Explication des Requêtes - Création de la Base de Données et des Tables pour la Logistique de Transport

Ce document fournit des explications détaillées sur les requêtes utilisées pour créer une base de données de logistique de transport. Il comprend la création des tables "entrepots" et "expeditions" ainsi que leurs relations.

Création de la Base de Données

Création de la base de données "transport_logistique"

sql

```
CREATE DATABASE IF NOT EXISTS transport_logistique;
```

Cette requête crée une base de données nommée "transport_logistique" si elle n'existe pas déjà. C'est dans cette base de données que seront stockées les tables liées à la logistique de transport.

Création de la Table "entrepots"

Description de la table "entrepots"

La table "entrepots" est conçue pour enregistrer les informations relatives aux entrepôts.

sql

```
CREATE TABLE entrepots (

    id INT AUTO_INCREMENT PRIMARY KEY,

    nom_entrepot VARCHAR(255),

    adresse VARCHAR(255),

    ville VARCHAR(255),

    country_code CHAR(2),
```

```
    FOREIGN KEY (country_code) REFERENCES countries(code)
);
```

- `id`: Identifiant unique de l'entrepôt, auto-incrémenté.
- `nom_entrepot`: Nom de l'entrepôt.
- `adresse`: Adresse de l'entrepôt.
- `ville`: Ville où se situe l'entrepôt.
- `country_code`: Code du pays de l'entrepôt faisant référence à la table "countries".

Création de la Table "expeditions"

Description de la table "expeditions"

La table "expeditions" enregistre les détails des expéditions.

sql

```
CREATE TABLE expeditions (
    id INT AUTO_INCREMENT PRIMARY KEY,
    date_expedition DATE,
    date_reception DATE,
    id_entrepot_source INT,
    id_entrepot_destination INT,
    poids DECIMAL(10, 2),
    statut VARCHAR(255),
    id_country_source CHAR(2),
    id_country_destination CHAR(2),
    FOREIGN KEY (id_entrepot_source) REFERENCES entrepots(id),
    FOREIGN KEY (id_entrepot_destination) REFERENCES entrepots(id),
    FOREIGN KEY (id_country_source) REFERENCES countries(code),
    FOREIGN KEY (id_country_destination) REFERENCES countries(code)
```

```
);
```

- `id`: Identifiant unique de l'expédition, auto-incrémenté.
- `date_expedition`: Date de l'expédition.
- `date_reception`: Date de réception de l'expédition.
- `id_entrepot_source` et `id_entrepot_destination`: Références aux entrepôts source et destination.
- `poids`: Poids de l'expédition.
- `statut`: Statut actuel de l'expédition.
- `id_country_source` et `id_country_destination`: Références aux pays source et destination.

Ces requêtes permettent de créer une base de données pour gérer les entrepôts, les expéditions, et leurs relations. La relation entre les tables est établie via les clés étrangères.

Explication des Requêtes - Insertion et Mise à Jour de Données

Ce document fournit des explications sur les requêtes utilisées pour insérer de nouvelles données dans les tables "entrepots" et "expeditions", ainsi que pour mettre à jour une donnée dans la table "expeditions".

Ajout de données dans la table "entrepots"

Ajout de données dans la table "entrepots"

sql

```
INSERT INTO entrepots (nom_entrepot, adresse, ville, country_code)
VALUES
    ('Entrepot A', 'Adresse A', 'Ville A', 'FR'),
    ('Entrepot B', 'Adresse B', 'Ville B', 'DE'),
    ('Entrepot C', 'Adresse C', 'Ville C', 'IT'),
    ('Entrepot D', 'Adresse D', 'Ville D', 'ES'),
    ('Entrepot E', 'Adresse E', 'Ville E', 'TR'),
    ('Entrepot F', 'Adresse F', 'Ville F', 'PT'),
    ('Entrepot G', 'Adresse G', 'Ville G', 'GR'),
    ('Entrepot H', 'Adresse H', 'Ville H', 'NL'),
    ('Entrepot I', 'Adresse I', 'Ville I', 'BE'),
    ('Entrepot J', 'Adresse J', 'Ville J', 'SE'),
    ('Entrepot K', 'Adresse K', 'Ville K', 'FR'),
    ('Entrepot L', 'Adresse L', 'Ville L', 'GR');
```

Ces requêtes insèrent de nouvelles données dans la table "entrepots", avec les détails de différents entrepôts dans différentes villes associés à des pays via le code de pays.

Ajout de données dans la table "expeditions"

Ajout de données dans la table "expeditions"

sql

```
INSERT INTO expeditions (date_expedition, date_reception,
id_entrepot_source, id_entrepot_destination, poids, statut,
id_country_source, id_country_destination) VALUES

    -- Liste des expéditions insérées ici

    ...
```

Ces requêtes insèrent de nouvelles données dans la table "expeditions", indiquant les détails de diverses expéditions entre entrepôts situés dans différents pays, avec des dates, des poids, des statuts et des informations sur les pays sources et de destination.

Mise à jour de données dans la table "expeditions"

Mise à jour du champ "date_reception" dans la table "expeditions"

sql

```
UPDATE expeditions SET date_reception = '2023-01-16' WHERE id = "1";
```

Cette requête met à jour le champ "date_reception" pour l'expédition avec l'identifiant "1", indiquant la date à laquelle cette expédition a été reçue.

Mise à jour du champ "statut" dans la table "expeditions"

sql

```
UPDATE expeditions SET statut = 'En transit' WHERE id = "2";
```

Cette requête met à jour le champ "statut" pour l'expédition avec l'identifiant "2", changeant son statut en "En transit".

Requêtes SQL

Afficher tous les entrepôts

```
SELECT * FROM entrepots;
```

Cette requête récupère toutes les données présentes dans la table "entrepots" de la base de données. Elle affiche l'ensemble des informations relatives à chaque entrepôt enregistré.

Afficher toutes les expéditions

```
SELECT * FROM expeditions;
```

Cette requête extrait toutes les informations de la table "expeditions" de la base de données. Elle affiche l'intégralité des données enregistrées pour chaque expédition.

Afficher toutes les expéditions en transit (statut "En cours")

```
SELECT * FROM expeditions WHERE statut = 'En cours';
```

Cette requête filtre les expéditions en fonction de leur statut. Seules les expéditions en cours, identifiées par le statut "En cours", seront affichées.

Afficher toutes les expéditions livrées (statut "Livré")

```
SELECT * FROM expeditions WHERE statut = 'Livré';
```

Cette requête permet de sélectionner et d'afficher les expéditions ayant atteint le statut "Livré".

Afficher tous les pays

```
SELECT * FROM countries;
```

Cette requête extrait toutes les données de la table "countries" de la base de données. Elle présente la liste complète de tous les pays enregistrés.

Requête 1 : Afficher les expéditions en transit depuis l'Europe vers l'Asie

sql

```sql
SELECT *

FROM expeditions AS e

JOIN entrepots AS source ON e.id_entrepot_source = source.id

JOIN entrepots AS dest ON e.id_entrepot_destination = dest.id

JOIN countries AS source_country ON e.id_country_source =
source_country.code

JOIN countries AS dest_country ON e.id_country_destination =
dest_country.code

WHERE source_country.continentcode = 'EU'

  AND dest_country.continentcode = 'AS'

  AND e.statut = 'En transit';
```

Cette requête utilise les tables "expeditions", "entrepots", et "countries" pour afficher les détails des expéditions en transit. Elle utilise des jointures pour lier les entrepôts sources et de destination à leurs pays respectifs et filtre les expéditions initiées depuis un entrepôt en Europe vers un entrepôt en Asie.

Requête 2 : Afficher les entrepôts ayant envoyé des expéditions à destination dans le même pays

sql

```sql
SELECT DISTINCT source.*

FROM expeditions AS e

JOIN entrepots AS source ON e.id_entrepot_source = source.id

JOIN entrepots AS dest ON e.id_entrepot_destination = dest.id

WHERE source.country_code = dest.country_code;
```

Cette requête sélectionne les entrepôts sources qui ont expédié des produits vers des entrepôts de destination situés dans le même pays. Elle utilise des jointures entre les tables

"expeditions" et "entrepots" pour comparer les codes pays des entrepôts sources et de destination.

Requête 3 : Afficher les entrepôts ayant envoyé des expéditions à destination dans un pays différent

sql

```
SELECT DISTINCT source.*

FROM expeditions AS e

JOIN entrepots AS source ON e.id_entrepot_source = source.id

JOIN entrepots AS dest ON e.id_entrepot_destination = dest.id

WHERE source.country_code != dest.country_code;
```

Cette requête sélectionne les entrepôts sources qui ont envoyé des produits vers des entrepôts de destination situés dans un pays différent. Elle utilise des jointures pour comparer les codes pays des entrepôts source et de destination.

Requête 4 : Afficher les expéditions en transit depuis un pays commençant par "F" et pesant plus de 500 kg

sql

```
SELECT *

FROM expeditions AS e

JOIN entrepots AS source ON e.id_entrepot_source = source.id

JOIN countries AS source_country ON e.id_country_source =
source_country.code

WHERE source_country.name LIKE 'F%'

  AND e.poids > 500

  AND e.statut = 'En transit';
```

Cette requête sélectionne les expéditions en transit initiées par des entrepôts situés dans un pays dont le nom commence par la lettre "F" et dont le poids est supérieur à 500 kg.

Requête 5 : Nombre total d'expéditions pour chaque combinaison de pays d'origine et de destination

sql

```
SELECT id_country_source, id_country_destination, COUNT(*) AS
nombre_expeditions

FROM expeditions

GROUP BY id_country_source, id_country_destination;
```

Cette requête effectue un regroupement par combinaison de pays d'origine et de destination, puis compte le nombre total d'expéditions pour chaque combinaison.

Requête 6 : Afficher les entrepôts ayant expédié au cours des 30 derniers jours avec un poids total supérieur à 1000 kg

sql

```
SELECT e.id, e.nom_entrepot, SUM(exp.poids) AS poids_total

FROM entrepots e

INNER JOIN expeditions exp ON e.id = exp.id_entrepot_source

WHERE exp.date_expedition >= CURDATE() - INTERVAL 30 DAY

GROUP BY e.id

HAVING SUM(exp.poids) > 1000;
```

Cette requête affiche les entrepôts ayant expédié au cours des 30 derniers jours, tout en ayant un poids total des expéditions supérieur à 1000 kg. Elle utilise une jointure et une agrégation pour calculer le poids total des expéditions par entrepôt.

Requête 7 : Afficher les expéditions livrées avec un retard de plus de 2 jours ouvrables

sql

```
SELECT *

FROM expeditions
```

```
WHERE statut = 'Livré' AND DATEDIFF(date_expedition, CURDATE()) > 2;
```

Cette requête sélectionne les expéditions livrées avec un retard de plus de 2 jours ouvrables en comparant la date d'expédition à la date actuelle.

Requête 8 : Nombre total d'expéditions

sql

```
SELECT COUNT(*) AS nombre_total_expeditions

FROM expeditions;
```

Cette requête simple compte le nombre total d'expéditions dans la table "expeditions".

Ces requêtes utilisent des jointures, des filtres, des fonctions d'agrégation (COUNT, SUM), et des conditions pour effectuer des analyses spécifiques sur les données des expéditions et des entrepôts.

BONUS

1. Création de la table "clients"

sql
```sql
CREATE TABLE clients (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nom VARCHAR(255),
    adresse VARCHAR(255),
    ville VARCHAR(255),
    pays VARCHAR(255)
);
```

Cette instruction crée une nouvelle table appelée "clients" avec des colonnes pour l'identifiant, le nom, l'adresse, la ville et le pays du client.

2. Création de la table de jointure "expeditions_clients"

sql
```sql
CREATE TABLE expeditions_clients (
    id_expedition INT,
    id_client INT,
    FOREIGN KEY (id_expedition) REFERENCES expeditions(id),
    FOREIGN KEY (id_client) REFERENCES clients(id)
);
```

Elle crée une table de jointure pour lier les expéditions et les clients, en utilisant des clés étrangères pour référencer les ID des expéditions et des clients dans d'autres tables.

3. Modification de la table "expeditions" pour ajouter une colonne "id_client"

sql
```sql
ALTER TABLE expeditions
ADD COLUMN id_client INT,
ADD FOREIGN KEY (id_client) REFERENCES clients(id);
```

Cela modifie la table "expeditions" en ajoutant une nouvelle colonne "id_client" et établit une clé étrangère pour référencer les clients.

4. Ajout de données dans la table "clients"

sql
```sql
INSERT INTO clients (nom, adresse, ville, pays) VALUES
    (...); -- (les données des clients sont insérées ici)
```

Elle insère des données (nom, adresse, ville, pays) dans la table "clients".

5. Ajout de données dans la table "expeditions_clients"

sql

```
INSERT INTO expeditions_clients (id_expedition, id_client) VALUES
    (...); -- (les associations entre les expéditions et les clients
sont insérées ici)
```

Elle insère des associations entre les expéditions et les clients dans la table de jointure "expeditions_clients".

6. Requête pour afficher le nombre d'expéditions envoyées et reçues par chaque client

Cette requête effectue des jointures et des agrégations pour compter le nombre d'expéditions envoyées et reçues par chaque client, en utilisant des sous-requêtes pour distinguer les expéditions envoyées et reçues.

7. Requête pour afficher les détails de chaque expédition avec les noms des clients impliqués

Cette requête utilise des jointures pour relier les expéditions, les clients expéditeurs et destinataires, et affiche les détails de chaque expédition (ID, poids, noms des clients expéditeur et destinataire, statut de l'expédition).

Ces requêtes utilisent des jointures, des agrégations et des sous-requêtes pour effectuer des opérations complexes impliquant plusieurs tables. Elles permettent de relier les expéditions, les clients et les associations entre les deux pour obtenir des informations détaillées sur les expéditions et les clients impliqués.