

Bases de données – TP 1

Licence 3 d’informatique 2011–2012

Premier TP

Utilisation de `psql`

PostgreSQL (www.postgresql.org) est installé sur la machine `nivose`. Pour y accéder, il faut donc vous connecter à `nivose` :
`ssh nivose` (utiliser votre mot de passe Unix).

Pour pouvoir appeler PostgreSQL, lors de la première séance il faut ajouter `/usr/local/pgsql/bin` à votre `PATH`. Le mieux est de l’ajouter à votre fichier `.bash_profile` pour l’avoir de manière permanente :

```
cat > .bash_profile
export PATH=/usr/local/pgsql/bin:$PATH
```

(suivi de `Ctrl-D` pour sortir de `cat`). Enfin, il faut sortir de `nivose` (`exit`) et se reconnecter pour que la modification prenne effet.

Vous avez chacun un nom d’utilisateur et une base de données : le nom d’utilisateur et le nom de la base est votre login sous Unix, le mot de passe est

Pour entrer dans votre base, tapez :

```
psql nomUtilisateur nomBase
ou plus simplement
psql
et donnez le mot de passe .....
```

Pour changer votre mot de passe, tapez :
`alter user votre_login with password 'NouveauPass';`

Pour obtenir de l’aide sur les commandes propres à PostgreSQL, taper
`<votre login> => \?`

Pour obtenir de l’aide sur les commandes SQL :
`<votre login> => \h`

Pour exécuter une liste de commandes SQL sauvegardée dans un fichier appelé `commandes.sql` :
`<votre login> => \i commandes.sql`

Pour quitter PostgreSQL, il faudra taper
`<votre login> => \q`

`psql` n’est pas sensible à la casse des lettres (le fait que ce soit une majuscule ou une minuscule). Vous pouvez utiliser la tabulation pour obtenir une complétion automatique.

Exercice 1 – Création d'une table

1. Créer la table `client` avec les attributs :
 - `nom` : chaîne de caractères représentant le nom du client,
 - `prenom` : chaîne de caractères représentant le prénom du client,
 - `adr` : chaîne de caractères représentant l'adresse du client,
 - `ss` : chaîne de caractères représentant le numéro de sécurité sociale du client.
2. Entrer ensuite 3 données dans la table `client` avec la commande `INSERT` et afficher la table avec la commande `SELECT`.
3. Effacer la table avec la commande `DROP TABLE`.
4. Créer de nouveau la table `client` avec les mêmes attributs que précédemment et en ajoutant l'attribut `id_client`, entier représentant l'identifiant du client. Entrer ensuite 3 données dans la table `client` dont deux ont le même identifiant.
5. On souhaite ajouter la contrainte d'unicité pour l'attribut `id_client`, c'est-à-dire qu'on ne veut pas que deux entrées dans la table puissent avoir le même identifiant. Utiliser la commande `ALTER TABLE nom_table ADD UNIQUE (attribut)`.
Que se passe-t-il et pourquoi? Modifier la base en utilisant la commande `DELETE` pour qu'il n'y ait plus de problème.
6. On souhaite en fait que l'attribut `id_client` soit la clé primaire de la table `client`. Utiliser la commande `ALTER TABLE nom_table ADD PRIMARY KEY (attribut)`.
7. Effacer la table et la recréer directement avec `id_client` comme clé primaire (sans passer par `ALTER TABLE`) et en obligeant que le nom soit renseigné (contrainte `NOT NULL`).
8. Essayer d'insérer des valeurs dans la table `client` avec des `id_client` identiques, des noms vides, etc.

Exercice 2 – Création d'une base pour la gestion d'un magasin pour la location de films

1. Se rappeler de la modélisation de la location de film dans le TD1.
2. Donner les tables nécessaires pour mettre en œuvre la base.
3. Créer alors les tables en précisant à chaque fois la clé primaire et en utilisant les clés étrangères pour représenter les relations.

Exercice 3 – Création de bases pour les autres exercices du TD1. Faire comme dans l'exercice précédent pour les autres exercices du TD1.