📘 MCD enrichi pour application intelligente de matching d'emploi

Sources : Adzuna, France Travail, Formulaire Candidat
Objectif : Construire une architecture flexible pour un moteur de matching emploi / candidat enrichi en NLP.

🧠 Explication globale

Ce modèle permet :

- De centraliser des données d'offres hétérogènes
- De croiser les préférences multiples d'un candidat (lieu, contrat, entreprise…)
- De générer un score d'adéquation avec des suggestions de formation
- De visualiser les tendances marché par domaine, lieu, etc.

📎 Utilisation de dbdiagram.io

Pour visualiser le MCD sous forme graphique : 👉 https://www.dbdiagram.io/home

🔽 À copier-coller directement sur dbdiagram.io

▶ Clique ici pour afficher le code du MCD (.io)

// 🔷 Table des offres d'emploi Table OffreEmploi { id_offre int [pk] // identifiant unique de l'offre (source: id) id_contrat int [ref: > Contrat.id_contrat] // FK vers type de contrat id_lieu int [ref: > Lieu.id_lieu] // FK vers localisation id_date int [ref: > Date_calendar.id_date] id_entreprise int [ref: > Entreprise.id_entreprise] // FK vers l'employeur id_domaine int [pk] // table domaine titre text // source: (title / intitule) description text // source: description date_publication datetime // source: created / dateCreation : normalisation du format date_mise_a_jour datetime source varchar(50) // valeur fixe : "adzuna" ou "france_travail" source_url varchar(100) niveau_seniorite varchar(20) // ex: Junior, Senior — déduit via NLP sur le titre ou description salaire_min int // Adzuna : salary_min | FranceTravail : à extraire via NLP (contexteTravail) : a calculer sur moyenne domaine_data+lieu+seniorité salaire_max int // idem teletravail_possible boolean // détecté dans le texte (mots clés: "remote", "télétravail", etc.) score_attractivite float // score calculé pondérant salaire, techno, remote, contrat, localisation code_ROME int // Code Rome }

// 🔷 Table des dates Table Date_calendar { id_date int [pk] full_date date // ex: 2025-04-10 jour int // 1 à 31 mois int // 1 à 12 mois_nom varchar(20) // Avril trimestre int // 1 à 4 annee int semaine int // numéro de semaine jour_semaine varchar(20) // Lundi, Mardi, etc. }

// 🔷 Table des localisations géographiques Table Lieu { id_lieu int [pk] libelle text // ex: "33 - BORDEAUX" — source: location_area / lieuTravail.libelle code_postal varchar(10) // à extraire via regex sur libelle ville varchar(50) // idem departement varchar(50) // idem region varchar(50) // Adzuna : location_area_0 | FranceTravail : à construire via mapping CP/region pays varchar(50) // valeur fixe : "France" latitude float // source directe (France Travail) longitude float // idem }

// 🔷 Table des entreprises Table Entreprise { id_entreprise int [pk] nom text // source: company_display_name / entreprise.nom id_type_entreprise int [ref: > TypeEntreprise.id_type] // nouveau : FK id_domaine_entreprise int [ref: > DomaineEntreprise.id_domaine_ent] // nouveau : FK tranche_effectif varchar(50) // source: trancheEffectifEtab (à fiabiliser via API) }

// 🔷 Type d'entreprise (Start-up, PME, etc.) Table TypeEntreprise { id_type int [pk] nom varchar(30) // ex: "Start-up", "PME", "ETI", "Grand Groupe" }

// 🔷 Domaine d'activité de l'entreprise Table DomaineEntreprise { id_domaine_ent int [pk] nom varchar(50) // ex: "Banque", "Retail", "Transport", "Santé", etc. }

// 🔷 Table des contrats Table Contrat { id_contrat int [pk] type_contrat varchar(30) // source: contract_type (CDI, CDD, Freelance) | NLP sur description temps_travail varchar(30) // NLP sur contexteTravail ou champ dédié alternance boolean // détecté via NLP (présence du mot "alternance") horaires text // source: contexteTravail.horaires }

// 🔷 Table des compétences techniques (standardisées) Table CompetenceTech { id_competence int [pk] nom text // extrait via NLP sur description type varchar(30) // classification manuelle : langage, outil, framework, cloud...

}

// 🔷 Table des formations recommandées Table Formation { id_formation int [pk] nom text plateforme text // ex: OpenClassrooms, Coursera, Udemy cout int // en euros niveau text // Débutant, Intermédiaire, Avancé duree varchar(30) // ex: "2 semaines", "10h" lien text // URL directe }

// 🔷 Table de liaison Formation <-> Compétence Table Formation_Competence { id_formation int [ref: > Formation.id_formation, primary key] id_competence int [ref: > CompetenceTech.id_competence, primary key] }

// 🔷 Table de liaison Offre <-> Compétence Table Offre_CompetenceTech { id_offre int [ref: > OffreEmploi.id_offre, primary key] id_competence int [ref: > CompetenceTech.id_competence, primary key] //exigence boolean // A ENLEVER NLP : si compétence obligatoire ou souhaitée }

// 🔷 Domaine data (standardisé) Table DomaineData { id_domaine int [pk] nom varchar(30) // ex: "ML", "BI", "Data Eng", "Data Analyst", etc. }

// 🔷 Table de liaison CompétenceTech <-> DomaineData Table Competence_Domaine { id_competence int [ref: > CompetenceTech.id_competence, primary key] id_domaine int [ref: > DomaineData.id_domaine, primary key] poids int // calcul dynamique en fonction de la tendance du marché. Ex : offre present dans 80% des offres VBI mais 20% du ML . A renfrocer avoir le booleen // Cette table permet de relier une compétence à plusieurs domaines // Exemple : Python → ML + Data Eng + BI }

// 🔷 Table des candidats (mise à jour avec des FK vers d'autres tables) Table Candidat { id_candidat int [pk] email text // donné utilisateur mobilite boolean // l'utilisateur est-il mobile ? niveau_experience varchar(30) // Junior, Senior — donné utilisateur salaire_min_souhaite int // souhait utilisateur remote_souhaite text // ex: "jamais", "1 ou 2j", "remote total" — texte libre }

// 🔷 Table des soft skills Table Soft_skills { id_soft_skills int [pk] nom_skill varchar(50) // }

// 🔷 Table de liaison Candidat <-> Compétence Table Candidat_Competence { id_candidat int [ref: > Candidat.id_candidat, primary key] id_competence int [ref: > CompetenceTech.id_competence, primary key] niveau int // niveau perçu ou auto-évalué : 1 (débutant) à 5 (expert) }

// 🔷 Domaine data préféré du candidat (s'il peut en choisir plusieurs) Table Candidat_DomaineData { id_candidat int [ref: > Candidat.id_candidat, primary key] id_domaine int [ref: > DomaineData.id_domaine, primary key] // Permet à un candidat d'avoir plusieurs domaines data préférés }

// 🔷 Localisations préférées du candidat (multi-lieux) Table Candidat_Lieu { id_candidat int [ref: > Candidat.id_candidat, primary key] id_lieu int [ref: > Lieu.id_lieu, primary key] type_pref varchar(20) // optionnel : "principale", "secondaire", "remote" }

Table Candidat_Contrat { id_candidat int [ref: > Candidat.id_candidat, primary key] id_contrat int [ref: > Contrat.id_contrat, primary key] // Permet de choisir plusieurs types de contrat souhaités (ex : CDI + Freelance) }

Table Candidat_TypeEntreprise { id_candidat int [ref: > Candidat.id_candidat, primary key] id_type int [ref: > TypeEntreprise.id_type, primary key] // Ex : je veux bosser en start-up OU ETI }

Table Candidat_DomaineEntreprise { id_candidat int [ref: > Candidat.id_candidat, primary key] id_domaine_ent int [ref: > DomaineEntreprise.id_domaine_ent, primary key] // Ex : secteurs préférés : Santé + Banque }

// 🔷 Table de matching entre offre et candidat Table MatchingCandidatOffre { id_matching int [pk] id_candidat int [ref: > Candidat.id_candidat] id_offre int [ref: > OffreEmploi.id_offre] score_global float // score final basé sur plusieurs critères score_tech float // score uniquement sur la correspondance des compétences manques text // liste des compétences manquantes suggestion_formation text // texte libre ou lien vers catalogue }

// 🔷 Localisations préférées du candidat (multi-lieux) Table Candidat_formation { id_candidat int [ref: > Candidat.id_candidat, primary key] id_formation int [ref: > Formation.id_formation, primary key] type_pref varchar(20) // optionnel : "principale", "secondaire", "remote" }

// 🔷 Localisations préférées du candidat (multi-lieux) Table offre_soft_skills { id_offre int [ref: > OffreEmploi.id_offre, primary key] id_soft_skills int [ref: > Soft_skills.id_soft_skills, primary key] type_pref varchar(20) // optionnel : "principale", "secondaire", "remote" }

🎯 Objectifs du modèle

- Fournir un moteur de **recommandation d'offres** intelligent et personnalisé.
- Identifier les **compétences manquantes** pour un candidat et recommander des **formations ciblées**.
- Analyser les **tendances du marché** (salaires, technos, régions) par domaine data.
- Gérer des **préférences complexes et multiples** côté candidat (plusieurs lieux, contrats, types d'entreprises…).
- Intégrer des données **structurées + enrichies via NLP**.

📦 Composants clés

🔷 1. Offres d'emploi

- `OffreEmploi` contient les informations principales de chaque offre.
- Reliée à :
- - `Contrat` (CDI, Freelance…)
- - `Lieu` (géolocalisation)
- - `Entreprise`
- - `DomaineData` (ML, BI…)
- - `Date_calendar` (calendrier analytique)
- Les compétences associées à chaque offre sont dans `Offre_CompetenceTech`.

🔷 2. Candidats

- `Candidat` stocke les données personnelles + préférences.
- Table flexible : toutes les **préférences multiples** sont gérées via des tables de liaison :
- - `Candidat_Contrat`
- - `Candidat_TypeEntreprise`
- - `Candidat_DomaineEntreprise`
- - `Candidat_DomaineData`
- - `Candidat_Lieu`
- Les compétences sont listées dans `Candidat_Competence` avec un **niveau de maîtrise (1 à 5)**.

🔷 3. Matching intelligent

- `MatchingCandidatOffre` est le **pivot du moteur de recommandation**.
- Contient :
- - `score_tech` : matching de compétences
- - `score_global` : matching global pondéré
- - `manques` : liste des compétences absentes
- - `suggestion_formation` : lien ou nom de formation recommandée