

Table of Contents

Report	2
Inventaire 3Bassins	4
Inventaire 3Bassins au 30-09-2024	8
Engen Mon Caprice	11
Divers problèmes	12
Papangue	14
FraisApprocheReception	15
Vito 11	16
Méthodes mémo techniques	18
Frais d'approche	22
Calcul frais d'approche	32

Report

Il manque le commentaire de fin de contrôle de sous caisse

The screenshot shows a report window titled 'rpContrôleSousCaisseDetail'. The main title is 'Contrôle de sous-caisse'. Below the title, there are several input fields with labels: 'Sous-caisse de : [Utilisateur création.Utilisateur]', 'Début : [Date de début]', 'Utilisateur arrêt : [Utilisateur arrêt.Utilisateur]', 'Fin : [Date de fin]', 'Utilisateur clôture : [Utilisateur clôture.Utilisateur]', 'Clôture : [Date de clôture]', 'Dépôt : [Utilisateur clôture.Dépôt par défaut pour l]', and 'Nombre de validation : [Nombre de validation]'. Below these fields is a section titled 'lbLibelléRésultat'. Under this section, there is a table with the following content:

Encaissements au comptant et des clients comptants	
lbModesRèglements	
Ventes (+)	[Contrôle.Encaissements]

Report-ContôleSousCaisse.png

Se placer le rpxxxDetail

- Add sub band

⚠ dans notre exemple c'est le sbCommentaire

- name the Sub band (sbCommentaire) icon tools (symbole de la clé)

The screenshot shows a sub-band configuration window titled 'sbCommentaire'. It contains a text field with the expression 'Commentaire : [Contrôle].[Commentaire]' and a gear icon for settings.

sbCommentaire.png

- décocher Visible
- Dans Expression Editor
- Ajouter **[Contrôle].[Commentaire]**
-

Dans scripts

- `IbCommentaireBilletageValidé.Text = billetageValidé.Commentaire;`

Inventaire 3Bassins

Résultat envoyé par Ivalis

Exemple de produit non présent dans les fichiers envoyés par Ivalis

Référence	Désignation	Achat	Quantité	Total
0490000025545	CAMEL ACTIVATE CIGARILLOS PURPLE	6.22	-81.00	-503.82

Référence non présente dans les fichiers envoyés par Ivalis

Référence	Désignation	Achat	Quantité	Total
0490000029895	BIERE AFFLIGEM BLONDE	1.06	53.00	56.18

Référence présente dans le fichier des inconnus de Ivalis , mais non présent dans l'inventaire, édité au 30 Septembre 2024. Référence existe dans la base

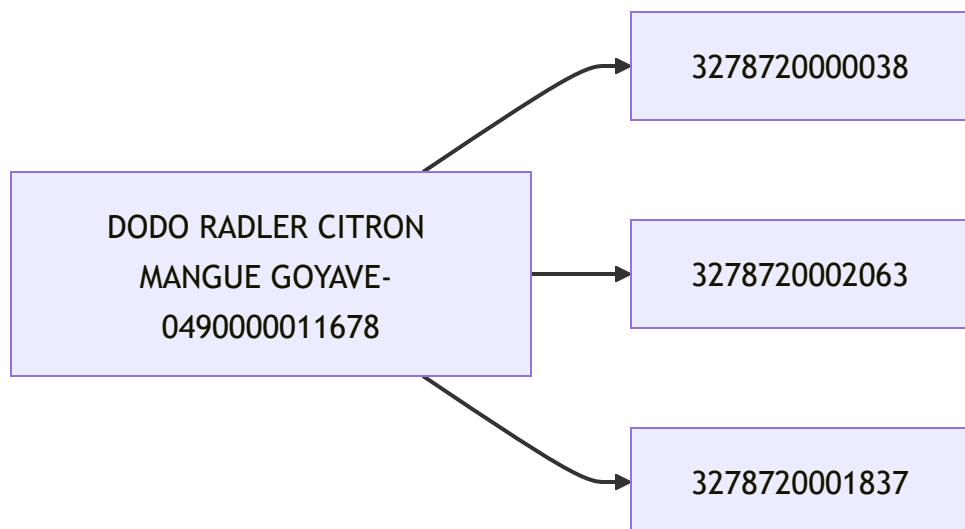
Référence	CodeBarre	Désignation	Achat	Quantité	Total
041508963985	041508963985	SAN PELLEGRINO 1L	1.25	10+19	?
0490000024531	8850161160790	COCOMAX	1.46	18	26.28
0490000028126	8850161161247	COCOMAX GRANDE BOUTEILLE	2.67	?	?

0490000024531	COCOMAX	1.45 €	18.00	26.15 €
0490000028126	COCOMAX GRANDE BOUTEILLE	2.65 €	-77.00	-203.74 €

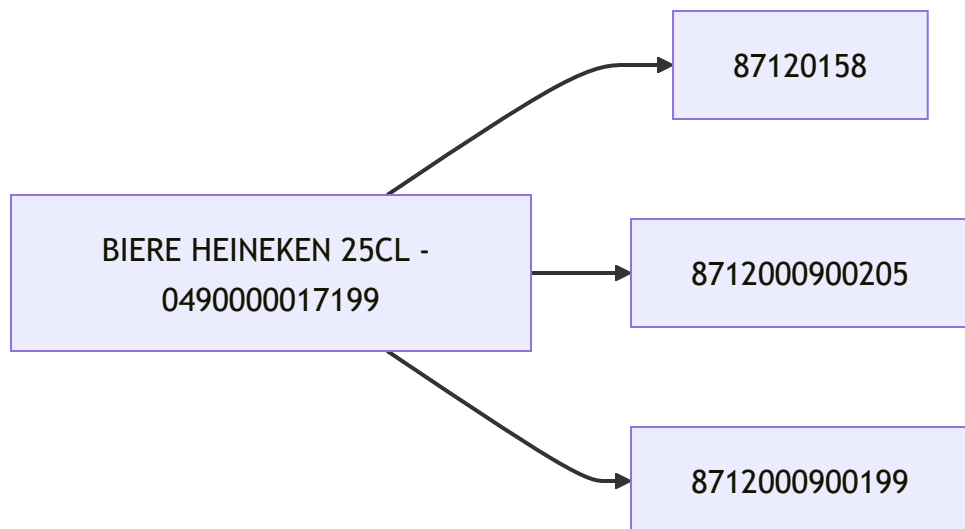
COCOMAX.png

Les références 0490000024531 & 0490000028126

La DODO RADLER CITRON GOYAGE MANGUE (0490000011678) est associé aux codes barres (3278720000038,3278720002063,3278720001837)



La référence **0490000017199** BIERE HEINEKEN 25CL

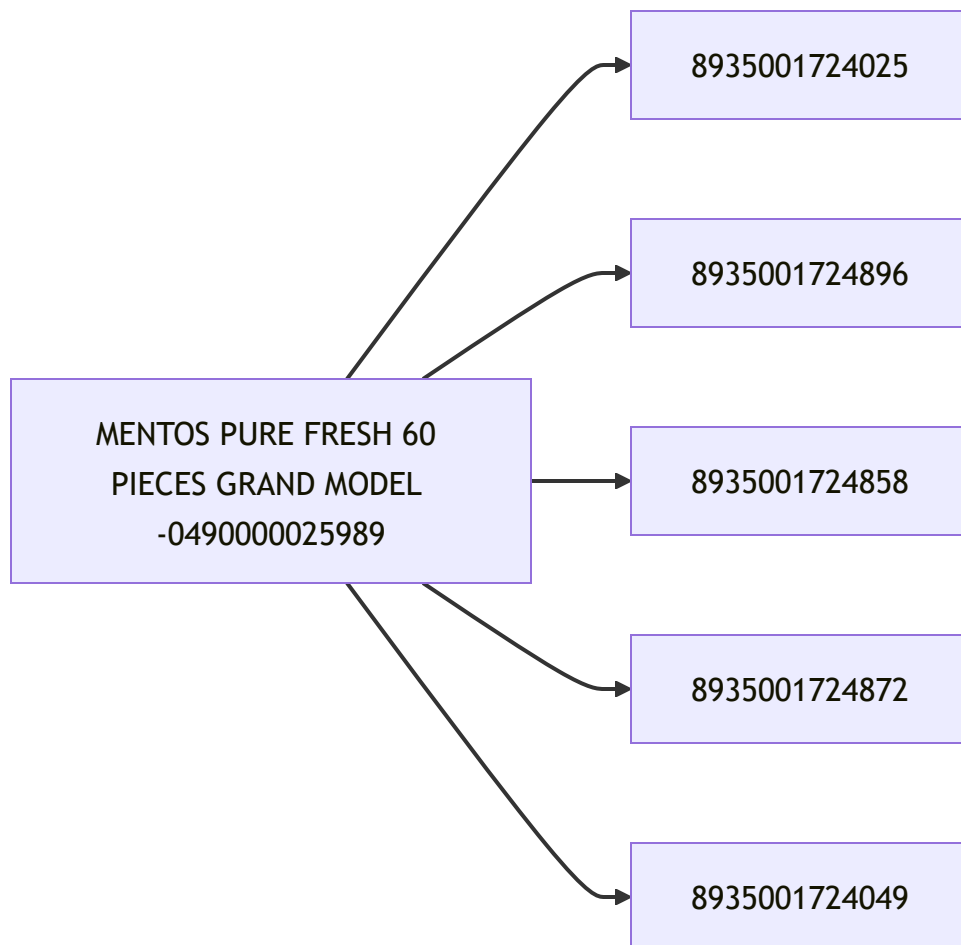


⚠ le code barre **8712000900205** associé à la référence **0490000017199** a été inventorié par Ivalis pour une quantité de 42

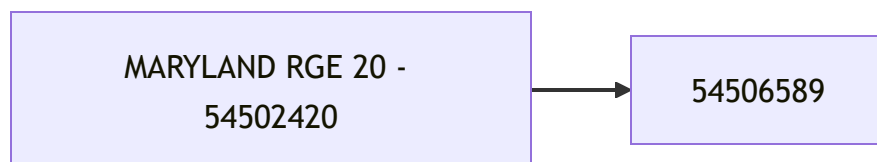
⚠ La liste des mouvements de stock pour ce produit
Une coorection de stock le 30-09-2024 à 12:08

0490000026672 - DODO CAN 50 CL - Non inventorié Pas de code barre associé

0490000027808 - VERRINE DESSERT - Non inventorié Pas de code barre associé



•



01-10-2023 au 31-09-2024

Réception : 70 (dernière réception 15-2024 quantité = 30)

Ventes : 379 (dernière vente 28-09-2024 quantité = 1)

Non inventorié par Ivalis

0490000026580



5400827009418

 MARYLAND RGE 30

Inventorié : 16

Réception : 32

Vente : 179

0490000029246



5400827026057


 MARYLAND RGE 40

Inventorié : 3

0490000027921



5902198160724

 0490000027921 (BUM BUM)

01-10-2023 au 30-09-2024

Vente = 1738

pas d'inventaire réalisé par Ivalis

Inventaire 3Bassins au 30-09-2024

Méthode : Tous les produits avec date d'inventaire le 30-09-2024 et une sortie au delà du 30-09-2024

Les différents codebarres associés à cette référence

Référence	CodeBarre	Désignation
049000002598 9	893500172402 5	MENTOS PURE FRESH 60 PIECES GRAND MO DEL
049000002598 9	893500172489 6	MENTOS PURE FRESH 60 PIECES GRAND MO DEL
049000002598 9	893500172485 8	MENTOS PURE FRESH 60 PIECES GRAND MO DEL
049000002598 9	893500172487 2	MENTOS PURE FRESH 60 PIECES GRAND MO DEL
049000002598 9	893500172404 9	MENTOS PURE FRESH 60 PIECES GRAND MO DEL

Les mouvements de stock

- Des ventes ont été réalisées après le 30-09-2024
- Cette référence n'est pas présente dans les fichiers envoyés par ivalis (Résultats inventaires et inconnus)

Référence	CodeBarre	Désignation
0490000027570		NOUILLE INSTANT PORC

Les mouvements de stock

- Des ventes ont été réalisées après le 30-09-2024
- Cette référence n'est pas présente dans les fichiers envoyés par ivalis (Résultats inventaires et inconnus)

Référence	CodeBarre	Désignation
0490000023817	8723900213500	GLADSTONE TABAC ROUGE

- Des ventes ont été réalisées après le 30-09-2024
- Cette référence n'est pas présente dans les fichiers envoyés par ivalis (Résultats inventaires et inconnus)

Référence	CodeBarre	Désignation
0490000027808	NULL	VERRINE DESSERT

- Des ventes ont été réalisées après le 30-09-2024
- Cette référence n'est pas présente dans les fichiers envoyés par ivalis (Résultats inventaires et inconnus)

Référence	Code barre	Désignation
3073786155186	3073780972192	LA VACHE QUI RIT
3073786155186	3073780972291	LA VACHE QUI RIT
3073786155186	3451790515910	LA VACHE QUI RIT
3073786155186	3251350124008	LA VACHE QUI RIT

- Aucun inventaire réalisé pour cette référence

Dans l'état de stock joint , les références dont les quantités sont négatives , n'ont pas fait l'objet d'un inventaire. Les exemples ci-dessus extraites de l'état d'inventaire ont eu des ventes après le 30-09-2024.

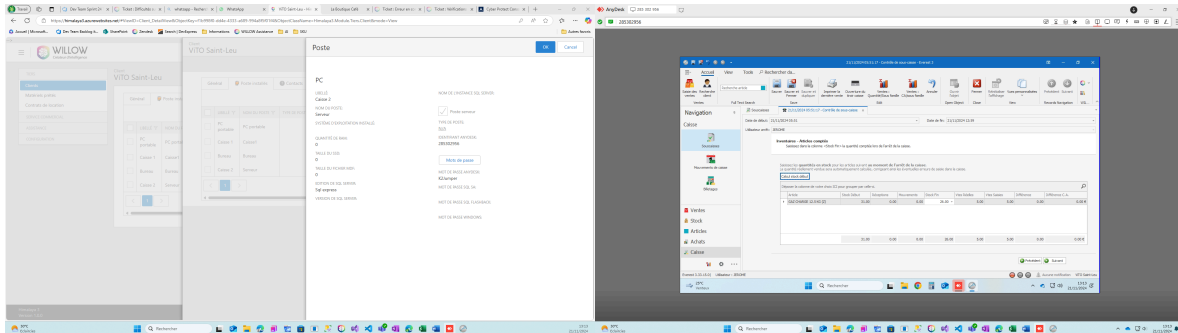
Les articles inventoriés par Ivalis ont bien été comptabilisé dans l'inventaire. Par contre, les articles non inventoriés n'ont pas été remis à zéro. La solution serait de lister les articles non inventoriés. Stock-> Liste des articles non inventoriés

- Date de l'inventaire du 30-09-2024 au 30-09-2024
- Action rechercher

Une liste des articles non inventoriés sera produite. Ensuite sélectionner l'action Créer un inventaire -> Inventaire normale.

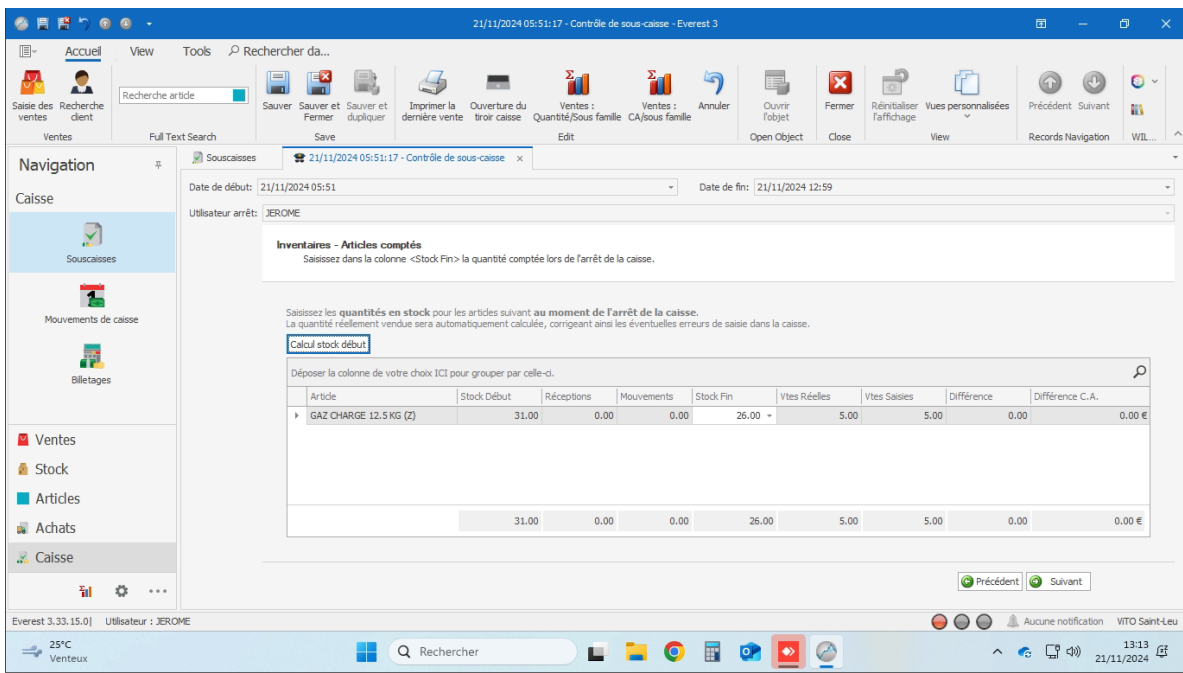
Engen Mon Caprice

deux BI on été facturé,un seul bi devrait être facturé. Comment faire pour annuler la facture ?



Stock Gaz.png

Divers problèmes



StockVitoSaintLeu.png

Start typing here...

Mobile Pneu

Solde : 0.00 € Encours : 0.00 € Situation : 0.00 € Compte : Points : 0 Détails : Sélectionner un Client :
 Le compte du client est soldé Utilisables : désactivé Utiliser : NOWAK SIHOE MARIE 2206-604

FORFAIT Recherche

PETITES FOUR... PRESTATIONS ...

Legend 3

L'erreur suivante est survenue :

Passer l'état de « GetObjectNonReentrant » de l'état « CommitTransactionNonReentrant, CommitChangesToDataLayer » est interdite en raison de l'état 'CommitChangesToDataLayer' pour 'DevExpress.Xpo.LinqUnitOfWork(2588)'. Très probablement, vous essayez d'ouvrir un objet à l'opération de changement, tandis que l'objet précédent

OK

Article non sélectionné

1.00 Montant :

Désignation						
7 PLAQ AV NISSAN PIXO		0.00 %	45.00 €			
DC FERODO DISQUE DE FREIN AV NISSAN PIXO/SUZUKI ALTO	2.00	0.00 %	150.00 €			
ITRAC E.TRAC 155/65R14 75T TRACMAX X-PRIVILET2	2.00	0.00 %	150.00 €			
TAXE ECO	2.00	0.00 %	3.12 €			
PETITES FOURNITURES DIVERSES	1.00	0.00 %	0.00 €			
PRESTATIONS	1.00	0.00 %	150.00 €			

6 SUN=9.00

Annuler :

Stock : Aucun article Total TTC : 0.00 € Total TTC net : 0.00 €

Contacts : NISSAN PIXO DFB...

127 % de MPOULE STOP 2PLOT 12V21W
 18 % de LEICHTLAUF HIGH TECH 5W40 SL
 18 % de TAXE ECO
 18 % de PRESTATIONS DIVERSES 8.5%

7 8 9
 4 5 6
 1 2 3
 0 .

Valider

TTC net: 498.12 €

JxlOqdQpxg.png

Papangue

$$\forall x \in X, \quad \exists y \leq \epsilon$$

$$\cos(2\theta) = \cos^2 \theta - \sin^2 \theta$$

$$k = \textcolor{red}{x} - \textcolor{blue}{2}$$

valeur

FraisApprocheReception

Ajout d'un fras d'approche fixe sur une réception

- FraisApprocheFixe (ligne 448)
- TotalHTDocument (ligne 225 - Réception)
-

Vito 11

Dans le journal OF - OD cabinet comptable Il y a des écritures de comptes bancaires qui devraient être affectées au journal de banque 50

Code	ID
OF	159374
50	13292

Compte de banque

ID	COMPTE	JOURNAL
13293	51250000	13292
3035244	51250001	13292

ID	COMPTE	DEBIT	CREDIT
6982378	77100000	0	828.14
6982379	51250000	828.14	0
6982421	77100000	0	488.25
6982422	51250000	488.25	0
6982451	77100000	0	1506.91
6982452	51250000	1506.91	0
6982491	76800000	0	2037.75
6982492	51250001	2037.75	0
6982503	7710000	0	683.16
6982504	51250000	683.16	0

Action à réaliser sur la base de Vito11

UPDATE COMPTABILITEGENERALE

SET journal=13292

WHERE id **IN**

(6982378,6982379,6982421,6982422,6982451,6982452,6982491,6982492,6982503,6982504)

Méthodes mémo techniques

Quel est le bon sens des écritures comptables

Je vais vous donner deux codes mnémotechniques simples pour retenir facilement le sens des écritures pour les achats et les ventes.

Code **DIED** (pour les Dépenses/Achats)

D: Ce qui rentre est au Débit

I: Les Immobilisations rentrent → Débit

E: Les Entrées en stock rentrent → Débit

D: Les Dépenses rentrent → Débit

Code **CRIC** (pour les Recettes/Ventes)

C: Ce qui sort est au Crédit


R: Les Recettes sortent → Crédit

I: Les sorties d'Immobilisations sortent → Crédit


C: Les sorties de stock (Cessions) sortent → Crédit

En pratique :

Pour les ACHATS (DIED) :

 Achats/Stocks/Immobilisations (Débit car ça rentre) à Fournisseurs/Banque (Crédit)

Pour les VENTES (CRIC) :

 Clients/Banque (Débit) à Ventes/Stocks/Immobilisations (Crédit car ça sort)

Un autre moyen mnémotechnique simple :

 Les Dépenses sont au Débit Les Recettes sont au cRédit

Taux de marque vs taux de marge

La différence principale entre taux de marque et taux de marge :

Taux de marque = $(\text{Prix de vente HT} - \text{Prix d'achat HT}) / \text{Prix de vente HT} \times 100$

Taux de marge = $(\text{Prix de vente HT} - \text{Prix d'achat HT}) / \text{Prix d'achat HT} \times 100$

Exemple avec un produit :

- Prix d'achat HT : 100€
- Prix de vente HT : 150€
- Marge : 50€

Donc :

- Taux de marque = $50/150 \times 100 = 33,33\%$
- Taux de marge = $50/100 \times 100 = 50\%$

Le taux de marque est toujours inférieur au taux de marge. Pour convertir :

- Taux de marge = $\text{Taux de marque} / (1 - \text{Taux de marque})$
- Taux de marque = $\text{Taux de marge} / (1 + \text{Taux de marge})$

Taux de marge et taux de marque

Le **taux de marge** et le **taux de marque** sont deux indicateurs financiers couramment utilisés dans la gestion d'une entreprise pour évaluer la rentabilité des ventes, mais ils diffèrent dans leur mode de calcul et leur interprétation.

Taux de marge

Le **taux de marge** mesure le rapport entre la marge brute (bénéfice réalisé sur un produit ou service) et le **coût d'achat** du produit.

Formule: $\text{Taux de marge} = \left(\frac{\text{Marge brute}}{\text{Coût d'achat}} \right) \times 100$

Où :

- **Marge brute** = Prix de vente HT - Coût d'achat HT

Interprétation: Il indique le pourcentage de bénéfice réalisé par rapport au coût d'achat. Plus le taux est élevé, plus l'entreprise génère une marge importante pour chaque euro dépensé en achat.

Taux de marque

Le **taux de marque** mesure le rapport entre la marge brute et le **prix de vente**.

Formule: $\text{taux de marque} = \left(\frac{\text{marge brute}}{\text{prix de vente}} \right) \times 100$

Interprétation: Il montre le pourcentage du prix de vente qui correspond à la marge brute. Il est particulièrement utile pour évaluer la part du chiffre d'affaires qui reste à l'entreprise après avoir couvert ses coûts d'achat.

Différence clé

- **Taux de marge:** centré sur le coût d'achat (analyse des dépenses).
- **Taux de marque:** centré sur le prix de vente (analyse des revenus).

Exemple

Un produit a :

- Coût d'achat : 50 € HT
- Prix de vente : 100 € HT

1. **Marge brute:** $100 - 50 = 50$ €, €

2. **Taux de marge:** $\left(\frac{50}{50} \right) \times 100 = 100$ %

3. **Taux de marque:** $\left(\frac{50}{100} \right) \times 100 = 50$ %

- Le taux de marge est de **100 %**, car la marge brute est égale au coût d'achat.
- Le taux de marque est de **50 %**, car la marge brute représente la moitié du prix de vente.

Ces indicateurs sont complémentaires : le **taux de marge** aide à évaluer la rentabilité des achats, tandis que le **taux de marque** donne une vision sur la structure du chiffre d'affaires.

Frais d'approche

Les frais d'approche sont l'ensemble des coûts supplémentaires qui s'ajouteront au prix d'achat initial d'une marchandise pour la rendre disponible à son lieu de vente.

Quel est l'importance des frais d'approche ?

- Ils permettent de déterminer le coût complet d'acquisition
- Ils sont essentiels pour calculer le **prix de revient réel**
- ils influencent directement la marge commerciale.

En comptabilité, les frais d'approche sont généralement incorporés au coût d'entrée des marchandises dans le stock

Ces frais comprennent généralement :

1. Les frais de transport

- Transport maritime, aérien ou terrestre
- Frais de manutention
- Assurance transport

2. Les frais de douane et taxes

- Droits de douane
- Tva à l'importation
- Redevances diverses (Octroi de mer,...)

3. Les Frais logistiques

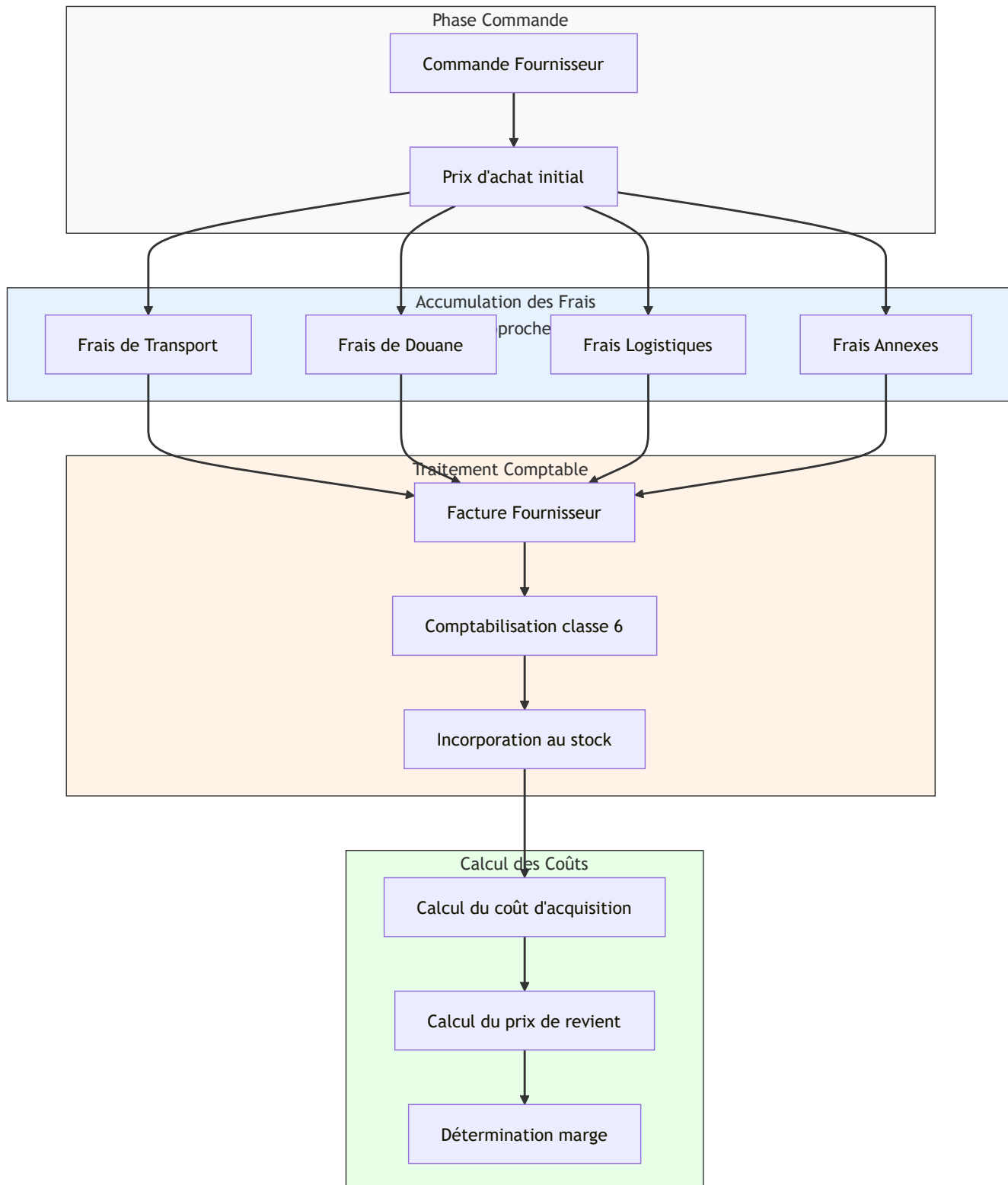
- Frais de stockage temporaires
- Frais de documentation

4. Autres frais annexes

- Commissions d'intermédiaires

- Frais de change

Schéma global du flux



Exemple d'implémentation en c#

```
using System;
using System.Collections.Generic;
using System.Linq;

namespace GestionFraisApproche
{
    public class Article
    {
        public int Id { get; set; }
        public string Reference { get; set; }
        public string Designation { get; set; }
        public decimal PrixAchat { get; set; }
        public decimal Poids { get; set; }
        public decimal Volume { get; set; }
        public Dictionary, decimal> FraisApproche { get; set; } = new
        Dictionary, decimal>();
        public decimal CoutTotal { get; set; }

        public Article(string reference, string designation, decimal
        prixAchat)
        {
            Reference = reference;
            Designation = designation;
            PrixAchat = prixAchat;
            CoutTotal = prixAchat;
        }
    }

    public class FraisApproche
    {
        public decimal Transport { get; set; }
        public decimal Douane { get; set; }
        public decimal Logistique { get; set; }
        public decimal Autres { get; set; }
        public Dictionary, decimal> Details { get; set; } = new
```



```

Dictionary, decimal>());

        public decimal CalculerTotalFrais() => Transport + Douane +
Logistique + Autres;
    }

    public enum CleRepartition
    {
        Valeur,
        Poids,
        Volume
    }

    public class Commande
    {
        public string Numero { get; set; }
        public DateTime Date { get; set; }
        public List Articles { get; set; } = new List();
        public FraisApproche FraisGlobaux { get; set; } = new
FraisApproche();
        public CleRepartition CleRepartition { get; set; } =
CleRepartition.Valeur;

        public Commande(string numero, DateTime date)
        {
            Numero = numero;
            Date = date;
        }
    }

    public class EcritureComptable
    {
        public string Journal { get; set; }
        public DateTime Date { get; set; }
        public string CompteDebit { get; set; }
        public string CompteCredit { get; set; }
        public decimal Montant { get; set; }
        public string Libelle { get; set; }
    }

```

```

    }

    public class GestionnairesFraisApproche
    {
        private Dictionary, Commande> _commandes = new Dictionary,
Commande>();
        private const decimal TVA = 0.20m;

        public Commande CreerCommande(string numero, DateTime date)
        {
            var commande = new Commande(numero, date);
            _commandes[numero] = commande;
            return commande;
        }

        public void AjouterArticle(string numeroCommande, Article
article)
        {
            if (_commandes.TryGetValue(numeroCommande, out var
commande))
            {
                commande.Articles.Add(article);
            }
            else
            {
                throw new KeyNotFoundException($"Commande
{numeroCommande} non trouvée");
            }
        }

        public void AjouterFrais(string numeroCommande, string
typeFrais, decimal montant, string description)
        {
            if (!_commandes.TryGetValue(numeroCommande, out var
commande))
                throw new KeyNotFoundException($"Commande
{numeroCommande} non trouvée");
        }
    }

```

```

        switch (typeFrais.ToLower())
        {
            case "transport":
                commande.FraisGlobaux.Transport += montant;
                break;
            case "douane":
                commande.FraisGlobaux.Douane += montant;
                break;
            case "logistique":
                commande.FraisGlobaux.Logistique += montant;
                break;
            default:
                commande.FraisGlobaux.Autres += montant;
                break;
        }

        commande.FraisGlobaux.Details[description] = montant;
    }

    public void RepartirFrais(string numeroCommande)
    {
        if (!_commandes.TryGetValue(numeroCommande, out var
commande))
            throw new KeyNotFoundException($"Commande
{numeroCommande} non trouvée");

        decimal baseRepartition = 0;
        var coefficients = new List();

        // Calcul de la base de répartition
        switch (commande.CleRepartition)
        {
            case CleRepartition.Valeur:
                baseRepartition = commande.Articles.Sum(a =>
a.PrixAchat);
                coefficients = commande.Articles.Select(a =>
a.PrixAchat / baseRepartition).ToList();
                break;

```

```

        case CleRepartition.Poids:
            baseRepartition = commande.Articles.Sum(a =>
a.Poids);
            coefficients = commande.Articles.Select(a => a.Poids
/ baseRepartition).ToList();
            break;
        case CleRepartition.Volume:
            baseRepartition = commande.Articles.Sum(a =>
a.Volume);
            coefficients = commande.Articles.Select(a =>
a.Volume / baseRepartition).ToList();
            break;
    }

    // Application des frais
    for (int i = 0; i < commande.Articles.Count; i++)
    {
        var article = commande.Articles[i];
        var coefficient = coefficients[i];

        article.FraisApproche["transport"] =
commande.FraisGlobaux.Transport * coefficient;
        article.FraisApproche["douane"] =
commande.FraisGlobaux.Douane * coefficient;
        article.FraisApproche["logistique"] =
commande.FraisGlobaux.Logistique * coefficient;
        article.FraisApproche["autres"] =
commande.FraisGlobaux.Autres * coefficient;

        article.CoutTotal = article.PrixAchat +
article.FraisApproche.Values.Sum();
    }
}

public List<Article> GenererEcrituresComptables(string numeroCommande)
{
    if (!_commandes.TryGetValue(numeroCommande, out var
commande))

```

```

        throw new KeyNotFoundException($"Commande
{numeroCommande} non trouvée");

        var ecritures = new List();

        // Écriture principale d'achat
        decimal totalAchat = commande.Articles.Sum(a =>
a.PrixAchat);
        ecritures.Add(new EcritureComptable
        {
            Journal = "AC",
            Date = commande.Date,
            CompteDebit = "607000",
            CompteCredit = "401000",
            Montant = totalAchat,
            Libelle = $"Facture {commande.Numero}"
        });

        // Écritures des frais d'approche
        if (commande.FraisGlobaux.Transport > 0)
        {
            ecritures.Add(new EcritureComptable
            {
                Journal = "AC",
                Date = commande.Date,
                CompteDebit = "608100",
                CompteCredit = "401000",
                Montant = commande.FraisGlobaux.Transport,
                Libelle = $"Transport {commande.Numero}"
            });
        }

        // Autres frais similaires...

        return ecritures;
    }

    public decimal CalculerPrixRevient(Article article, decimal

```

```

margeStandard)
    {
        return article.CoutTotal * (1 + margeStandard);
    }
}

// Extension pour faciliter les tests et l'utilisation
public static class Exemple
{
    public static void DemonstrationUtilisation()
    {
        var gestionnaire = new GestionnairesFraisApproche();

        // Création d'une commande
        var commande = gestionnaire.CreerCommande("CMD001",
DateTime.Now);

        // Ajout d'articles
        var article1 = new Article("REF001", "Article 1", 1000m) {
Poids = 10m };
        var article2 = new Article("REF002", "Article 2", 2000m) {
Poids = 20m };

        gestionnaire.AjouterArticle("CMD001", article1);
        gestionnaire.AjouterArticle("CMD001", article2);

        // Ajout des frais
        gestionnaire.AjouterFrais("CMD001", "transport", 500m,
"Transport maritime");
        gestionnaire.AjouterFrais("CMD001", "douane", 150m, "Droits
de douane");
        gestionnaire.AjouterFrais("CMD001", "logistique", 200m,
"Stockage temporaire");

        // Répartition des frais
        gestionnaire.RepartirFrais("CMD001");

        // Génération des écritures

```

```
        var ecritures =  
gestionnaire.GenererEcrituresComptables("CMD001");  
    }  
}  
}
```

Calcul frais d'approche

Lien entre frais d'approche et prix unitaire moyen pondéré

Le frais d'approche et le prix unitaire moyen pondéré (PUMP) sont liés dans le cadre de la gestion des coûts et des stocks. Voici le lien entre ces deux notions :

1. Définition des frais d'approche

Les frais d'approche incluent tous les coûts supplémentaires nécessaires pour que des marchandises ou des matières premières soient disponibles à leur emplacement d'utilisation. Cela peut inclure :

- Les frais de transport
- Les droits de douane
- Les assurances
- Les coûts de manutention, etc.

Ces frais s'ajoutent au coût d'achat pour obtenir le coût total d'acquisition.

2. Prix unitaire moyen pondéré (PUMP)

Le PUMP est une méthode de valorisation des stocks. Il s'agit du coût moyen d'une unité de produit en fonction du coût total des marchandises disponibles (coût d'achat + frais d'approche) et de la quantité totale.

La formule est la suivante :

$$\text{PUMP} = \frac{\text{Valeur totale du stock (achats + frais d'approche)}}{\text{Quantité totale en stock}}$$

3. Lien entre les deux

Les frais d'approche sont inclus dans la valeur totale du stock et donc dans le calcul du PUMP. Plus les frais d'approche sont élevés, plus le PUMP augmente. En d'autres termes :

- Augmentation des frais d'approche → Augmentation du coût total des marchandises

→ Augmentation du PUMP.

- Réduction des frais d'approche → Réduction du coût total des marchandises → Diminution du PUMP.

Exemple concret

- Achat de 100 unités à 10 € chacune (coût initial : 1 000 €).
- Frais d'approche : 200 €.
- Quantité totale : 100 unités.

Le PUMP sera :

$$\text{PUMP} = \frac{1000 + 300}{100} = 13\text{€}$$

Si les frais d'approche augmentent à 300 €, le PUMP deviendra :

Ainsi, la gestion des frais d'approche a un impact direct sur la valorisation des stocks via le PUMP.

Calcul du pump version Legend

Etape 1

```
article.Pump = PumpUtils.GetPump(article, DateTime.Now, forceCalcul);
```

Etape 2

```
private static decimal GetPumpMouvementsStock(Article article, DateTime date)
{
    decimal pump = Decimal.Zero, stock = Decimal.Zero;

    IQueryable mouvementsStock = GetMouvementsStock(article, date);
    foreach (MouvementStock mouvementStock in mouvementsStock)
    {
        if (mouvementStock.TypeDocument == TypeDocument.Réception)
        {
```

```

        pump = CalculPump(stock,
                           pump,
                           mouvementStock.Quantité,
                           mouvementStock.ValeurUnitaire,
                           GetFraisApproche(mouvementStock));
    }
    stock += mouvementStock.Quantité;
}

return pump;
}

```

Etape 3

```

private static decimal CalculPump(decimal ancienStock, decimal
ancienPump, decimal quantitéReçue, decimal prixAchat, decimal
fraisApproche)
{
    decimal coûtRevientUnitaire = prixAchat * (1 + fraisApproche);

    if (ancienStock = Decimal.Zero || ancienPump = Decimal.Zero)
        return coûtRevientUnitaire;

    decimal nouveauStock = ancienStock + quantitéReçue;
    if (nouveauStock = Decimal.Zero)
        return coûtRevientUnitaire;

    decimal pump = ((ancienPump * ancienStock) + (coûtRevientUnitaire *
quantitéReçue)) / nouveauStock;
    return pump > Decimal.Zero ? pump : Decimal.Zero;
}

```