01/2024

RAPPORT INTÉGRATION DE DONNÉES

Rayan BEN YACOUB
Furkan NARIN
Meriam BOUMEDIENE
BUT 2AFA VCOD

SOMMAIRE

INTRODUCTION

Nil, la plateforme de e-commerce de livres basée en France, souhaite mieux comprendre le comportement des utilisateurs sur son site www.nil.fr. Chaque livre sur la plateforme a une page dédiée, et les clients peuvent commander directement depuis ces pages.

Dans ce projet, nous proposerons une modélisation relationnelle complète, détaillant les faits, les dimensions, et les attributs. En utilisant SQL sur RStudio, nous construirons un entrepôt de données pour extraire des mesures cruciales, permettant à Nil d'optimiser ses opérations et d'analyser le comportement des utilisateurs.

MODELISATION RELATIONNELLE

Le choix des dimensions et des attributs dépendent des informations que nous souhaitons stocker. Pour ce projet, nous avons retenu 6 dimensions et 3 faits afin de répondre aux exigences spécifiques de la plateforme.

Ainsi, voici les tables de dimensions :

Table “Livre” :
Les attributs de base du livre sont inclus pour stocker des informations essentielles comme le titre ou l’auteur.

Table “Client” :
Stock les informations sur le client.

Table “Catégorie” :
Permet à l’utilisateur de gérer les catégories des livres afin d’explorer des livres par catégorie depuis la page d’accueil.

Table “PageDetail” :
Stock les détails spécifiques à l’exploration de livre comme les avis, le résumé
.
Table “Maison d’édition” :
Représente les maisons d’édition des livres.

Table “SociétéLivraison”:
Permet de gérer les sociétés de livraison et ainsi suivre le traitement des commandes.

Et les tables de faits :

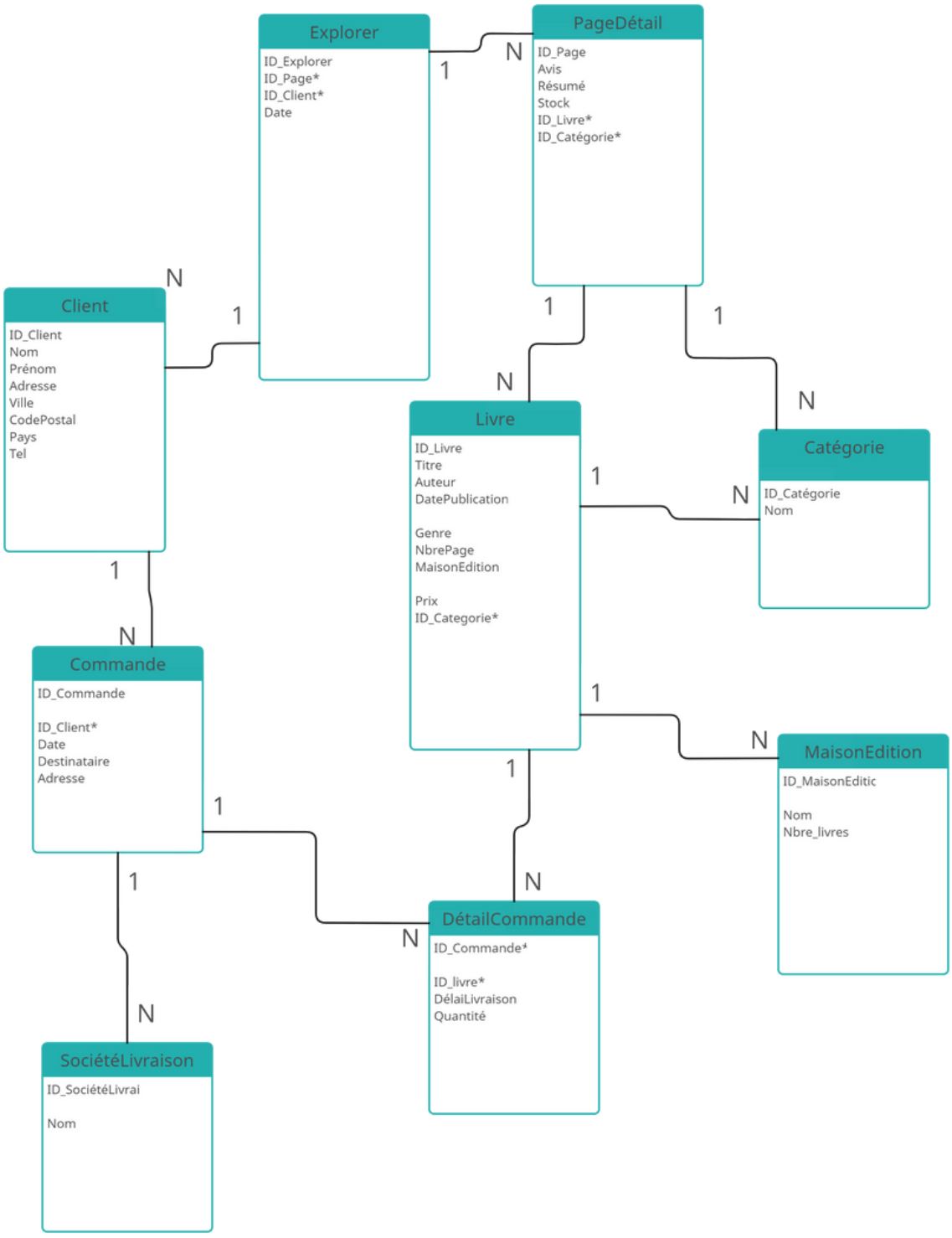
Table “Explorer” :
Elle permet de faire le lien entre Client et la page détail des livres.

Table “Commander” :
Stock les informations sur les commandes des clients et suit le traitement des commandes passées sur le site.

Table “DetailCommande”:
Contient les détails spécifiques sur les articles inclus dans chaque commande comme la quantité achetée.

MODELISATION RELATIONNELLE

Voici la modélisation relationnelle :



MODELISATION RELATIONNELLE

Granularité :

La table de fait “Commande” a une granularité de transaction puisque que nous enregistrons chaque commande passée par client en tant qu’entité distincte. Chaque ligne dans la table représentera une transaction unique. Ainsi, la table “Détail Commande” suivra la même logique.

Type de gestion des changements:

Table “Livre” :
Type 1 : écrasement de l’ancienne valeur par la nouvelle
Les informations de base des livres comme le titre ou l’auteur changent rarement donc il n’est pas crucial de conserver un historique complet.

Table “Client” :
Type 2 : ajout d’une ligne ou type 1 (à voir si les infos clients importants)
Les informations des clients changent rarement mais l’historique des modifications peut rester important.

Table “Catégorie” :
Type 3 : ajout d’une colonne
Les informations des catégories changent rarement mais il est important de pouvoir associer l’ancienne valeur à la nouvelle.

Table “PageDetail” :
Type 2 : ajout d’une ligne
Les détails des pages peuvent changer et il est important de conserver un historique des modifications. Ce type permet de suivre les différentes versions au fil du temps avec des ID unique.
.
Table “Maison d’édition” :
Type 3 : ajout d’une colonne
Les informations de maison d’édition changent rarement mais il est important de pouvoir associer l’ancienne valeur à la nouvelle.

Table “SociétéLivraison”:
Type 3 : ajout d’une colonne
Les informations des sociétés de livraison changent rarement mais il est important de pouvoir associer l’ancienne valeur à la nouvelle.

MODELISATION RELATIONNELLE

Dimensions douteuses : dimension pour laquelle le même attribut peut apparaître plusieurs fois

La table ”Client” est une dimension douteuse car un client peut apparaître plusieurs fois dans différentes lignes de la table (ex: même nom et prénom)

La table “Livre” peut être également une dimension douteuse si des livres ont le même titre.

Nous n’avons pas de dimensions générées ou de dimensions causales.

DEVELOPPEMENT SQL

Ensuite, nous avons pu développer notre entrepôt de données à partir de cette modélisation.

Voici un exemple de création d'une table :

```
##CREATION TABLE CATEGORIE
dbExecute(bd, "CREATE TABLE Categorie (
    id_categorie BIGINT,
    Nom VARCHAR(55),
    CONSTRAINT pk_catid PRIMARY KEY (id_categorie)
);")
dbListFields(bd, "Categorie")
```

Insertion de données dans une table :

```
# Table Categorie
categories_data <- data.frame(
  id_categorie = 1:7,
  Nom = c("Science-fiction", "Policier", "Romance", "Aventure", "Realiste", "Fantastique", "Historique")
)

for (i in 1:nrow(categories_data)) {
  query <- sprintf("INSERT INTO Categorie VALUES (%d, '%s');", categories_data$id_categorie[i], categories_data$Nom[i])
  dbExecute(bd, query)
}
dbReadTable(bd,"categorie")
```

DEVELOPPEMENT SQL

Nous avons maintenant listé deux mesures par niveau d'additivité et rédigé les requêtes SQL associées.

Fait additif : additionnable suivant toutes les dimensions

- **Mesure 1 :** Quantité Vendue

Cette mesure est additionnable car la quantité vendue mesure la même entité (la quantité d'un livre) dans chaque transaction. L'addition de ces quantités sur différentes dimensions (produits, clients, périodes, etc.) donne un total significatif qui représente la quantité totale vendue à travers l'ensemble des transactions.

- **Mesure 2 :** Nombre Total de Commandes

Peu importe la dimension, le nombre total de commandes est additionnable. On peut agréger ce nombre sur n'importe quelle dimension sans perdre la signification de la mesure.

Fait semi additif : additionnable seulement suivant certaines dimensions

- **Mesure :** Niveau de stock

Le niveau de stock est semi additif car il peut être agrégé de manière significative selon la dimension temporelle (par exemple, à la fin de chaque mois) , mais pas nécessairement selon d'autres dimensions.

- **Mesure 2** : Nombre de client

Cette mesure est semi additive, car elle peut être additionnée selon la dimension temporelle, par exemple mais non selon la dimension produit.

Fait non additif : Non additionnable, peu importe la dimension

- **Mesure 1** : délai de livraison moyen

Le délai de livraison n'est pas additive, car additionner les délais sur différentes dimensions ne fournit pas une mesure globale significative de la performance globale de la logistique.

- **Mesure 2** : Taux de livre Vendu par catégorie

Le Taux de Livres Vendus par Catégorie est une mesure non additive car c'est un ratio et tout les ratios sont des faits non additif

DEVELOPPEMENT SQL

Requête SQL :

Fait additif

- Mesure 1 : Quantité Vendue

```
SELECT sum(Quantite) AS QteTotal
        FROM detailcommande;
```

- Mesure 2 : Nombre Total de Commandes

```
SELECT COUNT (DISTINCT id_commande) AS NbTotalCommande
        FROM Commande;
```

Fait semi additif

- Mesure 1 : Niveau de stock

```
SELECT SUM(STOCK) AS stockTotal
            FROM PageDetail;
```

- Mesure 2 : Nombre de clients

```
SELECT COUNT (DISTINCT id_client) AS NbTotalClient
FROM Client;
```

Fait non additif

- Mesure 1 : délai de livraison moyen

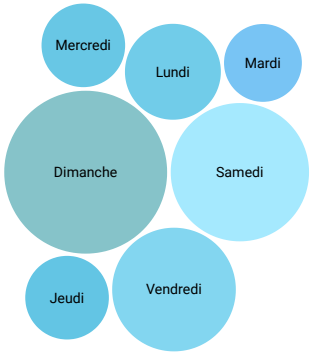
```
SELECT AVG(DELAILIV)
        FROM detailcommande;")
```

- Mesure 2 : Taux de livre Vendu par catégorie

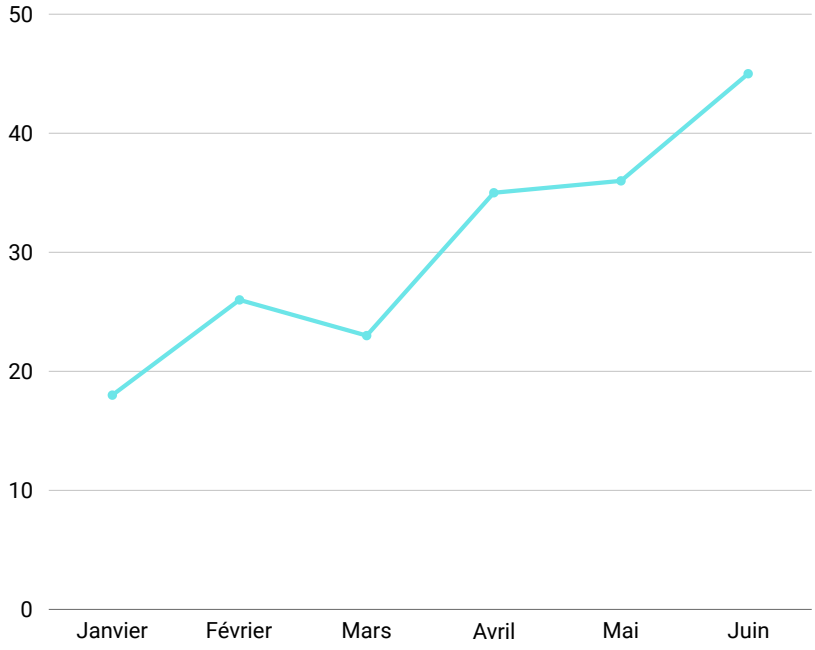
```
SELECT SUM(quantite)FROM detailcommande; #quantité totale vendue

SELECT  l.categorie, SUM(d.quantite ) as TotalVendu,
(SUM(d.quantite)/ (SELECT SUM(d.quantite)FROM detailcommande))*100 AS TauxVendu
        FROM detailcommande AS d
        LEFT JOIN livre AS l ON l.id_livre = d.id_livre
        GROUP BY 1;
```

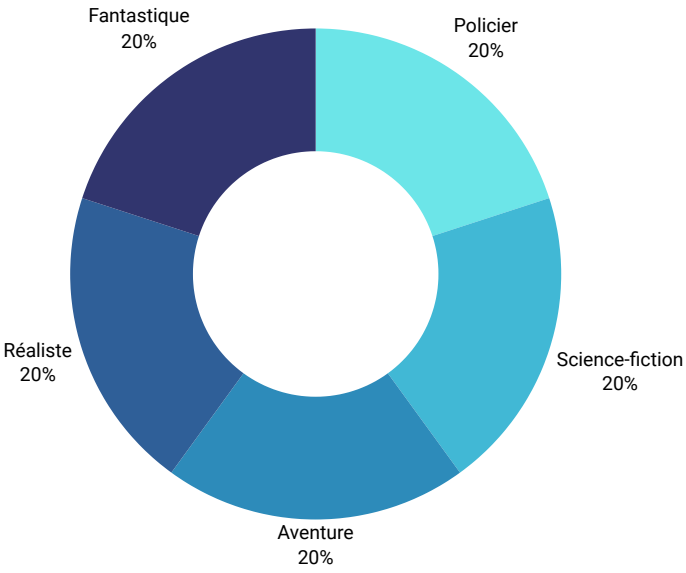
MAQUETTE REPORTING



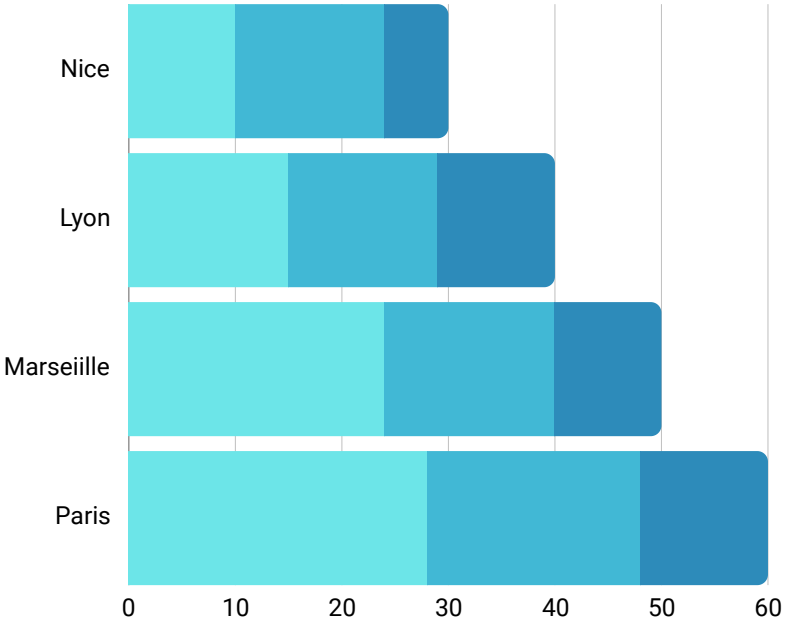
Tendance mensuelle des ventes : premier semestre



Répartition des ventes par catégorie de livres



Distribution des commandes par société de livraison et par villes



Carte des jours les plus actifs en Exploration