Université du Littoral Côte d'Opale

1ère année Master en Informatique

Base de données avancées
JDBC – Multi-bases

I. Écrivez un programme Java qui se connecte à une base de données relationnelle (~~MySQL ou~~ ORACLE) pour exécuter les commandes SQL suivantes puis afficher les résultats :

```
#
# Structure for the `vol` table :
#
CREATE TABLE vol (
NumVol VARCHAR(8),
Heure_depart DATETIME,
Heure_arrive DATETIME,
Ville_depart VARCHAR(20),
Ville_arrivee VARCHAR(20),
PRIMARY KEY (NumVol)
);
INSERT INTO vol VALUES ('AF118', '08:30', '10:57','Paris','Athens');
INSERT INTO vol VALUES ('AF212','09:21','14:10','Paris','Moscow');
INSERT INTO vol VALUES ('AF178','12:56','14:15','Paris','London');
INSERT INTO vol VALUES ('AF010','07:53','14:19','Paris','New York');
INSERT INTO vol VALUES ('AF012','07:58','20:10','Paris','Los Angeles');
INSERT INTO vol VALUES ('AF001','22:10','12:00','Paris','Tahiti');
INSERT INTO vol VALUES ('SA854','22:00','10:14','Singapore','Athens');
INSERT INTO vol VALUES ('AA111','15:45','21:10','Beijing','Singapore');
INSERT INTO vol VALUES ('SA012','07:57','11:26','Sydney','Singapore');
INSERT INTO vol VALUES ('AF109','07:39','14:10','Tahiti','Sydney');
INSERT INTO vol VALUES ('AA517','23:57','07:12','Honolulu','Tokyo');
INSERT INTO vol VALUES ('JA014','15:35','19:00','Tokyo','Beijing');
INSERT INTO vol VALUES ('JA115','21:26','10:10','Los Angeles', 'Tokyo');
INSERT INTO vol VALUES ('AA015','20:50','07:00','New York', 'Lima');
INSERT INTO vol VALUES ('AA515','07:20','12:38', 'Lima', 'Los Angeles');
INSERT INTO vol VALUES ('AA516','15:20','21:38', 'Beijing', 'Athens');
INSERT INTO vol VALUES ('AF218','21:12','09:16','Beijing','Paris');
INSERT INTO vol VALUES ('AA118','07:15','13:10','New York', 'Paris');
INSERT INTO vol VALUES ('TA215','08:00','10:10','Tunis','Paris');
INSERT INTO vol VALUES ('OA005','14:20','17:00','Athens','Paris');
INSERT INTO vol VALUES ('PA022','10:12','23:55','Lima','Paris');
INSERT INTO vol VALUES ('AF002','15:52','00:12','Tokyo','Paris');
```

II. Ecrivez un programme Java qui, pour chaque enregistrement de la table Vol, insère un nouvel enregistrement pour le vol retour associé : au Numvol on ajoute le caractère 'R' à la fin, les horaires sont recalculés en considérant un départ 2 heures après l'arrivée (ou sont laissés vides...), la ville de départ et la ville d'arrivée sont échangées.

Par exemple, à partir de l'enregistrement :

AF118 08:30 10:57 Paris Athens

On insérera l'enregistrement :

AF118R 12:57 15:24 Athens Paris

III. Écrivez un programme Java qui se connecte à une base de données relationnelle (MySQL) pour créer une table « Escales » avec les données indiquées ci-dessous (MySQL) puis les afficher.

La table Escales :

Num_escale	Ville_escale	Duree_escale_minimum
1	Moscou	5 H
2	Singapour	5 H
3	Sydney	4 H
4	Tahiti	4 H
5	Honolulu	4 H
6	Los Angeles	5 H
7	New York	4 H
8	Beijing	3 H
8	Lima	3 H
8	London	3 H

IV. En utilisant la table Vol (base de données Oracle), écrivez un programme Java qui propose des vols pour un tour du monde au départ de Paris (ou d'une ville choisie par l'utilisateur), avec des escales et des durées prévisionnelles d'escale définies dans la table Escales (MySQL).

Éventuellement, enregistrez les données du tour du monde dans une base de données graphe.

Pour cet exercice, je vous propose d'utiliser l'interface RowSet (qui est une extension de ResultSet) déconnecté. C'est-à-dire que pendant tout le temps de la manipulation des données, les données utilisées proviennent du rowset déconnecté, et pas directement de la base.

NB : Ce n'est pas le cas ici mais pour enregistrer les modifications de l'utilisateur le rowset doit être reconnecté à la base de données. Lors de cette reconnexion, le rowset enregistre uniquement les lignes modifiées par l'utilisateur. Les rowsets sont

particulièrement utiles lorsque les données de la base doivent transiter entre plusieurs couches distantes ; dans de nombreuses applications, l'interface utilisateur est située sur une autre machine que l'ordinateur qui héberge le SGBD.

ATTN : L'enregistrement des modifications dans la base à la reconnexion du rowset doivent tenir compte des éventuels conflits dus à des modifications effectuées par d'autres utilisateurs pendant la déconnexion du rowset. S'il y a des conflits, le programme demande à l'utilisateur (par une fenêtre de dialogue) s'il veut conserver les données actuellement dans la base ou les écraser par les modifications qu'il a effectuées. Pour tester la gestion des conflits, vous modifierez en direct dans la base, certaines des lignes que vous modifiez en parallèle par l'interface graphique de l'application Java.