Rapport

Projet Job Market Research

Pipeline de collecte, traitement et visualisation des données du marché de l'emploi Data

Réalisé par :

- Souhail Khalmadani
- Houssam Ettakifi
- Youness Azroul
- Ouassim Moussaoui
- Meryem El Fatine
- Hiba Asghar
- Souleimane Bentalha

Encadré par : Mr Said Abouks

VERSIONS DU DOCUMENT

Version	**Contenu de la version**	**Date de mise à jour / Auteur**
V0.1	Initialisation du document	04/08/2025 - MEL

Table des matières

1 Ingestion de Données

L'ingestion de données constitue la première étape du pipeline et consiste à collecter, organiser et stocker les données brutes provenant de plusieurs sources...

1.1 Collecte automatisée des données

La collecte est réalisée via un processus de web scraping dynamique développé en `Python` à l'aide de `Selenium`. Cet outil a été choisi pour sa capacité à interagir avec des pages web complexes utilisant du contenu chargé de manière asynchrone (JavaScript, scroll infini, pagination).

Sources principales ciblées :

— Emploi.ma

— Rekrute

— Bayt

— Maroc Annonces

En complément, l'API `Corsignal` est intégrée pour récupérer des annonces structurées et actualisées.

Champs extraits pour chaque annonce :

— Titre du poste

— Nom de l'entreprise

— Localisation (ville, région, pays)

— Type de contrat

— Date de publication

— Description complète

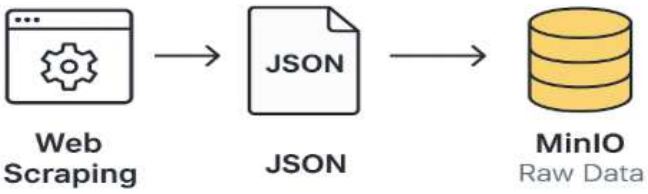
— Compétences mentionnées

— Secteur d'activité

— Niveau d'études et expérience requis

— URL de l'annonce

La collecte est exécutée de manière planifiée et intègre des délais aléatoires pour limiter la détection par les systèmes anti-scraping.



1.2 Stockage brut dans MinIO

Les données extraites sont sauvegardées au format `JSON` structuré, selon un schéma commun garantissant l'homogénéité entre toutes les sources.

Le stockage repose sur `MinIO`, un système de stockage objet compatible S3, pour sa scalabilité, sa fiabilité et sa capacité à séparer les données brutes des données enrichies.

Organisation des buckets :

— `webscraping/` : Données brutes issues du scraping.

— `traitement/` : Données enrichies prêtes pour le traitement analytique.

Structure de rangement :

```
/webscraping/2025-03-15/emploi_ma.json
/webscraping/2025-03-15/rekrute.json
...
```

Cette organisation facilite la traçabilité, la reprise sur erreur et la réutilisation des données.

1.3 Prétraitement initial

Avant stockage définitif, un prétraitement léger est appliqué :

— Suppression des doublons (basée sur l'URL et les métadonnées principales).

— Filtrage des annonces incomplètes ou hors périmètre.

— Harmonisation minimale des formats (dates, encodage UTF-8).

Ce nettoyage précoce garantit la qualité minimale requise avant les étapes avancées d'enrichissement.

1.4 Exemple JSON brut (Annexe)

En annexe, un exemple de fichier `JSON` brut est fourni, illustrant la structure type d'une annonce après extraction :


```
{
  "title": "Data Analyst",
  "company": "ABC Consulting",
  "location": "Casablanca, Maroc",
  "contract_type": "CDI",
  "publication_date": "2025-03-15",
  "description": "Analyse des données clients et création de dashboards Power BI."
  "skills": ["Python", "SQL", "Power BI"],
  "sector": "IT",
  "education_level": "Bac+5",
  "experience": "2 ans",
  "job_url": "https://emploi.ma/offre/data-analyst-12345",
  "source": "Emploi.ma"
}
```


1.5 Capture d'écran MinIO (Annexe)

En annexe, une capture d'écran de l'interface `MinIO` montre l'organisation hiérarchique des dossiers et buckets, confirmant la séparation entre données brutes (`webscraping`) et données enrichies (`traitement`).

Outils utilisés :

— Python 3.9+

— Selenium (web scraping dynamique)

— JSON (format standardisé d'échange de données)

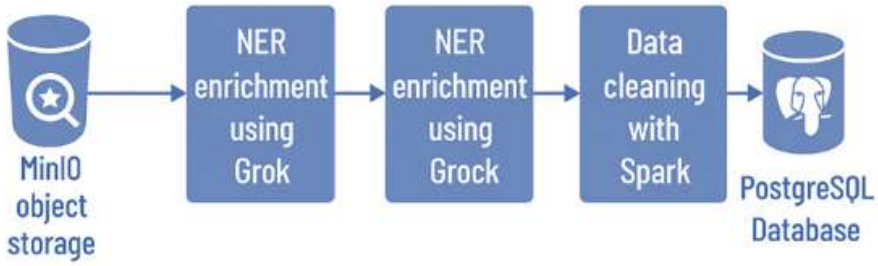
— MinIO (stockage objet S3-compatible)

2 Traitement et Enrichissement des Données

Architecture du Pipeline de Traitement

Après l'étape d'extraction des données via web scraping, le processus entre dans une phase critique de traitement et d'enrichissement. Cette partie du pipeline vise à transformer les données brutes extraites en informations structurées et exploitables, prêtes pour l'analyse et la visualisation. Le traitement s'articule autour de

trois composants principaux : l'enrichissement par reconnaissance d'entités nommées (NER), le nettoyage et formatage des données, et leur préparation en format tabulaire.



Pipeline de traitement des offres d'emploi

2.1 Enrichissement par NER

2.1.1 Skillner (Étape 1)

La première phase d'enrichissement utilise `Skillner`, une bibliothèque `Python` spécialisée dans l'extraction de compétences à partir de textes d'offres d'emploi, basée sur `spaCy` et une base de données de compétences prédéfinie.

Architecture technique :

- **Modèle NLP** : `spaCy` avec le modèle `en_core_web_lg`
- **Matching** : `PhraseMatcher` pour l'identification de patterns
- **Base de données** : `SKILL_DB` contenant les compétences et leurs types
- **Stockage** : Sauvegarde automatique vers `MinIO`

2.1.2 Grok (Étape 2)

La seconde phase utilise `Grok`, le modèle de langage développé par `xAI`, pour classifier automatiquement les profils data (Analyst, Engineer, Scientist, etc.) après l'extraction initiale par Skillner.

Entraîné sur un jeu de données spécialisé de descriptions d'offres étiquetées selon les différents métiers de la donnée, Grok constitue un modèle de langage avancé adapté à la classification de profils data.

Le classifieur prend en entrée le titre et la description. Il renvoie le profil le plus probable avec un score de confiance et des alternatives.

Cette classification permet d'agréger les tendances par profil métier et de mieux comprendre la répartition des besoins en compétences data sur le marché de l'emploi marocain.

2.2 Nettoyage et Formatage avec Apache Spark

2.2.1 Choix technologique

`Apache Spark` a été sélectionné pour cette étape critique en raison de ses capacités de traitement distribué et de sa robustesse dans la manipulation de grands volumes de données.

Avantages de Spark :

— Traitement parallèle et distribué

— Optimisations automatiques des requêtes

— Gestion efficace de la mémoire

— Écosystème riche de bibliothèques de traitement

2.2.2 Opérations de nettoyage

Validation des champs obligatoires :

— Présence des champs critiques : `job_url`, `titre`, `via`, `publication_date`

— Filtrage des enregistrements avec valeurs nulles ou vides

Déduplication :

— Suppression des doublons basée sur `job_url`

— Garantie d'unicité des données traitées

Normalisation des formats :

— Dates : formats multiples supportés

— Compétences : transformation en format plat avec type

— Secteurs : séparation des valeurs multiples

— Texte : nettoyage (trim) et typage `StringType`

2.2.3 Formatage et structuration

La phase de formatage prépare les données pour leur intégration dans `PostgreSQL`. Les fichiers enrichis sont lus depuis le bucket MinIO `ner` via le protocole `S3A`, avec exclusion automatique des fichiers de taille inférieure à 10 octets.

Transformations appliquées :

— Validation via schéma global prédéfini

— Renommage systématique des colonnes
— Aplatissement des compétences via `UDF`
— Conversion en tableaux (`arrays`) des champs multi-valeurs
— Valeur `"Unspecified"` pour les champs manquants

Optimisations PostgreSQL :

— JSON format compatible avec le système de chargement
— Nom de fichier unique avec timestamp + UUID

2.3 Préparation Tabulaire avec PipelineLoader

2.3.1 Architecture en étoile (Star Schema)

Le `PipelineLoader` transforme les fichiers `JSON` nettoyés vers un modèle dimensionnel en étoile comprenant des tables de dimensions et une table de faits centrale.

Tables et vues

Rechercher dans les tables (10)

Rechercher

Table	Moteur	Interclassement	Longueur des données²	Longueur de l'index²
dim_compagnie	table		16,384	32,768
dim_contrat	table		49,152	32,768
dim_date	table		8,192	32,768
dim_niveau_etudes	table		49,152	32,768
dim_niveau_experience	table		49,152	32,768
dim_skill	table		49,152	32,768
dim_source	table		49,152	32,768
dim_titre	table		16,384	32,768
fact_offre	table		65,536	114,688
offre_skill	table		40,960	16,384
10 au total		en_US.utf8	393,216	393,216

2.3.2 Processus de chargement dimensionnel

Stratégie ETL :

— **Extract** : Depuis le bucket MinIO `traitement`
— **Transform** : Normalisation, création de clés
— **Load** : Insertion avec contraintes référentielles

Gestion des dimensions :

— SCD Type 1 : Insertion ou mise à jour des dimensions

— Intégrité référentielle assurée

Optimisations :

— Index sur clés étrangères

— Contraintes `CHECK` sur types de compétences

— Clés primaires composées pour les tables de liaison

3 Contrôle Qualité et Monitoring

Le pipeline intègre des mécanismes de qualité et de monitoring pour garantir la fiabilité des données.

Métriques de qualité

— Filtrage automatique des fichiers valides

— Comptage avant/après nettoyage

— Validation des champs obligatoires

— Traçabilité via noms de fichiers uniques

— Sauvegarde intermédiaire à chaque étape

Gestion des erreurs

— Reprise spécialisée : `Skillner` (continuation), `Grok` (retry avec backoff)

— Validation des schémas dans Spark

— Contraintes d'intégrité et rollback PostgreSQL

— Logs rotatifs multi-niveaux avec messages explicites

4 Orchestration des Tâches Asynchrones

L'orchestration est assurée par `Celery` (gestion de tâches) et `Redis` (broker de messages) pour une exécution distribuée.

Composants clés :

— **Redis** : Faible latence, haute disponibilité, gestion `ACK/NACK`

— **Celery** : Exécute le scraping, nettoyage, enrichissement, insertion

— **Flower** : Interface de monitoring des workers

— **Redis Commander** : Inspection manuelle des files Redis

Fonctionnalités Celery :

— Retry automatique

— `Rate limiting`

— Planification avec `Celery Beat`

5 Conteneurisation et Déploiement

La conteneurisation avec `Docker` isole les services et simplifie le déploiement.

Services conteneurisés

— **Scraper** : Image `Python` avec `Selenium` + `ChromeDriver`

— **PySpark** : Image avec `OpenJDK`, `PySpark`, accès MinIO

— **PostgreSQL** : Initialisation via `docker-entrypoint-initdb.d`

— **Celery & Redis** : Workers dédiés à chaque tâche

— **MinIO** : Stockage S3-compatible des données brutes/enrichies

6 Visualisation des Données

6.1 Objectifs

— Fournir une interface analytique intuitive pour explorer les compétences demandées dans le domaine de l'Intelligence Artificielle (IA) et du Big Data.

— Permettre aux décideurs (recruteurs, formateurs, analystes RH) d'identifier les tendances technologiques, les profils recherchés, et les besoins émergents du marché.

— Supporter l'analyse exploratoire grâce à des filtres interactifs et des visualisations adaptées à différents angles d'analyse.

6.2 Superset

Pourquoi Superset ?

Outil *open-source*, performant, supportant `PostgreSQL`, adapté aux pipelines conteneurisés (`Docker`) et facilement personnalisable (`CSS`, thèmes, filtres dynamiques).

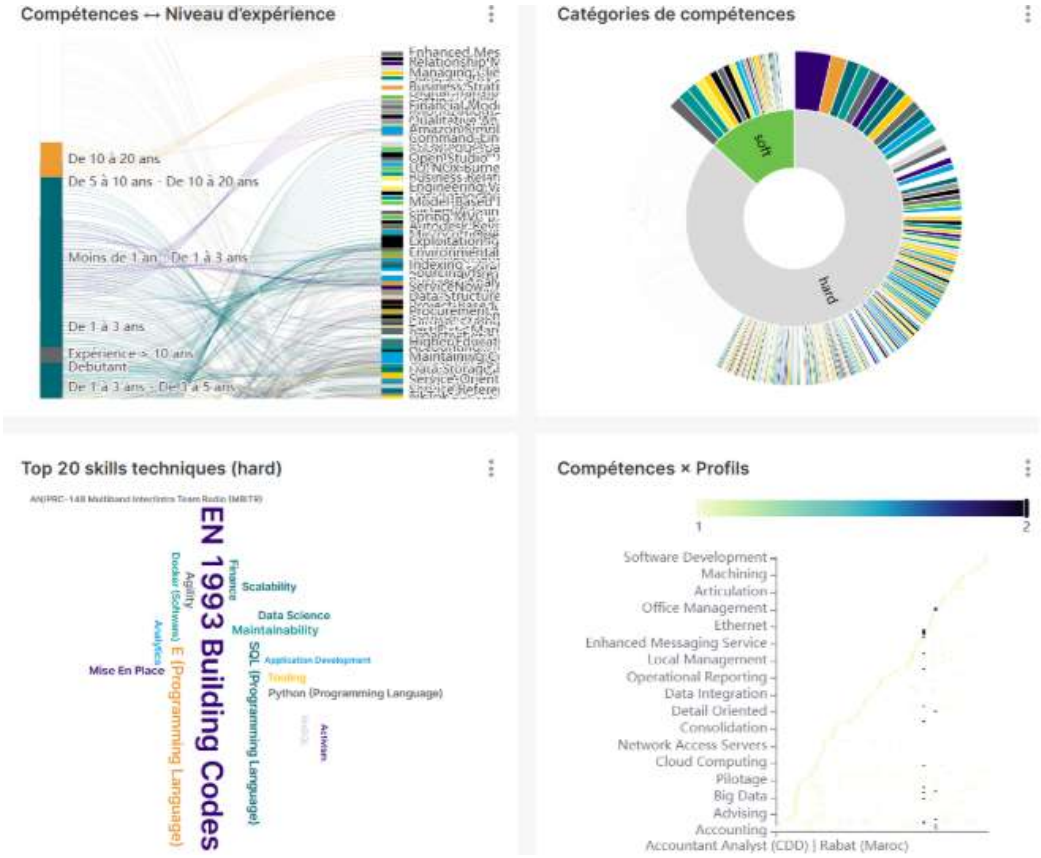
Connexion à la base analytique PostgreSQL :

Connexion directe à la base analytique (modèle en étoile) construite à partir du traitement Spark + PipelineLoader.

Mode d'intégration :

Superset est conteneurisé via un service `Docker` dédié (`superset`) et configuré via `superset_config.py` pour supporter les fonctionnalités modernes (filtres natifs, `cross-filters`, palette personnalisée...).

6.3 Dashboards



Cartographie Compétences Data

Ce dashboard a pour objectif de cartographier les compétences recherchées dans le domaine de la donnée selon les profils, les niveaux d'expérience et les catégories. Il s'appuie sur des données enrichies par NER (Skillner) et classification de profils (Grok).

Visualisations principales :

— Word Cloud – Top 20 skills techniques (hard)

— Heatmap – Compétences × Profils

— Sankey Diagram – Compétences Niveau d'expérience

— Sunburst – Catégories de compétences (hiérarchie)

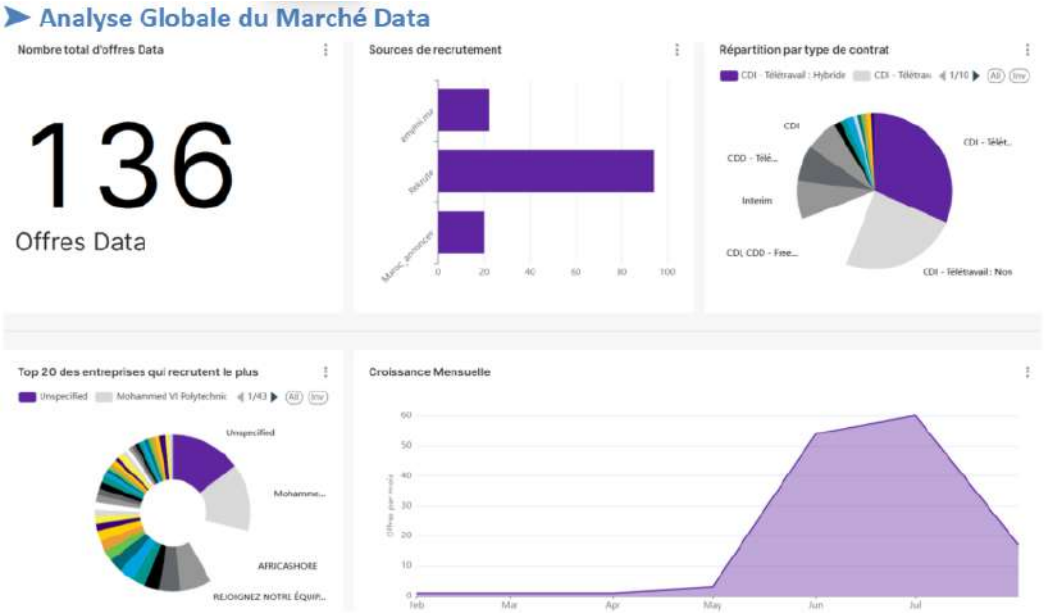
Filtres interactifs utilisés :

— Sélecteur Profil Data

— Slider Niveau d'expérience

— Multi-select Secteurs

Analyse Globale du Marché Data



Ce dashboard fournit une vision macro du marché de l'emploi dans les métiers de la data, avec un suivi des volumes, sources, types de contrats et employeurs majeurs.

Visualisations principales :

— Nombre total d'offres data – Big Number

— Sources de recrutement – Bar Chart

— Répartition par type de contrat – Pie Chart

— Top 20 des entreprises qui recrutent le plus – Pie Chart

— Croissance mensuelle – Line Chart

Filtres utilisés :

— Source

— Type de contrat

— Date

— Niveau d'expérience

— Entreprise

6.4 Design

— Utilisation d'un thème de couleur personnalisé `"DxC"` via `superset_config.py`.

— Activation de fonctionnalités avancées : `DASHBOARD_NATIVE_FILTERS`, `ENABLE_ECHARTS`, `GENERIC_CHART_AXES`, pour améliorer l'expérience utilisateur.

6.5 Résultats et interprétations

— Visualisation dynamique et modulaire intégrée au pipeline existant.

— Extensible pour comparer l'évolution temporelle des compétences ou géolocaliser les offres.

— Ouvre la voie à la mise en place d'un observatoire du marché de l'emploi data.

7 Conclusion

Le pipeline présenté dans ce document illustre une architecture complète et modulaire pour la collecte, le traitement, l'enrichissement et la valorisation des données du marché de l'emploi dans les métiers de la donnée.

Chaque composant a été soigneusement sélectionné pour répondre à des exigences de robustesse, de scalabilité et de maintenabilité : depuis le scraping dynamique avec Selenium jusqu'au traitement distribué avec Apache Spark, en passant par l'enrichissement sémantique via Skillner et Grok. Le stockage structuré dans MinIO, l'organisation tabulaire en étoile avec PostgreSQL et l'orchestration des tâches avec Celery garantissent une chaîne de traitement fluide, automatisée et traçable.

La visualisation finale avec Superset permet de transformer cette richesse informationnelle en outils décisionnels accessibles, offrant des perspectives précieuses sur les tendances du marché, les compétences recherchées et l'évolution des besoins.

Ce système constitue un socle solide pour la mise en place d'un observatoire dynamique du marché de l'emploi, et peut être facilement étendu ou adapté à d'autres domaines sectoriels. Il combine des choix technologiques modernes et open-source avec une approche analytique centrée sur la qualité et l'interprétabilité des données.

Annexes

Annexe A : Exemple de fichier JSON brut


```
{
  "title": "Data Analyst",
  "company": "ABC Consulting",
  "location": "Casablanca, Maroc",
  "contract_type": "CDI",
  "publication_date": "2025-03-15",
  "description": "Analyse des données clients et création de dashboards Power BI."
  "skills": ["Python", "SQL", "Power BI"],
  "sector": "IT",
  "education_level": "Bac+5",
  "experience": "2 ans",
  "job_url": "https://emploi.ma/offre/data-analyst-12345",
  "source": "Emploi.ma"
}
```

Annexe B : Capture d'écran MinIO