



ENSA KHOURIBGA

Big Data Warehouse NoSQL – Simulation de flux de commandes multicanal



Réalisé par :

Akilou Illa Abdourazak
Boumhand Yassine
Chaoui Khalil

Encadré par :

Pr. Mostafa Saadi

Filière : IID2



Sommaire

01	Introduction et Contexte	05	Modélisation NoSQL dans MongoDB
02	Architecture Générale du Système	06	Analyse Décisionnelle et Agrégations
03	Simulation de Production de Données	07	Reporting et Visualisation
04	Processus d'Intégration des Données	08	Conclusion et Perspectives

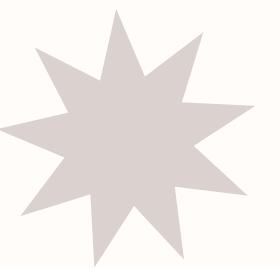
Introduction

CONTEXTE DU PROJET

- Développement rapide des technologies numériques
- Volumes de données massifs et variés
- Entreprises multicanales :
 - Site web
 - Application mobile
 - Boutiques physiques
- Besoin de collecter, stocker et analyser les commandes clients
- Limites des bases de données relationnelles classiques



PROBLÉMATIQUE



Défis des Bases Relationnelles :

- Rigidité du schéma
- Complexité des relations (jointures)
- Scalabilité limitée
- Difficulté avec données hétérogènes

Solution NoSQL :

- Flexibilité de modélisation
- Meilleure scalabilité
- Gestion de données semi-structurées (JSON)



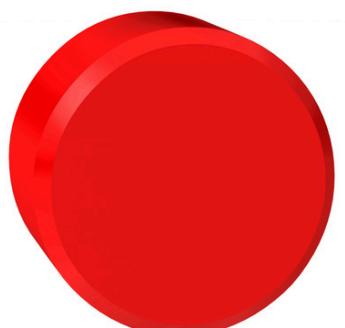
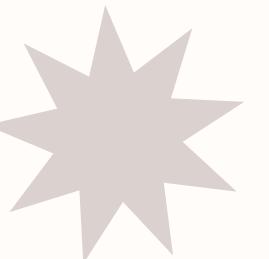
Qu'est-ce que NoSQL ?

DÉFINITION : NoSQL (Not Only SQL)

- Famille de bases de données non-relationnelles
- Pas de schéma fixe et prédéfini
- Adapté aux environnements Big Data

Types de bases NoSQL :

- Orientées documents (MongoDB)
- Clé-valeur (Redis)
- Orientées colonnes (Cassandra)
- Orientées graphes (Neo4j)



PRÉSENTATION DE MONGODB

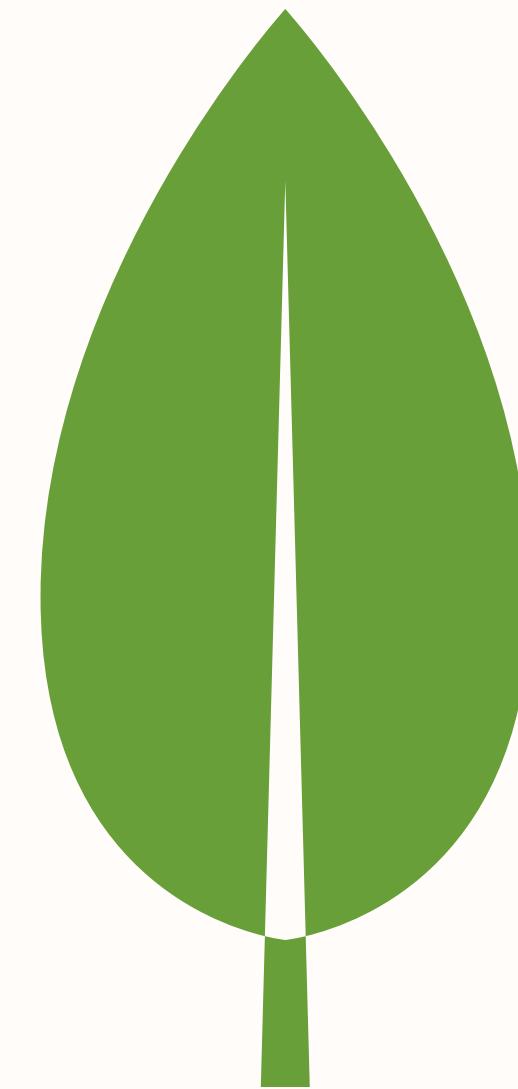
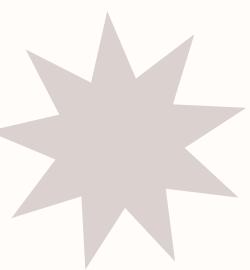
Base de données NoSQL orientée documents

Caractéristiques clés :

- Flexibilité du schéma
- Haute performance lecture/écriture
- Scalabilité horizontale
- Idéal pour Big Data

Types de bases NoSQL :

- Format BSON (Binary JSON)
- Collections au lieu de tables
- Documents au lieu de lignes
- Pas de jointures complexes

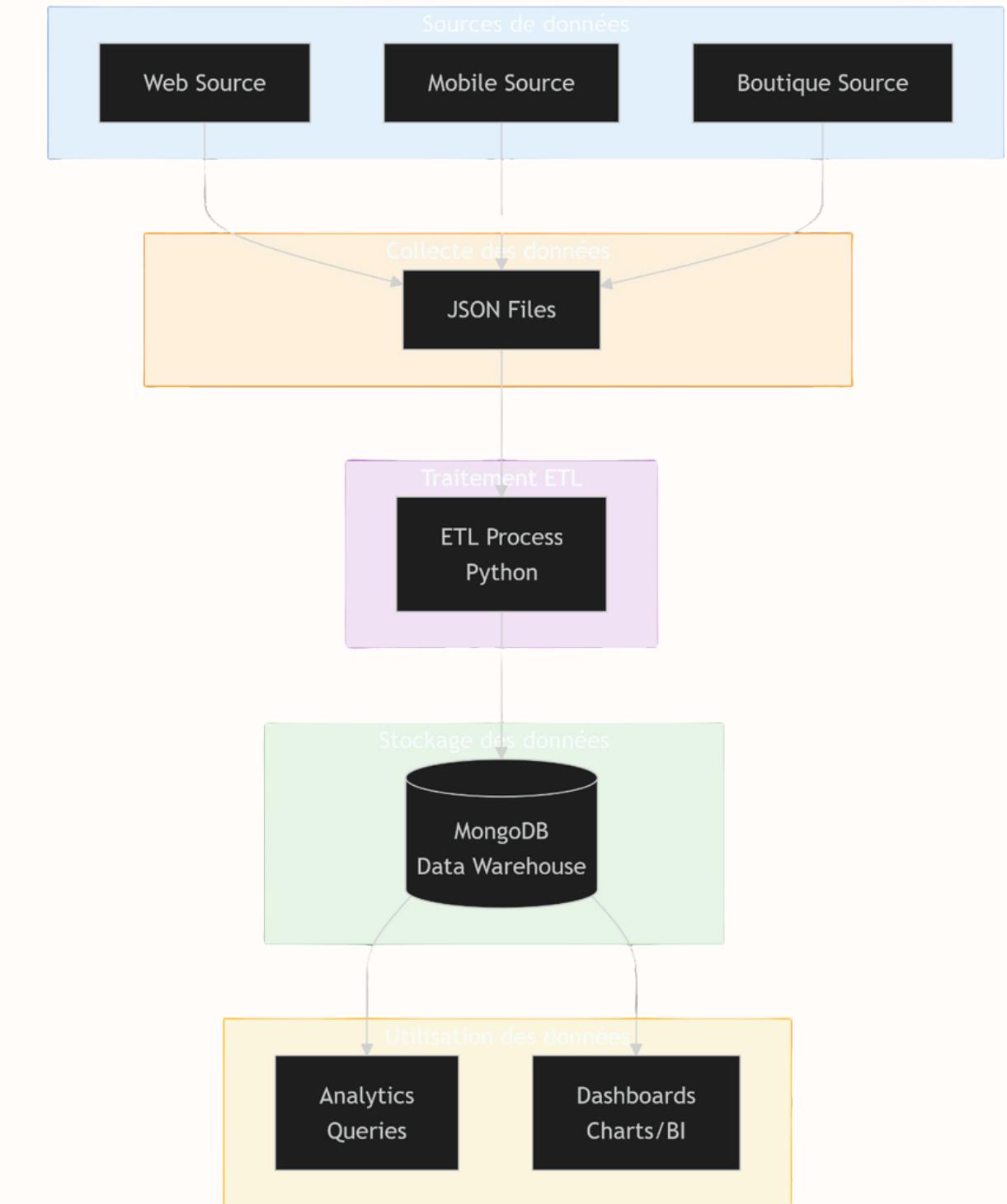
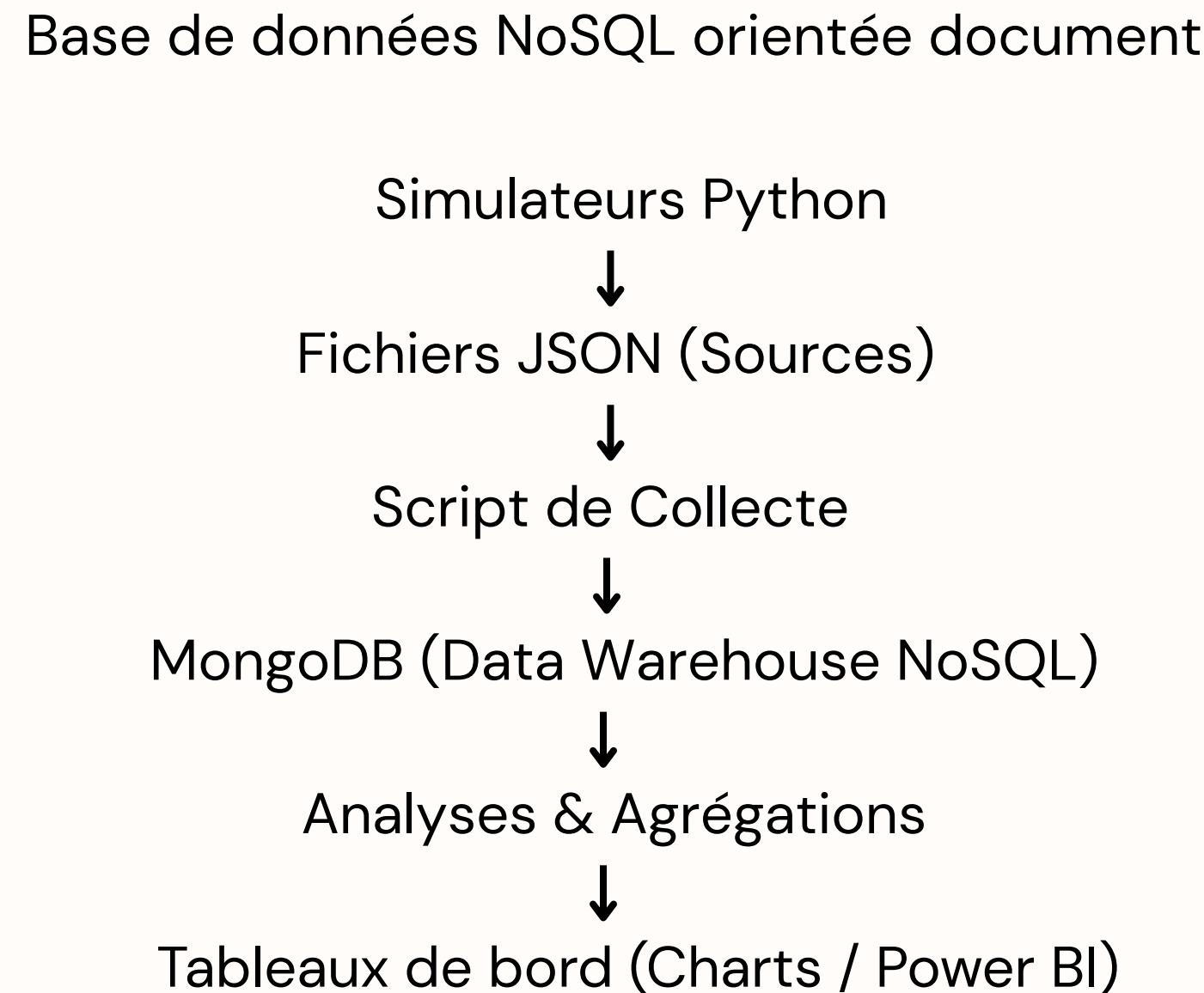


OBJECTIFS DU PROJET

- Simuler un environnement de vente multicanal
 - Collecter les commandes des différentes sources
 - Construire un Big Data Warehouse NoSQL avec MongoDB
 - Réaliser des analyses décisionnelles
 - Produire des tableaux de bord pour visualiser les KPI



ARCHITECTURE GÉNÉRALE DU SYSTÈME



LES TROIS SOURCES DE DONNÉES

1. Site Web

- Commandes en ligne
- Avec adresse de livraison

2. Application Mobile

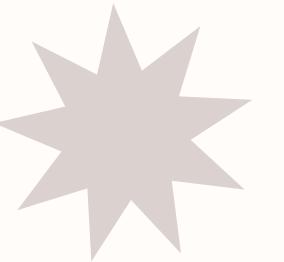
- Achats sur place
- Sans adresse de livraison

3. Boutique Physique

- Données de localisation
- Avec adresse de livraison



COMPOSANTS DE L'ARCHITECTURE



Couche de Collecte :

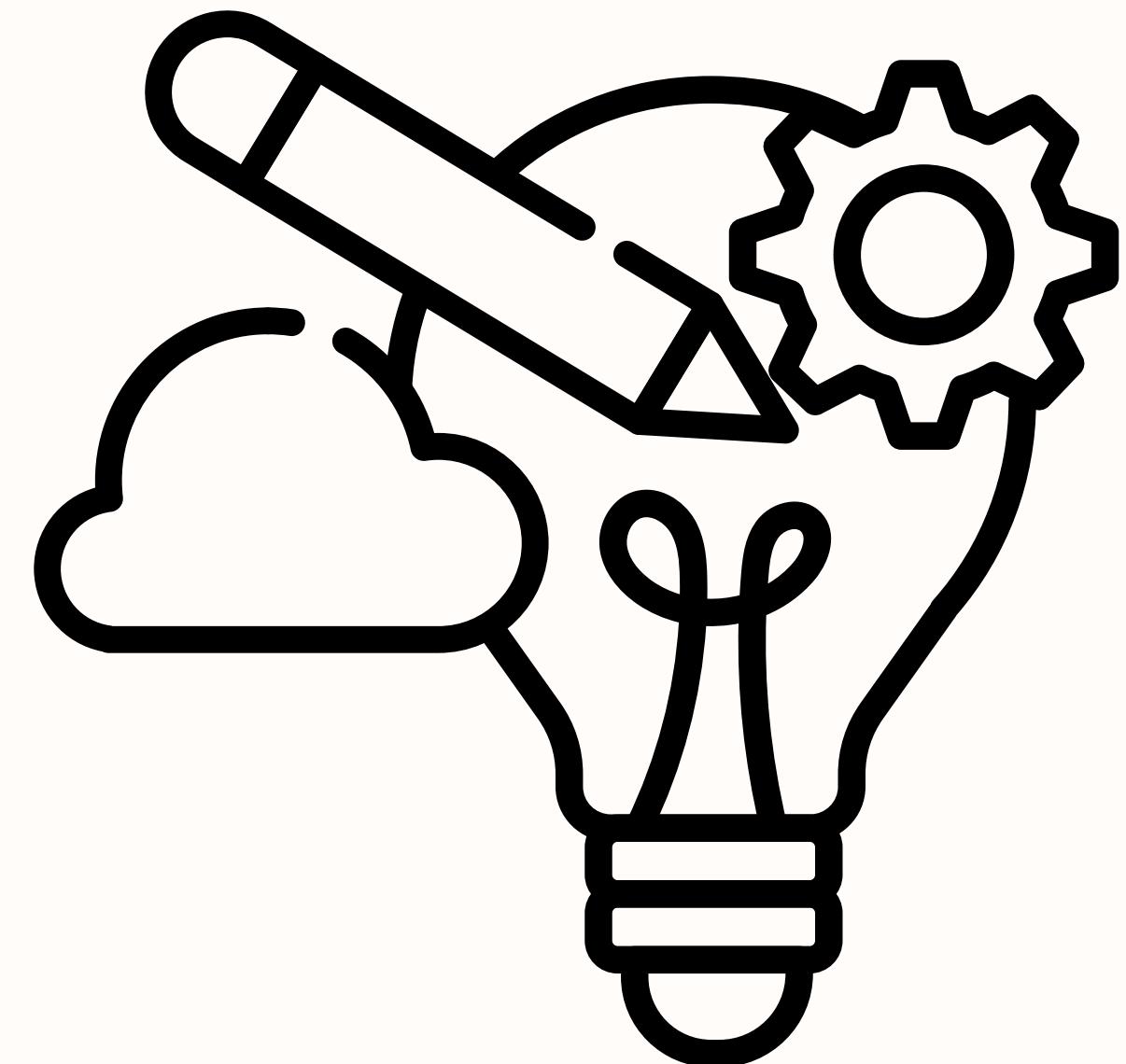
- Surveillance continue des répertoires
- Détection automatique de nouveaux fichiers
- Validation des données

Entrepôt NoSQL :

- MongoDB comme Data Warehouse centralisé
- Collection unique "commandes"

Couche d'Analyse :

- Framework d'agrégation MongoDB
- Calcul des indicateurs clés (KPI)



SIMULATION DE PRODUCTION DE DONNÉES

Objectif :

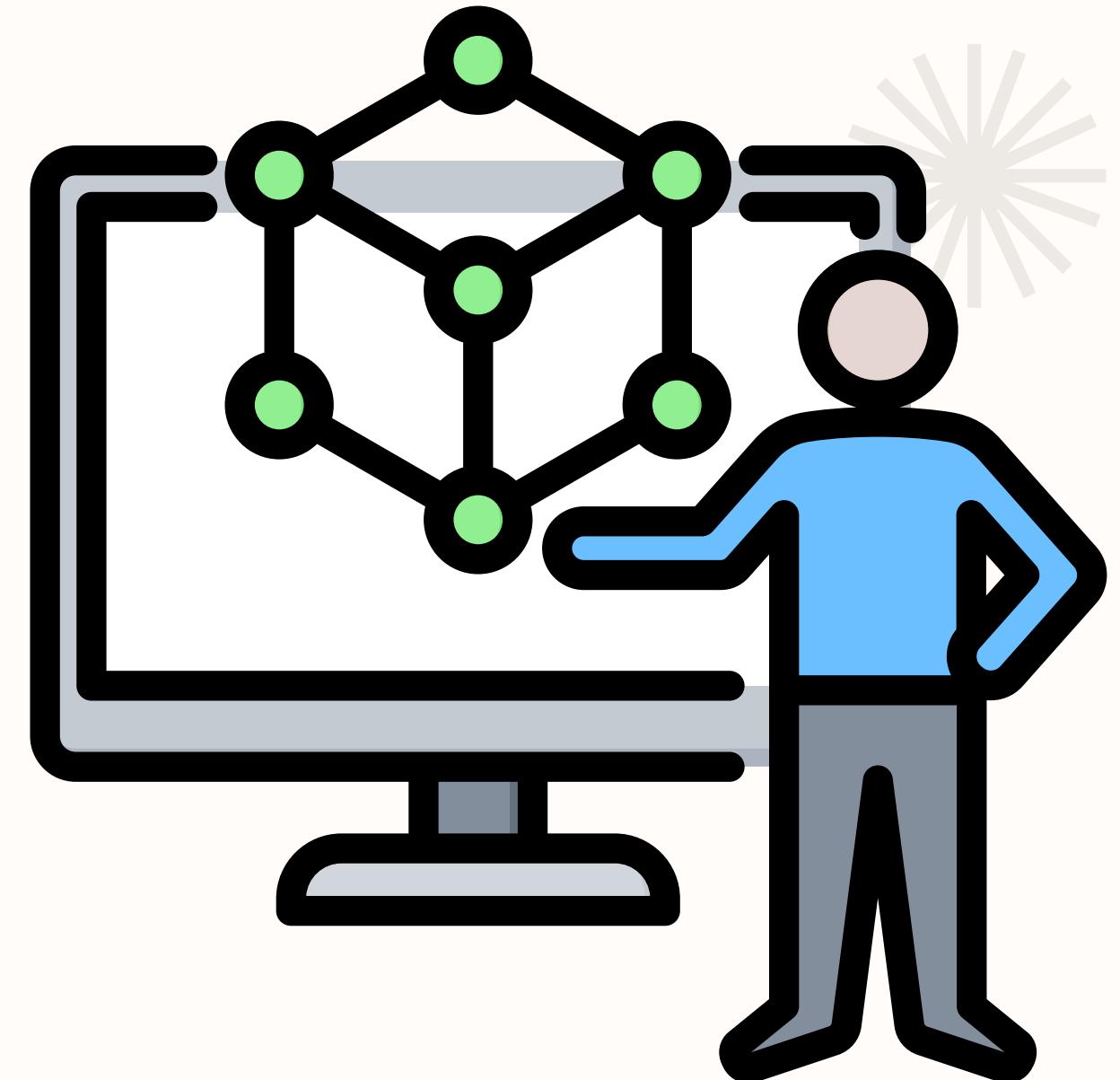
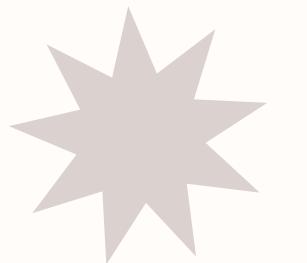
- Reproduire le comportement d'un système multicanal réel

Approche :

- 3 scripts Python indépendants
- Génération aléatoire de commandes
- Format JSON
- Flux continu (2 à 5 secondes)

Volume généré :

- 500+ commandes par source
- Total : 1500+ commandes



STRUCTURE D'UNE COMMANDE JSON



Informations essentielles :

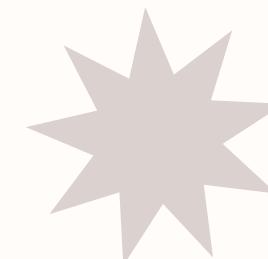
- id_commande : Identifiant unique
- date : Date et heure
- canal : Source (web/mobile/boutique)
- statut : confirmée ou annulée
- produits : Liste des articles
 - quantité, prix_unitaire
 - product_id, nom, catégorie
- montant_total : Total de la commande

```
1  {
2      "id_commande": "BOU-20251231160430-2125",
3      "canal": "boutique_physique",
4      "date_commande": "2025-12-31T16:04:30.245094",
5      "client": {
6          "nom": "Vincent Mercier",
7          "email": "tetienne@example.net",
8          "telephone": "+33 3 23 04 53 53"
9      },
10     "boutique": "Marrakech Gueliz",
11     "produits": [
12         {
13             "nom_produit": "Chargeur sans fil",
14             "quantite": 2,
15             "prix_unitaire": 35.0,
16             "prix_total": 70.0
17         },
18         {
19             "nom_produit": "Souris Gaming Razer",
20             "quantite": 3,
21             "prix_unitaire": 80.0,
22             "prix_total": 240.0
23         },
24         {
25             "nom_produit": "Sac à dos ordinateur",
26             "quantite": 1,
27             "prix_unitaire": 60.0,
28             "prix_total": 60.0
29         }
30     ],
31     "montant_total": 370.0,
32     "statut": "confirmée",
33     "mode_paiement": "especes",
34     "vendeur_id": "V682"
35 }
```

SCRIPTS PYTHON DE SIMULATION

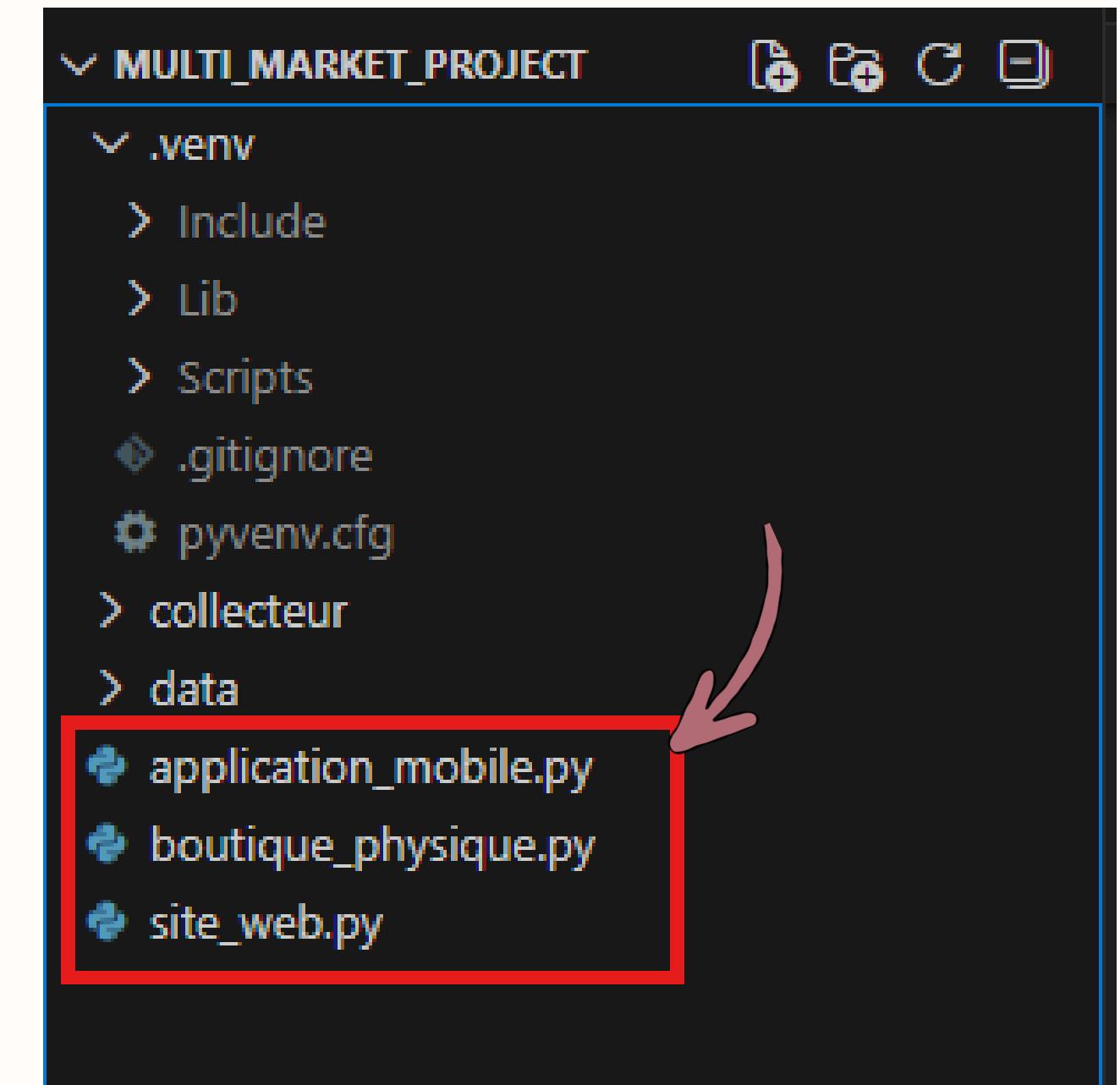
Trois scripts développés :

- *site_web.py*
- *application_mobile.py*
- *boutique_physique.py*



Fonctionnalités :

- Génération aléatoire de commandes
- Sauvegarde en fichiers JSON
- Nouvelle commande toutes les 2-5 secondes
- Stockage dans répertoires dédiés



STRUCTURE DES RÉPERTOIRES

```
/data/
└── sources/
    ├── site_web/
    ├── application_mobile/
    └── boutique_physique/
└── archive/
```



Avantages :

- Identification claire de l'origine
- Facilite la détection des nouveaux fichiers
- Organisation structurée

```
PROJETBINOSQL
└── collecteur
    └── collecteur.py
└── data
    ├── archive
    └── sources
        ├── application_mobile.py
        ├── boutique_physique.py
        └── site_web.py
```

PROCESSUS D'INTÉGRATION DES DONNÉES

Objectif :

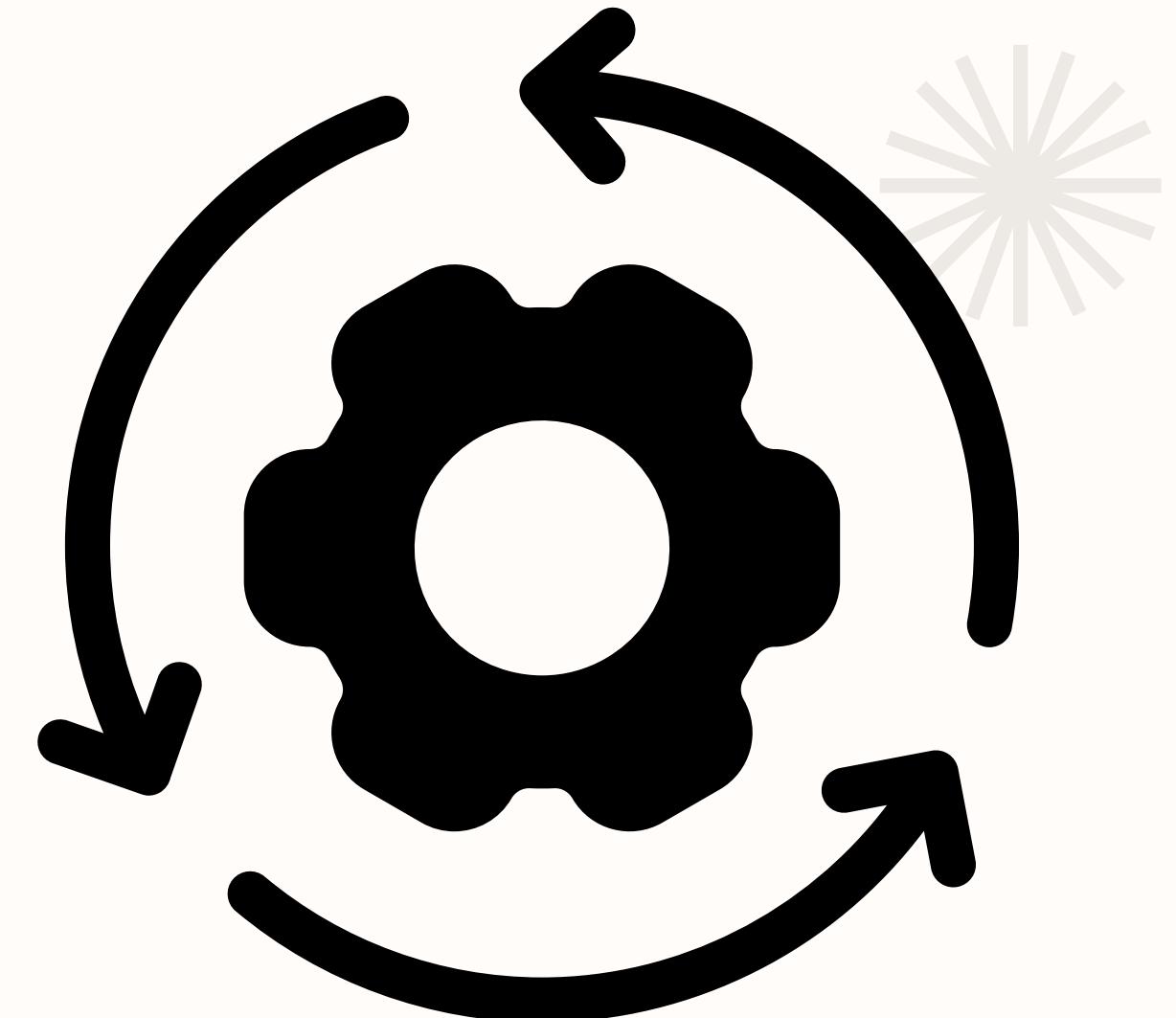
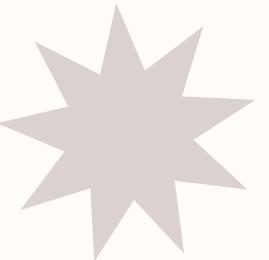
- Transformer les fichiers JSON en données exploitables

Étapes du processus :

- Surveillance des répertoires sources
- Validation des fichiers JSON
- Insertion dans MongoDB
- Archivage des fichiers traités

Fonctionnement :

- Quasi temps réel (< 10 secondes)
- Traitement automatique et continu



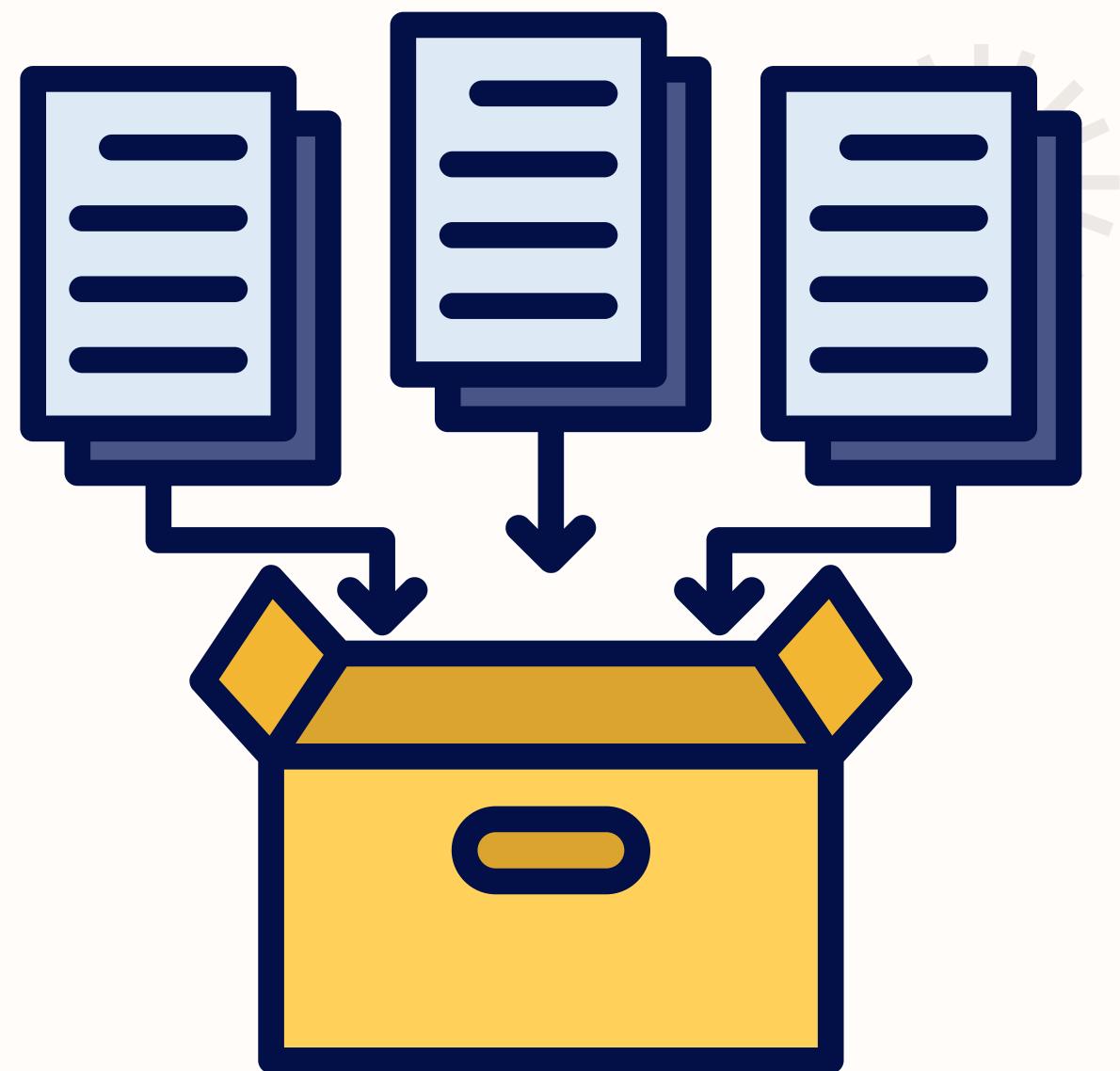
SCRIPT DE COLLECTE

Fonctionnalités :

- *Surveillance continue des 3 répertoires*
- *Détection automatique nouveaux fichiers*
- *Validation format JSON*
- *Robustesse (ignore fichiers corrompus)*

Traitement :

- Lecture du fichier JSON
- Validation de la structure
- Insertion dans MongoDB
- Déplacement vers archive



INTÉGRATION DANS MONGODB



Configuration :

- Base de données : *multi_market*
- Collection : *commandes*

Connexion PyMongo :

- Client MongoDB local
- Insertion document par document
- Gestion des erreurs
- Déplacement vers archive

Résultat :

- Centralisation de toutes les commandes dans un entrepôt unique

The screenshot shows the MongoDB Compass interface connected to 'localhost:27017'. The database list includes 'DemoDB', 'admin', 'config', 'local', and 'multi_market'. The 'commandes' collection under 'multi_market' is highlighted with a pink border.

- ▶ DemoDB
- ▶ admin
- ▶ config
- ▶ local
- ▶ multi_market
 - ▶ commandes

MODÉLISATION DES DONNÉES DANS MONGODB

Approche orientée documents :

- *Une commande = Un document unique*
- *Toutes les infos dans le même document*
- *Toutes les infos dans le même document*
- *Documents imbriqués (produits)*

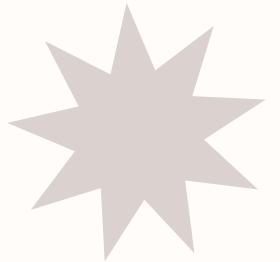
Avantages :

- Pas de jointures
- Requêtes simplifiées
- Meilleures performances
- Flexibilité du schéma

STRUCTURE D'UN DOCUMENT "COMMANDE"

Informations essentielles :

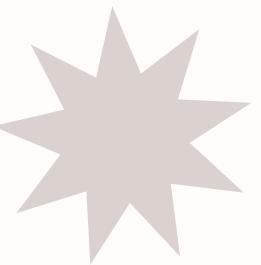
- id_commande : Identifiant unique
- date_commande : Date et heure
- canal : Source (web/mobile/boutique)
- statut : confirmée ou annulée
- produits : [] (tableau de produits)
- montant_total : Total de la commande



Documents imbriqués :

- Centralisation de toutes les commandes dans un entrepôt unique

FLEXIBILITÉ DU MODÈLE NoSQL



Différences entre canaux :

- *Web/Mobile* : Avec adresse de livraison
- *Boutique* : Sans adresse de livraison

Solution MongoDB :

- Champs optionnels gérés naturellement
- Pas de schéma rigide imposé
- Évolution facile du modèle

Comparaison avec relationnel :

- SQL : Nécessite plusieurs tables + jointures
- NoSQL : Document unique auto-suffisant

ANALYSE DÉCISIONNELLE



Objectif :

Transformer les données brutes en informations exploitables

Indicateurs clés (KPI) calculés :

1. Chiffre d'affaires par mois et par canal
2. Top 10 des produits les plus vendus
3. Taux de commandes annulées par canal
4. CA moyen par commande

Outil : Framework d'agrégation MongoDB

Principe du Pipeline

Suite d'étapes de transformation des données

Opérateurs utilisés :

- \$match : Filtrer les documents
- \$group : Regrouper et agréger
- \$project : Sélectionner/transformer champ
- \$sort : Trier les résultats

Principe

Équivalent SQL :

WHERE → GROUP BY → SELECT → ORDER BY

CHIFFRE D'AFFAIRES PAR MOIS ET PAR CANAL

Objectif :

- Calculer le CA mensuel pour chaque canal de vente

Méthode :

- Filtrer commandes Confirmé uniquement
- Extraire mois de la date
- Calculer montant total par produit
- Regrouper par mois + canal

Intérêt décisionnel :

- Identifier les canaux performants
- Suivre l'évolution temporelle
- Orienter la stratégie commerciale



```
$match > $addFields > $group > $project > $sort
```

RESULTS

```
chiffre_affaires : 7565
canal : "application_mobile"
mois : 12

chiffre_affaires : 39680
canal : "boutique_physique"
mois : 12

chiffre_affaires : 119425
canal : "site_web"
mois : 12
```

TOP 10 DES PRODUITS LES PLUS VENDUS

Objectif :

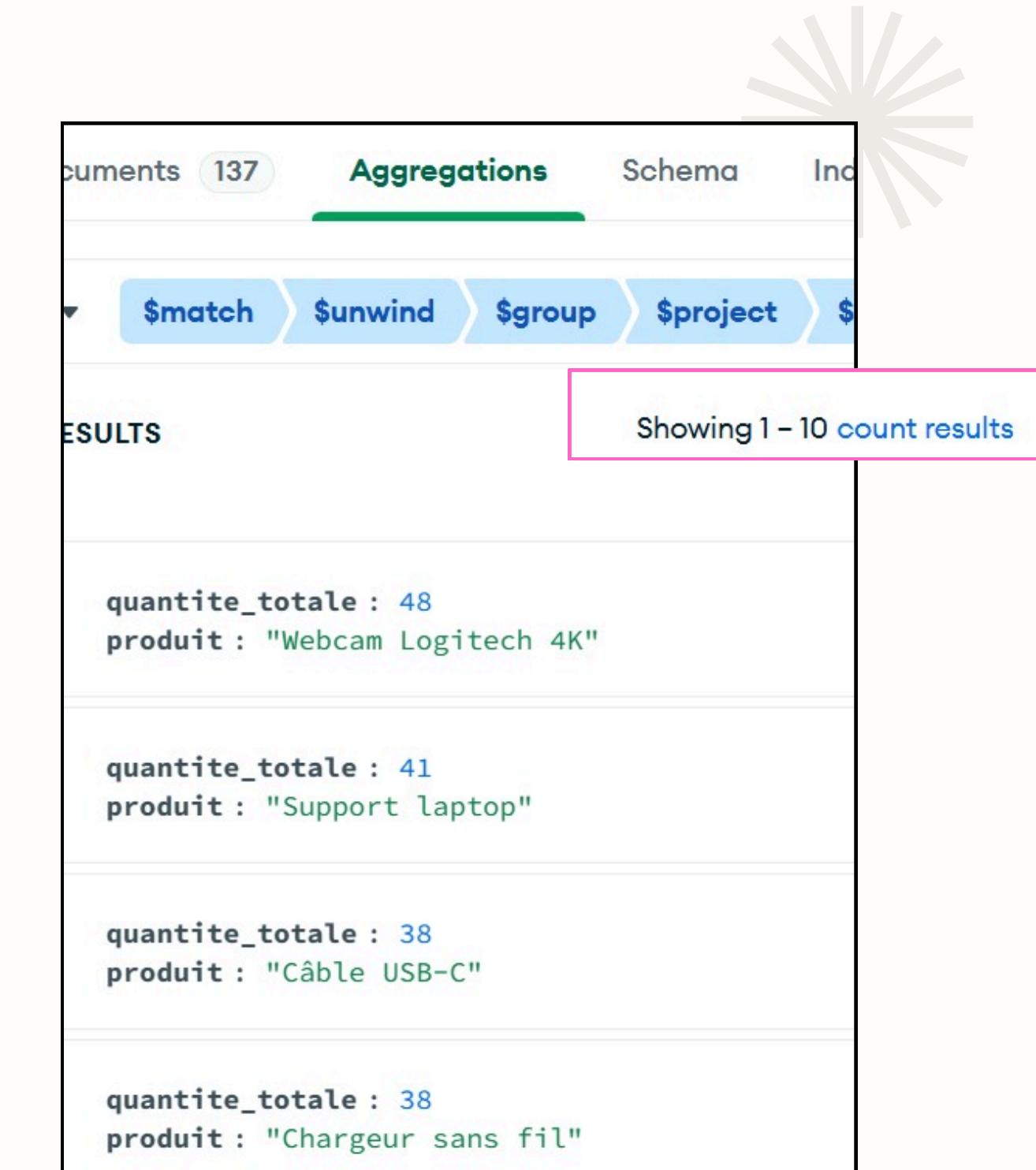
- Identifier les produits les plus populaires

Méthode :

- Décomposer tableau produits
- Regrouper par nom de produit
- Sommer les quantités vendues
- Trier par ordre décroissant
- Limiter aux 10 premiers

Utilité :

- Comprendre préférences clients
- Optimiser gestion des stocks
- Cibler actions marketing



The screenshot shows a MongoDB aggregation pipeline interface. At the top, there are tabs for 'Documents' (137), 'Aggregations' (selected), 'Schema', and 'Index'. Below the tabs is a pipeline stage editor with stages: '\$match', '\$unwind', '\$group', '\$project', and '\$sort'. A pink box highlights the '\$project' stage. To the right of the pipeline is a status bar with 'Showing 1 - 10 count results'. The results section displays four documents:

```
quantite_totale : 48
produit : "Webcam Logitech 4K"

quantite_totale : 41
produit : "Support laptop"

quantite_totale : 38
produit : "Câble USB-C"

quantite_totale : 38
produit : "Chargeur sans fil"
```

TAUX D'ANNULATION & CA MOYEN



Taux de commandes annulées par canal :

- *Ratio Annulé / Total commandes*
- *Par canal de vente*
- *Indicateur de qualité de service*

CA moyen par commande

- CA total / Nombre de commandes Confirmé
- Par canal
- Évalue la valeur moyenne des paniers

Apport décisionnel :

- Déetecter problèmes opérationnels
- Comparer comportements d'achat

REPORTING ET VISUALISATION

Objectif final :

- Présenter les résultats sous forme visuelle claire

Outils utilisés :

1. MongoDB Charts

- Outil natif MongoDB
- Crédit rapide de graphiques

2. Power BI

- Fonctionnalités avancées
- Tableaux de bord interactifs

Transformation ::

- Données brutes → Informations visuelles exploitables



TABLEAUX DE BORD RÉALISÉS

Avec MongoDB chart

Visualisations créées :

- CA total par mois et par canal
- Top 10 produits vendus
- Taux d'annulation par canal
- CA moyen par commande

Types de graphiques :

- 📊 Barres : CA par mois et par canal
- 📈 Lignes : Taux de commandes annulées
- 〽 Circulaires : top 10 des produits les plus vendus
- 📊 CA moyen par commande pour chaque canal



TABLEAUX DE BORD RÉALISÉS

Avec Power BI

Visualisations créées :

- CA total par mois et par canal
- Top 10 produits vendus
- Taux d'annulation par canal
- CA moyen par commande

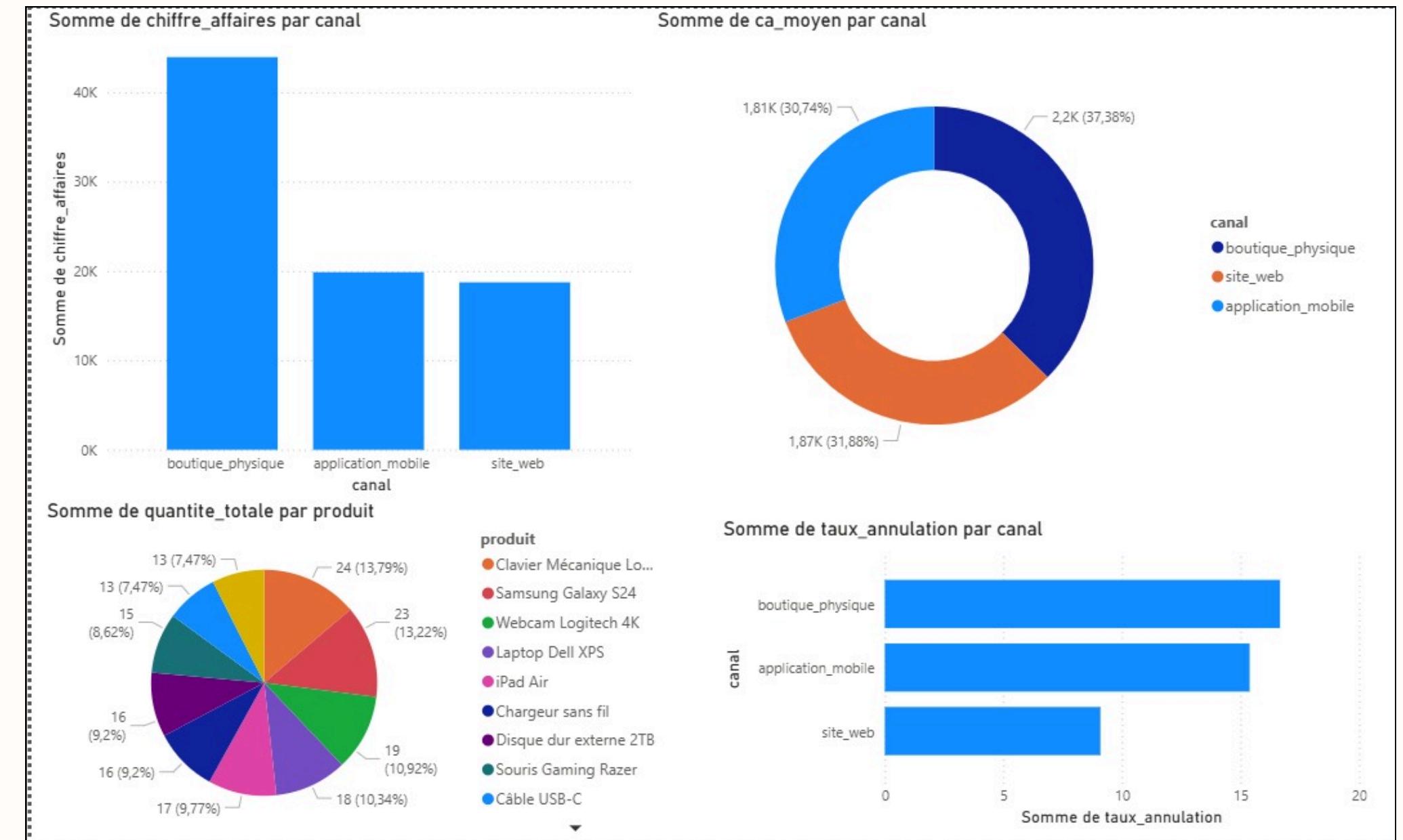
Types de graphiques :

📊 Barres : CA par mois et par canal

📈 Lignes : Taux de commandes annulées

🥧 Circulaires : top 10 des produits les plus vendus

📊 CA moyen par commande pour chaque canal



APPORTS DU PROJET



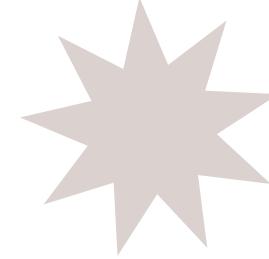
Compétences acquises :

- *Modélisation NoSQL orientée documents*
- *Gestion de flux de données continus*
- *Framework d'agrégation MongoDB*
- *Construction d'un Data Warehouse NoSQL*
- *Reporting et visualisation décisionnelle*

Résultats obtenus :

- Système fonctionnel multi-canaux
- 1500+ commandes intégrées
- 4 analyses décisionnelles majeures
- Tableaux de bord interactifs

CONCLUSION



MongoDB : Solution efficace pour Big Data Warehouse

- *Flexibilité de modélisation*
- *Performance des agrégations*
- *Gestion de données hétérogènes*
- *Scalabilité horizontale*

Perspectives d'évolution

- *Ajout de nouveaux canaux de vente*
- *Analyses prédictives (Machine Learning)*
- *Alertes automatiques sur KPI*
- *Dashboard temps réel*

Le NoSQL s'impose comme une technologie incontournable pour les systèmes décisionnels modernes

ENSA KHOURIBGA

Merci !