Nguyen Bang
Lerner Thomas
Redouin Noé
Durocher Violette

MLD Projet NF18 semaine 2

1) Relations :

- **Employe** (#idEmploye:int, nom:string, prenom:string, agence=>Agence(idAgence));
- **AgentTechnique** (#idEmploye=>Employe);
- **AgentCommercial** (#idEmploye=>Employe);
- **Agence** (#idAgence:int, nom:string, adresse:string, siret:string, mail:string, telephone:string);

- **SocieteEntretien** (#siret:string, nom:string);
- **Entretien** (#idEntretien:int, date_entretien:date, date_controle:date, resultat:string, societe=>SocieteEntretien(siret), agent_tech=>AgentTechnique(idEmploye));

- **Location** (#idContrat:int, date_debut:date, date_fin:date, km_parcourus:float, vehicule=>Vehicule(immat), agent_com=>AgentCommercial(idEmploye), entretien=>Entretien(idEntretien), facturation=>Facturation(idFacturation));
- **LocationParticulier** (#idContrat=>Location, particulier=>Particulier(idClient));
- **LocationProfessionnel** (#idContrat=>Location, entreprise=>Entreprise(idClient));
- **Facturation** (#idFacturation:int, clientParticulier=>LocationParticulier(particulier), clientProfessionnel=>LocationProfessionnel(entreprise), agent_com=>AgentCommercial(idEmploye), montant:money, date_payement:date, moyen_reglement:string, etat:string);

- **ValidationFinale** (#agent_com=>AgentCommercial(idEmploye), #location=>Location(idContrat), date_validation:date, resultat_validation:boolean);

- **Particulier** (#idClient:int, num_bancaire:string, mail:string, tel:string, adresse:string, num_permis:string, date_naissance:date, nom:string, prenom:string);
- **Entreprise** (#idClient:int, num_bancaire:string, mail:string, tel:string, adresse:string, nom:string, siret:string);
- **Conducteur** (#num_permis:string, nom:string, prenom:string, date_naissance:date, entreprise=>Entreprise(idClient), location=>LocationProfessionnel(idContrat));

- **Véhicule** (#immat:string, #modele=>Modele(nom), carburant=>TypeCarburant(nom), couleur:string, nb_km:float; agence=>Agence(idAgence), agent_tech=>AgentTechnique(idEmploye));
  /// nb_km not null

- **Option** (#nom:string);
- **AssociationOptionsVehicule** (#vehicule=>Vehicule(immat), #option=>Option(nom));

- **TypeCarburant** (#nom:string);
- **Marque** (#nom:string);
- **Modele** (#nom:string, #marque=>Marque(nom), #categorie=>Categorie(nom));

2) Complément

Vues sur les classes filles LocationParticulier et LocationProfessionnel.

vLocationParticulier = Projection (JointureNaturelle (Location, LocationParticulier), idContrat, date_debut, date_fin, km_parcourus, vehicule, facturation, particulier)
vLocationProfessionnel = Projection (JointureNaturelle (Location, LocationProfessionnel), idContrat, date_debut, date_fin, km_parcourus, vehicule, facturation, entreprise)

3) Contraintes

A - Contraintes liées à la cardinalité

a. Cardinalité 1..1

- Employe-Agence:
Employe.agence NOT NULL

- Entretien-SocieteEntretien:
Entretien.societe NOT NULL

- Entretien-AgentTechnique:
Entretien.agent_tech NOT NULL

- Location-Vehicule:
Location.vehicule NOT NULL

- Location-Entretien:
Location.entretien NOT NULL

- Location-Facturation:
Location.facturation NOT NULL

- LocationParticulier-Particulier:
LocationParticulier.particulier NOT NULL

- LocationProfessionnel-Entreprise:
LocationProfessionnel.entreprise NOT NULL

- Facturation-AgentCommerical:
Facturation.agent_com NOT NULL

- Facturation-Client(Professionnel ou Particulier):

Facturation.client NOT NULL

- Conducteur-Entreprise:

Conducteur.entreprise NOT NULL

- Conducteur-Location:

Conducteur.location NOT NULL

- Vechicule-TypeCarburant:

Vehicule.carburant NOT NULL

- Vehicule-Agence:

Vehicule.agence NOT NULL

- Vehicule-AgentTechnique:

Vehicule.agent_tech NOT NULL

b. Contrainte de cardinalité 1 dans les associations 1:N

(On veut que tous les éléments de R1 soient référencés au moins une fois dans R2, donc il n'existe pas de tuple de R1 qui ne soit pas référencé par R2.a, donc PROJECTION(R1,a) = PROJECTION(R2,a))

- Agence-Employé:

Projection (Agence.idAgence) = Projection (Employe.idAgence)

- Facturation-Location:

Projection (Location.facturation(idFacturation)) = Projection (Facturation.idFacturation)

- Entreprise-Conducteur:

Projection (Entreprise.idClient) = Projection (Conducteur.entreprise(idClient))

- Entreprise-LocationProfessionnel:

Projection (LocationProfessionnel.entreprise(idClient)) = Projection (Entreprise.idClient)

- Véhicule-TypeCarburant:

Projection (TypeCarburant.nom) = Projection (Vehicule.carburant(nom))

c. Associations 1:1

- Entretien-Location:

(C'est location qui contient la clé étrangère vers Entretien, donc on rajoute la contrainte entretien UNIQUE)

Location.entretien UNIQUE

- LocationParticulier-Particulier:

LocationParticulier.particulier UNIQUE

- LocationProfessionnel-Conducteur:

Conducteur.location UNIQUE

B - Contraintes liées à l'héritage

a. Héritage par référence, classe mère non abstraite

- Employe(AgentCommercial, AgentTechnique): *(Pas de clé locale, mais on doit préciser que l'héritage est exclusif)*

INTERSECTION (PROJECTION(AgentCommercial.idClient),
PROJECTION(AgentTechnique.idClient)) = {}

b. Héritage par référence, classe mère abstraite

- Location-(LocationParticulier, LocationProfessionnel): *(Pas de clé locale, mais on doit préciser que tous les tuples de la classe mère sont référencés par les classes filles)*

PROJECTION(Location.idContrat) = PROJECTION(LocationParticulier.idContrat) UNION
PROJECTION(LocationProfessionnel.idContrat)

c. Héritage par les classes filles avec clé au niveau de la classe classe mère abstraite

- Client-(Particulier, Entreprise): *(Clés locales à préciser + on doit vérifier l'unicité de la clé mère au niveau des classes filles)*

Particulier.num_permis KEY
Entreprise.siret KEY
INTERSECTION (PROJECTION(Particulier.idClient), PROJECTION(Entreprise.idClient)) = {}

Au niveau de la relation Facturation, on doit également imposer une contrainte sur les clés étrangères clientParticulier et clientProfessionnel (le client est de l'un ou l'autre type) :
Facturation.clientParticulier OR Facturation.clientProfessionnel

C - Contraintes liées au contenu

a. Contraintes check-age

On devra vérifier lors de l'implémentation que tous les conducteurs et les particuliers ont plus de 21 ans.

b. Enumérations

Certains attributs ne peuvent prendre des que des valeurs d'un domaine limité (comme par exemple le résultat de l'entretien, le modèle de véhicule). Cela pourra être spécifié lors de l'implémentation.

c. NOT NULL non liés à la cardinalités

Certains attributs doivent peut-être porter la contrainte de non nullité, pour préserver la clarté des données rentrées dans la base de données (par exemple, le nom et le prénom des employés). Nous pourrons revoir cela plus tard.

Début de la liste des attributs non nuls (à compléter) :

Agence : tous.
Employe : tous
SocieteEntretien : tous
Entretien : idEntretien, date_entretien
Client : idClient, num_bancaire, mail, adresse, nom
Conducteur : numPermis, nom, prenom, date_naissance
Location : idContrat, date_debut, km_parcourus
Facturation: idFacturation, etat

4) Points de clarification :

- ❖ Dans la relation entre Entreprise, LocationProfessionel et Conducteur, nous pensons à la situation où il existe un conducteur qui peut être lié à une location d'une entreprise différente de son entreprise actuelle. La solution peut être :
- Soit on supprime le lien entre Entreprise et LocationProfessionel, pour que l'entreprise puisse seulement effectuer des locations par intermédiaire des conducteurs.
- Soit on ajoute une contrainte afin que LocationProfessionel.entreprise et Conducteur.entreprise soient identiques.

- ❖ Dans la classe Facturation, nous avons mis des clés étrangères afin de pouvoir retrouver les clients correspondants pour une facturation. Cependant, le client associé à la facturation doit être le même que celui qui a effectué la location liée à cette facturation. On a donc décidé de mettre 2 clés étrangères vers la classe LocationParticulier et LocationProfessionnel plutôt que vers Entreprise et Particulier directement. Nous nous sommes demandé si ce n'était pas redondant dans ce cas, comme la classe Location est déjà liée à la classe Facturation.