

## ANNEXE TP2

On peut paramétrer la gestion de la concurrence au niveau système et au niveau des transactions.

Le niveau d'isolation d'une transaction peut être indiqué par :

```
SET TRANSACTION ISOLATION LEVEL { SERIALIZABLE | READ COMMITTED } ;
```

```
SET TRANSACTION { READ ONLY | READ WRITE } ;
```

cette instruction doit être la première de la transaction.

### **read committed**

c'est le niveau par défaut. Toute modification validée avant le début de l'instruction en cours est visible par cette instruction. La sérialisabilité de la transaction n'est pas garantie.

### **read only**

ce niveau est propre à Oracle. Une transaction **read only** ne voit que les modifications validées avant son début. Une telle transaction ne peut pas modifier la base et elle n'est jamais bloquée.

### **serializable**

Ce niveau est comme **read only**, mais en plus la transaction  $T$  peut modifier des nuplets. Si  $T$  tente de modifier (**delete** ou **update**) un nuplet dont la dernière version a été produite par une autre transaction validée après le démarrage de  $T$ , Oracle considère qu'il y a un problème potentiel de sérialisabilité et provoque donc l'annulation de l'instruction et la propagation d'une erreur Oracle (-08177).

L'exécution d'une instruction de modification de table ou contrainte forme automatiquement une transaction : elle provoque une validation implicite en début et en fin d'exécution.

## Pose implicite de verrous par Oracle

**insert**, **delete** ou **update** provoque un verrouillage intentionnel **row exclusive** de la table, puis verrouille les nuplets impliqués.

**select ... for update** fait de même mais pose un verrou intentionnel **row share** sur la table. Si une telle exécution tente de poser un verrou sur un nuplet déjà verrouillé par une autre transaction, elle est bloquée jusqu'à la terminaison de cette autre transaction. Lors du déblocage l'ensemble des nuplets sélectionnés est réévalué.

## Pose explicite de verrous sous Oracle

Il est parfois nécessaire de verrouiller explicitement les données avec la commande **lock table**.

```
lock table <table> in { exclusive | share } mode [nowait] ;
```

### exclusive

les autres transactions ne peuvent que lire la table (mises à jour et poses de verrou bloquées).

### share

bloque, pour les autres transactions, la pose du verrou intentionnel **row exclusive**, et la pose de verrou **exclusive**.

### nowait

plutôt que de bloquer l'exécution si la table est déjà verrouillée, Oracle abandonne l'instruction en signalant l'erreur -54 : la transaction peut alors faire autre chose avant de retenter le verrouillage.

	X	SRX	S	RX	RS	mot clef	commentaire
<b>X</b>	-	-	-	-	-	<b>exclusive</b>	
<b>SRX</b>	-	-	-	-	+	<b>share row exclusive</b>	
<b>S</b>	-	-	+	-	+	<b>share</b>	
<b>RX</b>	-	-	-	+	+	<b>row exclusive</b>	intentionnel <b>update, delete, insert</b>
<b>RS</b>	-	+	+	+	+	<b>row share</b>	intentionnel <b>select ... for update</b>

