TP9 : PROJET SPEEDCAR

Objectif du TP :

1. **Conception d'un BD**
   - Modélisation du MCD (Entité – Association – cardinalités)
2. **Déploiement de la BD**
   - Génération du script de création de la base de données (PostgreSql)
   - Modification du script (ajout des contraintes)
   - Création de la base de données sur Dbeaver
   - Création d'un jeu d'essai
3. **Exploitation de la BD**
   - Vues
   - Procédures – Fonctions
   - Triggers

Vous devez déposer vos réponses sur Moodle dans un dépôt

CONTEXTE

Une compagnie de transport de personnes *SPEEDCAR* envisage la mise en place d'une plateforme destinée aux particuliers. Cette application offrira des services similaires à ceux proposés par REBU :

- La plateforme répertorie les chauffeurs et les passagers, conservant diverses informations telles que l'adresse e-mail, le prénom, le nom, le « hash » du mot de passe, etc. Il est possible qu'un chauffeur en repos puisse également être passager.

- *SPEEDCAR* effectue une vérification de tous ses chauffeurs, exigeant une inscription préalable qui ne se transforme en compte chauffeur qu'après validation. *SPEEDCAR* souhaite une répartition géographique de ses chauffeurs….

- Pour afficher le périmètre d'intervention d'un chauffeur, l'adresse principale est conservée sous forme de longitude et latitude, avec le chauffeur responsable de choisir son périmètre d'intervention, généralement représenté par un cercle.

- Un passager propose une course partant d'un lieu et se rendant à un autre lieu, avec les emplacements stockés sous forme de couples longitude/latitude. La course a lieu à une date et une heure précises, le logiciel fournissant une estimation de l'heure d'arrivée.

- Les chauffeurs peuvent consulter les courses dans leur périmètre d'intervention, proposer leurs services en tant que candidats, et contacter le passager. Le

passager choisit ensuite le chauffeur en consultant les profils et les tarifs fournis par la plateforme. Le chauffeur est alors déclaré indisponible le temps de sa course.

- Dans la première version, le tarif du chauffeur est composé d'un montant fixe et d'un tarif kilométrique.
- La longueur du trajet pour chaque course est stockée, arrondie au kilomètre le plus proche grâce à l'API Google Maps. Cette longueur sert de base au calcul du tarif pratiqué par le chauffeur.
- *SPEEDCAR* prélève une commission d'apporteur d'affaires sur chaque candidature de chauffeur, avec un montant fixe fonction de la longueur du trajet, déterminé selon un barème progressif (par exemple, 5€ pour un trajet de 0 à 50 km exclus, 9€ pour un trajet de 50 à 100 km exclus, etc.), démontrant ainsi le souci de la société pour le respect de ses partenaires, chauffeurs et passagers.

PARTIE 1 : CONCEPTION

Modéliser la base de données du logiciel sous forme d'un MCD d'après les informations fournies.

Avoir recours si nécessaire à l'héritage et aux contraintes interrelation

>> image du MCD

PARTIE 2 : MISE EN OEUVRE

Générer le script SQL de création votre base de données pour PostgSql.

Garantir l'intégrité des données en complétant le script par l'expression des contraintes de domaine (not null – default – check - unique)

>> script de la bdd

En utilisant Dbeaver, créer le schéma et les tables.

>> image du modèle E/R de Dbeaver

PARTIE 3 : VUE

Une vue peut permettre de simplifier l'accès à certaines informations ou permet de préparer des résultats intermédiaires en combinant des données de plusieurs tables.

Produire les vues suivantes : >> script de la vue + résultat

1. **CoursesDetails** : qui rassemble des informations sur les courses, les passagers et les chauffeurs. Cette vue doit inclure les courses qui n'ont pas encore de chauffeur candidat.

2. **TarificationsChauffeurs** : qui permet de visualiser les tarifications associées à chaque chauffeur.

3. **CandidaturesCourses** : qui rassemble des informations sur les candidatures aux différentes courses.

4. **CommissionsCourses** : qui permet de visualiser les commissions associées à chaque course en fonction de la longueur du trajet.

PARTIE 4 : Procèdures et fonctions

1. **CalculerTarifCourse** : La fonction prendra l'ID de la course en paramètre et retournera le tarif total en fonction de la tarification du chauffeur associé.

2. **NombreCandidaturesMoisChauffeur** : La fonction prendra l'ID du chauffeur en paramètre et retournera le nombre de candidatures associées au cours d'un mois spécifié.
   Faire appel à la fonction EXTRACT pour obtenir le mois et l'année de la course.

3. **MettreAJourPerimetreChauffeur** : (2 solutions)
   a. Une procédure qui prendra l'ID du chauffeur et les nouvelles coordonnées (longitude et latitude) en paramètres, mettra à jour le périmètre d'intervention du chauffeur.
   b. Une fonction qui prendra l'ID du chauffeur et les nouvelles coordonnées (longitude et latitude) en paramètres, mettra à jour le périmètre d'intervention du chauffeur et renverra un message de succès.

4. **MettreAJourTarificationChauffeur** : La procédure prendra l'ID du chauffeur et les nouvelles informations de tarification, puis mettra à jour la tarification du chauffeur.

PARTIE 5 : Triggers

On souhaite stocker pour chaque chauffeur sa disponibilité ou non en temps réel, la date de sa dernière course, le nombres de courses effectué

Prévoir les triggers afin de :

1. T_**VerifierDisponibiliteChauffeur** : Vérifie si un chauffeur est disponible avant d'ajouter une nouvelle course.

2. T_**MajNombreCoursesChauffeur** : Ce déclencheur met à jour le nombre de courses d'un chauffeur chaque fois qu'une nouvelle course lui est affectée.

3. **T_ArchiverChauffeur** : archiver un chauffeur dans une table d'archive lorsqu'il est supprimé de la table des chauffeurs. Les données personnelles seront anonymisées.

4. T_**VerifierAgeMinimumPassager** : vérifier si un passager a l'âge minimum requis avant d'enregistrer une course. (au moins 18 ans)

PARTIE 5 : RGPD

1. **AnonymiserPassagersInactifs** : Créer une procédure stockée pour anonymiser les passagers qui n'ont pas effectué de course depuis plus d'un an en mettant à jour leurs informations personnelles (par exemple, en remplaçant leur adresse e-mail, prénom et nom par des valeurs anonymes).

2. **ArchiverCoursesPlusDeDeuxAns** : Créer une procédure pour archiver les courses qui ont plus de 2 ans.
   a. Créer une table ArchiveCourse avec une structure identique à courses
   b. La procédure sélectionne les courses et les insère dans la nouvelle table, puis supprime les courses achivées de la table d'origine

3. **trg_EnregistrerMotDePasseHistorique** :
   a. Créer une table pour stocker les mots de passe hashés des utilisateurs
   b. Créer un déclencheur (trigger) qui insérera dans cette table chaque modification du mot de passe dans la table Utilisateur.
   Le système devra empêcher l'insertion du nouveau mot de passe s'il est identique à l'ancien.