ENSI CAEN
ÉCOLE PUBLIQUE D'INGÉNIEURS
CENTRE DE RECHERCHE

1ère année Informatique

– Bases de Données/ PostgreSQL –
Sujet n°6

Sujet	THEME	Durée TD	Durée TP
6	Création et suppression de tables, contraintes d'intégrité, indexes, contrôles des accès, dictionnaire de données, importation de données dans les tables, sauvegarde, restauration		**2h**

Myriam Mokhtari-Brun

ENSICAEN **Myriam Mokhtari-Brun**
Informatique 1ère **année**

TP Bases de Données N°6
- Création et Suppression de tables -
- Contraintes d'intégrité -
- Indexes -
- Contrôle des accès -
- Dictionnaire de données -
- Chargement -
- Exportation/importation -

I. CREATION ET SUPPRESSION DE TABLES

Donner le script de création des tables de *Clinique* (au moins 4 tables : type_animal, departement, client, animal) :

- Les contraintes à respecter sont celles indiquées dans le sujet qui décrit la base de Données *Clinique* : s'inspirer de la Description de la base, de la Liste des informations et du Modèle Physique de Données.
- Pour les contraintes d'intégrité référentielles, préciser s'il doit y avoir des suppressions en cascade ou non.

Exécutez ce script sous *psql* de POSTGRES. Au besoin, corrigez-le et relancez son exécution.

- Avant chaque "CREATE TABLE ..." du script, ajouter l'ordre de destruction de la table (DROP TABLE ...), pour pouvoir relancer l'exécution du script sans avoir à gérer les erreurs dues aux tables déjà créées.

- Contrôler la création des tables par utilisation :
  - de la commande *\d* sous *psql* pour avoir toutes les tables
  - de la commande *\d nom_table* sous *psql* pour avoir tous les champs de la table *nom_table*
  - du dictionnaire de données : SELECT sur la table système pg_indexes pour avoir les indexes
    SELECT sur la table système pg_tables pour avoir les tables

- Remplissez vos tables avec quelques lignes cohérentes et d'autres incohérentes (c-à-d ne vérifiant pas les contraintes d'intégrité précisées lors des créations.
- Observez les messages d'erreurs produits par POSTGRES. Corriger ces lignes pour qu'elles puissent être insérées dans les tables.

II. CREATION ET SUPPRESSION D'INDEXES

- Index existants

  Quels sont les index générés automatiquement par POSTGRES ? (utiliser la vue PG_INDEXES du dictionnaire de données).

- Ajout d'index
  - Créer les indexes sur les clés étrangères de chaque table. Utilité ?
  - Créer un index *nomcl_ind* sur le nom du client dans la table *client.* Cet index est bien utile si on souhaite rapidement parcourir la table par ordre alphabétique des noms de client.

  - Créer un index sur les valeurs décroissantes de la date de visite et pour chaque date sur l'ordre croissant des numéros d'animaux, utile si l'on veut lister rapidement les visites selon cet ordre.

  - Supposons que l'instruction suivante soit souvent exécutée. Elle permet de comptabiliser par client le nombre d'animaux de poids important nés sur une période données :

```
SELECT numcl, COUNT(idani)
FROM animal
WHERE daten BETWEEN TO_DATE('12-06-2004','DD-MM-YYYY') AND current_date
  AND poids > 1300
GROUP BY numcl;
```

    Créer un index utile à l'exécution répétée de cette requête.

III. ACCORD ET RETRAIT DE PRIVILEGES

- Privilège de lecture de la table *client* :
  Lecture autorisée à l'ensemble des utilisateurs et à certains utilisateurs nommés. Tests.
  Reprise du privilège à l'ensemble des utilisateurs, sauf aux utilisateurs nommés. Tests.

- Privilège d'insertion dans la table *client* :
  Insertions autorisées à votre binôme voisin. Tests.
  Reprise du privilège. Tests.

- Privilèges de modification de la table *animal* :
  Modification autorisée à votre binôme voisin. Tests
  Reprise du privilège. Tests.

- Privilège de référence sur la table *traitement* :
  Référence autorisée à votre binôme voisin. Tests.
  Reprise du privilège. Tests.

- Privilège de lecture de la table *client* en cascade :
  Lecture autorisée à votre binôme voisin, en lui permettant de redistribuer ce privilège à d'autres. Tests.
  Reprise du privilège. Tests.

- Reprise de tous les privilèges. Tests.

IV. CHARGEMENT DES TABLES

Le chargement des tables peut être effectué avec la commande **COPY** sous psql à partir des fichiers textes situés sous le répertoire indiqué par votre enseignant. Cette commande a été mise dans chacun des fichiers *.sql indiqués ci dessous :

Table	**Fichier**
departement	departement.sql
client	client.sql
type_animal	type_animal.sql
animal	animal.sql
visite	visite.sql
detail_visite	detail_visite.sql
medicament	medicament.sql
traitement	traitement.sql

- Ouvrir un des fichiers *.sql et comprendre son contenu.
- Pour exécuter chaque fichier, dans un terminal, aller sous psql et taper : \i *nomfichier.sql*
- Attention à l'ordre de chargement de la liste des tables (comprendre pourquoi ?).
- Dans les fichiers sql, attention au format des dates, à l'ordre des colonnes, au format des nombres réels (format américain avec point décimal), …

- Des erreurs peuvent être détectées par l'outil du chargement. Par exemple : doublon dans la colonne clé primaire, valeur de clé étrangère ne faisant référence à aucune ligne dans la table "maître", date erronée, contraintes de valeurs non respectées, etc.... Les corriger mais ne relancer le chargement que pour les tables non correctement remplies (interdit de relancer d'abord le script de création des tables, utiliser plutôt des DELETE FROM .... et/ou des ALTER TABLE ....).

- Vérifier que tous les enregistrements ont bien été chargés dans leur table destination (avec un SELECT)

V. SAUVEGARDE, RESTAURATION (sous POSTGRES)

- Sauvegarder de la base par l'utilitaire **pg_dump** sous unix.
- Détruire exprès la table *detail_visite*.
- Restaurer cette table détruite en utilisant **pg_restore** sous unix. Vérifier.

VI. EXPLOITATION DE LA BASE

Dorénavant, cette base de données vous appartient. Vous pouvez l'utiliser comme bon vous semble.