Projet Atelier Intégration des Données

Titres et Niveaux : M1 EISI / M1 CDPIA / M1 CYBER
Référence du module : TRDE703 Atelier Intégration des Données

Spécificité de cet exercice : Utilisation de données massives (Big Data)

Source de données : Openfoodfacts - Datalake (MongoDB) ou CSV
ETL : Apache Spark
Langage : Java ou Python (Spark est dispo sous Python avec PySpark)
Datawarehouse : BDD relationnelle SQL (MySQL, Postgresql…)

Objectif pédagogique

Thème : Construire un datamart “OpenFoodFacts Nutrition & Qualité” avec Apache Spark → MySQL

Mettre en place une solution ETL chargée de collecter les données depuis un Datalake ou une source de données massives permettant de répondre à une problématique métier exposée.

Contexte & objectifs

Par groupes de 2 à 3, vous allez mettre en place une chaîne d’intégration de données de bout en bout à partir des données OpenFoodFacts (OFF) :
collecte (bulk), qualité des données, modélisation en étoile et chargement dans un datamart MySQL pour l’analyse (SQL). L’ETL doit être Apache Spark (Java ou PySpark pour Python).

Vous êtes autorisés à utiliser ChatGPT. L’exigence technique et la rigueur de vos choix doivent être au niveau M1.

Données sources (OFF)

- Exports complets : JSONL/CSV, mis à jour quotidiennement. Utilisez un export complet pour le seed.
    - fr.openfoodfacts.org
    - static.openfoodfacts.org
- API de lecture (v1/v2/v3) : utile pour vérifications ponctuelles ou enrichissements ciblés (ex. quelques codes-barres).
- Champs et formats : OFF expose de nombreux champs (ex. last_modified_t, nutriscore_grade, nutriments.sugars_100g, *_tags, *_datetime).

- Taxonomies (catégories, additifs, allergènes, etc.) : endpoints dédiés pour référentiels multilingues.

Livrables attendus

- Repo (Git) structuré :
  - /docs (README, data-dictionary, schémas), /etl (code Spark), /sql (DDL/DML), /tests, /conf.
- Pipeline Spark reproductible : initial load (export complet). Log des métriques de qualité.
- Datamart MySQL (étoile ou flocon contrôlé) + scripts DDL/DML.
- Cahier de qualité : règles, coverage, anomalies, before/after.
- Jeu de requêtes analytiques (SQL) répondant à des questions métiers.
- Note d'architecture : choix techniques, schémas, stratégie d'upsert.

Périmètre fonctionnel (exemples de KPI)

- Répartition Nutri-Score par catégorie / marque / pays.
- Évolution (par date de modification) de la complétude des nutriments.
- Taux d'anomalies (valeurs hors bornes, incohérences d'unités).
- Classement des marques par "qualité nutritionnelle moyenne" (médiane sugars_100g, salt_100g, etc.).
- Top catégories avec le plus d'additifs en moyenne.

Architecture & contraintes

- Ingestion (Bronze) : lire JSON (ou CSV). Extractions des champs clefs (code, noms, nutriments 100g, tags…).
- Conformation (Silver) : normaliser types/units, flatten des structures, dédoublonnage par code-barres et choix de la langue (fr > en > fallback).
- Modélisation (Gold) : tables dimensionnelles + fact table(s).

Cible : MySQL 8 via Spark JDBC.

Qualité : produire des métriques (complétude, unicité, cohérence, référentiels) et rapports (CSV/JSON + tableau de bord SQL).

Modèle de données proposé (datamart)

Dimensions (extrait)

- dim_time(time_sk, date, year, month, day, week, iso_week)
- dim_brand(brand_sk, brand_name)

- dim_category(category_sk, category_code, category_name_fr, level, parent_category_sk)
- dim_country(country_sk, country_code, country_name_fr)
- dim_product
    - product_sk (PK), code (EAN-13… unique nat. key), product_name, brand_sk, primary_category_sk, countries_multi (JSON), effective_from, effective_to, is_current
- (Optionnel) dim_nutri(nutri_sk, nutriscore_grade, nova_group, ecoscore_grade)

Faits

- fact_nutrition_snapshot
    - fact_id (PK), product_sk, time_sk, (mesures pour 100g) energy_kcal_100g, fat_100g, saturated_fat_100g, sugars_100g, salt_100g, proteins_100g, fiber_100g, sodium_100g,
    - attributs : nutriscore_grade, nova_group, ecoscore_grade, completeness_score (0-1), quality_issues_json (JSON)
- (Optionnel) bridge_product_category(product_sk, category_sk) si vous conservez le N-N.

Note champs OFF : les mesures existent généralement sous nutriments.<nom>_100g et les scores/grades (nutriscore_grade, ecoscore_grade, nova_group) + horodatages last_modified_t.

Règles de qualité (exemples)

- Unicité : un code-barres ↔ un produit courant (is_current = 1).
- Complétude (pondérée) : présence de product_name, au moins N nutriments clés, au moins 1 catégorie normalisée, marque renseignée.
- Bornes : nutriments 100g dans des intervalles raisonnables (ex. 0 ≤ sugars_100g ≤ 100).
- Unités : harmoniser kcal/kJ, g/mg, sel/sodium (sel ≈ 2.5 × sodium).
- Taxonomies : rattacher catégories / additifs / allergènes via référentiels OFF.
- Langues : product_name_fr prioritaire si présent.

Requêtes analytiques à produire (SQL)

- Top 10 marques par proportion de produits Nutri-Score A/B.
- Distribution Nutri-Score par niveau 2 de catégorie.
- Heatmap (table) pays × catégorie : moyenne sugars_100g.
- Taux de complétude des nutriments par marque.
- Liste anomalies (ex. salt_100g > 25 ou sugars_100g > 80).
- Évolution hebdo de la complétude (via dim_time).

Exigences techniques (ETL Spark)

- Langage : Java ou Python (PySpark).
- Lecture JSON/CSV avec schéma explicite (pas d'inférence magique en prod).
- Nettoyage : trim/normalize, cast des types, unités, null-safe.
- Référentiels : charger taxonomies (catégories, additifs) et join en broadcast.
- Dédoublonnage : par code, garder l'enregistrement le plus récent (last_modified_t).
- SCD2 produit : comparer hash d'attributs suivis → fermer l'ancienne ligne (effective_to, is_current=0) et ouvrir la nouvelle.
- Chargement MySQL : via JDBC (batch size, truncate-insert pour petites dimensions, upsert via INSERT … ON DUPLICATE KEY UPDATE pour bridge et faits).
- Métriques : publier un JSON par run (nb produits lus, filtrés, créés/mis à jour, % complétude, anomalies par règle).

Règles d'évaluation (100 pts)

- Collecte & incrémental (20) : bulk, idempotence.
- Qualité & métriques (20) : règles solides, reporting clair, anomalies expliquées.
- Modèles Datamarts (20) : étoiles cohérentes, clés, index.
- ETL Spark (25) : code clair/testé, perfs (partitionnement, broadcast), upserts maîtrisés.
- Analytique SQL (10) : requêtes pertinentes, résultats commentés.
- Docs & reproductibilité (5) : README, schémas, how-to run, journal des prompts.

Pistes "bonus" (pour aller plus loin)

- Conformité multilingue : résolution des noms produit/catégorie par priorité de langue.
- Détection d'anomalies (exemple : IQR) sur nutriments.
- Petit dashboard (connecté à MySQL).
- Historisation.