

Livrable Test Unitaire



Groupe :

Orianne Boussard

Antoine Favereau

Guillaume Dubois de Lavigerie

Enzo Miragliotta

Le 21 janvier 2022

Table des matières

Tests requêtes SQL	3
Supprimer un client.....	3
Modifier un client	3
Créer un client	4
Afficher un client	4
Créer un Personnel.....	4
Calculer le chiffre d'affaires sur un mois en particulier	5
Identifier les produits sous le seuil de réapprovisionnement.....	5
Calculer le montant total des achats pour un client	6
Identifier les 10 articles les moins vendus	6
Calculer la valeur commerciale du stock.....	7
Test des différentes Interfaces.....	8
Test Unitaire appliqué à la classe CLmappStock	12
Class CLmappStock	12
Test des Méthodes : setNom_couleur et getNom_couleur	13
Autres méthodes :	13
Class CLserviceGestionStock.....	14
Test de la Méthode : ajouterUnArticle.....	14
Test de la Méthode : suppUnArticle.....	15
Test de la Méthode : modifierUnArticle.....	15

Tests requêtes SQL

Pour commencer, nous allons tester les différentes requêtes SQL sur SQL serveur qui vont être utiliser par la suite dans les différentes classes de notre Programme.

Ces différents tests permettront par la suite lors de la programmation d'avoir une certitude sur le fait que le problème ne vienne pas des requêtes SQL.

Supprimer un client

- Requête

```
Update Client SET Supprimer_client = 'True' WHERE ID_client = '1';
```

- Résultat attendu

Pour les ID_client égal à 1 la colonne Supprimer_client doit être à True

- Résultat Obtenue

	ID_client	Nom_client	Prenom_client	Date_naissance	Supprimer_client
1	2	Descarte	Pierre	2006-02-15 00:00:00.000	1
2	1002	dg	georges	2003-06-05 00:00:00.000	0

Test validé

Modifier un client

- Requête

```
UPDATE Client SET Nom_client = 'dupont', Prenom_client = 'Jean', Date_naissance = '08-02-2005' WHERE ID_client = '2';
```

- Résultat attendu

Modifier les informations du client numéro 2

- Résultat Obtenue

	ID_client	Nom_client	Prenom_client	Date_naissance	Supprimer_client
1	2	dupont	Jean	2005-02-08 00:00:00.000	1
2	1002	dg	georges	2003-06-05 00:00:00.000	0

Test validé

Créer un client

- Requête

```
INSERT INTO Client(Nom_client, Prenom_client, Date_naissance, Supprimer_client)
VALUES('dg', ' georges', '05-06-2003', 'False');
```

- Résultat attendu

Création d'un client avec les différentes informations écrites dans la requête.

- Résultat Obtenue

	ID_client	Nom_client	Prenom_client	Date_naissance	Supprimer_client
1	2	dupont	Jean	2005-02-08 00:00:00.000	1
2	1002	dg	georges	2003-06-05 00:00:00.000	0
3	1003	grbhty	georges	2003-06-05 00:00:00.000	0

Test validé

Afficher un client

- Requête

```
SELECT Nom_client, Prenom_client, Date_naissance, numero_rue, extension_rue, nom_rue, nom_ville,
code_postal FROM Client, Adresse, Villes WHERE Client.ID_client = Adresse.ID_client AND
Adresse.ID_ville = Villes.ID_ville ;
```

- Résultat attendu

Affichage des différentes informations liés à l'ID d'un client

- Résultat Obtenue

Nom_client	Prenom_client	Date_naissance	numero_rue	extension_numero	nom_rue	nom_ville	code_postal
------------	---------------	----------------	------------	------------------	---------	-----------	-------------

Test validé

Créer un Personnel

- Requête

```
INSERT INTO
Personnel(Nom_personnel, Prenom_personnel, Date_embauche, numero_rue, nom_rue, extension_nu
mero, Supprimer_personnel, ID_personnel_Superviser) VALUES('" + gilette', 'timeo', '" +
06-04-2005', '2', 'fontaine', ' bis ', 'False', (SELECT ID_personnel FROM Personnel WHERE
Nom_personnel = 'jack' AND Prenom_personnel = 'lucas' AND Supprimer_personnel =
'False'));
```

- Résultat attendu

Création d'un personnel sans superviseur

- Résultat Obtenue

	ID_personnel	Nom_personnel	Prenom_personnel	Date_embauche	adresse_personnel	ID_personnel_Superviser
1	1	Marmoude	Mohamed	1894-06-14 00:00:00.000	juvien	NULL

Test validé

Calculer le chiffre d'affaires sur un mois en particulier

- Requête

```
SELECT SUM(montant_paiement) as 'chiffre affaire sur un mois' FROM (select
montant_paiement, MONTH(date_solde_paiement) as 'date' from Paiement) as tab where
tab.date = 6;
```

- Résultat attendu

Le chiffre d'affaires sur le mois de JUIN

- Résultat Obtenue

	chiffre affaire sur un mois
1	500

Test validé

Identifier les produits sous le seuil de réapprovisionnement

- Requête

```
SELECT nom_article FROM Articles WHERE seuil_reapprovisionnement>quantite_stock;
```

- Résultat attendu

Affichage des articles sous le seuil de réapprovisionnement

- Résultat Obtenue

	nom_article
1	EnzoLarticle

Test validé

Calculer le montant total des achats pour un client

- Requête

```
Select SUM(prix_article_HT * quantite_article * taux_TVA) AS Montant_Total_Achat,  
(Select Reference_commande FROM Commande WHERE Reference_commande = 'MLOP12') AS  
Reference_commande,(Select Nom_client FROM Client Where ID_client = (Select ID_client  
FROM Commande WHERE Reference_commande = 'MLOP12')) as Nom_Client FROM Articles INNER  
JOIN Contenir ON Articles.Designation = Contenir.Designation WHERE Reference_commande  
= (Select Reference_commande FROM Commande WHERE Reference_commande = 'MLOP12' AND  
Supprimer_commande = 'True') GROUP BY Reference_commande;
```

- Résultat attendu

Montant total des achats de la référence commande MLOP12

- Résultat Obtenue

	Montant_Total_Achat	Reference_commande	Nom_Client
1	540	MLOP12	Julien

Test validé

Identifier les 10 articles les moins vendus

- Requête

```
Select Top 10 Sum(quantite_article) as Nombre_Vendu,Designation FROM Contenir Group By  
Designation Order by Nombre_Vendu ASC;
```

- Résultat attendu

Les 10 articles les moins vendus dans l'ordre croissant ici 8 produit devront être afficher car dans la BDD il n'y a que 8 articles

- Résultat Obtenue

	Nombre_Vendu	Designation
1	1	MLA98
2	3	LKQA784
3	3	K87AZA
4	10	J154A
5	14	2PC11AZ
6	28	MK10P
7	30	15K2M
8	50	MPKD1

Test validé

Calculer la valeur commerciale du stock

- Requête

```
SELECT SUM(prix_article_HT*quantite_stock*taux_TVA)as 'valeur commerciale stock' FROM Articles;
```

- Résultat attendu

La somme de tous les articles en stock

- Résultat Obtenue

	valeur commerciale stock
1	25005

Test validé

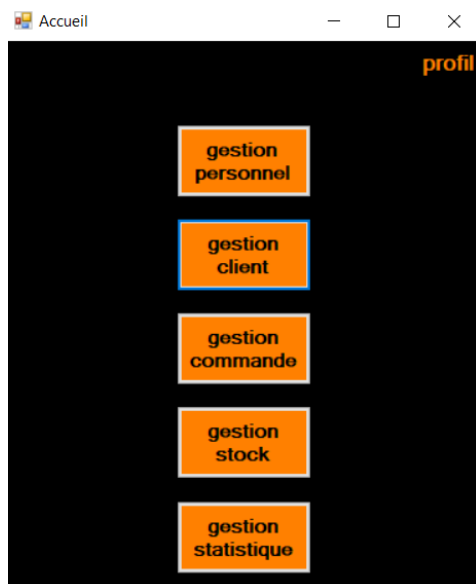
Test des différentes Interfaces

Page de connexion :



The screenshot shows a window titled 'Accueil' with a black background. In the center, there are two orange input fields. The first is labeled 'Nom' and the second is labeled 'Prenom'. Below these fields is an orange button with the text 'Valider' in black. The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

Page d'accueil :



The screenshot shows a window titled 'Accueil' with a black background. In the top right corner, there is a link labeled 'profil'. In the center, there is a vertical list of five orange buttons with black text. The buttons are labeled 'gestion personnel', 'gestion client', 'gestion commande', 'gestion stock', and 'gestion statistique'. The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

Gestion personnel :

Gestion personnel

ID

Nom

Prenom

Date d'embauche

Prenom supérieur

Nom supérieur

Adresse

n°

ext

rue

CP

Ville

ID_personnel	Nom_personnel	Prenom
2	fvresg	vfdsg
*		

retour ajouter modifier supprimer

Gestion client :

Gestion client

ID

Nom

Prenom

Date d'anniversaire

Adresses livraison

numero_rue	extension_nu
*	

Adresses facturation

numero_rue	extension_nu
*	

ID_client	Nom_client	Prenom
1002	dg	georges
1003	grbity	georges
*		

retour ajouter modifier supprimer

Gestion commande :

Gestion commande

Référence Commande

Nom Client

Prenom Client

Nom Personnel

Prenom Personnel

Date Livraison

Date Emission

Articles

Designation	quantite_article
*	

ListeArticles

designation	nom_article	prix_article_HT
bgdh	e	54
1GFBHF	cable long	10
bgdh	e	54
bgdnhngt	a	12
bnqd	e	20
frezgt	cable	10
fsgr	f	200
gtr	c	50
gtrksgh	b	100
grdath	bhgt	25

retour Ajouter modifier supprimer

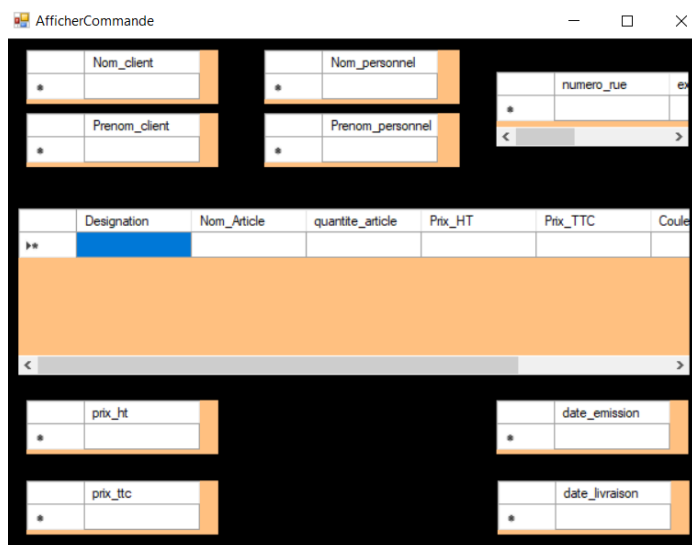
Afficher Commande Valider Commande

Valider commande :



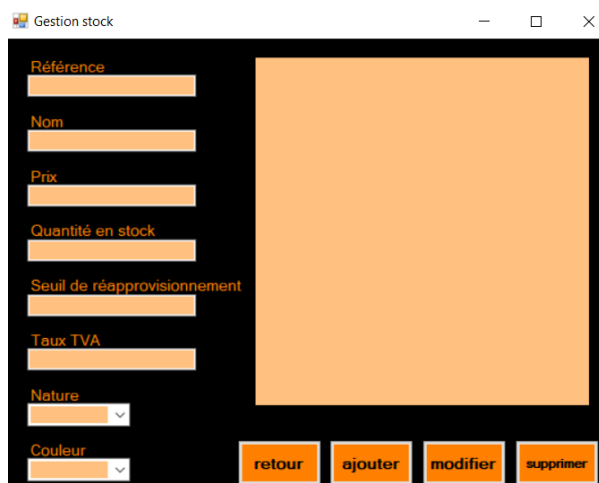
The screenshot shows a window titled "Gest..." with a dark background. It contains four orange input fields stacked vertically, each with a label above it: "Montant du Paiement", "Mode de Paiement", "Date du Paiement", and "Date Solde du Paiement". At the bottom, there is an orange button labeled "Valider".

Afficher commande :



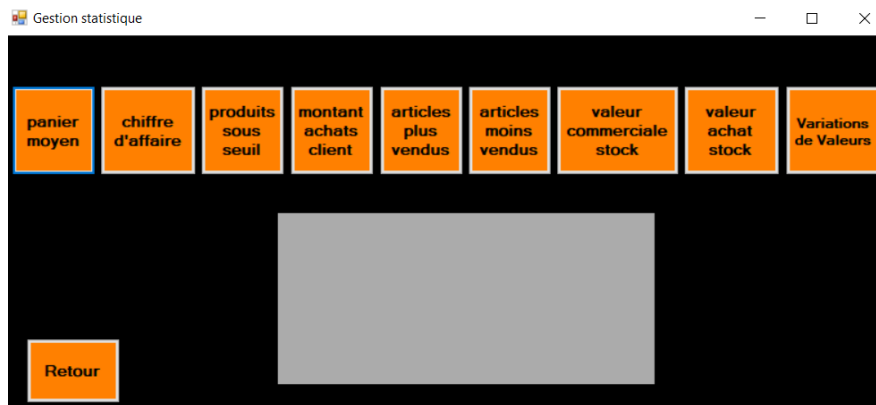
The screenshot shows a window titled "AfficherCommande" with a dark background. It contains several orange input fields and a table. At the top, there are three rows of fields: "Nom_client" and "Nom_personnel" (each with a small asterisk icon), "Prenom_client" and "Prenom_personnel" (each with a small asterisk icon), and "numero_rue" (with a small asterisk icon). Below these is a horizontal scroll bar. In the center, there is a table with the following headers: "Designation", "Nom_Article", "quantite_article", "Prix_HT", "Prix_TTC", and "Couleur". The first row of the table is highlighted in orange. Below the table is another horizontal scroll bar. At the bottom, there are four orange input fields: "prix_ht" and "date_emission" (each with a small asterisk icon), and "prix_ttc" and "date_livraison" (each with a small asterisk icon).

Gestion stock :



The screenshot shows a window titled "Gestion stock" with a dark background. It contains a list of orange input fields on the left side, each with a label above it: "Référence", "Nom", "Prix", "Quantité en stock", "Seuil de réapprovisionnement", "Taux TVA", "Nature" (with a dropdown arrow), and "Couleur" (with a dropdown arrow). To the right of these fields is a large orange rectangular area. At the bottom, there are four orange buttons: "retour", "ajouter", "modifier", and "supprimer".

Gestion statistique :



Test Unitaire appliqué à la classe CLmappStock

Class CLmappStock

Description :

Cette classe permet d'affecter toutes les valeurs des différentes variables en lien avec le stock. (La référence, la nature du produit, la couleur, le nom de l'article, le taux de tva, la quantité en stock, le seuil de réapprovisionnement et le prix de l'article.

Code :

Class CLmappStock.h

```
namespace NS_Comp_Mappage
{
    ref class CLmappStock
    {
    private:
        System::String^ Reference;
        System::String^ Nature_produit;
        System::String^ Nom_couleur;
        System::String^ Nom_article;
        System::String^ TVA;
        System::String^ Quantite_stock;
        System::String^ Seuil_rea;
        System::String^ Prix_article;

    public:
        System::String^ Select(void);
        System::String^ Insert(void);
        System::String^ Delete(void);
        System::String^ Update(void);
        void setNom_couleur(System::String^);
        void setNature_produit(System::String^);
        void setTVA(System::String^);
        void setSeuil_rea(System::String^);
        void setQuantite_stock(System::String^);
        void setReference(System::String^);
        void setNom_article(System::String^);
        void setPrix_article(System::String^);
        System::String^ getNom_couleur(void);
        System::String^ getNature_produit(void);
        System::String^ getTVA(void);
        System::String^ getSeuil_rea(void);
        System::String^ getQuantite_stock(void);
        System::String^ getPrix_article(void);
        System::String^ getReference(void);
        System::String^ getNom_article(void);
    };
}
```

Class CLmappStock.cpp

```
System::String^ NS_Comp_Mappage::CLmappStock::Select(void)
{
    return "SELECT nom_article,prix_article_HT,quantite_stock,seuil_reapprovisionnement,taux_TVA,(SELECT nom_natu";
}
System::String^ NS_Comp_Mappage::CLmappStock::Insert(void)
{
    return "INSERT INTO Articles(Designation, nom_article, prix_article_HT , quantite_stock, seuil_reapprovisi";
}
System::String^ NS_Comp_Mappage::CLmappStock::Update(void){
    return "UPDATE Articles SET nom_article = " + this->Nom_article + ", prix_article_HT = " + this->Prix_artic";
}
System::String^ NS_Comp_Mappage::CLmappStock::Delete(void)
{
    return "UPDATE Articles SET Supprimer_article = 'False' WHERE Designation = " + this->Reference + " ";
}

void NS_Comp_Mappage::CLmappStock::setNom_couleur(System::String^ couleur){ this->Nom_couleur = couleur;}
void NS_Comp_Mappage::CLmappStock::setNature_produit(System::String^ nature){this->Nature_produit = nature;}
void NS_Comp_Mappage::CLmappStock::setTVA(System::String^ tva){this->TVA = tva;}
void NS_Comp_Mappage::CLmappStock::setSeuil_rea(System::String^ seuil){this->Seuil_rea = seuil;}
void NS_Comp_Mappage::CLmappStock::setQuantite_stock(System::String^ quantite){this->Quantite_stock = quantite;}
void NS_Comp_Mappage::CLmappStock::setPrix_article(System::String^ prix){this->Prix_article = prix;}
void NS_Comp_Mappage::CLmappStock::setReference(System::String^ reference){this->Reference = reference;}
void NS_Comp_Mappage::CLmappStock::setNom_article(System::String^ nom){this->Nom_article = nom;}

System::String^ NS_Comp_Mappage::CLmappStock::getNom_article(void) { return this->Nom_article; }
System::String^ NS_Comp_Mappage::CLmappStock::getReference(void) { return this->Reference; }
System::String^ NS_Comp_Mappage::CLmappStock::getPrix_article(void) { return this->Prix_article; }
System::String^ NS_Comp_Mappage::CLmappStock::getQuantite_stock(void) { return this->Quantite_stock; }
System::String^ NS_Comp_Mappage::CLmappStock::getSeuil_rea(void) { return this->Seuil_rea; }
System::String^ NS_Comp_Mappage::CLmappStock::getTVA(void) { return this->TVA; }
System::String^ NS_Comp_Mappage::CLmappStock::getNature_produit(void) { return this->Nature_produit; }
System::String^ NS_Comp_Mappage::CLmappStock::getNom_couleur(void) { return this->Nom_couleur; }
```

Test des Méthodes : **setNom_couleur** et **getNom_couleur**

```
void NS_Comp_Mappage::CLmappStock::setNom_couleur(System::String^ couleur){ this->Nom_couleur = couleur;}
```

La méthode **setRéférence** doit nous permettre d'enregistrer une valeur dans une variables.

```
System::String^ NS_Comp_Mappage::CLmappStock::getNom_article(void) { return this->Nom_article; }
```

La méthode **getRéférence** doit nous permettre de retourner la valeur d'une variable.

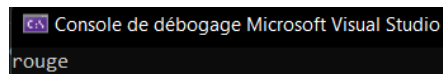
Résultat attendu :

Pour ce test, grâce à notre main ci-dessous nous allons donner une valeur de variable via la méthode **setnom_couleur** et la retourner via la méthode **getNom_couleur**. Nous allons donc voir si la variable donner et bien celle retourner grâce à un print sur la console. **Résultat attendu : « rouge »**

Main :

```
int main(array<System::String ^> ^args)
{
    NS_Comp_Mappage::CLmappStock testCouleur;
    testCouleur.setNom_couleur("rouge");
    System::String^ resultat = testCouleur.getNom_couleur();
    System::Console::WriteLine(resultat);
    return 0;
}
```

Résultat obtenu :



Console de débogage Microsoft Visual Studio
rouge

Test validé

Autres méthodes :

De même pour les autres méthodes **setReference**, **getReference**, **setNature_produit**, **getNature_produit**, **setNom_article**, **getNom_article**, **setTVA**, **getTVA**, **setQuantite_stock**, **getQuantite_stock**, **setSeuil_rea**, **getSeuil_rea** et **setPrix_article**, **getPrix_article**. Toutes ces méthodes sont identiques aux méthodes vues ci-dessus donc elles aussi fonctionnent.

*Pour les méthodes **Select**, **Insert**, **Delete**, **Update** nous les testerons dans la Classe **CLserviceGestionStock**.*

Class CLserviceGestionStock

Description :

Cette classe est une classe qui offre comme service d'ajouter un article, supprimer un article, modifier un article et de sélectionner tous les articles en utilisant les différentes méthodes de la classe de mappage : CLmappStock.

Code :

Class CLserviceGestionStock.h

```
namespace NS_Comp_Svc
{
    ref class CLserviceGestionStock
    {
    private:
        NS_Comp_Mappage::CLmappStock^ oMappStock;
    public:
        CLserviceGestionStock(void);
        System::Data::DataSet^ selectionnerTousLesArticles(System::String^);
        void ajouterUnArticle(System::String^, System::String^, System::String^, System::String^,
            System::String^, System::String^, System::String^, System::String^);
        void suppUnArticle(System::String^);
        void modifierUnArticle(System::String^, System::String^, System::String^, System::String^,
            System::String^, System::String^, System::String^, System::String^);
    };
}
```

Test de la Méthode : **ajouterUnArticle**

Méthode :

```
System::String^ NS_Comp_Svc::CLserviceGestionStock::ajouterUnArticle(System::String^ reference, System::String^ nom,
{
    System::String^ sql;

    this->oMappStock->setReference(reference);
    this->oMappStock->setNature_produit(nature);
    this->oMappStock->setNom_couleur(couleur);
    this->oMappStock->setNom_article(nom);
    this->oMappStock->setTVA(tva);
    this->oMappStock->setQuantite_stock(quantite);
    this->oMappStock->setSeuil_rea(seuil);
    this->oMappStock->setPrix_article(prix);
    sql = this->oMappStock->Insert();
    return sql;
}
```

Main :

```
int main(array<System::String ^> ^args)
{
    NS_Comp_Svc::CLserviceGestionStock test;
    System::String^ resultat = test.ajouterUnArticle("A3", "chaise", "rouge", "chaisou", "10", "100", "20", "50");
    System::Console::WriteLine(resultat);
    return 0;
}
```

Résultat attendu :

L'affichage sur la console de la requête permettant d'ajouter un nouvelle article qui a comme référence A3.

Résultat obtenue :

```
Microsoft Visual Studio
INSERT INTO Articles(Designation, nom_article, prix_article_HT, quantite_stock, seuil_reapprovisionnement, taux_TVA, Sup
primer_article, ID_nature_article, ID_couleur) VALUES('A3','chaise',rouge,chaisou,'10',100,'True', (SELECT ID_nature_ar
ticle FROM Nature_Article WHERE nom_nature_article = '20'), (SELECT ID_couleur FROM Couleurs WHERE nom_couleur = '50'));
```

Test validé

Test de la Méthode : **suppUnArticle**

Méthode :

```
System::String^ NS_Comp_Svc::CLserviceGestionStock::suppUnArticle(System::String^ reference)
{
    System::String^ sql;

    this->oMappStock->setReference(reference);
    sql = this->oMappStock->Delete();
    return sql;
}
```

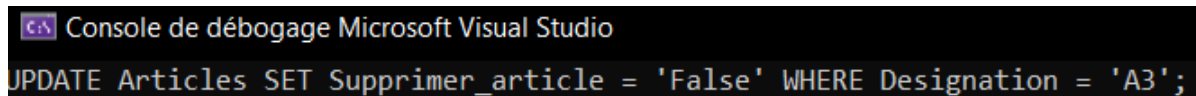
Main :

```
int main(array<System::String ^> ^args)
{
    NS_Comp_Svc::CLserviceGestionStock test;
    System::String^ resultat = test.suppUnArticle("A3");
    System::Console::WriteLine(resultat);
    return 0;
}
```

Résultat attendu :

L'affichage sur la console de la requête permettant de supprimer l'article qui a comme référence A3.

Résultat obtenue :



Console de débogage Microsoft Visual Studio

```
UPDATE Articles SET Supprimer_article = 'False' WHERE Designation = 'A3';
```

Test validé

Test de la Méthode : **modifierUnArticle**

Méthode :

```
System::String^ NS_Comp_Svc::CLserviceGestionStock::modifierUnArticle(System::String^ reference, System::String^ nom,
{
    System::String^ sql;

    this->oMappStock->setReference(reference);
    this->oMappStock->setNature_produit(nature);
    this->oMappStock->setNom_couleur(couleur);
    this->oMappStock->setNom_article(nom);
    this->oMappStock->setTVA(tva);
    this->oMappStock->setQuantite_stock(quantite);
    this->oMappStock->setSeuil_rea(seuil);
    this->oMappStock->setPrix_article(prix);

    sql = this->oMappStock->Update();
    return sql;
}
```

Main :

```
int main(array<System::String ^> ^args)
{
    NS_Comp_Svc::CLserviceGestionStock test;
    System::String^ resultat = test.modifierUnArticle("A3", "chaise", "noir", "chaisou", "10", "100", "20", "50");
    System::Console::WriteLine(resultat);
    return 0;
}
```

Résultat attendu :

L’affichage sur la console de la requête permettant de supprimer l’article qui a comme référence A3 en modifiant sa couleur en noir.

Résultat obtenue :

```
UPDATE Articles SET nom_article = 'chaise', prix_article_HT = noir, quantite_stock = chaisou, seuil_reapprovisionnement = 10, taux_TVA = 100, ID_nature_article = (SELECT ID_nature_article FROM Nature_Article WHERE nom_nature_article = '20') , ID_couleur = (SELECT ID_couleur FROM Couleurs WHERE nom_couleur = '50') WHERE Designation = 'A3';
```

Test validé