WITM20040400 - Marco Chan Witzel, LECM27050400 - Martin Lecerf, HATL17120400 - Lilian Hatton, POCC05020400 - Clément Pochart

Documentation - IntroDB

Exercice 1

1.1) La première partie crée une procédure stockée NouvelleAireStationnement qui insère une nouvelle université dans la table universite, puis crée un nouvel espace de stationnement dans la table espace_stationnement associé à cette université.
Ensuite, elle crée trois allées dans chaque espace de stationnement et dix places dans chaque allée. Les informations de l'université sont fournies en tant que paramètres à la procédure.

1.2) La deuxième partie de l'exercice crée une table log_aire_stationnement pour enregistrer chaque tentative d'insertion dans la table espace_stationnement.
Un déclencheur LogAireStationnement est créé pour insérer une entrée dans la table log_aire_stationnement chaque fois qu'une nouvelle ligne est insérée dans la table espace_stationnement. Cette entrée contient le nom de l'université, le sigle et la date et l'heure de la tentative d'insertion.

Cet exercice permet d'apprendre la syntaxe concernant les procédures stockées et les déclencheurs. La difficulté a été la compréhension et la réalisation des tests fonctionnels.

Exercice 2

2.1) NouvelEtudiant : Cette procédure prend en entrée plusieurs informations sur un étudiant (nom, prénom, code permanent, numéro de plaque, email, téléphone, supprimé, université) et insère ces informations dans la table `etudiant`. Avant l'insertion, la procédure vérifie que les informations ne sont pas nulles et qu'elles respectent certains formats (par exemple, le code permanent doit respecter le format ABCD00000000). De plus, si l'université n'existe pas dans la base de données, une erreur est signalée.

2.2) AfficherEtudiant : Cette procédure prend en entrée l'ID d'un étudiant et affiche toutes les informations de cet étudiant à partir de la table etudiant.

2.3) ModifierEtudiant : Cette procédure prend en entrée l'ID d'un étudiant et plusieurs autres informations (nom, prénom, code permanent, numéro de plaque, email, téléphone, université). Si une information n'est pas nulle, la procédure met à jour cette information pour l'étudiant dans la table sinon la colonne est ignorée. Comme la création des étudiants la procédure vérifie que les informations respectent certains formats avec des expressions régulières et que l'université existe dans la base de données. Chaque fois qu'une information d'un étudiant est modifiée, une nouvelle ligne est insérée dans cette table historique_etudiant avec les anciennes informations de l'étudiant et la date de modification.

2.4) SupprimerEtudiant : Cette procédure prend en entrée l'ID d'un étudiant et met à jour le champ `supprime` de cet étudiant à 1 dans la table `etudiant`, ce qui signifie que l'étudiant est supprimé.

Exercice 3

L'exercice 3 concerne la création d'une procédure stockée et d'un déclencheur en SQL pour gérer les réservations de places de stationnement.
Cet exercice simule bien l'utilisation de la base de données dans la peau d'un étudiant.
Voici une description détaillée de son fonctionnement :

`ReserverPlaceStationnement` : Cette procédure stockée prend en entrée l'ID d'un étudiant, une date et une heure de début et de fin.
Elle est conçue pour réserver une place de stationnement pour un étudiant à une date et une heure spécifiques.
- La procédure commence par vérifier si les données d'entrée sont valides. Si l'ID de l'étudiant, la date de début ou la date de fin sont nuls, une erreur est signalée.
- Ensuite, la procédure vérifie si l'étudiant a un cours pendant les heures de stationnement réservées. Si ce n'est pas le cas, une violation de stationnement est enregistrée dans une table `violation_stationnement` et une erreur est signalée.
- Si l'étudiant a un cours pendant les heures de stationnement réservées, la procédure cherche une place de stationnement disponible. Si aucune place n'est disponible, une erreur est signalée.
- Enfin, si une place est disponible, la procédure affiche les détails de la réservation, y compris l'espace de stationnement, l'allée, la place, le type de place, le montant à payer, la date et l'heure d'arrivée et de départ.

`UpdateAvailableSpaces` : C'est un déclencheur qui est activé après chaque insertion dans la table `place_reservee`.
Lorsqu'une réservation est effectuée, ce déclencheur met à jour le nombre de places disponibles dans l'allée correspondante.

Exercice 4

4.1) GenererIdentifiantEtudiant : On commence par compter le nombre d'étudiants existants. Ensuite, elle ajoute 1 à ce nombre pour obtenir un identifiant unique. Pour s'assurer que l'identifiant a toujours une longueur de 6 chiffres, elle utilise la fonction LPAD pour ajouter des zéros au début du nombre, puis elle ajoute le préfixe "ETU-" à l'identifiant.

4.2) Vue_Aires_Stationnement : Cette vue affiche des informations sur les aires de stationnement de chaque université. Pour chaque combinaison d'université, d'espace de stationnement et d'allée, elle compte le nombre de places disponibles et le nombre de places réservées.

4.3) MettreAJourDisponibilite : C'est un événement qui est déclenché toutes les 5 minutes. Il met à jour la disponibilité des places de stationnement. Il commence par identifier les places dont toutes les réservations sont terminées. Pour ce faire, il crée une table temporaire PlacesDisponibles qui contient les identifiants des places dont toutes les réservations sont terminées. Ensuite, il met à jour la disponibilité de ces places à Oui dans la table place. Enfin, il supprime la table temporaire PlacesDisponibles.

Exercice 5

L'exercice 5 consiste en une procédure stockée SQL nommée`RapportAiresStationnement`.
Cette procédure génère un rapport sur les aires de stationnement pour chaque université. Voici une description détaillée de son fonctionnement :

`RapportAiresStationnement` : Cette procédure ne prend aucun argument en entrée. Elle génère un rapport sur les aires de stationnement pour chaque université. Voici les étapes de son fonctionnement :
- Elle déclare plusieurs variables pour stocker les informations sur chaque université et les statistiques sur les aires de stationnement.
- Elle déclare un curseur pour parcourir toutes les universités dans la table `universite`.
- Elle ouvre le curseur et entre dans une boucle pour parcourir les universités.
- Pour chaque université, elle calcule plusieurs statistiques, y compris le nombre d'étudiants, le nombre d'espaces de stationnement, le nombre d'agents de surveillance, le nombre d'allées, le nombre de places, le nombre de places pour handicapés, le nombre de places disponibles, le nombre de places réservées, le nombre moyen de réservations en 2023, la date ayant eu le plus de réservations et la date ayant eu le moins de réservations.
- Elle affiche des statistiques pour chaque université.
- Elle continue à parcourir les universités jusqu'à ce qu'il n'y en ait plus.
- Enfin, elle ferme le curseur.

Cet exercice permet de visualiser les statistiques des données et est donc essentiel.

Exercice 6

Enfin, nous avons réalisé la sauvegarde selon l’aide fournie et à travers MySQL.