleDate, ProductCategory;
```

Voici le chart obtenu dans le Dashboard :

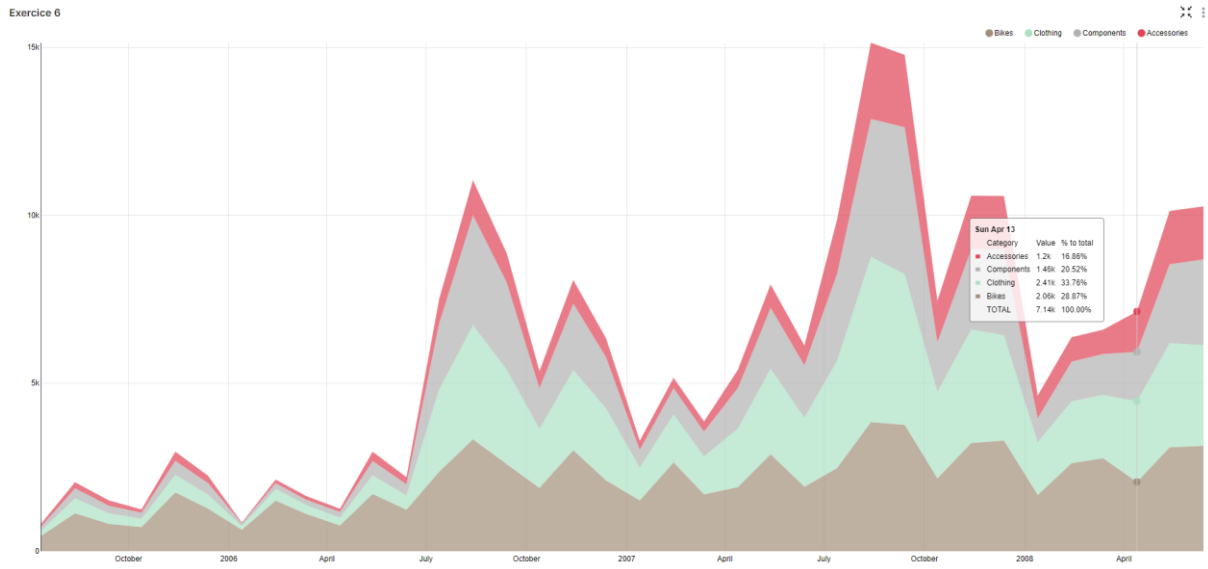


Figure 1 : Exercice 6 first look

Exercice 7

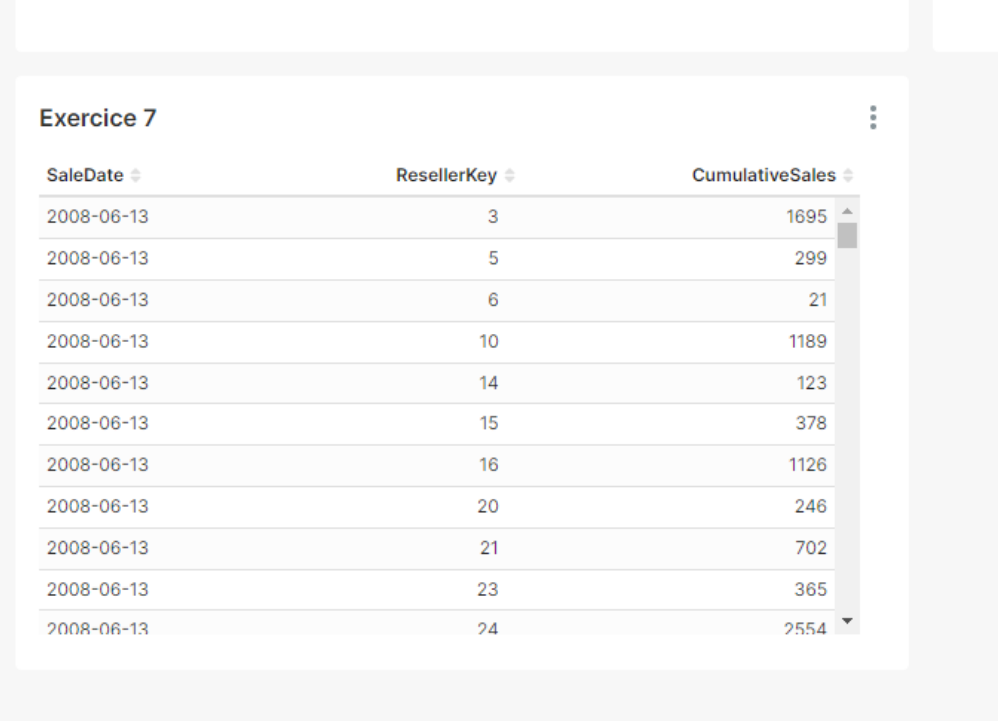
Pour cet exercice, nous avions comme objectif de :

Afficher la somme cumulée des ventes par jour de chaque revendeur et le disposer dans un tableau

Requête SQL :

```
SELECT
    d.FullDateAlternateKey AS SaleDate,
    fs.ResellerKey,
    SUM(fs.OrderQuantity) OVER (PARTITION BY fs.ResellerKey ORDER BY
d.FullDateAlternateKey) AS CumulativeSales
FROM
    factresellersales fs
JOIN
    dimdate d ON fs.DueDateKey = d.DateKey
ORDER BY
    fs.ResellerKey, d.FullDateAlternateKey;
```

Voici le chart obtenu dans le Dashboard :



Exercice 8

Pour cet exercice, nous avions comme objectif de :

Afficher par mois le nombre de ventes cumulatives de chaque revendeur avec la somme tout au bout (avec certaines contraintes de couleurs)

Requête SQL :

```
SELECT
    fs.ResellerKey,
    strftime('%Y-%m', d.FullDateAlternateKey) AS MonthOfSale,
    SUM(fs.OrderQuantity) AS QuantitySold
FROM
    factresellersales fs
JOIN
    dimdate d ON fs.DueDateKey = d.DateKey
GROUP BY
    fs.ResellerKey, MonthOfSale
ORDER BY
    fs.ResellerKey, MonthOfSale;
```

Voici le chart obtenu dans le Dashboard :

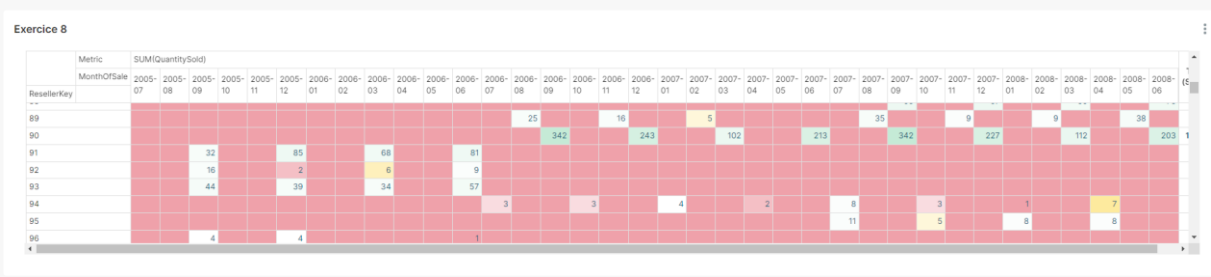


Figure 3 : Exercice 8 Global overview

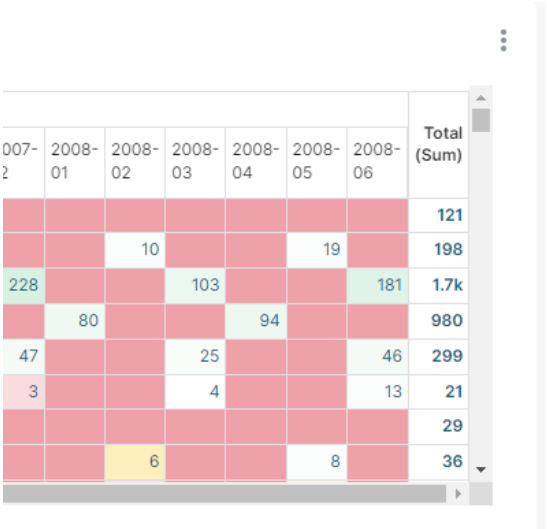


Figure 2 : Total sum view in Table

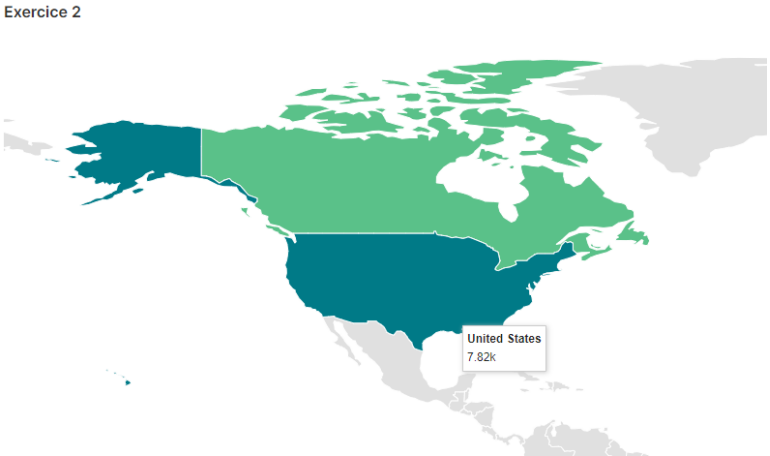
Exercice 9

Avantage de la Visualisation sur le Web Comparée à un Document Imprimé :

Le premier avantage de la visualisation des données sur le Web par rapport à un document imprimé est l'interactivité. Les visualisations Web peuvent être interactives, permettant aux utilisateurs de filtrer, de trier, et d'explorer les données de manière dynamique. Cela rend les données plus accessibles et compréhensibles pour un plus large public.

Exemples :

- **Filtrage et Sélection** : Sur le Web, les utilisateurs peuvent cliquer sur des éléments



spécifiques d'un même graphique pour filtrer et voir des détails supplémentaires (on parle ici des légendes). Par exemple, dans un graphique de ventes, cliquer sur un produit spécifique pourrait afficher ses ventes au fil du temps ou par région. Durant le TP, nous avons pu voir plusieurs cas comme ça notamment dans la Map de l'exercice 2. On peut cliquer sur un pays et avoir le nombre de clients dans ce même pays.

- **Mises à Jour des données en temps réel** : Les graphiques dans le Web peuvent être connectées à des sources de données en direct, permettant des mises à jour en temps réel. Cela est particulièrement utile pour surveiller des indicateurs importants de performance ou des tendances du marché qui évoluent rapidement. Comme un exemple concret, dans le tp, nous pouvons imaginer une société qui au lieu de devoir chaque jour mettre à jour les graphiques. Ils le feront automatiquement afin de voir l'évolution des ventes pour chaque « resseler ».

Expliquer ce qu'a indirectement apporté le modèle de communication de Shannon (1948) dans le domaine de visualisation des données.

Afin d'expliquer son apport indirect il faut commencer par comprendre ce qu'est le modèle de communication de Shannon 1948. Shannon c'est concentré sur la façon dont l'information est transmise d'un point A à un point B. En y identifiant qui fait quoi. Nous avons toutes ces étapes-là :

- **La source d'information** : d'où provient le message.
- **L'émetteur** : qui encode le message en un signal.
- **Le canal** : par lequel le signal est transmis.
- **Le récepteur** : qui décode le message.
- **La destination** : où le message arrive.

Ce modèle a indirectement influencer le domaine de la visualisations des données. Voici dans quels aspects :

1. **Transmission efficace de l'information** : On parle ici de transmettre des données de manière claire et efficace minimisant toutes sortes de problèmes qui pourrait arriver comme des perturbations et du bruit. Le but étant dont de créer des graphiques qui transmettent des données claires et pas de manière ambiguë ou compliquer à comprendre.
2. **Encodage et décodage** : Ce sous-entendu, veut dire que dans le monde de visualisation, les données encodées sont les données qui se trouve dans les graphiques. Pour qu'ensuite les utilisateurs la décodent (comprenne) les informations qui s'y trouvent. Le choix des graphs est donc crucials.
3. **Traitement du bruit** : Le concept de bruit peut être interprété comme toute complexité inutile qui empêche la compréhension claire des données. Une bonne visualisation des données doit donc minimiser ce bruit visuel pour permettre une bonne interprétation des données.

Pour ce résumé ce modèle, le but est de transmettre l'information de manière efficace et claire, en minimisant les perturbations ou les ambiguïtés.

Une échelle a été appliquée sur les séries de données temporelles du premier schéma de la Figure 4, résultant à l'affichage du deuxième schéma. Expliquer quelle est cette échelle et décrire quelle information elle donne par rapport au premier schéma.

Nous avons 2 graphiques de séries temporelles. L'échelle appliquée pour passer du premier au deuxième est une échelle logarithmique. Cette transformation est utilisée pour linéariser les croissances exponentielles, ce qui rends les données plus faciles à analyser et à comparer, surtout lorsque les variations sont très grandes.

Dans le premier graphique, la courbe indique une croissance exponentielle. Lorsqu'on applique une échelle logarithmique, cette croissance exponentielle apparaît comme une ligne droite dans le deuxième graphique. Cela indique que le taux de croissance est constant sur une échelle exponentielle. Cette application de l'échelle logarithmique permet de transformer une série temporelle exponentielle en une série temporelle qui a une relation linéaire. Cela aide également à identifier les anomalies ou les déviations par rapport à la tendance exponentielle attendue.

Vue globale du dashboard