**Rapport projet « Prévisions et Dashboard des ventes 2019 »**

**1. Introduction et contexte**

En 2019, notre client a enregistré l’ensemble de ses ventes sur son site e-commerce, générant des données précieuses pour améliorer la gestion des stocks et optimiser la chaîne logistique. Conscient de la nécessité d’anticiper les besoins et de disposer d’outils décisionnels en temps réel, le client a souhaité :

* Centraliser ses données de commandes sous un système de gestion de base de données relationnelle (SGBDR).
* Mettre en place un tableau de bord interactif pour visualiser les indicateurs clés de performance (KPI) et explorer les tendances mensuelles.
* Développer un modèle prédictif pour estimer, à l’échelle journalière, le volume de commandes futur.

Ce rapport retrace l’ensemble des étapes réalisées, du choix des technologies à la mise en production des livrables.

**2. Choix des technologies et milieu de travail**

1. **SGBDR : MySQL avec DBeaver**
   * **Pourquoi ?**
     + Interface graphique intuitive pour les opérations d’import et de requêtage.
     + Support natif du format CSV et options avancées de chargement en masse (LOAD DATA).
     + DirectQuery possible pour Power BI Desktop.
   * **Mise en place** : installation locale de MySQL (XAMPP/WampServer) et configuration d’une connexion DBeaver pour l’administration et l’import des données.
2. **Nettoyage des données : Python + pandas**
   * Traitement systématique des 12 fichiers CSV mensuels (Janvier à Décembre 2019).
   * Correction des anomalies :
     + Mauvais séparateurs (;/,) et en-têtes parasites.
     + Formats de date incohérents ou indéfinis.
     + Lignes vides ou incomplètes.
   * Génération de nouveaux fichiers Sales\_<Mois>\_2019\_clean.csv prêts à l’import.
3. **Reporting : Power BI Desktop**
   * Mode **DirectQuery** pour interroger directement la base MySQL en temps réel.
   * Utilisation du langage **DAX** pour créer des colonnes calculées (montant total, mois, ville).
   * Création de visuels interactifs et filtres pour les conseillers.
4. **Modélisation IA : Python**
   * Extraction des données journalières via SQLAlchemy et pandas.
   * Implémentation de modèles de série temporelle : SARIMA, Holt‑Winters et LSTM.
   * Évaluation automatique des performances (MAE, RMSE) et export du meilleur modèle.

**3. Construction de la base de données MySQL**

**3.1. Création de la base et de la table**

* Nom de la base : **ecommerce\_2019** (charset utf8mb4, collation utf8mb4\_unicode\_ci).
* Table **orders\_2019** avec les colonnes :
  + order\_id (INT, PK logique),
  + product (VARCHAR),
  + quantity\_ordered (INT),
  + price\_each (DECIMAL),
  + order\_date (DATETIME),
  + purchase\_address (VARCHAR).
* Index sur order\_date et product pour accélérer les requêtes temporelles et de regroupement.

**3.2. Import des données**

* **DBeaver ▸ Import Data** : multi‑sélection des CSV nettoyés.
* Paramètres :
  + Header row : 1 (ignore la ligne d’en-tête).
  + Délimiteur ,, quote ", ligne terminée par \n.
  + Format de date : MM/dd/yy HH:mm.
* Vérification du nombre total de lignes (~185 000).

**3.3. Gestion des doublons**

* Sauvegarde rapide de la table brute (orders\_2019\_backup).
* Création d’une table **orders\_2019\_dedup** remplie par SELECT DISTINCT de toutes les colonnes.
* Remplacement atomique via RENAME TABLE dans une transaction.
* Vérification : comparaison du nombre de lignes avant et après.

**4. Nettoyage avancé des CSV**

Pour chaque mois :

1. **Chargement brut** du fichier avec pandas (lecture ligne à ligne).
2. **Parsing** manuel des enregistrements :
   * Suppression des guillemets et remplacement des ; par espaces.
   * Split sur la virgule, reconstruction de l’adresse complète.
   * Conversion des champs en types natifs (int, float, datetime).
3. **Filtrage** des lignes corrompues : suppression de toutes les lignes où un champ critique est manquant ou mal formé.
4. **Export** d’un CSV propre et typé (\*\_clean.csv).
5. **Packaging** : création d’une archive ZIP contenant toutes les versions clean pour un import unique.

**5. Requêtes de contrôle et KPI**

Après import et déduplication, quatre requêtes SQL ont permis de valider la cohérence des données :

1. **Total des commandes** :
2. SELECT COUNT(DISTINCT order\_id) AS total\_commands
3. FROM orders\_2019;
4. **Chiffre d’affaires 2019** :
5. SELECT ROUND(SUM(quantity\_ordered \* price\_each),2) AS CA\_2019
6. FROM orders\_2019;
7. **Produit le plus vendu** :
8. SELECT product, SUM(quantity\_ordered) AS qty
9. FROM orders\_2019
10. GROUP BY product
11. ORDER BY qty DESC
12. LIMIT 1;
13. **Nombre de produits différents** :
14. SELECT COUNT(DISTINCT product) AS distinct\_products
15. FROM orders\_2019;

Les résultats ont été communiqués au client et intégrés en KPI dans Power BI.

**6. Conception du Dashboard Power BI**

**6.1. Connexion et modèle de données**

* **Get Data ▸ MySQL** (ecommerce\_2019 en DirectQuery).
* Création de colonnes DAX :
  + **MONTANT\_TOTAL** = [quantity\_ordered] \* [price\_each].
  + **Mois** = FORMAT(order\_date, "yyyy-MM") pour l’axe temporel.
  + **Ville** : extraction de la chaîne entre la 1ᵉ et la 2ᵉ virgule avec SUBSTITUTE + PATHITEM.

**6.2. Visuels et filtres**

1. **Cartes (Card)** : produits vendus (Sum), C.A. (Sum), commandes (DistinctCount).
2. **Graphique combiné** : colonnes = C.A. mensuel, ligne = quantités.
3. **Top 5 villes** : barres horizontales filtrées sur les 5 principales.
4. **Top 5 produits** : deux barres (quantité et C.A.), Top N=5.
5. **Segments (Slicers)** : Ville et product pour filtrer l’ensemble des visuels.

**6.3. Mise en forme et ergonomie**

* Palette harmonisée (3 teintes).
* Titres explicites et descriptions d’axe.
* Thème *Minimal* et mise en page aérée.
* Vérification des interactions (Edit interactions).

**7. Développement du script de prévision IA**

**7.1. Extraction et préparation**

* Requête SQL journalière :SELECT DATE(order\_date) AS order\_day, SUM(quantity\_ordered) AS qty ....
* Chargement dans pandas, indexation par date.
* Split : entraînement sur les 335 premiers jours, test sur les 30 derniers.

**7.2. Modèles évalués**

1. **SARIMA** : paramètres (1,1,1)(1,1,1,7) pour capturer saisonnalité hebdomadaire.
2. **Holt-Winters** : modèle additionnel trend + saisonnalité hebdo = 7.
3. **LSTM** : réseau récurrent (look\_back=7), couche LSTM 50 unités, sortie dense.

**7.3. Évaluation et sélection**

* **Métriques** : MAE et RMSE calculés sur le jeu test.
* Sélection automatique du modèle avec le plus faible RMSE.

**7.4. Export des résultats**

* Sauvegarde du meilleur modèle (.pkl ou .h5).
* Génération d’un fichier CSV daily\_quantity\_forecast.csv contenant les 30 prévisions suivantes.
* Production d’un graphique de la série historique vs. prédictions.

**8. Livrables et déploiements**

1. **Scripts SQL** : création, import, déduplication, requêtes de contrôle.
2. **Fichiers CSV nettoyés** et archive ZIP.
3. **Fichier Power BI** (.pbix) avec le dashboard complet et interactif.
4. **Script Python** (daily\_quantity\_forecast.py) avec les modèles de prévision.
5. **Modèles enregistrés** (.pkl, .h5).

Cette solution a permis au client d’obtenir :

* Un environnement de reporting fiable et à jour.
* Des indicateurs de performance pour piloter l’activité.
* Une capacité prédictive sur le volume de commandes pour optimiser les stocks.