Cahier des charges - Groupe 4

Dump de la base de données disponible dans `dump.sql`

Membres du groupe :

- Thomas BARILLE
- Germain LEIGNEL

But de la base de données e-commerce

Le but de cette base de données est de gérer un système de ecommerce en stockant et en organisant les informations relatives aux clients, aux produits, aux catégories, aux méthodes de paiement, aux commandes, aux adresses et aux images. La base de données permettra aux clients de naviguer dans les catégories, de rechercher des produits, de passer des commandes, de laisser des avis et d'effectuer des paiements.

Tables

- adress : Informations d'adresse des clients.
- image : Liens vers les images des produits.
- users : Informations sur les utilisateurs.
- product : Informations sur les produits.
- category : Catégories de produits.
- payment_method : Méthodes de paiement disponibles.
- review : Avis des clients sur les produits.
- orders : Informations sur les commandes passées.
- product_category : Table de liaison entre les produits et les catégories.
- illustrates : Table de liaison entre les images et les produits.
- fullfils : Table de liaison entre les produits et les commandes.

Vues

Produits les mieux notés

Cette vue affiche les produits ayant la meilleure note moyenne parmi tous les avis reçus.

```sql
CREATE VIEW best_rated_products AS
SELECT p.id, p.name, p.description, p.price, AVG(r.rating) AS average_rating
FROM product p
JOIN review r ON p.id = r.product_id
GROUP BY p.id, p.name, p.description, p.price
ORDER BY average_rating DESC;
```

Commandes par utilisateur

Cette vue affiche le nombre total de commandes et le montant total dépensé pour chaque utilisateur.

```sql
CREATE VIEW orders_by_user AS
SELECT u.id, u.username, COUNT(o.id) AS total_orders, SUM(o.price) AS total_spent
FROM users u
JOIN orders o ON u.id = o.users_id
GROUP BY u.id, u.username;
```

Ventes par catégorie

Cette vue montre le nombre total de ventes et le montant total généré pour chaque catégorie.

```sql
CREATE VIEW sales_by_category AS
SELECT c.id, c.name, COUNT(o.id) AS total_sales, SUM(o.price) AS total_revenue
FROM category c
JOIN product_category pc ON c.id = pc.category_id
JOIN product p ON pc.product_id = p.id
JOIN fullfils f ON p.id = f.product_id
JOIN orders o ON f.orders_id = o.id
GROUP BY c.id, c.name;
```

Produits les plus vendus

Cette vue affiche les produits les plus vendus, en termes de quantité et de chiffre d'affaires.

```sql
CREATE VIEW top_selling_products AS
SELECT p.id, p.name, p.description, p.price, COUNT(o.id) AS total_sales, SUM(o.price)
FROM product p
JOIN fullfils f ON p.id = f.product_id
JOIN orders o ON f.orders_id = o.id
GROUP BY p.id, p.name, p.description, p.price
ORDER BY total_sales DESC, total_revenue DESC;
```

Avis récents

Cette vue affiche les avis récents avec les informations du produit et de l'utilisateur.

```sql
CREATE VIEW recent_reviews AS
SELECT r.id, r.rating, r.comment, r.date_created, p.name AS product_name, u.username
FROM review r
JOIN product p ON r.product_id = p.id
JOIN users u ON r.user_id = u.id
ORDER BY r.date_created DESC;
```

Index

Recherche de produits

Pour accélérer la recherche de produits par leur nom, un index peut être créé sur la colonne name de la table product.

```sql
CREATE INDEX idx_product_name ON product(name);
```

Trier les avis par date

Pour accélérer l'affichage des avis triés par date, un index peut être créé sur la colonne date_created de la table review.

```
CREATE INDEX idx_review_date_created ON review(date_created);
```

Recherche de commandes par utilisateur

Pour accélérer la recherche des commandes par utilisateur, un index peut être créé sur la colonne users_id de la table orders.

```
CREATE INDEX idx_orders_users_id ON orders(users_id);
```