TP SÉCURITE DES DONNÉES IMIS 2021/2022 (B.NGUYEN)
BASES STATISTIQUES ET CHIFFREMENT

Objectif :

Ce TP est une introduction aux méthodes de chiffrements pour la gestion de données, exécutée sur Oracle et de génération de données anonymes. Dans une première partie nous abordons le problème de la gestion de données sensibles dans une base de données statistique. La seconde partie sera consacrée à l'utilisation des packages cryptographiques (chiffrement, déchiffrement, hachage, ...) disponibles au niveau du SGBD, que nous utiliserons pour protéger des mots de passes, une colonne sensible, et pour générer des données anonymes.

Préparation :

Dans ce TP (salle EX9), nous utilisons un docker qui nous fournit une version d'Oracle XE.

Voici la procédure pour lancer Oracle :

1) Récupérez le container docker pull `oracleinanutshell/oracle-xe-11g` (met 2 minutes à télécharger , le téléchargement a déjà fait par Pascal Pautrat sur chaque PC de la salle E09, faites-le avant le TP sur votre PC personnel linux , pour ne pas perdre de temps en début de TP)
2) Lancez le container : `docker run -d -p 49161:1521 oracleinanutshell/oracle-xe-11g`
3) Connectez vous à la base, en créant une connexion **en tant qu'administrateur** sur SQL Developper avec les paramètres suivants :

   hostname: localhost
   port: 49161
   sid: xe
   username: system
   password: oracle

4) Téléchargez le fichier `tp_sbd.zip` disponible sur mon site web ici : https://benjamin-nguyen.fr/ENS/IMIS/ et stockez les scripts en local quelque part où ils sont accessibles (par exemple en créant un répertoire `/home/tplocal/sbd`). Vous pouvez le récupérer en ligne de commande en faisant : wget https://benjamin-nguyen.fr/ENS/IMIS/tp_sbd.zip

Rappel : lorsque vous êtes connecté sur Oracle vous pouvez utiliser la commande `@/home/tplocal/sbd/STAT.sql` pour lancer un script PL/SQL localisé dans le répertoire `/home/tplocal/sbd/`

Note : Les scripts créent des nouveaux utilisateurs (par exemple l'utilisateur `USER_STAT` dans la question 1). Il est impératif de se connecter avec cet utilisateur dans une autre fenêtre, sans quoi si vous vous connectez en tant qu'administrateur, vous aurez bien évidemment accès à toutes les tables de la base de données.

PARTIE 1 - Introduction.

Dans cette partie, nous considérons la table statistique et la vue suivante:

- Table STATS: name varchar(50), gender char(1), age NUMBER, insurance varchar(50), leucocyte NUMBER. Le champ 'name' contient le nom d'un patient, 'gender' son genre (H ou F), 'age' son age, 'insurance' sa compagnie d'assurance (ex: MGEN, MATMUT, MAIF, ...), et 'leucocyte' son taux de leucocyte qui est considérée comme une données privée ultra sensible.
- Vue STATS_VIEW: gender char(1), insurance varchar(50), leucocyte NUMBER. Il s'agit d'une projection de la table STATS sur les colonnes gender, insurance et leucocyte.

Q1 : Utilisation de fonctions statistiques

Connectez-vous en tant qu'administrateur (utilisateur `system`).

Lancez le script STAT.sql

Connectez vous en tant qu'utilisateur USER_STAT (mot de passe : oracle)
Cet utilisateur a-t-il accès à la table STATS et à la vue STATS_VIEW ?

Cet utilisateur a accès aux fonctions WHERE_CLAUSE( ), ROW_COUNT( ), SUM_LEUCOCYTE( ) et peut interroger la vue STATS_VIEW uniquement au travers de ces fonctions. La première fonction permet de visualiser une clause WHERE formée à partir de la clause donnée en paramètre, la seconde donne le nombre de patients correspondant satisfaisant une clause WHERE donnée, et la troisième donne leur taux de leucocyte.

Vous pouvez lancer ces fonctions via la requête SQL suivante:

```
SELECT <propriétaire_fonction>.<nom_fonction> (' <contenu_clause_where> ' ) from dual;
```

Ces fonctions prennent en paramètre une chaine de caractères (clause <contenue_clause_where>) indiquant la clause WHERE d'une requête SQL posée sur une vue STATS_VIEW. La clause <contenue_clause_where> pourra être par exemple : assurance = "MATMUT" (attention: le symbole " est obtenu par deux guillemets simples successifs). Vous devrez consulter la table ALL_OBJETS pour trouver l'utilisateur propriétaire de ces fonctions permettant l'appel.

Quel est le nombre de patients assurés à la MATMUT, et la somme de leur taux de leucocyte ?
Quel est le nombre total de patients dans la base ?

Q2 : Attaque de fonctions statistiques non protégées

Vous savez que le patient Dubois est un homme qui est assuré à la MGEN.
Utiliser les fonctions statistiques pour obtenir le taux de leucocyte de Dubois.

Q3 : Attaque de fonctions statistiques protégées

Fermez la fenêtre USER_STAT et déconnectez bien USER_STAT. Si vous ne déconnectez pas USER_STAT, le script ne donnera pas les bons droits (on ne peut pas effacer un utilisateur connecté).
Connectez-vous en tant qu'administrateur.
Lancez le script STAT_PROTECT_1.sql

Connectez-vous en tant qu'utilisateur USER_STAT.
Relancer l'attaque conduite à la section précédente. Fonctionne-t-elle toujours ? Pourquoi ?
Trouver une séquence d'appels aux fonctions statistiques protégées permettant quand même d'obtenir le taux de leucocyte de Dubois. Vous pourrez par exemple calculer la somme des leucocytes de toutes les personnes *sauf* Dubois, puis la somme totale de leucocytes.

Q4 : Attaques de fonctions statistiques protégées

Fermez la fenêtre USER_STAT et déconnectez bien USER_STAT.
Connectez-vous en tant qu'administrateur.
Lancez le script STAT_PROTECT_2.sql

Connectez-vous en tant qu'utilisateur USER_STAT.
Relancer l'attaque conduite à la section précédente. Fonctionne-t-elle toujours ? Pourquoi ?
Trouver une séquence d'appels aux fonctions statistiques protégées permettant d'obtenir quand même le taux de leucocyte de Dubois.

PARTIE 2 - Anonymat et Chiffrement (package Oracle de chiffrement)

Dans cette partie, nous utiliserons le package DBMS_OBFUSCATION_TOOLKIT.

Q5 : Pseudonymat par chiffrement

Connectez-vous en tant qu'administrateur.
Créez une table CLEF avec un champs VAL de type CHAR(8) dans laquelle vous stockerez une clé (secrète) que vous choisirez.
Le script CRYPT.sql crée les fonctions ENCRYPT et DECRYPT que vous pourrez utiliser.

Lancez le script CRYPT.sql

Si le script a été correctement modifié, vous obtenez en sortie:

```
Function created.
PADING('TEST')
------------------------------------
------------------------------------
test----
Function created.
ENCRYPT('TEST','12345678')
------------------------------------
------------------------------------
2FD0D87C2F466C21
Function created.
DECRYPT('2FD0D87C2F466C21','12345678')
------------------------------------
------------------------------------
test
```

Tester ces fonctions en utilisant la table DUAL sur d'autres exemples.
Insérez le contenu de la table STATS dans la table STAT_ANONYM en remplaçant les noms des patients par une valeur anonyme obtenue par chiffrement du nom avec votre clé secrète.

Q6 : Attaque au pseudonymat par chiffrement

Connectez-vous en tant qu'utilisateur USER_STAT.
En supposant que cet utilisateur connaisse le nom d'un patient comme précédemment, peut-il retrouver les données correspondant à ce patient dans la table STAT_ANONYM ?

Lancez le script LISTING.sql

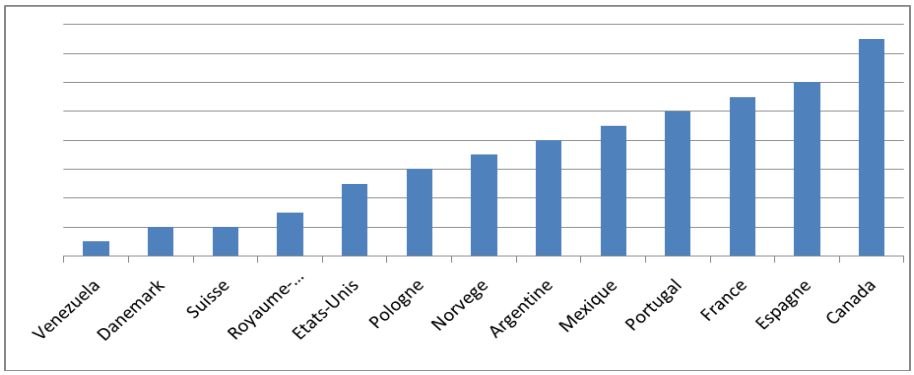
Une table LISTING contenant une série de noms potentiels avec leur genre, age, et compagnie d'assurance, vous est fournie dans le script LISTING.sql.
Dé-anonymiser la table STAT_ANONYM en utilisant LISTING.sql.
Combien de lignes de la table STAT_ANONYM parvenez-vous à dé-anonymiser ?
Est-il possible a l'aide du contrôle d'accès de se protéger contre ce type d'attaque ? Si oui, comment, si non pourquoi ?

Q7 : Chiffrement d'une colonne sensible

Connectez-vous en tant qu'administrateur.

Lancez le script CLIENT.sql

Ce script crée la table CLI avec 91 clients.
Créer une table CLI_CRYPT sur le même modèle que la table CLI, mais en chiffrant la colonne PAYS de la table CLI avec la clé secrète.
Donnez les droits nécessaires à l'utilisateur USER_STAT pour qu'il puisse accéder à la table CLI_CRYPT.
Connectez-vous en tant qu'utilisateur USER_STAT.
Vous avez accès à au graphique suivant montrant la distribution des pays dans la clientèle.



A partir de ce graphique, déchiffrer la colonne pays.

Q8 : **Chiffrement d'une colonne sensible**

Connectez-vous en tant qu'administrateur.
Chiffrez à nouveau la colonne PAYS pour résister à cette attaque.
Connectez-vous en tant qu'utilisateur USER_STAT.
Montrez que l'attaque de Q7 n'est plus opérante.

Q9 : **2-Anonymat (bonus)**

On considère que l'attribut : NomCli est un attribut **identifiant**. On considère que Tel est un attribut **identifiant**. On considère que NumCli est un attribut **non-sensible**. On considère que l'ensemble {Pays} est un **quasi-identifiant**.

Connectez-vous en tant qu'administrateur.
Modifiez le contenu de la table STAT_ANONYM de manière à la rendre 2-anonyme. Vous indiquerez les modifications que vous faites avec des requêtes de type UPDATE.
Connectez-vous en tant qu'utilisateur USER_STAT.
Montrer que l'attaque de Q7 n'est plus opérante.