Rapport de Projet BDR

Cseres Leonard, Laydu Aude, Gerber Tristan

1er décembre 2024

Table des matières

1 Introduction **2**

2 Cahier des charges **2**
2.1 Objectif de l'application 2
2.2 Fonctionnalités principales 2

3 Conception **2**
3.1 Entités Principales 2
3.2 Modélisation Conceptuelle (Schéma EA) 3
3.3 Modélisation Relationnelle 3
3.4 Contraintes et Intégrités Référentielles 4
3.4.1 Stratégies ON DELETE et ON UPDATE 4
3.4.2 Triggers de Validation 7
3.4.3 Contraintes Métier 8

4 Implémentation **10**
4.1 Technologies Utilisées 10
4.2 Structure de l'Application 10
4.3 Requêtes SQL 10

5 Tests **10**
5.1 Stratégie de Test 10
5.2 Résultats des Tests 10

6 Conclusion **10**
6.1 Bilan 10
6.2 Perspectives 10

7 Annexes **10**
7.1 A. Guide d'Installation et de Déploiement 10
7.2 B. Manuel Utilisateur 10
7.3 C. Schémas Complémentaires 10

1 Introduction

Dans le cadre du projet de BDR, nous avons:

1. Créé le cahier des charges
2. Éffectué la modélisation
3. Implémenté physiquement la bdd
4. Codé une application

2 Cahier des charges

Consulter le cahier des charges au format PDF

2.1 Objectif de l'application

Développer une application pour permettre aux recruteurs de gérer efficacement les processus de recrutement, incluant la gestion des candidats, des postes et des entretiens.

2.2 Fonctionnalités principales

- Gestion des candidats: ajout, modification, suppression, suivi des interactions, statut.
- Gestion des offres d'emploi: création, gestion des offres, suivi des candidatures.
- Gestion des entretiens: planification et suivi des retours.
- Suivi du processus de recrutement: vue d'ensemble des candidats par poste.
- Authentification de base pour les recruteurs.

3 Conception

3.1 Entités Principales

- **Personne**: Entité racine représentant tous les acteurs du système
- **Candidat**: Spécialisation de Personne, représentant les candidats
- **Recruteur**: Spécialisation de Personne, représentant les recruteurs
- **Interaction**: Modélise les différents types de communications
- **Offre**: Représente les opportunités d'emploi
- **Domaine**: Catégorise les compétences et secteurs d'activité

3.2 Modélisation Conceptuelle (Schéma EA)

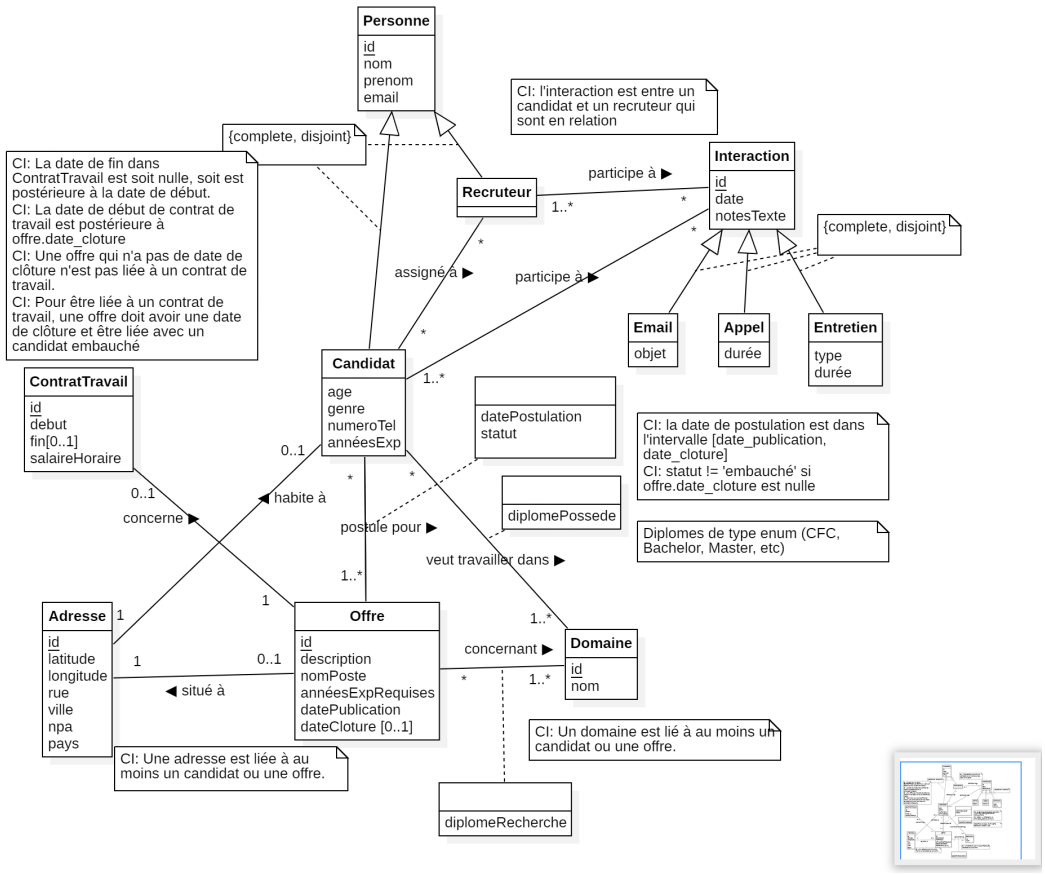

Figure 1: Schema EA

3.3 Modélisation Relationnelle

Personne(<u>id</u>, nom, prenom, email)

Recruteur(<u>idPersonne</u>)
Recruteur.idPersonne reference Personne.id

Adresse(<u>id</u>, latitude, longitude, rue, ville, npa, pays)

Candidat(<u>idPersonne</u>, age, genre, numeroTel, annéesExp, idAdresse)
Candidat.idPersonne reference Personne.id
Candidat.idAdresse reference Adresse.id Candidat.idAdresse NOT NULL UNIQUE

Recruteur_Candidat(<u>idRecruteur, idCandidat</u>)
Recruteur_Candidat.idRecruteur reference Recruteur.idPersonne
Recruteur_Candidat.idCandidat reference Candidat.idPersonne

Interaction(id, date, notes_texte)

Recruteur_Interaction(idRecruteur, idInteraction)
Recruteur_Interaction.idRecruteur reference Recruteur.idPersonne
Recruteur_Interaction.idInteraction reference Interaction.id

Candidat_Interaction(idCandidat, idInteraction)
Candidat_Interaction.idCandidat reference Candidat.idPersonne
Candidat_Interaction.idInteraction reference Interaction.id

Email(idInteraction, objet)
Email.idInteraction reference Interaction.id

Appel(idInteraction, durée)
Appel.idInteraction reference Interaction.id

Entretien(idInteraction, type, durée)
Entretien.idInteraction reference Interaction.id

Offre(id, descriptionOffre, nomPoste, annnéesExpRequises, datePublication, dateCloture, idAdresse)
Offre.idAdresse reference Adresse.id
Offre.idAdresse NOT NULL UNIQUE

ContratTravail(id, début, fin, salaireHoraire, idOffre)
ContratTravail.idOffre reference Offre.id
ContratTravail.idOffre NOT NULL, UNIQUE

Candidat_Offre(idCandidat, idOffre, datePostulation, statut)
Candidat_Offre.idCandidat reference Candidat.idPersonne
Candidat_Offre.idOffre reference Offre.id

Domaine(id, nom)

Offre_Domaine(idOffre, idDomaine, diplomeRecherche) Offre_Domaine.idOffre reference Offre.id Offre_Domaine.idDomaine reference Domaine.id

Candidat_Domaine(idCandidat, idDomaine, diplomePossede) Candidat_Domaine.idCandidat reference Candidat.idPersonne Candidat_Domaine.idDomaine reference Domaine.id

3.4 Contraintes et Intégrités Référentielles

3.4.1 Stratégies ON DELETE et ON UPDATE

3.4.1.1 Candidat

```
CONSTRAINT FK_Personne
FOREIGN KEY (idPersonne)
REFERENCES Personne(id)
ON DELETE CASCADE
ON UPDATE CASCADE

CONSTRAINT FK_Adresse
FOREIGN KEY (idAdresse)
```

```
REFERENCES Adresse(id)
ON DELETE RESTRICT
ON UPDATE NO ACTION
```

- **Personne:** Si une personne est supprimée, le profil de candidat est également supprimé. Cela garantit qu'un candidat ne peut pas exister sans une identité personnelle de base.
- **Adresse:** L'adresse ne peut pas être supprimée si un candidat y est associé, afin de préserver l'intégrité des données historiques.

3.4.1.2 Recruteur

```
FOREIGN KEY (idPersonne)
REFERENCES Personne(id)
ON DELETE CASCADE
ON UPDATE CASCADE
```

- Similaire au Candidat: si la personne est supprimée, son profil de recruteur est également supprimé.

3.4.1.3 Recruteur_Candidat

```
CONSTRAINT FK_Recruteur
FOREIGN KEY (idRecruteur)
REFERENCES Recruteur(idPersonne)
ON DELETE RESTRICT
ON UPDATE CASCADE

CONSTRAINT FK_Candidat
FOREIGN KEY (idCandidat)
REFERENCES Candidat(idPersonne)
ON DELETE RESTRICT
ON UPDATE CASCADE
```

- Empêche la suppression d'un recruteur ou d'un candidat s'il existe des relations professionnelles entre eux.
- Autorise la mise à jour des identifiants si nécessaire.

3.4.1.4 Interactions (Emails, Appels, Entretiens)

```
FOREIGN KEY (idInteraction)
REFERENCES Interaction(id)
ON DELETE CASCADE
ON UPDATE CASCADE
```

- Les interactions spécialisées (email, appel, entretien) sont supprimées si l'interaction parent est supprimée.
- Permet de maintenir la cohérence des types d'interactions.

3.4.1.5 Recruteur_Interaction et Candidat_Interaction

```
CONSTRAINT FK_Recruteur/Candidat
FOREIGN KEY (idRecruteur/idCandidat)
REFERENCES Recruteur/Candidat(idPersonne)
ON DELETE RESTRICT
ON UPDATE CASCADE

CONSTRAINT FK_Interaction
FOREIGN KEY (idInteraction)
REFERENCES Interaction(id)
ON DELETE RESTRICT
ON UPDATE CASCADE
```

- Empêche la suppression d'un recruteur ou candidat s'il a des interactions historiques.
- Interdit la suppression d'interactions liées à des personnes.

3.4.1.6 Offre

```
CONSTRAINT FK_Adresse
FOREIGN KEY (idAdresse)
REFERENCES Adresse(id)
ON DELETE RESTRICT
ON UPDATE NO ACTION
```

- L'adresse associée à une offre ne peut pas être supprimée, préservant la localisation historique de l'offre.

3.4.1.7 Contrat_Travail

```
CONSTRAINT FK_Contrat_Travail
FOREIGN KEY (idOffre)
REFERENCES Offre(id)
ON DELETE RESTRICT
ON UPDATE NO ACTION
```

- Un contrat ne peut pas être lié à une offre supprimée, garantissant l'intégrité des références.

3.4.1.8 Candidat_Offre

```
CONSTRAINT FK_Candidat
FOREIGN KEY (idCandidat)
REFERENCES Candidat(idPersonne)
ON DELETE RESTRICT
ON UPDATE CASCADE

CONSTRAINT FK_Offre
FOREIGN KEY (idOffre)
REFERENCES Offre(id)
ON DELETE RESTRICT
ON UPDATE NO ACTION
```

- Empêche la suppression d'un candidat ayant postulé à des offres.
- Interdit la modification de l'offre une fois les candidatures enregistrées.

3.4.1.9 Offre_Domaine et Candidat_Domaine

```
CONSTRAINT FK_Offre/Candidat
FOREIGN KEY (idOffre/idCandidat)
REFERENCES Offre/Candidat(id)
ON DELETE RESTRICT
ON UPDATE NO ACTION

CONSTRAINT FK_Domaine
FOREIGN KEY (idDomaine)
REFERENCES Domaine(id)
ON DELETE RESTRICT
ON UPDATE NO ACTION
```

- Préserve l'intégrité des liens entre offres/candidats et domaines.
- Empêche la suppression de domaines sans considérer leurs relations existantes.

Ces choix de contraintes visent à:

- Maintenir l'intégrité référentielle
- Prévenir les suppressions accidentelles
- Conserver un historique fiable
- Permettre des mises à jour contrôlées des données

3.4.2 Triggers de Validation

3.4.2.1 `check_personne()`

- **Objectif:** Garantir qu'une personne ne peut être créé que si un Candidat ou un Recruteur est aussi crée dans la transaction
- **Déclenchement:** Après l'insertion dans la table Personne
- **Action:** Lève une exception si la condition n'est pas respectée

3.4.2.2 `check_personne_exists()`

- **Objectif:** Garantir qu'un Candidat ou un Recruteur ne peut être créé que si l'identifiant de personne existe déjà
- **Déclenchement:** Après l'insertion dans les tables Candidat et Recruteur
- **Vérification:** Vérifie que l'`idPersonne` n'est pas null et existe dans la table Personne
- **Action:** Lève une exception si la condition n'est pas respectée

3.4.2.3 `check_interaction()`

- **Objectif:** Garantir qu'une interaction ne peut être créé que si une interactions spécialisées (Email, Appel, Entretien) est aussi crée dans la transaction
- **Déclenchement:** Après l'insertion dans la table Interaction
- **Action:** Lève une exception si la condition n'est pas respectée

3.4.2.4 `check_interaction_exists()`

- **Objectif:** Assurer que les interactions spécialisées (Email, Appel, Entretien) sont liées à une interaction principale existante
- **Déclenchement:** Après l'insertion dans les tables Interaction_Email, Interaction_Appel, Interaction_Entretien
- **Vérification:** Confirme que l'`idInteraction` n'est pas null et existe dans la table Interaction
- **Action:** Lève une exception si la condition n'est pas respectée

3.4.2.5 `check_contrat_embauche()`

- **Objectif:** Empêcher la création d'un contrat de travail sans candidat embauché
- **Déclenchement:** Avant l'insertion dans la table Contrat_Travail
- **Vérification:** Vérifie qu'il existe au moins un candidat avec le statut 'Embauché' pour l'offre
- **Action:** Lève une exception si aucun candidat n'a été embauché

3.4.2.6 `check_fin_after_cloture()`

- **Objectif:** Garantir que la date de fin du contrat est postérieure à la date de clôture de l'offre
- **Déclenchement:** Avant l'insertion ou la mise à jour dans la table Contrat_Travail
- **Vérification:** Compare la date de fin du contrat avec la date de clôture de l'offre
- **Action:** Lève une exception si la date de fin est antérieure ou égale à la date de clôture

3.4.2.7 `check_candidat_offre_constraints()`

- **Objectif:** Valider les candidatures selon plusieurs contraintes
  1. Un statut 'Embauché' nécessite une date de clôture d'offre
  2. La date de postulation doit être entre la date de publication et la date de clôture de l'offre
- **Déclenchement:** Avant l'insertion ou la mise à jour dans la table Candidat_Offre
- **Vérification:**
  - Vérifie la cohérence du statut 'Embauché' avec la date de clôture
  - Contrôle que la date de postulation est dans la période valide de l'offre
- **Action:** Lève une exception si l'une des conditions n'est pas respectée

3.4.2.8 `check_domaine_link()`

- **Objectif:** S'assurer qu'un domaine est lié soit à un candidat, soit à une offre
- **Déclenchement:** Après l'insertion ou la mise à jour dans la table Domaine
- **Vérification:** Compte les liens avec des candidats et des offres
- **Action:** Lève une exception si aucun lien n'est trouvé

Tous ces triggers visent à maintenir la cohérence et l'intégrité des données du système de gestion de candidatures.

3.4.3 Contraintes Métier

3.4.3.1 Validation d'Âge des Candidats

```
CHECK (age >= 16 AND age < 100)
```

- **Objectif:** Garantir un intervalle d'âge réaliste pour les candidats
- **Plage:** Entre 16 et 99 ans
- **Logique:** Exclut les candidats potentiellement trop jeunes ou irréalistement âgés

3.4.3.2 Validation des Coordonnées Géographiques (Adresse)

```
CHECK (latitude > -90 AND latitude < 90)
CHECK (longitude > -180 AND longitude < 180)
```

- **Objectif:** Assurer des coordonnées géographiques valides
- **Latitude:** Comprise entre -90° et 90° (pôles Nord et Sud)
- **Longitude:** Comprise entre -180° et 180° (méridien de Greenwich)
- **Logique:** Correspond aux limites physiques de la Terre

3.4.3.3 Validation des Dates de Contrat de Travail

```
CHECK (fin IS NULL OR fin > debut)
```

- **Objectif:** Garantir la cohérence des dates de contrat
- **Condition:** Si une date de fin est spécifiée, elle doit être postérieure à la date de début
- **Flexibilité:** Permet des contrats sans date de fin (contrats indéterminés)

3.4.3.4 Validation du Salaire Horaire

```
CHECK (salaireHoraire > 0)
```

- **Objectif:** Assurer un salaire horaire positif
- **Condition:** Le salaire doit être strictement supérieur à zéro
- **Logique:** Exclut les valeurs nulles ou négatives

3.4.3.5 Validation des Dates de Publication et Clôture d'Offre

```
CHECK (dateCloture IS NULL OR dateCloture > datePublication)
```

- **Objectif:** Garantir la logique temporelle des offres
- **Condition:** Si une date de clôture existe, elle doit être postérieure à la date de publication
- **Flexibilité:** Autorise des offres sans date de clôture

4 Implémentation

4.1 Technologies Utilisées

4.2 Structure de l'Application

4.3 Requêtes SQL

5 Tests

5.1 Stratégie de Test

5.2 Résultats des Tests

6 Conclusion

6.1 Bilan

6.2 Perspectives

7 Annexes

7.1 A. Guide d'Installation et de Déploiement

7.2 B. Manuel Utilisateur

7.3 C. Schémas Complémentaires