COLLÈGE DE
BOIS-DE-BOULOGNE

Enseignants : Diallo Abdourahmane, Mohammed Salim Meflah
Sujet : Travail pratique 2 : Sécurité des applications
Correction : sur 100
Valeur : 25% du semestre
Date de remise : vendredi 10 décembre, fin de la journée (23h59) Mode de remise : via Léa

Objectif

Ce travail pratique consiste à sécuriser une base de données Oracle dans un contexte de gestion hospitalière. L'étudiant devra créer une base pluggable, configurer les utilisateurs, activer un audit complet, chiffrer et déchiffrer des colonnes sensibles, effectuer une sauvegarde avec Oracle Data Pump, puis développer une petite application Python avec PyQt6 pour interagir avec la base de manière sécurisée. Le travail se réalise en binôme.

PARTIE 1 : Création de la Base de Données Oracle

Créer une base pluggable, un utilisateur administrateur et concevoir le schéma de la base.

1. Créer une base pluggable nommée Hopital_db dans un environnement Oracle multitenant.
2. Créer un utilisateur nommé Hopital_admin et lui attribuer les privilèges nécessaires pour la gestion de la base.
3. Créer les tables suivantes dans le schéma Hopital_admin : Patients, Medecins et Consultations.
4. Insérer au moins quatre enregistrements de test dans chaque table.
5. Afficher la structure des tables et les données insérées.

PARTIE 2 : Audit avancé de la Base de Données

Mettre en place un audit complet pour surveiller les connexions, les modifications de données et les changements de privilèges.

6. Activer l'audit global au niveau du système Oracle.
7. Configurer l'audit pour suivre les connexions réussies et échouées à la base.
8. Configurer l'audit pour enregistrer toutes les opérations (INSERT, UPDATE, DELETE) sur les tables Patients et Consultations.

9. Activer l'audit sur la gestion des rôles et privilèges attribués aux utilisateurs.
10. Créer une table personnalisée nommée Audit_Action contenant les informations suivantes : (audit_id, username, action, objet, date_action).
11. Créer un trigger d'audit qui enregistre automatiquement toute modification sur la table Patients dans la table Audit_Action.
12. Effectuer quelques opérations sur la base et vérifier les enregistrements dans les logs d'audit et dans la table Audit_Action.

PARTIE 3 : Chiffrement et Déchiffrement des Données Sensibles

Appliquer le chiffrement et le déchiffrement sur des colonnes contenant des données sensibles.

13. Identifier dans le schéma les colonnes contenant des données sensibles (par exemple : num_secu, diagnostic, traitement).
14. Écrire un script SQL de chiffrement en utilisant le package DBMS_CRYPTO.
15. Écrire le script de déchiffrement correspondant pour récupérer les données en clair.
16. Vérifier que le déchiffrement permet bien de retrouver les valeurs d'origine.
17. Documenter les résultats et joindre les captures d'écran montrant le texte avant chiffrement, après chiffrement, et après déchiffrement.
18. Chiffrement transparent TDE Assurez-vous que les utilisateurs autorisés peuvent accéder et déchiffrer les données d'une manière transparente avec l'algorithme AES128 ou AES256. Utilisez Oracle Wallet pour gérer les clés de chiffrement transparent.
19. **Comparaison :** Quelle différence y'a-t-il entre les méthodes de chiffrement

PARTIE 4 : Sauvegarde et Restauration avec Oracle Data Pump

Sauvegarder et restaurer le schéma de la base de données avec les outils expdp et impdp.

20. Créer un répertoire logique Oracle pour stocker les sauvegardes Data Pump.
21. Effectuer une sauvegarde complète du schéma Hopital_admin avec la commande expdp.
22. Supprimer temporairement une des tables du schéma (ex. Consultations) pour simuler une perte de données.
23. Restaurer la table supprimée en utilisant la commande impdp.
24. Vérifier la restauration et documenter les étapes par des captures d'écran.

PARTIE 5 : Application Python avec PyQt6

Développer une application graphique en Python permettant de se connecter à la base Oracle et d'interagir avec les données de manière sécurisée.

25. Installer les modules nécessaires : PyQt6, cx_Oracle et hashlib.

26. Créer une interface graphique comprenant un formulaire de connexion, un tableau d'affichage des données, un champ de saisie SQL, un champ de chiffrement SHA256 et une section affichant les journaux d'audit.
27. Utiliser des requêtes paramétrées pour éviter les injections SQL.
28. Tester la robustesse de l'application avec des requêtes valides et malveillantes.

Critères d'Évaluation

Critère	Points
Création de la base de données et du schéma	10
Insertion et affichage des données	5
Configuration et vérification de l'audit	20
Chiffrement et déchiffrement des données	15
Sauvegarde et restauration avec Data Pump	10
Application PyQt6 fonctionnelle	15
Sécurité et tests d'injection SQL	10
Rapport clair et structure : page de garde, table de matière, introduction, conclusion, et clarté du rapport.	15
Total	100

Livrables

• Scripts SQL complets (création, audit, chiffrement, sauvegarde)

• Script Python de l'application PyQt6

• Captures d'écran de toutes les étapes principales dans le rapports

• Rapport synthétique présentant les résultats et observations

Important : ajouter vos initiales dans tous les objets que vous créez