**Suivi des commandes des revendeurs**

# Distributech

Cahier des charges

Nathalie BEDIEE

# Table des matières

# 1. Introduction

**Nom du projet :** Suivi des commandes des revendeurs
Ce document décrit les besoins, contraintes et objectifs visant à améliorer la gestion des commandes et des stocks chez Distributech.

## 1.1. Motivations

### 1.1.1. Le client

L'entreprise **Distributech** exerce une activité de grossiste en équipements électroniques. Elle collabore avec un réseau de **revendeurs régionaux** qui se réapprovisionnent régulièrement auprès d'elle afin de maintenir leurs stocks.

### 1.1.2. Le problème

**Manque de centralisation des informations**.

Actuellement, les **commandes** sont simplement transmises sous forme de **fichiers CSV** envoyés par les revendeurs, **sans intégration centralisée**.
Ce type de format, bien que simple à manipuler, peut poser des problèmes de **cohérence** ou de **complétude** :

- colonnes manquantes
- valeurs non conformes
- formats de date incohérentes
- ...

Ces anomalies nécessitant souvent une **étape de validation et de normalisation** avant intégration.

Les stocks de Distributech sont suivis dans **une base SQLite** unique, représentant un **stock partiel** basé sur les **mouvements de production**, sans lien direct avec les commandes.

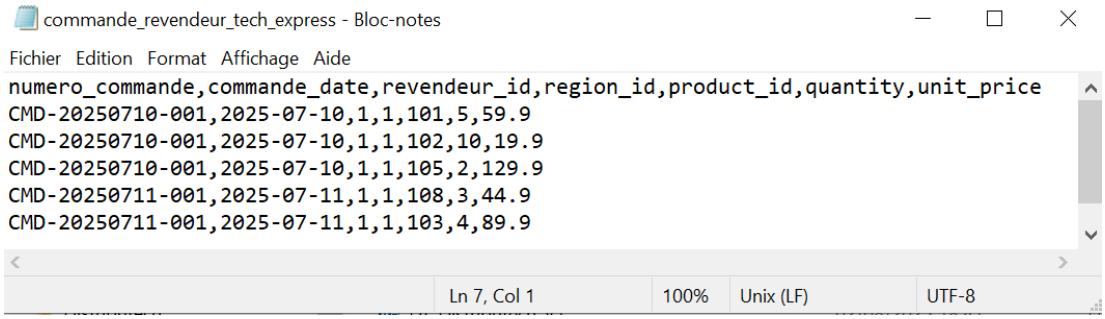
**Limites constatées :**

- **Stock partiel :** SQLite ne représente pas le stock réel (pas d'impact des commandes), donc pas de stock global consolidé.
- **Suivi logistique limité :** difficile de connaître les niveaux de stock consolidés à une date ou une période.
- **Suivi commercial incomplet :** pas de vue consolidée des volumes commandés ni du chiffre d'affaires (par région/période).

### 1.1.3. L'existant

➢ **Flux de commandes**
Les commandes sont transmises sous forme de **fichiers CSV hebdomadaires** envoyés par les revendeurs.

- **Exemple de contenu** (fichier de commandes) :




- **Caractéristiques actuelles** :
  - Séparateur : ,
  - Champs obligatoires : tous sauf region_id
    (la région peut être déduite via la table revendeur de SQLite)
  - Format des dates : AAAA-MM-JJ
  - Types attendus: quantite (integer), prix_unitaire (float)

- **Règles métier** :
  - Une commande est associée à un seul revendeur
  - chaque commande à un numéro de commande unique
  - aucune valeur négative

➢ **Mouvements de production**
La **Base SQLite** joue un rôle central : elle sert de source de référence pour le catalogue des produits, la liste des régions et revendeurs : Elle conserve l'historique des **réapprovisionnements** (mouvements de production) effectué par Distributech.

**Structure des tables**

- **region**
  - region_id (integer, clé primaire) : identifiant unique de la région
  - region_name (text) : nom de la région
    *Exemples : Île-de-France, Bretagne…*

- **revendeur**
  - revendeur_id (integer, clé primaire) : identifiant unique du revendeur
  - revendeur_name (text) : nom du revendeur
  - region_id (integer, clé étrangère vers région) : référence région associée

  *Un revendeur est lié à une région unique*

- **produit**
  - product_id (integer, clé primaire) : identifiant unique du produit
  - product_name (text) : nom du produit
    *Exemple : Casque Bluetooth, Chargeur USB-C…*
  - cout_unitaire (real) : coût d'achat pour le revendeur

- **production**
  - production_id (integer, clé primaire, auto-incrémenté)
  - product_id (integer, clé étrangère vers produit)
  - quantity (integer) : quantité de produits réapprovisionnés
    *Strictement positif*
  - date_production (text, format AAAA-MM-JJ)
    *≤ date du jour (pas de dates futures)*

  *Chaque entrée correspond à un mouvement d'entrée de stock (réassort).*

### 1.1.4. Le besoin non satisfait

1. **Absence de consolidation des données** : les commandes (CSV hebdomadaires) et les mouvements de stock (SQLite) sont gérés séparément, sans vue unifiée.

2. **Difficulté de reconstitution du stock** : l'état courant ou passé du stock par produit ne peut être obtenu qu'en recalculant manuellement les entrées et sorties, ce qui est fastidieux et source d'erreurs.

3. **Manque d'automatisation** : les processus actuels nécessitent des manipulations manuelles répétitives (intégration des CSV, mise à jour SQLite), entraînant perte de temps et risque d'incohérences.

4. **Limitation du suivi commercial** : aucune consolidation ne permet d'analyser facilement le chiffre d'affaires par région ou l'évolution des ventes dans le temps.

5. **Manque de traçabilité** : en l'absence de système centralisé, il est difficile d'assurer un suivi fiable (ex. doublons, erreurs de saisie, cohérence des dates et identifiants).

### 1.1.5. Les objectifs

- **Centraliser les données** : regrouper dans une base unique l'ensemble des informations issues des fichiers CSV (commandes), de la base SQLite (stocks, revendeurs, régions, produits), afin d'éviter la dispersion des données et d'assurer une cohérence globale.

- **Automatiser le processus ETL** : mettre en place un pipeline robuste (Extract, Transform, Load) capable d'intégrer périodiquement les nouvelles données, de valider leur qualité et de normaliser les formats.

- **Générer une base SQL consolidée et maintenable** : disposer d'un modèle relationnel structuré permettant de conserver l'historique des commandes et des mouvements de stock tout en garantissant des mises à jour simples et fiables.
- **Offrir une visualisation claire :** rendre possible l'analyse de l'évolution des stocks au niveau global (pas de répartition régionale, car le stock de production est centralisé) et des produits commandés par région ou sur une période, pour répondre aux besoins logistiques et commerciaux.
- **Améliorer le suivi commercial** : permettre le calcul et l'analyse du chiffre d'affaires global et par région, afin de suivre l'évolution des ventes et d'appuyer la prise de décision.
- **Renforcer la traçabilité et la qualité des données** : assurer le suivi des sources (fichier, ligne d'origine), détecter et rejeter les anomalies (doublons, incohérences), pour garantir des analyses fiables.

## 1.2. Contexte

### 1.2.1. Précisions sur le client

**Distributech** est un **grossiste** en équipements électroniques.

L'entreprise collabore avec plusieurs **revendeurs régionaux**, chacun rattaché à une zone géographique, et leur fournit un **catalogue de produits commun**.

L'entreprise a **besoin** d'une solution fiable pour **centraliser ses commandes** et ses **mouvements de stock**.

### 1.2.2. Marché

Le secteur de la **distribution** repose sur des **échanges** fréquents entre un **fournisseur** et plusieurs **partenaires régionaux**.

Les **données** commerciales sont transmises sous forme de **fichiers standardisés**, le plus souvent au format CSV, afin de **faciliter l'intégration** et le traitement automatisé.

# 2. Documentation

## 2.1. Terminologie métier

- **Revendeur :** client régional rattaché à une zone géographique, achetant des équipements auprès de Distributech.
- **Région** : zone géographique unique à laquelle chaque revendeur est associé.

- **Produit :** article du catalogue commun de Distributech, identifié par un code unique, un nom et un coût unitaire.
- **Mouvement de production :** opération enregistrée dans la base SQLite, correspondant à un réapprovisionnement (entrée de stock) pour un produit donné, à une date précise.
- **Commande :** ensemble d'articles demandés par un revendeur.
- **Ligne de commande** : détail d'une commande, correspondant à un produit précis, une quantité et un prix unitaire.
- **Stock :** quantité disponible d'un produit à un instant donné.

### 2.2. Bibliographie succincte

- **Spécifications des fichiers CSV** transmis par les revendeurs.
- **Base SQLite** utilisée pour le suivi des mouvements de production.
- **Bases de données relationnelles** : principes d'organisation des données (tables, relations, contraintes).
- **Méthode Merise** : approche de modélisation des données (MCD, MLD) pour concevoir le schéma cible.
- **Processus ETL (Extract, Transform, Load)** : notions générales sur l'extraction, la transformation et le chargement de données issues de sources hétérogènes.

## 3. Profil des utilisateurs finaux

### 3.1. En utilisation

Les utilisateurs finaux sont les équipes logistiques et commerciales de Distributech. Un tableau de bord leur permettra d'accéder aux informations consolidées sans avoir à interroger directement la base SQL.

- **Profil attendu** : utilisateurs métier (logistique, commercial), sans compétences informatiques avancées.
- **Compétences nécessaires** : savoir lancer le tableau de bord et consulter les résultats affichés, ainsi qu'ouvrir les fichiers CSV générés dans un tableur (ex : Excel).
- **Limites** : ils n'accèdent pas directement à la base ni aux vues SQL.

## 3.2. En exploitation

Les responsables de l'exploitation auront pour mission de déclencher et de superviser le processus ETL, garantissant ainsi la mise à jour régulière de la base consolidée et la disponibilité des données pour les utilisateurs finaux.

- **Profil attendu** : personnel technique (informatique/administration des systèmes).
- **Compétences nécessaires** : savoir exécuter et surveiller un processus automatisé, interpréter les journaux d'exécution (logs) et gérer les incidents simples (ex. fichier manquant, données incohérentes).
- **Limites** : ils n'ont pas besoin de compétences métier en logistique ou commercial, leur rôle étant centré sur la fiabilité et la disponibilité du système.
- **Évolution possible** : le déclenchement de l'ETL peut être manuel ou automatisé via un ordonnanceur, afin de s'intégrer dans le fonctionnement régulier de l'entreprise.

## 3.3. En maintenance

Le personnel chargé de la maintenance devra assurer la pérennité et l'évolutivité du système.
Son rôle est d'intervenir en cas d'anomalie et d'adapter la solution aux évolutions futures des besoins ou des données sources.

- **Profil attendu** : technicien ou développeur spécialisé en bases de données et en intégration de données.
- **Compétences nécessaires** :
  - assurer la maintenance corrective (résolution des anomalies techniques),
  - assurer la maintenance adaptative (évolution du modèle de données, prise en charge de nouveaux formats ou sources),
  - maintenir la documentation technique à jour.
- **Engagement attendu** : garantir la continuité de service et la qualité des données consolidées dans le temps.

## 3.4. Environnement d'utilisation

Le système sera utilisé dans un environnement bureautique standard, accessible aux différents profils identifiés (utilisateurs métier, exploitation, maintenance).

- **Environnement matériel** : postes de travail équipés d'un ordinateur standard, connectés au réseau interne de l'entreprise.

- **Environnement logiciel** :
  - système d'exploitation Windows ou Linux,
  - tableur (ex : Excel) pour l'exploitation des exports CSV,
  - système de gestion de base de données relationnelle (MySQL) pour stocker et interroger les données consolidées.

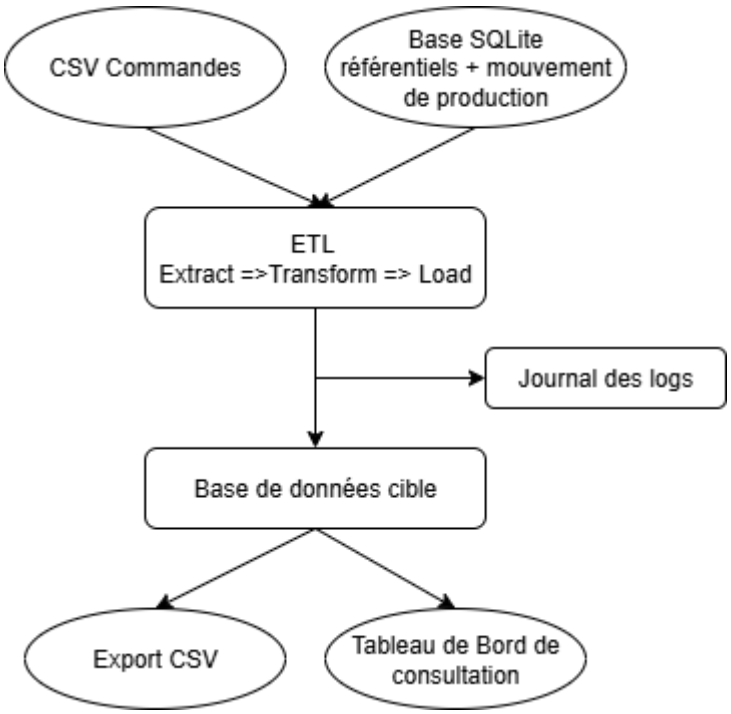
# 4. Fonctions à réaliser

### 4.1.1. Ce que le système doit faire

Le système attendu devra couvrir l'ensemble des fonctionnalités décrites ci-dessous. Elles sont présentées par ordre de priorité, avec pour chacune un critère de validation permettant leur vérification lors du recettage.

| Prio | Fonction | Description | Critère de performance |
|---|---|---|---|
| 1 | **Intégration des commandes** | Lire et traiter automatiquement les fichiers CSV hebdomadaires transmis par les revendeurs. | Toutes les commandes valides intégrées, anomalies détectées rejetées et consignées dans le log. |
| 2 | **Intégration des données SQLite** | Lire et intégrer les données de référence (régions, revendeurs, produits) ainsi que les mouvements de production. | Toutes les données valides intégrées, anomalies détectées rejetées et consignées dans le log. |
| 3 | **Nettoyage et validation** | Vérifier la cohérence des données, supprimer les doublons, contrôler formats et valeurs. | Aucune incohérence en base après traitement, toutes les anomalies consignées dans le log. |
| 4 | **Insertion en base SQL** | Alimenter le schéma relationnel cible et lier les entités. | Toutes les entités correctement insérées et liées. |
| 5 | **Traçabilité et logs** | Conserver l'origine des données (fichier, ligne) et générer un journal d'exécution détaillé. | 100 % des fichiers traités et des anomalies consignés dans le log. |
| 6 | **Export stock fin d'ETL** | Générer automatiquement un CSV de l'état des stocks par produit à l'issue du chargement. | Fichier CSV produit à chaque exécution, quantités correctes pour tous les produits valides. |
| 7 | **Stock à une date donnée** | Calculer le stock global par produit pour une date fournie à partir de l'historique mouvements + commandes. | Résultat conforme aux données sources |
| 8 | **Analyse par région** | Produire commandes et chiffre d'affaires par région, sur une période ou à une date donnée. | Résultats conformes aux données sources. |
| 9 | **Tableau de bord de consultation** | Fournir aux utilisateurs un accès simple aux principales informations (stock à une date, commandes et chiffre d'affaires sur une période), avec possibilité d'exporter les résultats. | Actions exécutables sans SQL, exports ouverts dans un tableur (ex : Excel). |

**Schéma de synthèse des flux :**




### 4.2. Ce que le système ne doit pas faire

- Modifier les fichiers CSV ou la base SQLite d'origine.
- Gérer une interface utilisateur graphique.
- Insérer en base des données non conformes.

## 5. Contrainte du système

### 5.1. Contraintes matérielles

- Le système doit fonctionner sur un **PC ou un serveur standard** (pas d'infrastructure dédiée requise).
- Un **espace disque suffisant** doit être prévu pour stocker :
  - les fichiers CSV reçus périodiquement,
  - la base SQLite source,
  - et la base de données cible centralisée.

## 5.2. Contraintes logicielles

- Le système devra être développé avec un **langage de programmation adapté au traitement de données**.
- Il devra s'appuyer sur des **bibliothèques pour la manipulation de fichiers CSV et l'accès à une base relationnelle**.
- Le système devra être **exploitable aussi bien sous Windows que sous Linux**

## 5.3. Contraintes fonctionnelles

- Les données sont traitées **localement** (aucune dépendance à un service externe).
- Le système doit fonctionner en **mode batch** : une fois lancé, l'ETL enchaîne automatiquement l'ensemble des étapes (extraction, validation, transformation, insertion en base, génération du stock global, export CSV et log) sans intervention de l'utilisateur pendant l'exécution.

## 5.4. Contraintes d'ergonomie

- L'utilisation du système doit rester **accessible à des profils non techniques** (ex. logisticiens, responsables commerciaux).
- L'accès aux données se fait via un **tableau de bord textuel** (menu console), conçu pour limiter les manipulations complexes.
- Les résultats doivent pouvoir être **exportés au format CSV**, afin d'être consultés facilement dans un tableur (ex. Excel).
- Les messages d'erreur et les logs doivent être **clairs et compréhensibles**, pour faciliter l'exploitation et l'identification des anomalies.

## 5.5. Autres facteurs de qualité exigés par le client

- **Fiabilité des traitements** : intégration correcte des données conformes et rejet explicite des anomalies.
- **Simplicité et maintenabilité du code** : permettre des évolutions ultérieures (nouvelles règles métier, nouvelles sources).

# 6. Aspects contractuels

## 6.1. Délais et protocole de livraison

**Durée totale estimée : 3 semaines**

- **Sprint 1 – Modélisation** : conception du modèle relationnel (**MCD et MLD**) à partir des besoins exprimés.
- **Sprint 2 – Extract** : développement du module d'import des fichiers CSV et de la base SQLite, pré-validation des données.
- **Sprint 3 – Transform** : développement du module de nettoyage, contrôles, enregistrement des anomalies dans les logs.
- **Sprint 4 – Load** : développement du module d'insertion en base SQL, gestion des erreurs, mapping des identifiants.
- **Sprint 5 – Gestion des stocks** : calcul du stock courant, export CSV de l'état des stocks, reporting d'indicateurs (KPI).
- **Sprint 6 – Tableau de bord** : mise à disposition d'une interface simple (cas d'école : exécution depuis un environnement Python, avec possibilité d'évolution future vers une interface web).
- **Sprint 7 – Améliorations** : évolutions post-MVP (contrôle de la base existante, stockage du dernier ID intégré, etc.).

**Livrables :**

- Programmes de traitement des données (ETL complet).
- Base SQL relationnelle (avec **MCD et MLD documentés**).
- CSV de l'état des stocks.
- Tableau de bord de consultation des données consolidées.
- Cahier des charges et documentation technique.

## 6.2. Aspect juridique et commercial

- Les données traitées dans le cadre du projet sont **confidentielles** et ne doivent en aucun cas être diffusées.
- Le client **Distributech** détient la **propriété intellectuelle complète** du produit final (programmes, base de données, documentation).
- Le projet doit respecter la réglementation en vigueur. Le **RGPD** n'est pas applicable ici car les données manipulées ne contiennent pas d'informations personnelles.
- L'utilisation de **logiciels ou bibliothèques tierces** devra se limiter à des solutions libres ou déjà couvertes par le client.
- Une **période de garantie et de maintenance corrective** minimale devra être assurée après livraison, afin de corriger d'éventuels dysfonctionnements identifiés en exploitation.