section 2

Rapport du projet III :

Gestion de Stock d’un Supermarché

Sommaire

Introduction

a. Contexte

Dans un environnement commercial en constante évolution, la gestion efficace des stocks est cruciale pour assurer la disponibilité des produits, minimiser les coûts et maximiser la satisfaction des clients. Ce projet vise à développer un système de gestion des stocks pour un supermarché, permettant de suivre les produits, les commandes vers les fournisseurs, les fournisseurs eux-mêmes, ainsi que les niveaux de stock.

Le Système proposé répondra à plusieurs besoins critiques du supermarché:

- **Suivi des produits**: Enregistrer et gérer les informations sur chaque produit, y compris les description, les prix, et les quantités disponibles.
- **Gestion des commandes vers les fournisseurs:** Suivre les commandes passées aux fournisseurs, gérer les délais de livraison, et s'assurer que les stocks sont réapprovisionnés en temps voulu.
- **Gestion des fournisseurs:** Maintenir une base de données des fournisseurs, incluant leurs informations de contact, les conditions commerciales, et l'historique des transactions.
- **Suivi des stocks:** Surveiller les niveaux de stock en temps réel, générer des alertes pour les réapprovisionnements, et produire des rapports détaillés pour la gestion et la prise de décision.

b. Objectifs

L’objectif principal de ce projet est de créer un système automatisé qui améliorera l’efficacité opérationnelle, réduira les erreurs humaines et permettra une gestion proactive des stocks. En outre, le système offrira des fonctionnalités de reporting avancées pour aider la direction à prendre des décisions informées et stratégiques.

Le présent rapport décrit en détail le processus de développement du système, depuis l’analyse des besoins jusqu’à la mise en œuvre, en passant par la conception et l'intégration des différentes composantes.

Méthodologie

Pour parvenir à un système qui répond pleinement aux exigences susmentionnées, nous avons choisi de procéder méthodiquement

1. Choix des entités et définition des associations

Suivant les besoins critique du super marché, nous avons réussi a répertorié 04 entités et trois associations lors de notre analyse. Il s'agit de:

Entités

- **produit**: id_produit, nom, description, prix, stock, id_fournisseur, id_categorie
- **Fournisseurs**: id_fournisseur, nom_fournisseur, contact, adresse, telephone
- **Commandes**: id_commande, id_produit, quantite, date_commande
- **Categories**: id_categorie, nom_categorie

Associations

Association	Cardinalité	Explication
Contenir	(1.1, 0.N)	Une commande ne peut contenir qu'un seul produit ; et un produit peut être contenu dans zéro ou plusieur produit.
Appartenir	(1.1, 1.N)	Un produit appartient a une seule catégorie ; et une catégorie peut avoir plusieurs produits.
livrer	(1.1, 1.N)	un produit est livré par un seul fournisseur ; et un fournisseur peut livrer plusieurs produits.

2. création de la base de données et des tables

Suivant le rendu de notre MCD présenté dans la documentation technique, nous avons créé une base de donnée "gestion-stockDB" ; des tables à base du langage SQL et nous leur avons attribué les caractéristiques suivantes.

a. **Commandes**

Caractéristique	Nom du champ	type	NULL?
cle de référence de la table	id_commande	int primary key	non
indique la quantité du produit commandé	quantite	int	non
indique la date auquel la commande a été passée	date_commande	date	non

b. **Produits**

Caractéristique	Nom du champ	type	NULL?
cle de référence de la table	id_produit	int primary key	non
indique la quantité du produit commandé	nom_produit	varchar	non
indique la description du produit	description	varchar	non

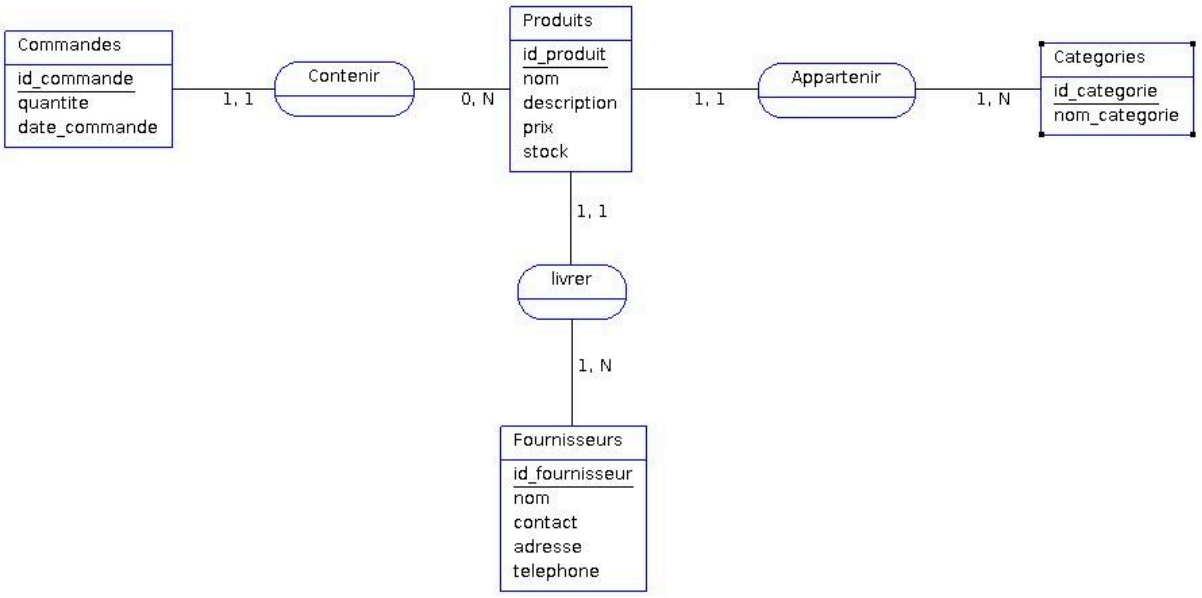
indique le prix du produit	prix	real	non
indique la quantité du produit restant en stock	stock	int	non

c. Fournisseurs

Caractéristique	Nom du champ	type	NULL?
cle de référence de la table	id_fournisseur	int primary key	non
indique le nom du fournisseur	nom_fournisseur	varchar	non
indique l'adresse email du fournisseur	contact	varchar	non
indique l'adresse du fournisseur	adresse	varchar	non
indique le numéro du fournisseur	telephone	varchar	non

d. Categories

Caractéristique	Nom du champ	type	NULL?
cle de référence de la table	id_categorie	int primary key	non
indique le nom du fournisseur	nom_categorie	varchar	non



3. Ajout des données

La création de nos différentes tables étant faite, nous avons pour une opérationnalisation de notre base de données inséré une série de données pour tester son bon fonctionnement.

Chacun des ces donnés remplit pleinement les caractéristiques que nous avons définies dans les tables ci-dessus. Nous avons respectivement dans notre base de données à l'heure actuelle.

- 03 Categories
- 03 Fournisseurs
- 03 Produits

4. Création des Opérations

Notre système en plus de stocker et gérer les produits et les fournisseurs fournit aussi une panoplie de requêtes, fonctions, procédures et autres pour faciliter les tâches de recherche de mise à jour et suppression de certaines données ayant une caractéristique bien précise.

Nous avons prévu des requête pour :

- Lister tous les produits d'une certaine catégorie
- trouver les fournisseurs d'un certain produit.
- lister les commandes en cours pour un produit
- compter le nombre total de commandes par produit

Nous avons des fonctions et procédure tel que :

- Une procédure pour ajouter une nouvelle commande
- Une fonction pour vérifier la disponibilité d’un produit

5. Mise en place des vues et triggers

Les vues sont des objets de la base de données, constitués d’un nom, et d’une requête de sélection. Une fois qu’une vue est définie, on peut l’utiliser comme on le ferait avec une table ; table qui serait constituée par la requête définissant la vue. Pour notre projet nous avons défini une vue pour afficher les détails d’une commande (nom du produit, nom du fournisseur, quantité, date de commande).

Les triggers (ou déclencheurs) sont des objets de la base de données. Attachés à une table, ils vont déclencher l'exécution d’une instruction, ou d’un bloc d'instruction, lorsqu’une, ou plusieurs lignes sont insérées, supprimées ou modifiées dans la table à laquelle ils sont attachés. Pour notre projet nous avons créé un trigger pour mettre à jour le stock d’un produit à l'ajout et à la suppression d'une commande.

Résultats

Dans cette section vous trouverez pour chaque étape de notre projet des captures d'écran montrant l'ensemble de nos actions sous forme de scripts SQL et le rendu lors de l'implémentation dans MySQL.

1. création de la base de données et des tables

```sql
#creation Base de donnée
drop database if exists Gestion_stock;

CREATE DATABASE Gestion_stock CHARACTER SET 'utf8';

USE Gestion_stock;

DROP TABLE IF EXISTS Categories;

CREATE TABLE Categories (
    id_categorie int(50) NOT NULL auto_increment,
    nom_categorie VARCHAR(20) NOT NULL,
    PRIMARY KEY (id_categorie)
) ENGINE = InnoDB;

DROP TABLE IF EXISTS Fournisseurs;

CREATE TABLE Fournisseurs (
    id_fournisseur int(50) NOT NULL auto_increment,
    nom_fournisseur VARCHAR(50) NOT NULL,
    contact VARCHAR(50) NOT NULL,
    adresse VARCHAR(50) NOT NULL,
    telephone VARCHAR(15) NOT NULL,
    PRIMARY KEY (id_fournisseur)
) ENGINE = InnoDB;

DROP TABLE IF EXISTS Produits;

CREATE TABLE Produits (
    id_produit int(50) auto_increment NOT NULL,
    nom_produit VARCHAR(50) NOT NULL,
    description VARCHAR(255) NOT NULL,
    prix REAL NOT NULL,
    stock INT(5) NOT NULL,
    id_categorie int not null,
    id_fournisseur int not null,
    PRIMARY KEY (id_produit),
    foreign key (id_categorie) references Categories(id_categorie),
    foreign key (id_fournisseur) references Fournisseurs(id_fournisseur)
) ENGINE = InnoDB;

DROP TABLE IF EXISTS Commandes;

CREATE TABLE Commandes (
    id_commande int (50) auto_increment NOT NULL,
    quantite INT (5) NOT NULL,
    date_commande DATE NOT NULL,
    id_produit int not null,
    PRIMARY KEY (id_commande),
    foreign key (id_produit) references Produits(id_produit)
) ENGINE = InnoDB;
```

- affichage des tables

```
mysql> SHOW TABLES;
+--------------------------+
| Tables_in_Gestion_stock  |
+--------------------------+
| Categories               |
| Commandes                |
| Fournisseurs             |
| Produits                 |
+--------------------------+
4 rows in set (0,00 sec)

mysql>
```

2. Ajout des données

```sql
#insertion des données
INSERT INTO
    Categories (nom_categorie)
VALUES ('Électronique'),
    ('Livres'),
    ('Vêtements');

INSERT INTO
    Fournisseurs (
        nom_fournisseur,
        contact,
        adresse,
        telephone
    )
VALUES
    ('Fournisseur A', 'contactA@example.com','123 Rue Exemple', '0123456789' ),
    ('Fournisseur B', 'contactB@example.com', '456 Rue Exemple','0123456790'),
    ('Fournisseur C', 'contactC@example.com','789 Rue Exemple','0123456791');

INSERT INTO
    Produits (
        nom_produit,
        description,
        prix,
        stock,
        id_categorie,
        id_fournisseur
    )
VALUES
    ('Ordinateur Portable', 'Ordinateur portable haute performance', 1200.00, 50, 1, 1 ),
    ('Livre de Programmation','Livre pour apprendre à programmer', 35.00, 150, 2,2),
    ('T-shirt', 'T-shirt en coton', 20.00, 200, 3, 3);

insert into Commandes (quantite, date_commande, id_produit)
values
        (20, now(), 1),
        (10, date_add(now(), interval 3 day), 2),
        (30, date_add(now(), interval 5 day), 3);
```

- **Affichage des Tableaux avec les données**

```
mysql> select*from Produits;
+------------+----------------------+-----------------------------------+------+-------+-------------+---------------+
| id_produit | nom_produit          | description                       | prix | stock | id_categorie | id_fournisseur |
+------------+----------------------+-----------------------------------+------+-------+-------------+---------------+
|          1 | Ordinateur Portable  | Ordinateur portable haute performance | 1200 |    50 |           1 |             1 |
|          2 | Livre de Programmation | Livre pour apprendre à programmer |   35 |   150 |           2 |             2 |
|          3 | T-shirt              | T-shirt en coton                  |   20 |   200 |           3 |             3 |
+------------+----------------------+-----------------------------------+------+-------+-------------+---------------+
3 rows in set (0,00 sec)

mysql>
```

```
mysql> select*from Fournisseurs;
+----------------+-----------------+----------------------+-----------------+------------+
| id_fournisseur | nom_fournisseur | contact              | adresse         | telephone  |
+----------------+-----------------+----------------------+-----------------+------------+
|              1 | Fournisseur A   | contactA@example.com | 123 Rue Exemple | 0123456789 |
|              2 | Fournisseur B   | contactB@example.com | 456 Rue Exemple | 0123456790 |
|              3 | Fournisseur C   | contactC@example.com | 789 Rue Exemple | 0123456791 |
+----------------+-----------------+----------------------+-----------------+------------+
3 rows in set (0,00 sec)

mysql>
```

```
mysql> select*from Commandes;
+--------------+----------+---------------+------------+------------------+
| id_commande | quantite | date_commande | id_produit | statut_commande |
+--------------+----------+---------------+------------+------------------+
|            1 |       20 | 2024-07-28    |          1 |                1 |
|            2 |       10 | 2024-07-31    |          2 |                1 |
|            3 |       30 | 2024-08-02    |          3 |                0 |
+--------------+----------+---------------+------------+------------------+
3 rows in set (0,00 sec)

mysql>
```

```
mysql> select*from Categories;
+---------------+----------------+
| id_categorie | nom_categorie |
+---------------+----------------+
|             1 | Électronique   |
|             2 | Livres         |
|             3 | Vêtements      |
+---------------+----------------+
3 rows in set (0,00 sec)

mysql>
```

3. création des requêtes

```sql
#Requête pour lister les produits par catégorie
Select Produits.nom_produit, Categories.nom_categorie
from Produits
join Categories
        on Produits.id_categorie = Categories.id_categorie
where Categories.nom_categorie = 'Électronique' ;


#Requête pour trouver les fournisseurs dun certain produit
Select Produits.nom_produit, Fournisseurs.nom_fournisseur
from Produits
join Fournisseurs
        on Produits.id_fournisseur = Fournisseurs.id_fournisseur
where Produits.nom_produit = 'Ordinateur Portable' ;


#Requête pour lister les commandes en cours pour un produit
Select Produits.id_produit as id ,Produits.nom_produit, Commandes.statut_commande as Encours
from Produits
join Commandes
        on Commandes.id_produit = Produits.id_produit
where Produits.nom_produit = 'Ordinateur Portable' ;


#Requête pour compter le nombre total de commandes par produit
Select Produits.id_produit as id ,Produits.nom_produit, COUNT(Commandes.id_commande) as nbre_commande
from Produits
join Commandes
        on Commandes.id_produit = Produits.id_produit

Group by (Commandes.id_commande);
```

- **affichage des résultats des requêtes**

```
+------------------------------------------------------------------------------+
| ------Requête pour compter le nombre total de commandes par produit------  |
+------------------------------------------------------------------------------+
| ------Requête pour compter le nombre total de commandes par produit------  |
+------------------------------------------------------------------------------+
1 row in set (0,00 sec)

+----+------------------------+---------------+
| id | nom_produit            | nbre_commande |
+----+------------------------+---------------+
|  1 | Ordinateur Portable    |             1 |
|  2 | Livre de Programmation |             1 |
|  3 | T-shirt                |             1 |
+----+------------------------+---------------+
3 rows in set (0,00 sec)
```

```
+------------------------------------------------------------------------------+
| -------Requête pour lister les commandes en cours pour un produit--------  |
+------------------------------------------------------------------------------+
| -------Requête pour lister les commandes en cours pour un produit--------  |
+------------------------------------------------------------------------------+
1 row in set (0,00 sec)

+----+---------------------+---------+
| id | nom_produit         | Encours |
+----+---------------------+---------+
|  1 | Ordinateur Portable |       1 |
+----+---------------------+---------+
1 row in set (0,00 sec)

+------------------------------------------------------------------------------+
```

```
+-------------------------------------------------------------------------------+
| --------Requête pour trouver les fournisseurs dun certain produit---------  |
+-------------------------------------------------------------------------------+
| --------Requête pour trouver les fournisseurs dun certain produit---------  |
+-------------------------------------------------------------------------------+
1 row in set (0,00 sec)

+---------------------+-----------------+
| nom_produit         | nom_fournisseur |
+---------------------+-----------------+
| Ordinateur Portable | Fournisseur A   |
+---------------------+-----------------+
1 row in set (0,00 sec)

+-------------------------------------------------------------------------------+
```

```
mysql> SOURCE /home/chrislandry/Bureau/projet_gestion_stock/Gestion-de-stock/requete.sql;
+-------------------------------------------------------------------+
| -------Requête pour lister les produits par catégorie-----------   |
+-------------------------------------------------------------------+
| -------Requête pour lister les produits par catégorie-----------   |
+-------------------------------------------------------------------+
1 row in set (0,00 sec)

+---------------------+----------------+
| nom_produit         | nom_categorie |
+---------------------+----------------+
| Ordinateur Portable | Électronique  |
+---------------------+----------------+
1 row in set (0,00 sec)
```

4. Définition des fonction et procédures de notre système

```
#procedure pour ajouter une commande

drop procedure if exists newCommande;
delimiter //
create procedure newCommande(in id_prod int, in qté int)
begin
        insert into Commandes value(0, qté, now(), id_prod, false);
end //
delimiter ;

# function pour verifier la disponibilites d'un produit

drop function if exists isProduct
delimiter //
create function isProduct(id_prod int)
returns boolean
deterministic
begin
        declare qte int;
    select stock into qte from Produits where id_produit = id_prod;
    if qte > 0 then
                return true;
        else
                return false;
        end if;
end //
delimiter ;
```

- **résultats de la procédure pour créer une nouvelle commande(4)**

```
mysql> CALL newCommande(1,20);
Query OK, 1 row affected, 1 warning (0,09 sec)

mysql> select*from Commandes;
```

```
mysql> select*from Commandes;
+-------------+----------+---------------+------------+-----------------+
| id_commande | quantite | date_commande | id_produit | statut_commande |
+-------------+----------+---------------+------------+-----------------+
|           1 |       20 | 2024-07-28    |          1 |               1 |
|           2 |       10 | 2024-07-31    |          2 |               1 |
|           3 |       30 | 2024-08-02    |          3 |               0 |
|           4 |       20 | 2024-07-28    |          1 |               0 |
+-------------+----------+---------------+------------+-----------------+
4 rows in set (0,00 sec)
```

- **résultats de la fonction pour vérifier la disponibilité d'un produit (isProduct)**

```
mysql> SELECT isProduct(1);
+---------------+
| isProduct(1) |
+---------------+
|             1 |
+---------------+
1 row in set (0,00 sec)

mysql>
```

5. mise en place des vues et triggers

```sql
drop trigger if exists addStock;
delimiter //
create trigger addStock
after insert on Commandes
for each row
begin
        declare currentStock int;
    select stock into currentStock from Produits where Produits.id_produit = new.id_produit;
    update Produits set stock = new.quantite + currentStock where Produits.id_produit = new.id_produit;
end //
delimiter ;


drop trigger if exists reduceStock;
delimiter //
create trigger reduceStock
after delete on Commandes
for each row
begin
    declare currentStock int;
    select stock into currentStock from Produits where Produits.id_produit = old.id_produit;
    update Produits set stock = currentStock - old.quantite where Produits.id_produit = old.id_produit;
end //
delimiter ;
```

```sql
drop view if exists CommandeDetails;
create view CommandeDetails as
select Produits.nom_produit, Fournisseurs.nom_fournisseur, Commandes.quantite, Commandes.date_commande, Commandes.id_commande
from Produits
join Fournisseurs
        on Produits.id_fournisseur = Fournisseurs.id_fournisseur
join Commandes
        on Commandes.id_produit = Produits.id_produit ;

select*from CommandeDetails
```

- **affichage des résultats de la vue**

```
+-----------------------------------------------------+
| ---View pour afficher les details dune commande---- |
+-----------------------------------------------------+
| ---View pour afficher les details dune commande---- |
+-----------------------------------------------------+
1 row in set (0,00 sec)

Query OK, 0 rows affected (0,11 sec)

+-------------+------------------------+-----------------+----------+---------------+
| id_commande | nom_produit            | nom_fournisseur | quantite | date_commande |
+-------------+------------------------+-----------------+----------+---------------+
|           1 | Ordinateur Portable    | Fournisseur A   |       20 | 2024-07-27    |
|           2 | Livre de Programmation | Fournisseur B   |       10 | 2024-07-30    |
|           3 | T-shirt                | Fournisseur C   |       30 | 2024-08-01    |
+-------------+------------------------+-----------------+----------+---------------+
3 rows in set (0,00 sec)
```

Conclusion

A travers notre base de données que nous avons créé, nous avons simplifié la collecte des données et amélioré l'efficacité de la gestion des stocks du supermarché. Il n’est plus nécessaire d'utiliser des formulaires papier pour le suivi des produits et des commandes passées aux différents fournisseurs.