

---

# IMPLÉMENTATION DES SGBDs L2

TP3

Verrous et transactions sur Oracle

---

## Schéma

Considérons le schéma de base de données suivant :

```
CCheque(client, solde)
CEpargne(client, solde)
DecisionCredit(dossier, client, decision)
```

## Exercices

**Avant de commencer, connectez-vous à Oracle sur votre machine virtuelle en suivant les instructions disponibles sur Moodle.**

### Exercice 1. Création du schéma

Créer les tables du schéma en choisissant le type de chaque colonne à l'aide des informations suivantes :

- **CCheque** et **CEpargne** contiennent le numéro du client propriétaire du compte et son solde qui peut être négatif.
- **DecisionCredit** contient le dossier des demandes de crédit de chaque client avec la décision associée (OK si elle est acceptée, KO sinon).

### Exercice 2. Problèmes de concurrence

Au cours de cet exercice, assurez-vous que suffisamment de données soient insérées dans la base pour répondre aux questions. On supposera que chaque client a déjà un compte chèque et un compte épargne.

*Question 1.* On souhaite implémenter la procédure **epargner(c, m)** qui permet de transférer le montant *m* du compte chèque d'un client vers son compte épargne, uniquement dans le cas où le solde du compte chèque reste positif. Pour se faire, nous allons utiliser le langage PL/SQL, en complétant la procédure ci-dessous :

```
— Exemple de bloc PL/SQL

CREATE OR REPLACE PROCEDURE epargner (c NUMBER, m NUMBER) AS
  — Quelques variables
  v_solde NUMBER;
BEGIN
  — récupérer le montant du compte chèque
  SELECT solde INTO v_solde FROM CCheque WHERE client=c;
  — Affichage
  DBMS_OUTPUT.PUT_LINE ( 'Solde Cheque : ' || v_solde );
END;
/
```

Pour voir les sorties de la procédure, il faut activer leur affichage avec :

```
set serveroutput on
```

Vous pouvez exécuter cette procédure avec la commande suivante :

```
execute epargner(1, 100);
```

*Question 2.* Modifier votre procédure pour `epargner(c NUMBER, m NUMBER, delai NUMBER)` en insérant la commande `DBMS_SESSION.sleep(delai);` pour simuler un délai entre certaines des opérations effectuée. Utilisez ce délai pour obtenir une exécution en parallèle de deux de ces transferts vers le compte épargne d'un même client telles que le solde chèque du client devienne négatif à cause d'une lecture non reproductible.

*Question 3.* Implémentez une nouvelle procédure `traitementCredit(c, m)`, qui inscrit une nouvelle décision de crédit. Si le montant de crédit demandé  $m$  est 3 fois supérieur à la somme des soldes de ces comptes chèques et épargne, le crédit est accepté, sinon, la demande de crédit est rejetée.

*Question 4.* Ajoutez aussi un délai à la procédure `traitementCredit`. Exécuter en parallèle un traitement de crédit avec un transfert, si bien que le crédit soit accepté ou refusé alors qu'il n'aurait pas dû l'être.

**Exercice 3.** Oracle permet d'appliquer un verrou sur une table dans une transaction pour limiter les opérations des autres transactions sur cette table. Il y a deux modes de verrous :

- **SHARE** permet les lectures depuis les autres transactions, mais bloque les écritures,
- **EXCLUSIVE** bloque les écritures et les lectures des autres transactions.

On pose un verrou en utilisant une commande de la forme suivante :

```
LOCK TABLE DecisionCredit IN SHARE MODE;
```

À la fin d'une transaction, tous les verrous déposés sont relâchés.

*Question 5.* Tester les compatibilités entre les différents modes de verrous. Est-ce qu'une transaction peut prendre un verrou **SHARE** ou **EXCLUSIVE** lorsqu'une autre transaction détient un verrou **SHARE** ou **EXCLUSIVE** sur la même table ?

*Question 6.* Introduisez des verrous dans les procédures de l'exercice précédent pour éviter les problèmes de concurrence que l'on a observé.

*Question 7.* Que se passe-t-il lors que vous effectuez des exécutions d'épargne en parallèle sur des comptes différents (utiliser un long délai pour vous rendre compte) ? Est-ce que cela vous semble nécessaire ?

*Question 8.* Vous pouvez utiliser une requête de la forme `SELECT .... FOR UPDATE` pour sélectionner les lignes sur lesquelles appliquer un verrou **EXCLUSIVE**. Utiliser ce type de commande pour éviter le problème observé à la question précédente.

*Question 9.* Sans définir de verrou, comment résoudre les problèmes de concurrence observés dans l'exercice précédent ?