IUT2 UNIVERSITÉ Grenoble Alpes Département INFO

1ère année – S2d
Année 2019-2020

M2106

Préambule

Création des relations de la base de données

On rappelle le schéma de la base de données permettant de gérer les informations de l'association "Le Tourmentin".

Adherent (numadh, nom, prenom, fonction, adresse, telephone, skipper, anneeadh)

Cotisation (numadh, anneecot, montant, paye)

Bateau (numbat, nombat, taille, typebat, nbplaces)

Agence (nomagence, telephone, fax, adresse)

Proprietaire (numadh, numbat)

Loueur (nomagence, numbat)

Activite (numact, typeact, depart, arrivee, datedebut, datefin)

Chefdebord (numact, numadh, numbat)

Equipage (numact, numadh, numbat)

Regate(numact, numregate, forcevent)

Resultat (numbat, numact, numregate, classement, points)

1. Créer un répertoire **~/M2106** et y recopier les fichiers *create.sql*, *drop.sql* et *insert.sql* (disponibles dans **/users/info/pub/s2/M2106/voile**).
2. Se placer dans le répertoire **~/M2106**, puis se connecter à votre base postgresql par la commande (par exemple pour l'utilisateur **users2d01**),

psql -h postgres-info -U users2d*01* bases2d*01*

ATTENTION A BIEN UTILISER VOTRE PROPRE LOGIN COMMUNIQUE PAR L'ENSEIGNANT !!!

3. Exécuter successivement les fichiers *drop.sql*, *create.sql* et *insert.sql* (**\i**) pour créer les tables de la base de données et y insérer les données.
4. Exécuter la commande SQL de mise à jour suivante pour donner votre login comme nom à l'adhérent numéro 10 :

```
update adherent set nom = getpgusername() where numadh = 10;
```

TP 1 – Procédures en PL/pgSQL

Fichier de réponses

Pour conserver trace de votre travail et pouvoir discuter de vos solutions avec votre enseignant, vous écrirez les énoncés des questions (en commentaires entre `/*` et `*/`) suivis de vos solutions dans un fichier `~/M2106/tp1.sql`. Nous vous recommandons d'éditer ce fichier puis d'effectuer des copier-coller dans le terminal où s'exécute votre connexion postgresql.

Organisation du bureau

Nous vous recommandons l'organisation suivante de votre bureau :

- Fenêtre 1 : sujet de votre tp, à gauche sur toute la hauteur de l'écran ;
- Fenêtre 2 : éditeur de texte pour la mise à jour de `tp1.sql`, à droite sur la moitié supérieure de l'écran ;
- Fenêtre 3 : terminal où s'exécute votre connexion postgresql, à droite sur la moitié inférieure de l'écran.

Question 1 : Ecrire une fonction `NbAnneesCot()` qui, étant donné un numéro d'adhérent, retourne le nombre d'années où il a cotisé (`paye='oui'`).
***Expériences* :** Appeler la fonction `NbAnneesCot()` pour l'adhérent numéro 1.

Question 2 : Ecrire une fonction `MonNumero()` qui, sans argument, donne le numéro d'adhérent de l'utilisateur connecté.
***Expériences* :** Créer (`insert`) une nouvelle sortie de numéro 9 (la supprimer au préalable si une activité existe déjà avec ce même numéro), partant dans 10 jours (`current_date + 10`) de Vannes et arrivant dans 12 jours à Vannes. S'inscrire (`insert`) en tant que chef de bord du bateau numéro 4 sur cette sortie, sans jamais connaître son propre numéro d'adhérent (pour cela utiliser la fonction `MonNumero()`). Tester (`select`) la réussite de votre commande.

Question 3 : Ecrire une fonction `MonBateau()` qui, étant donné un numéro d'activité, donne le numéro du bateau pour lequel l'utilisateur connecté est chef de bord. Si un tel bateau n'existe pas, lever une exception et afficher un message expliquant l'erreur rencontrée. Pour écrire cette fonction, vous utiliserez la fonction précédente `MonNumero()`.
***Expériences* :** Tester cette fonction avec l'activité de numéro 2 (pour laquelle vous êtes chef de bord d'un bateau) puis avec l'activité de numéro 5 (cas inverse).

Question 4 : Un skipper connecté souhaite reconduire tout son équipage d'une activité à une autre. Pour l'aider, écrire une procédure `ReconduireEquipe()` qui, étant donnés deux numéros d'activité, inscrit tous les membres d'équipage du bateau dirigé par l'utilisateur connecté pour la première activité, dans l'équipage que ce dernier dirige pour la deuxième activité donnée. Pour écrire cette fonction, vous utiliserez la fonction précédente `MonBateau()`.
***Expériences* :** Reconduire l'équipage que vous aviez sur l'activité 2 dans la nouvelle activité 9.
***Indication* :** On peut insérer dans une table le résultat d'une requête par la commande

```
insert into table_destination select … ;
```

Question 5 : Le secrétaire souhaite enregistrer un nouveau rallye de *n* régates sans avoir à renseigner le numéro de cette nouvelle activité. Pour l'aider, écrire une fonction `NouveauRallye()` qui, étant donnés un lieu de départ et d'arrivée, une date de début et de fin, et un nombre de régates, crée un nouveau rallye correspondant dont le numéro est supérieur au plus grand numéro d'activité enregistré jusque-là, puis crée les régates associées. Finalement cette fonction renverra le numéro de la nouvelle activité ainsi insérée.
***Expériences* :** Créer un nouveau rallye partant dans 30 jours de Vannes, arrivant dans 32 jours à Vannes, et comportant 4 régates, le tout grâce à la fonction `NouveauRallye()`.