Plateforme d'intelligence du marché de l'emploi (Data/AI)

Made by: Malak ASSABBAR
Essohanam Jouse KPATCHA

Supervised by: Pr. A. EL HADDADI

National School of Applied Sciences in Al Hoceima (ENSAH)
Année universitaire: 2025/2026
Janvier 2026

Table des matières

Chapitre 1

Introduction

1.1 Contexte et problématique

Le recrutement dans les métiers de la donnée (*Data Analyst*, *Data Engineer*, *Data Scientist*, *ML Engineer*) repose sur des signaux hétérogènes : intitulés variables, descriptions longues, compétences implicites, et sources multiples. L'objectif de Plateforme d'intelligence du marché de l'emploi (Data/AI) est d'automatiser la collecte, la structuration, puis l'analyse de ces informations afin de :

— mesurer la demande du marché (volumétrie, entreprises, localisation),
— identifier les compétences les plus requises et leurs tendances,
— proposer un système de recommandation d'offres en fonction d'un profil candidat,
— fournir des dashboards interactifs via une solution libre et auto-hébergée.

1.2 Objectifs du projet

Le projet a été réalisé comme un pipeline de bout en bout, organisé en phases :

1. Ingestion multi-sources (scraping) des offres ciblées Data/ML.
2. Nettoyage, harmonisation, déduplication, normalisation des champs.
3. Extraction de compétences (approche hybride regex + sémantique).
4. Modélisation et chargement dans Snowflake (schéma en étoile).
5. Data mart BI : vues optimisées pour exploration et agrégations.
6. Visualisation et reporting : dashboards dans Apache Superset (Docker).

1.3 Livrables

Les livrables produits incluent :

— des fichiers CSV d'export (données brutes et nettoyées),
— un schéma Snowflake (dimensions et faits) et des vues BI (VW_),
— un moteur de recommandation (similarité sémantique + matching compétences),
— un environnement Superset prêt à l'emploi (docker-compose),
— un ensemble de dashboards illustrés (captures intégrées au rapport).

Chapitre 2

Architecture générale

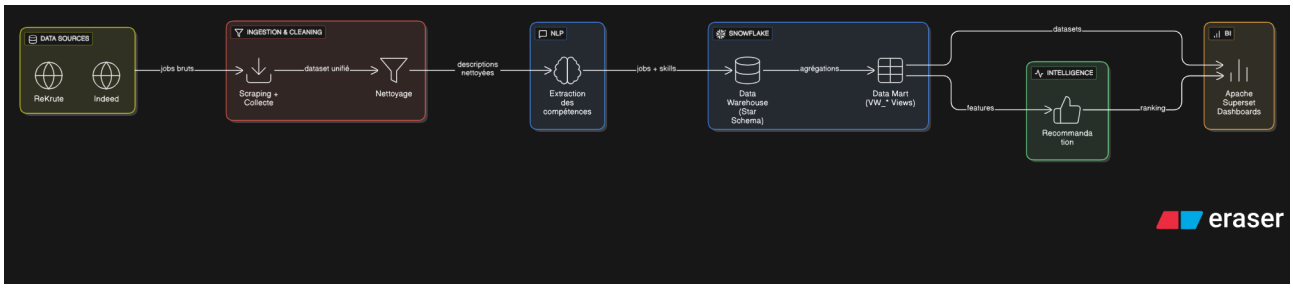

FIGURE 2.1 – Architecture du Projet

2.1 Vue d'ensemble

La plateforme est structurée en modules Python spécialisés et scripts SQL. L'approche privilégiée est un pipeline reproductible : chaque phase produit des artefacts (CSV, tables Snowflake, vues) consommés par la phase suivante.

2.2 Organisation du dépôt

Les principaux répertoires sont :

— **src/ingestion** : scrapers ReKrute et Indeed.
— **src/processing** : nettoyage et normalisation.
— **src/nlp** : extraction de compétences.
— **src/database** : chargement Snowflake.
— **src/recommandation** : moteur de recommandation.
— **scripts/** : schéma Snowflake et data mart (vues).
— **docker-compose.yml** : orchestration Superset + Postgres + Redis.

2.3 Flux de données

Le flux principal est :

1. Scraping → exports bruts (`indeed_jobs.csv`, `rekrute_jobs.csv`).
2. Nettoyage → dataset consolidé (`jobs_cleaned.csv`).
3. Extraction skills → dataset de relations job-skill (`jobs_skills.csv`).
4. Chargement Snowflake → dimensions + faits.

5. Vues BI → datasets Superset.
6. Dashboards Superset → exploration et reporting.

2.4 Choix technologiques

— **Python** : orchestration, scraping, traitement, NLP.
— **Selenium + undetected-chromedriver** : contournement de protections anti-bot sur Indeed.
— **BeautifulSoup + requests** : extraction HTML robuste sur ReKrute.
— **Sentence-BERT** : similarité sémantique pour compléter les regex.
— **Snowflake** : entrepôt cloud, schéma en étoile, vues analytiques.
— **Apache Superset** : BI open-source, dashboards web, connecteur Snowflake.
— **Docker Compose** : déploiement local reproductible de Superset.

Chapitre 3

Ingestion des données

3.1 Objectif d'ingestion

La phase d'ingestion vise à collecter des offres orientées Data/ML depuis plusieurs sources afin de réduire les biais d'une source unique. Les scrapers appliquent un filtrage "métier" (DATA/ML/AI) dès la collecte pour limiter le bruit.

3.2 Scraper ReKrute (requests + BeautifulSoup)

3.2.1 Principes

Le module ReKrute cible une page listant les métiers IT et explore la pagination. Chaque offre est ensuite visitée pour extraire des attributs structurés : titre, entreprise, localisation, description, URL et métadonnées.

3.2.2 Filtrage Data/ML

Un filtrage strict est réalisé via des expressions régulières (ex : `data`, `machine learning`, `ai`) et des exclusions (ex : `frontend`, `devops`). L'objectif est d'exclure les postes IT génériques non centrés sur la donnée.

3.2.3 Extraction de l'entreprise

Une extraction robuste s'appuie sur la balise `og:title` de la page, au format `[COMPANY] Job Title`. Un *fallback* texte est prévu si la métadonnée est absente.

3.3 Scraper Indeed (Selenium + undetected-chromedriver)

3.3.1 Contexte anti-bot

Indeed applique fréquemment des protections qui bloquent les scrapers classiques. La solution adoptée utilise `undetected_chromedriver` et une navigation interactive.

3.3.2 Multi-régions

Le scraper supporte plusieurs domaines (ex : ES, FR, UK) et collecte des liens de résultats via pagination. Les pages d'offres sont ensuite visitées pour extraire titre, entreprise, localisation et description.

3.3.3 Filtrage et limitation

Le filtrage Data/ML est appliqué sur le titre et la description (regex + exclusions). Un paramètre `max_offers` limite le nombre d'offres afin de maîtriser temps d'exécution et charge.

3.4 Artefacts produits

Les résultats bruts sont exportés en CSV dans `data/`. Ces fichiers constituent l'entrée de la phase de nettoyage.

Chapitre 4

Nettoyage et harmonisation

4.1 Objectifs

Le nettoyage vise à :

— harmoniser les colonnes entre sources,
— normaliser les titres et les champs textuels,
— supprimer les doublons (URL, couple titre/entreprise),
— produire un dataset consolidé pour les phases NLP et entrepôt.

4.2 Harmonisation des schémas

Les scrapers produisent des colonnes proches mais non identiques. Le module de nettoyage aligne les champs sur un schéma standard :

Champ	**Description**
job_id	identifiant technique unique
title	intitulé de poste
company	entreprise
location	localisation brute
url	lien de l'offre
description	description (tronquée)
publish_date	date de publication (ou fallback)
source	indeed / rekrute
scrape_date	date de scraping

4.3 Normalisation des titres

Les intitulés sont normalisés via des règles de remplacement (ex : *ml engineer* → *Machine Learning Engineer*) puis capitalisation standard.

4.4 Gestion des textes

Les descriptions sont nettoyées (espaces, caractères de contrôle) et tronquées pour garantir des tailles raisonnables en stockage et traitement.

4.5 Déduplication

Deux stratégies complémentaires sont appliquées :

— déduplication par URL,
— déduplication par (titre, entreprise) pour capter les duplicats multi-URLs.

4.6 Sortie

La sortie principale est `data/jobs_cleaned.csv` utilisée ensuite pour l'extraction de compétences et le chargement Snowflake.

Chapitre 5

Extraction des compétences (NLP)

5.1 Objectif

L'objectif est d'extraire automatiquement une liste de compétences pertinentes à partir des descriptions d'offres. Cette information alimente :

— la mesure de la demande (quelles compétences dominent),
— la construction du data mart (agrégations),
— le moteur de recommandation (matching profil/offre).

5.2 Approche hybride

Le module d'extraction combine :

— **Regex** sur une base de patterns (langages, outils, cloud, BI, etc.).
— **Sentence-BERT** (all-MiniLM-L6-v2) pour détecter des mentions implicites via similarité sémantique.

5.3 Taxonomie

Une taxonomie regroupe les compétences en catégories (ex : *Programming Languages*, *Data Engineering*, *Databases*, *Machine Learning*, etc.). Cette catégorisation facilite les graphiques par famille de compétences.

5.4 Sortie structurée

Pour chaque offre, le module produit une table relationnelle (job, skill) avec :

— un score de confiance (0–1),
— la position approximative de la première mention,
— la méthode (regex/bert/hybride).

5.5 Remarques de robustesse

— En absence de Sentence-Transformers, le système fonctionne en mode regex-only.
— Les patterns sont conçus pour éviter des collisions (ex : `java` vs `javascript`).

Chapitre 6

Entrepôt de données Snowflake

6.1 Modélisation : schéma en étoile

Le stockage analytique s'appuie sur un schéma en étoile :

— **Dimensions** : entreprises, localisations, compétences.
— **Faits** : offres d'emploi, relations offre-compétence.

6.1.1 Tables dimensionnelles

— `DIM_COMPANIES` : entreprise (nom, industrie, pays).
— `DIM_LOCATIONS` : localisation (nom, pays, région).
— `DIM_SKILLS` : compétence (nom, catégorie).

6.1.2 Tables de faits

— `FACT_JOBS` : offre (titre, description, source, dates, clés étrangères).
— `FACT_JOB_SKILLS` : pont (job_id, skill_id) avec score de confiance.

6.2 Chargement des données

Le chargement est orchestré par un module Python qui :

— lit les CSV nettoyés,
— extrait les valeurs uniques pour les dimensions,
— charge les dimensions, puis récupère les IDs générés,
— charge les faits en respectant les dépendances,
— calcule des champs dérivés (longueur description, ancienneté de l'offre).

6.2.1 Gestion des dates

Les dates sont converties au format chaîne (YYYY-MM-DD) avant insertion afin d'éviter les problèmes de casting (DATE/TIMESTAMP) côté Snowflake.

6.3 Statistiques (état actuel)

Au moment de la rédaction, les volumes Snowflake sont :

Table	Lignes
DIM_COMPANIES	220
DIM_LOCATIONS	9
DIM_SKILLS	183
FACT_JOBS	79
FACT_JOB_SKILLS	370

6.4 Qualité et complétion

Certaines dimensions (ex : industrie) peuvent être incomplètes selon les sources. Une stratégie d'enrichissement a été appliquée (valeurs par défaut et heuristiques simples) pour permettre des visualisations agrégées cohérentes.

Chapitre 7

Data Mart BI (Vues Snowflake)

7.1 Motivation

Les dashboards exigent des requêtes rapides et des champs déjà prêts pour l'analyse. Plutôt que d'interroger directement le schéma en étoile pour chaque graphique, un data mart est exposé via des vues `VW_`.

7.2 Vues principales

— `VW_JOBS_FULL_CONTEXT` : offre enrichie (entreprise, localisation, skills agrégées).
— `VW_SKILLS_DEMAND` : demande par compétence (volumes, pourcentage, confiance).
— `VW_JOBS_BY_TITLE` : distribution des titres normalisés.
— `VW_MARKET_OVERVIEW` : métriques globales (KPI).
— `VW_COMPANY_OPPORTUNITIES` : entreprises qui recrutent et leurs caractéristiques.
— `VW_REGIONAL_ANALYSIS` : analyse géographique (adaptée par localisation).
— `VW_SKILL_SPECIALIZATION` : compétences dominantes par titre.
— `VW_TRENDING_SKILLS` : tendances sur fenêtres temporelles.
— `VW_JOB_COMPLEXITY` : score de complexité (proxy).

7.3 Correction de la vue régionale

Une difficulté rencontrée concernait l'agrégation par (région, pays) lorsque ces champs sont absents/NULL dans les données. La vue a été ajustée pour regrouper par `LOCATION_NAME` et fournir des valeurs par défaut via `COALESCE`.

7.4 Avantages pour la BI

— Simplification : des champs prêts à l'emploi (ex : skills agrégées en liste).
— Performance : agrégations réalisées côté entrepôt.
— Cohérence : une source unique de vérité pour tous les dashboards.

Chapitre 8

Système de recommandation

8.1 Objectif

Au-delà de la BI descriptive, le projet inclut un moteur de recommandation permettant de classer des offres selon un profil candidat.

8.2 Principe de scoring

Le scoring combine deux composantes :

— **Match compétences** (pondéré majoritairement) : présence des compétences du candidat dans la description.
— **Similarité sémantique** : similarité cosinus entre l'embedding du profil et les embeddings des offres.

8.3 Embeddings (Sentence-BERT)

Les descriptions des offres sont encodées avec le modèle `all-MiniLM-L6-v2`. Les embeddings sont pré-calculés au chargement afin d'accélérer la recommandation.

8.4 Fusion et pondération

La note finale est une combinaison pondérée, typiquement 60% compétences et 40% sémantique. Cette pondération reflète l'importance des prérequis techniques explicites.

8.5 Sortie

Le moteur produit un DataFrame avec :

— titre, entreprise, localisation,
— score compétences, score sémantique, score combiné,
— tri décroissant et top-k résultats.

8.6 Usage

Un scénario d'exemple génère des recommandations et exporte un fichier `candidate_recommendations.` exploitable en BI.

Chapitre 9

Visualisation dans Apache Superset

9.1 Choix de l'outil

La visualisation est réalisée avec **Apache Superset** pour disposer d'une solution web open-source, déployable localement via Docker, et sans dépendance à un compte cloud propriétaire.

9.2 Déploiement Docker Compose

L'environnement comprend :

— **PostgreSQL** : base de métadonnées Superset,
— **Redis** : cache,
— **Superset** : interface web (port 8088).

Le script d'initialisation automatise :

— migrations DB (`superset db upgrade`),
— création de l'utilisateur admin,
— initialisation des rôles/permissions (`superset init`),
— installation du driver Snowflake (`snowflake-sqlalchemy`).

9.3 Connexion à Snowflake

Après installation du driver, Superset est connecté à l'entrepôt Snowflake. Cette connexion permet d'importer chaque table/vue comme *dataset* Superset.

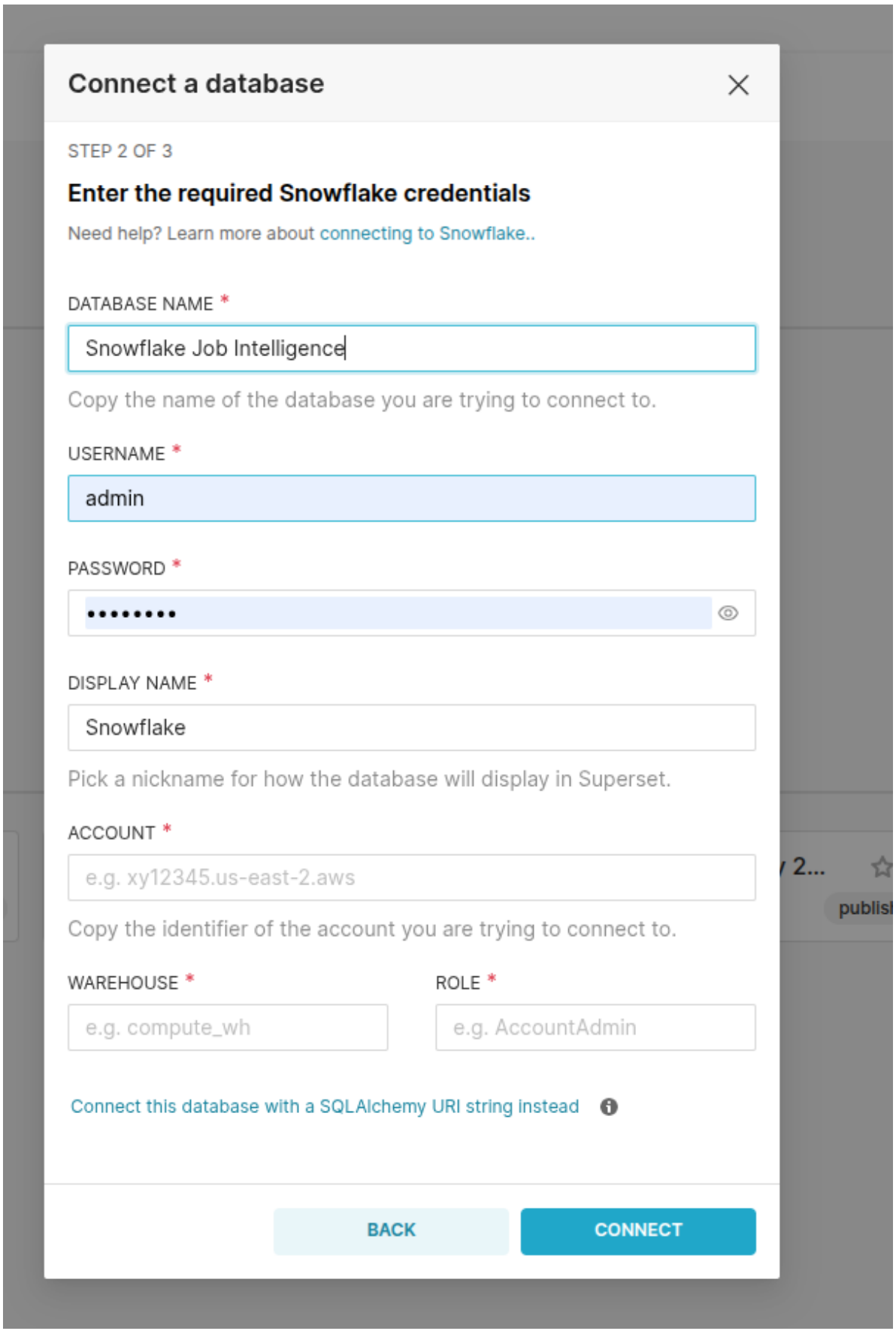

FIGURE 9.1 – Connexion Snowflake dans Superset

9.4 Création des datasets

Dans Superset, un dataset correspond à une table ou une vue. La création est répétée pour les tables de faits/dimensions et les vues BI.

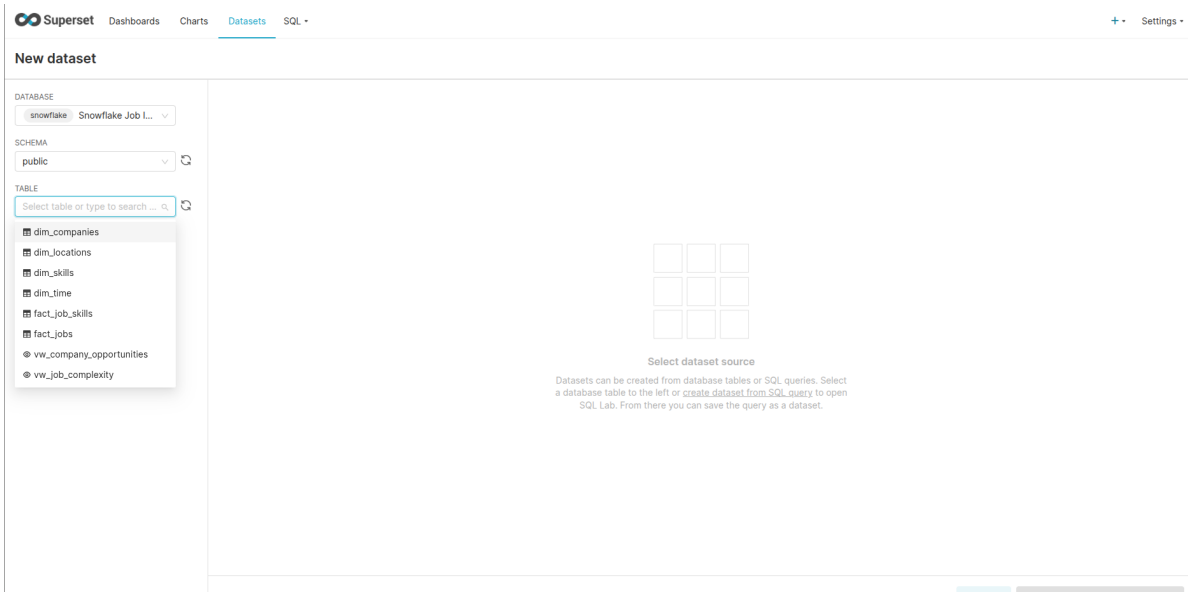

FIGURE 9.2 – Création d'un dataset (table/vue) dans Superset

9.5 Dashboards réalisés

Les dashboards s'appuient majoritairement sur les vues `VW_` afin de réduire la complexité des charts.

9.5.1 Dashboard 1 : Market Overview

KPIs (total jobs, entreprises, skills) et distributions (titres, sources, localisation).

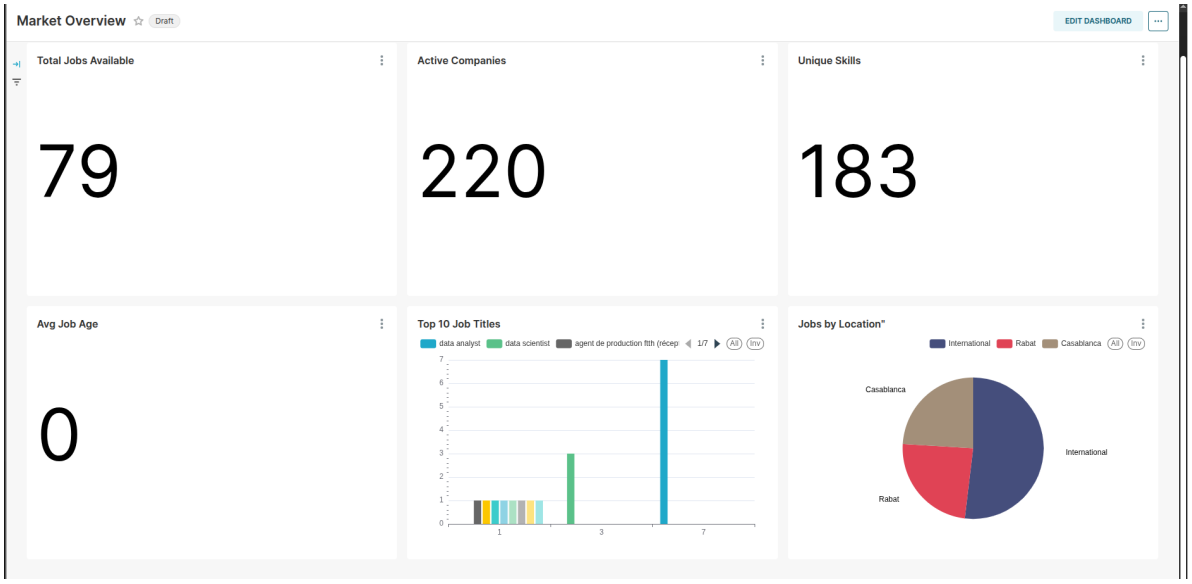

FIGURE 9.3 – Dashboard Market Overview

9.5.2 Dashboard 2–3 : Skills Demand et Company Opportunities

Analyse des compétences demandées et des entreprises recrutant, incluant des visualisations de type bar charts, tables et treemap. La dimension `INDUSTRY` a été enrichie lorsque manquante afin d'activer la visualisation "Companies by Industry".

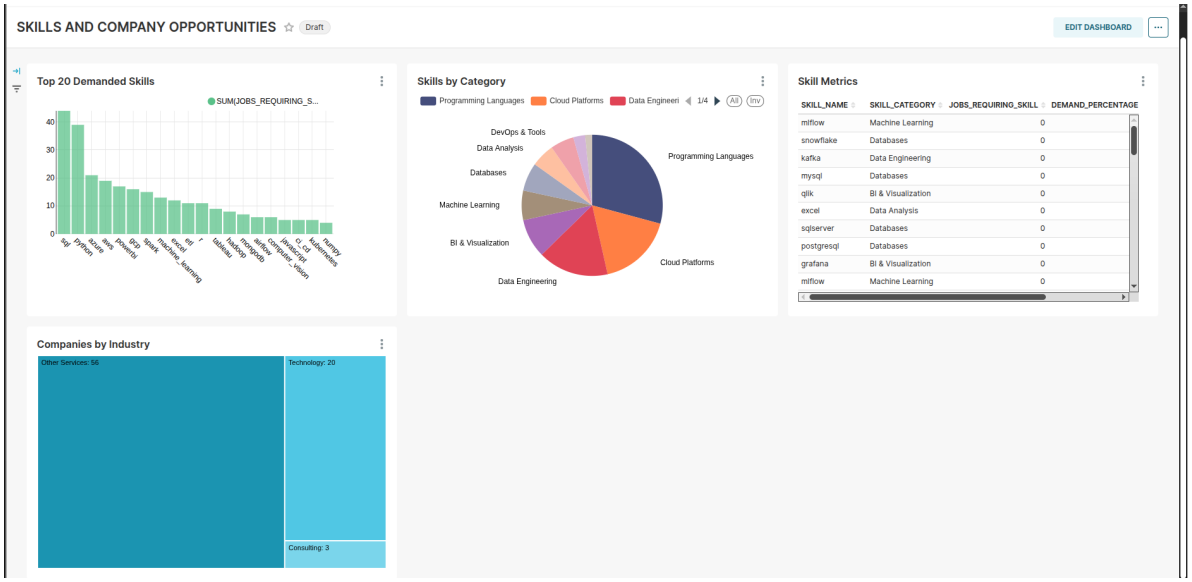

FIGURE 9.4 – Dashboards Skills Demand et Company Opportunities

9.5.3 Dashboard 4 : Job Details Explorer

Exploration interactive des offres (table + filtres). Les filtres sont mis en place via les **Native Filters** Superset (équivalent des slicers Power BI).

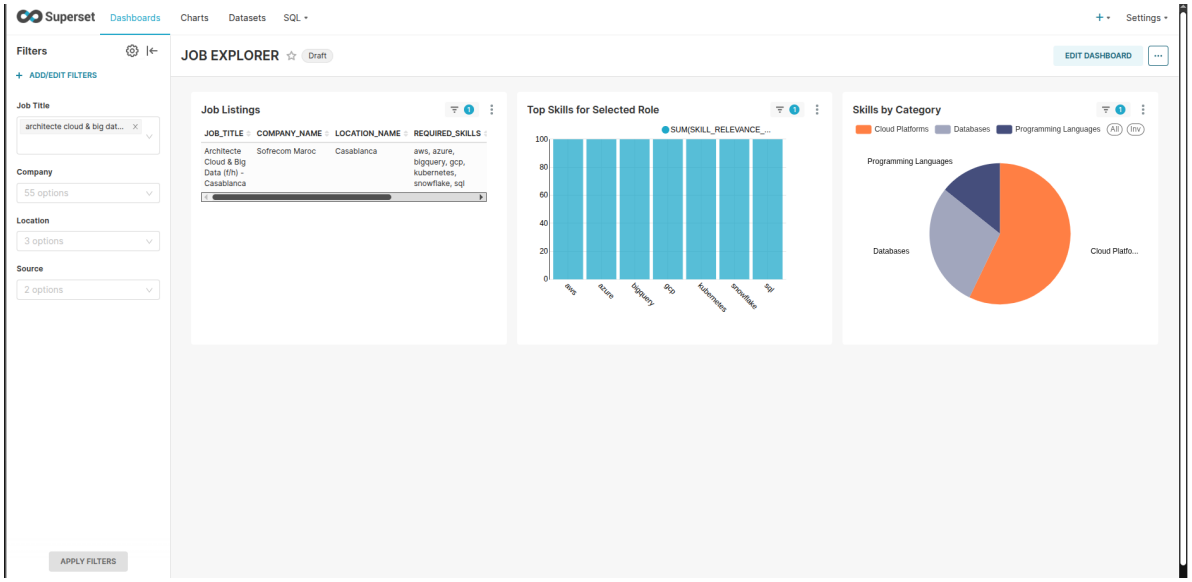

FIGURE 9.5 – Job Explorer avec filtres et liste des compétences

9.6 Bonnes pratiques de configuration des charts

— **Dimensions** : champs catégoriels (ex : titre, ville, entreprise).
— **Metrics** : agrégations numériques (COUNT, SUM, AVG).

— **Tri et Top-N** : utiliser `SORT BY` + `ROW LIMIT` pour les classements.

Chapitre 10

Conclusion et perspectives

10.1 Bilan

Plateforme d'intelligence du marché de l'emploi (Data/AI) met en place un pipeline complet d'intelligence du marché de l'emploi orienté Data/ML : collecte multi-sources, préparation de données, extraction automatique de compétences, stockage analytique Snowflake, data mart via vues, recommandations et dashboards Superset.

10.2 Limites

— Les scrapers peuvent être affectés par des changements d'interface (sélecteurs HTML) et des protections anti-bot.
— Les champs métiers (industrie, pays, région) dépendent de la richesse des sources et peuvent nécessiter enrichissement.
— Le NLP reste sensible à la qualité des descriptions et au vocabulaire (synonymes, abréviations).

10.3 Perspectives

— Ajouter une couche d'enrichissement géographique (normalisation pays/ville via référentiels).
— Étendre la taxonomie de compétences et gérer explicitement les synonymes.
— Automatiser la génération d'exports Superset (dashboards/charts) via API.
— Mettre en place une planification (cron) et un monitoring (logs, alerting) du pipeline.