

## **Traitement des requêtes sous Oracle**

### **Notes :**

1. Conserver les sources à des fins de ré-utilisation.
2. Conserver les résultats à des fins de démonstration.

### **Exemples de génération de plan:**

#### **1. Exemple de génération de plan SANS nommage de requête :**

```
SQL> explain plan for
  2 select * from produit where prod# = 700;
```

#### **Explicité.**

#### **2. Exemple d'affichage des résultats SANS nommage de requête :**

```
SQL> column ID format 99;
SQL> column parent format 99;
SQL> column Objet format A10;
SQL> column operation format A20 ;
SQL> column options format A10;
SQL>
  1 select '|', id "ID", '|', parent_id "parent", '|', operation, '|',
  2      object_name "Objet", '|', options, '|'
  3* from plan_table;
```

'	ID	'	parent	'	OPERATION	'	Objet	'	OPTIONS	'
	0				SELECT STATEMENT					
	1		0		TABLE ACCESS		PRODUIT		FULL	

#### **3. Exemple de génération de plan AVEC nommage de requête :**

```
SQL> explain plan set statement_id = 'Rqte1' for
  2 select * from produit where prod# = 700;
```

#### **Explicité.**

#### **4. Exemple d'affichage des résultats AVEC nommage de requête :**

```
  1 select '|', id "ID", '|', parent_id "parent", '|', operation, '|',
  2      object_name "Objet", '|', options, '|'
  3* from plan_table where statement_id = 'Rqte1'
```

```
SQL> /
```

ID	parent	OPERATION	Objet	OPTIONS
0		SELECT STATEMENT		
1	0	TABLE ACCESS	PRODUIT	FULL

Cette technique est adaptée lorsqu'on veut conserver des plans d'exécution car Oracle ne "vide pas" la PLAN\_TABLE. En effet, chaque plan généré est ajouté à ceux existants dans la PLAN\_TABLE.

On pourra évidemment afficher d'autres attributs, **comme des attributs ayant trait aux coûts de la requête** (Unité centrale, entrées-sorties, ...).

### Session Traitement des requêtes

- Créer les relations** suivantes (voir les scripts 1-Createdepot.sql, 2-Createproduit.sql, 3-Creastock.sql)

```
create table produit ( prod# int not null, libelle varchar(20), pu float not null)
create table depot ( dep# int not null, adr varchar(20), capacite int)
create table stock ( prod# int not null, dep# int not null, qte int not null)
```

- Créer les procédures d'instanciation** (SQL> **start** chemin-vers-le-source-de-la-procédure):  
**Sources des procédures fournis.**
  - Une procédure (Script : 12-ProcInitProd.sql) qui permet d'instancier la relation produit. Cette procédure aura comme argument *NbProd*, le nombre de produits à insérer. Les tuples de la relation sont de la forme <i, 'Produit\_i', i>,  $i \in [1 .. NbProd]$ .
  - Une procédure (11-ProcInitDep.sql) qui permet d'instancier la relation depot. De la même façon, cette procédure aura un argument *NbDep* désignant le nombre de dépôts à insérer. Les tuples de la relation sont de la forme <j, 'Depot Numero j', 10\*j>,  $j \in [1 .. NbDep]$ .
  - Une procédure (13-ProcInitStock.sql) qui permet d'instancier la relation stock Cette procédure aura comme argument *NbrStock* désignant le nombre moyen de dépôts dans lesquels est stocké un même produit. Les tuples de la relation sont de la forme <prod#, dep#, x\*prod#>.
- Instancier les relations** [SQL> **exec** NomProcédure(Paramètre effectif)] avec un nombre conséquent de produits (*NbProd* > 1500), de dépôts (*NbDep* >= 20) en considérant qu'un produit est stocké en moyenne dans 10 dépôts. **L'instanciation de la relation stock doit être faite après celles des relations produit et dépôt.**

**Exemples :**

- `exec InitProd(2000) ;`
- `exec InitDep(20) ;`
- `exec InitStock(10) ;`

4. **Engendrer et examiner les plans** d'exécution des requêtes ci-dessous. Intégrer dans l'examen des plans des attributs de coût. **Conserver les plans pour usage/comparaison ultérieurs.**
5. **Créer un index** sur produit.prod# et un autre sur (stock.prod#, stock.dep#).
6. Quels sont les nombres de niveaux d'index pour chaque index créé ? Expliquer.
7. Quel est le nombre de blocs libres de chaque table ? Expliquer.
8. **Engendrer, examiner les plans d'exécution des requêtes 5, 6, 7 et comparer ces plans** avec ceux engendrés sans les index.
9. **Collecter des statistiques** sur les tables et sur les index (cf. **ANALYZE TABLE** ou **ANALYZE INDEX**).
10. Quels sont les nombres de niveaux d'index pour chaque index créé ?
11. Quel est le nombre de blocs libres de chaque table ?
12. **Engendrer, examiner les plans d'exécution des requêtes 5, 6, 7 et comparer ces plans** avec ceux engendrés sans les index ni les statistiques.
13. **Faire le ménage :** Supprimer les tables.

### - Liste des requêtes -

**Rqte2 :** select \* from produit where prod# = 700 or prod# = 800

**Rqte3 :** select \* from produit where prod# in (700, 800)

**Rqte4 :** select d.adr from depot d, stock s  
where d.dep# = 4 and d.dep# = s.dep#  
and s.prod# > 100 and s.prod# < 500;

**Rqte5 :** select d.adr from depot d where d.dep# = 4  
and d.dep# in (select s.dep# from stock s  
where s.prod# > 100 and s.prod# < 500)

**Rqte6 :** select prod#, sum(qte) "Total en stock" from stock  
where dep# > 10 and dep# < 100 group by prod#  
order by prod#;

**Requête avec vue :** Créer la vue ci-dessous puis engendrer et examiner le plan de la requête 7.

create view V as (select libelle, prod# from produit where pu > 100)

**Rqte7 :** select V.prod#, V.libelle, p.pu, s.qte  
from produit p, V, stock s  
where p.prod# = V.prod# and V.prod# = s.prod#;