

Institut Supérieur du Numérique (SupNum) Nouakchott – Mauritanie

Niveau : L2 Matière : Bases de Données Avancées – Oracle

Durée : 1h30

Document non autorisé

Enseignant : Ingénieur Sidi Mahmoud KABER

SUJET D'EXAMEN - BASES DE DONNÉES AVANCÉES (ORACLE)

CONTEXTE : SYSTÈME DE COMPENSATION INTERBANCAIRE (SCI-RIM)

L'administrateur principal des bases de données de l'entité « **GIMTEL** », relevant de l'autorité de la Banque Centrale de Mauritanie (**BCM**), est chargé de la gestion, entre autres, des transactions de retrait par carte à puce entre les différentes banques nationales (**BPM**, **BNM**, **BAMIS**, etc.).

Votre mission est de mettre en place l'infrastructure de stockage, la sécurité et la logique métier (PL/SQL) de la base de données centrale **DB_SWITCH**.

Schéma Relationnel Simplifié

Pour cet examen, nous nous concentrerons sur les trois tables critiques suivantes :

1. **BANQUES** (Identifie les banques membres)
 - ID_BANQUE (NUMBER, PK) : Identifiant unique.
 - NOM_BANQUE (VARCHAR2) : Nom (ex : 'BPM').
 - SOLDE_COMPENSATION (NUMBER) : Solde de la banque auprès de la **BCM** (doit rester positif).
2. **CARTES** (Cartes à puce actives)
 - NUM_CARTE (VARCHAR2(16), PK) : Numéro de la carte.
 - ID_BANQUE (NUMBER, FK) : Banque émettrice.
 - DATE_EXPIRATION (DATE) : Date de fin de validité.
 - STATUT (NUMBER) : 1=Active, 0=Bloquée.
3. **TRANSACTIONS** (Historique des retraits)
 - ID_TX (NUMBER, PK) : Identifiant transaction.
 - NUM_CARTE (VARCHAR2, FK) : Carte utilisée.
 - MONTANT (NUMBER) : Montant du retrait en MRU.
 - DATE_TX (DATE) : Date et heure de l'opération.
 - ETAT (VARCHAR2) : 'SUCCES' ou 'ECHEC'.

PARTIE I : GESTION DU STOCKAGE ET DES TABLESPACES (5 Points)

Afin d'optimiser les performances (I/O) et la sécurité, vous devez séparer physiquement les données transactionnelles des données statiques et des journaux système.

Question 1 (2 pts) :

Écrivez la commande SQL permettant de créer un tablespace nommé TBS_GAB_DATA destiné à stocker les tables volumineuses (comme TRANSACTIONS).

- Le fichier de données doit se nommer gab_data01.dbf.
- La taille initiale est de **500 Mo**.
- Le fichier doit pouvoir s'étendre automatiquement par tranches de **50 Mo**.

Question 2 (1.5 pts) :

On souhaite isoler les index pour améliorer la performance des recherches lors des retraits. Écrivez la commande pour créer un tablespace TBS_GAB_INDEX de **200 Mo** fixe (sans extension automatique).

Question 3 (1.5 pts) :

Expliquez, en 3 lignes maximum, l'intérêt pour une banque de séparer le tablespace des données (TBS_GAB_DATA) de celui des index (TBS_GAB_INDEX) sur des disques physiques différents.

PARTIE II : SÉCURITÉ ET GESTION DES UTILISATEURS (4 Points)

La sécurité est primordiale. Seuls les agents agréés et les processus automatiques doivent accéder aux données.

Question 4 (1.5 pts) :

Créez un utilisateur Oracle nommé AGENT_SUPERVISEUR identifié par le mot de passe SupNum2024 ! Cet utilisateur doit utiliser le tablespace TBS_GAB_DATA par défaut.

Question 5 (1 pt) :

Le superviseur doit pouvoir se connecter à la base et créer des tables, mais il ne doit **pas** pouvoir supprimer la base de données (DROP DATABASE).

Donnez les commandes pour lui attribuer les privilèges système strictement nécessaires (rôles standards ou privilèges spécifiques).

Question 6 (1.5 pts) :

Pour renforcer la sécurité, créez un RÔLE nommé ROLE_AUDITEUR.

Attribuez à ce rôle uniquement le droit de **lecture** (SELECT) sur la table TRANSACTIONS et la table BANQUES. Attribuez ensuite ce rôle à l'utilisateur AGENT_SUPERVISEUR.

PARTIE III : DÉVELOPPEMENT PL/SQL AVANCÉ (11 Points)

C'est le cœur du système. Vous devez automatiser les contrôles et les traitements.

Question 7 : Fonction de vérification (3 pts)

Créez une fonction stockée nommée FN_VERIF_CARTE qui prend en paramètre un numéro de carte (p_num_carte).

La fonction doit renvoyer un **booléen** (ou un entier 1/0) :

- Retourne **TRUE** (ou 1) si la carte existe, si son STATUT est à 1 (Active) et si la DATE_EXPIRATION est supérieure à la date du jour (SYSDATE).
- Retourne **FALSE** (ou 0) dans tous les autres cas.

Question 8 : Procédure de traitement de retrait (5 pts)

Écrivez la procédure stockée SP_TRAITEMENT_RETRAIT qui gère une demande de retrait.

- **Paramètres d'entrée** : p_carte (Varchar2), p_montant (Number).
- **Algorithme à implémenter** :
 1. Vérifier si la carte est valide (utilisez la fonction FN_VERIF_CARTE). Si invalide, lever une exception personnalisée avec le message "Carte invalide ou expirée".
 2. Vérifier si le montant du retrait est positif et ne dépasse pas le plafond standard de 10 000 MRU. Sinon, erreur "Montant non autorisé".
 3. Insérer une ligne dans la table TRANSACTIONS avec l'état 'SUCCES' et la date actuelle.
 4. Valider la transaction (COMMIT).
 5. **Gestion des erreurs (EXCEPTION)** : En cas d'erreur (carte invalide ou autre problème technique), insérer la transaction avec l'état 'ECHEC' et ne pas propager l'erreur bloquante (le guichet doit juste afficher "Refusé").

Question 9 : Déclencheur (Trigger) de sécurité (3 pts)

Pour lutter contre la fraude, la direction demande une surveillance accrue.

Créez un déclencheur (Trigger) nommé TRG_ALERTE_GROS_RETRAIT.

- **Type** : Après insertion (AFTER INSERT) sur la table TRANSACTIONS.
- **Condition** : Se déclenche uniquement si le MONTANT est supérieur ou égal à 20 000 MRU.
- **Action** : Le trigger doit insérer une alerte dans une table JOURNAL_ALERTE (supposée existante avec les champs message et date_alerte) indiquant : "Retrait suspect sur la carte XXXXX".

Bonne chance et bonne réflexion !