Base de données - Cours II
Exercices - Option R

I - Gestion des Passagers

On voudrait ajouter une table de passagers à la base **BDVols.**

Un passager est décrit par son numéro, son nom et son prénom. Ces attributs seront respectivement représentés par :

- un champ **NumeroP** de type **INT**
- un champ **Nom** de type **VARCHAR(20)**.
- un champ **Prenom** de type **VARCHAR(20)**.

La table qui contient ces données se nomme **Passagers**.

---

1) Création de la table

Ecrire la requête permettant de créer la table passagers dans un fichier texte que vous importerez sous PhpMyAdmin.

Pour les questions suivantes vous aurez à recharger plusieurs fois ce fichier. Comme, il est impossible re créer une table qui existe déjà, insérez les mots **IF NOT EXISTS**, juste après **CREATE TABLE**. Avec cette option, la requête **CREATE TABLE** n'est pas exécutée si la table existe déjà.

D'autre part, à chaque chargement du fichier il faudrait que la table soit initialement vide. Pour cela placez la requête **DELETE FROM Passagers** juste après la requête **CREATE TABLE**.

2) Adjonction des passagers

Avec une seule requête **INSERT INTO**, ajoutez trois passagers dans la table de manière à obtenir:

NumeroP	Nom	Prenom
1	Thirion	Eric
2	Gaston	Lagaffe
3	Clapton	Eric

3) Correction de l'erreur à la deuxième ligne

Dans la deuxième ligne, le nom et le prénom sont inversés. Corrigez cette erreur en utilisant les requêtes **DELETE** et **INSERT**.

NumeroP	Nom	Prenom
1	Thirion	Eric
2	Lagaffe	Gaston
3	Clapton	Eric

4) Modification des noms et des prénoms

Sans utiliser les numéros des passagers et en utilisant uniquement la requête **UPDATE**, modifiez la table passager afin d'obtenir le résultat suivant:

NumeroP	Nom	Prenom
1	Thirion	Alex
2	Deferre	Gaston
3	Clapton	Alex

Le corrigé de cette exercice se trouve dans le fichier **Passagers.sql**

II - Modification de la table Avions

Modifiez la table **Avions** de manière à ce qu'elle contienne les trois colonnes suivantes :

NumeroAvion	TypeAvion	MiseEnService
1	Airbus/A320	1998
2	Airbus/A320	1990
3	Airbus/A320	1994
4	Airbus/A320	1983
5	Airbus/A300	1990
6	Airbus/A300	1985
7	Airbus/A300	1991

- La colonne **NumeroAvion** est l'ancienne colonne **NumeroA** renommée (son type était INT).
- La colonne **TypeAvion** est une concaténation des anciennes colonnes **Constructeur** et **Modèle**. On pourra utiliser ici la fonction **CONCAT(x,y)** (cf fonctions sur les chaines - cours I).
- La colonne **MiseEnService** est une nouvelle colonne qui contient l'année de mise en service de l'avion. Pour affecter des valeurs à cette colonne on pourra utiliser la fonction **RAND()** (cf fonctions mathématiques - cours I)

Pour cela, on effectuera successivement les opérations suivantes:

1. Adjonction d'une nouvelle colonne **TypeAvion** représentant le type de l'avion, en remplissant également cette colonne avec des données Constructeur/Modèle.
2. Adjonction d'une nouvelle colonne **MiseEnService** contenant l'année de mise en service de l'avion.
3. Suppression des anciennes colonnes **Compagnie**, **Capacite**, **Constructeu**r et **Modele**.
4. Renomage de la colonne **NumeroA** en **NumeroAvion**.

Le corrigé de cet exercice se trouve dans le fichier **Modif-Avion.sql**

III- Gestion des réservations

On voudrait à présent pouvoir réserver un vol pour un passager donné.

Une réservation est représentée par le numéro du passager qui a reservé et le numéro du vol qu'il a réservé.

Ces attributs seront respectivement représentés par :

- un champ **Passager** de type **INT**.
- un champ **Vol** de type **INT**.

La table contenant ces données se nomme **Reservations**.

Pour cet exercice, vous écrirez vos requêtes dans le fichier **Reservations.sql** du dossier **Etudiant**.

Ce fichier contient déjà des requêtes SQL pour créer la table de réservation.

Un autre fichier, **PassagersCelebres.sql** contient les requêtes de création de la table **Passagers** à partir d'un fichier texte **Passagers2.txt.**

Avant de commencer cet exercice, effectuez d'abord les opérations suivantes:

1. Modification du chemin du fichier **Passagers2.txt**

   Ouvrez le fichier **PassagersCelebres.sql** (dossier Etudiant)**.** Le remplissage de la table des passagers se fait par la requête **LOAD DATA INFILE.** Les données de la table sont lues à partir du fichier **Passagers2.txt** qui se trouve dans le même répertoire. Modifiez le chemin d'accès afin qu'il corresponde à l'emplacement du fichier **Passager2.txt** sur votre ordinateur.

2. Création de la table **Passagers** sous PhpMyAdmin.

   Après avoir sélectionné la base **BDVols**, importez le fichier **PassagersCelebres.sql**, afin de créer la table **Passagers**. Cette opération ne doit être effectuée qu'une seule fois.

3. Ouvrez le fichier **Reservations.sql** et passez à la question 1.

Remarque: vous constaterez que les créations des tables **Reservations** et **Passagers** se terminent par **ENGINE=InnoDB**. Cela signifie que ces tables seront créées en format **InnoDB**. Ce format est nécessaire pour le fonctionnement de l'intégrité référentielle.

1) Réservations

Pour le vol 11, effectuez des réservations pour les passagers 1,6, 11,16, 21, 31 et 41.

Pour le vol 34, effectuez des réservations pour les passagers 1, 7, 11, 17, 21, 32, et 41.

Pour pouvoir mieux visualiser les réservations on créera une vue nommée **VueReservations** possédant exactement les colonnes suivantes:

- **NV**: Numéro du vol.
- **Depart**: Ville de départ du vol.
- **Arrivee**: Ville d'arrivée du vol.
- **Jour**: Date du vol.
- **HD**: Heure de départ du vol.
- **HA**: Heure de d'arrivée du vol.
- **NP**: Numéro du passager.
- **Nom**: Nom du passager.
- **Prenom**: Prénom du passager.

La vue est trié par numéro de vol, puis par nom de passager. Vous obtiendrez alors le résultat suivant:

NV	Depart	Arrivee	Jour	HD	HA	NP	Nom	Prenom
11	Paris	Strasbourg	2001-09-04	10	11	31	Applegate	Christina
11	Paris	Strasbourg	2001-09-04	10	11	6	Cox	Courtney
11	Paris	Strasbourg	2001-09-04	10	11	16	Electra	Carmen
11	Paris	Strasbourg	2001-09-04	10	11	21	Hayek	Salma
11	Paris	Strasbourg	2001-09-04	10	11	1	Holmes	Katie
11	Paris	Strasbourg	2001-09-04	10	11	11	Monroe	Marilyn
11	Paris	Strasbourg	2001-09-04	10	11	41	Pfeiffer	Michelle
34	Strasbourg	Paris	2001-09-05	8	9	21	Hayek	Salma
34	Strasbourg	Paris	2001-09-05	8	9	1	Holmes	Katie
34	Strasbourg	Paris	2001-09-05	8	9	7	Lopez	Jennifer
34	Strasbourg	Paris	2001-09-05	8	9	32	Milano	Alyssa
34	Strasbourg	Paris	2001-09-05	8	9	11	Monroe	Marilyn
34	Strasbourg	Paris	2001-09-05	8	9	17	Moss	Carrie-Anne
34	Strasbourg	Paris	2001-09-05	8	9	41	Pfeiffer	Michelle

2) Propagation des mises à jours sur la vue des réservations

Vérifiez que toutes les modifications apportées aux réservations, aux vols ou aux passagers sont automatiquement propagées à la vues. Par exemple:

- Supprimez les réservations de Marylin Monroe. Puis vérifiez qu'elle n'apparaissent plus dans la vue.
- Modifiez le prénom de Katie Holmes. Remplacez le par Sherlock. Vérifiez qu'il est également modifié dans la vue.

3) Adjonction d'une contrainte de clé primaire

Pour l'instant, il est possible d'effectuer deux fois la même réservation (même passager - même vol).

Afin d'éviter ceci, ajoutez une contrainte de clé primaire sur deux colonnes (**Passager** et **Vol**) dans la table **Réservation**.

Vérifiez ensuite qu'il n'est plus possible d'effectuer deux fois la même réservation.

En réservant de nouveau le vol 11 pour le passager 16, vous obtiendrez le message d'erreur suivant:

Duplicate entry '16-11' for key 'PRIMARY'

Mettez ensuite cette instruction en commentaire, afin que l'erreur ne se reproduise plus aux prochains chargement du fichier.

---

4) Adjonction de contraintes d'intégrité référentielle

Ajoutez une contrainte d'intérgrité référentielle entre la colonne **Passagers** de la table **Reservations** et la colonne **NumeroP** de la table **Passagers**, avec les options **ON DELETE CASCADE** et **ON UPDATE CASCADE**.

Vérifiez tout d'abord que l'intégrité référentielle est à présent bien contrôlée par MySQL. En réservant un vol

pour un passager inexistant, vous devriez obtenir un message d'erreur.

Par exemple, si vous réservez le vol 11 pour le passager 2000, vous obtniendrez le message suivant:

Cannot add or update a child row: a foreign key constraint fails
(`bdvols`.`reservations`, CONSTRAINT `reservations_ibfk_1` FOREIGN KEY (`Passager`)
REFERENCES `passagers` (`NumeroP`))

Mettez ensuite cette instruction en commentaire, puis vérifiez que l'option **ON DELETE CASCADE** fonctionne bien. Par exemple, si on supprime le passager Michelle Pfeiffer (passager 41), on constatera que toutes les réservations de ce passager ont automaquement été supprimées. Attention: il faut vérifier cet effet sur la table **Reservations** et non pas sur la vue.

Après avoir mis l'instruction précédente en commentaire, vérifiez cette fois que l'option **ON UPDATE CASCADE** fonctionne bien. Vous pouvez par exemple remplacer le numéro du passager Applegate Christina (31) par 2031, puis vérifez que ce remplacement a automatiquement été répercuté dans la table **Reservations**.

Bug non élucidé: pour que cette opération fonctionne, il faut regénérer la base BDVols et recharger le fichier **PassagersCelebres.sql**.

Le corrigé de cet exercice se trouve dans le fichier **Reservations.sql**.

IV - Extraits d'études de cas

Cas Tholdi - 2009 - Option R - Dossier 4

Extraits du schéma relationnel de la base:

INTERVENIR (numEmp, numRepar, nbHeures)
numEmp, numRepar : Clé primaire
numEmp : Clé étrangère faisant référence à numero de la table EMPLOYE
numRepar : Clé étrangère faisant référence à numero de la table REPARATION

Dictionnaire des données partiel:

Table	*Code*	*Type*	*Longueur*
EMPLOYE	numero	Numérique	5
EMPLOYE	nom	Chaîne	25
REPARATION	numero	Numérique	5
REPARATION	dateDeb	Date	
INTERVENIR	nbHeures	Numérique	3

Écrire la requête SQL permettant de créer la table INTERVENIR en gérant les contraintes d'intégrité de clé primaire et de clés étrangères

Remarque: il faudra utiliser ici le type **NUMBER** qui existe en ORACLE, mais pas en MySQL. **NUMBER(c,d)** définit un nombre de **c** chiffres et **d** décimales.

Cas Asdomi - 2010 - Option D - Dossier 2

Extraits du schéma relationnel de la base:

INTERVENTION (dateIntervention, heureDébut, matriculeSalarié, duréeIntervention, nbKm,
noDossier, noVéhicule)

dateIntervention, heureDébut, matriculeSalarié : clé primaire
matriculeSalarié : clé étrangère en référence à matricule de SALARIE
noDossier : clé étrangère en référence à numéro de DOSSIER

noVéhicule : clé étrangère en référence à numéro de VEHICULE

Écrire le(s) ordre(s) SQL pour créer la table INTERVENTION et les contraintes d'intégrité qui concernent cette table.