

Indexation sous Oracle

UE fondements des bases de données - TP4

Vous trouverez de l'aide pour ce TP notamment sur le site d'Oracle :

- aller à <http://www.oracle.com/pls/db102/>
- Dans "Most Popular" cliquez sur "Performance tuning guide"
- si besoin est, enregistrez-vous pour accéder à l'aide.

1 Visualisation du plan d'exécution et du coût estimé d'une requête

Lors de l'envoi d'une requête SQL, celle-ci est analysée syntaxiquement et sémantiquement par Oracle. L'optimiseur de requêtes génère alors un plan d'exécution, puis la requête est exécutée. Il est possible de demander à Oracle de générer un plan d'exécution sans exécuter la requête, puis de visualiser ce plan d'exécution. La procédure est la suivante :

- Créer la table destinée à recevoir les détails du plan d'exécution. Cette requête de création peut-être trouvée dans l'aide en ligne d'oracle atteignable à partir de www.oracle.com.

```
create table PLAN_TABLE (  
statement_id varchar2(30),  
timestamp date,  
remarks varchar2(80),  
operation varchar2(30),  
options varchar2(30),  
object_node varchar2(128),  
object_owner varchar2(30),  
object_name varchar2(30),  
object_instance numeric,  
object_type varchar2(30),  
optimizer varchar2(255),  
search_columns number,  
id numeric,  
parent_id numeric,  
position numeric,  
cost numeric,  
cardinality numeric,  
bytes numeric,  
other_tag varchar2(255),  
partition_start varchar2(255),  
partition_stop varchar2(255),  
partition_id numeric,  
other long,  
distribution varchar2(30));
```

- Demander à Oracle le plan d'exécution retenu pour une requête, qu'il stocke dans la table PLAN_TABLE :

```
EXPLAIN PLAN  
Set statement_id='identifiant_requete'  
For une_requete;
```

- Puis interroger la table PLAN_TABLE de façon à obtenir un plan d'exécution lisible, par exemple :

```
SELECT LPAD(' ',2*(LEVEL-1))||operation||' '||options  
||' '||object_name  
||' '||DECODE(id, 0, 'Cost = '||position) "Query Plan"  
FROM plan_table  
START WITH id = 0 AND statement_id = 'identifiant_requete'  
CONNECT BY PRIOR id = parent_id AND statement_id = 'identifiant_requete';
```

1. Utilisez le script suivant pour créer des tables de tests Studio et film. La première recense un ensemble de studios avec leur numéro unique, et si oui(1) ou non(0) ils sont localisés en Europe. La deuxième un ensemble de films avec leur numéro unique et leur studio d'enregistrement.

Ces tables devront-être détruites à la fin du TP.

```
create table studio(NumStudio number(4), European number(1));  
declare  
i number (5);  
j number(2) ;  
begin  
j :=0;  
for i in 1..7500 loop
```

```

insert into studio values(i, j) ;
if j=1 then j := 0 ;
else j :=1 ;
end if ;
end loop ;
end ;
\

create table film(NumFilm number(5), NumStudio number(4));
declare
i number (5);
j number(2) ;
begin
for i in 1..5000 loop
insert into film values(i, 5000-i+1);
end loop;
for i in 5001..10000 loop
insert into film values(i, 10000-i+1);
end loop ;
for i in 10001..15000 loop
insert into film values(i, 15000-i+1);
end loop ;
end ;
\

```

2. Générer des statistiques sur les données, avec la commande `"EXEC DBMS_UTILITY.ANALYZE_SCHEMA('NOM_UTILISATEUR','COMPUTE')"`, afin de rendre l'optimiseur plus pertinent. Affichez (et observez) le plan d'exécution physique des requêtes suivantes, ainsi que le coût estimé par Oracle. *Attention, ces requêtes ne doivent pas être exécutées.*
 - `select * from studio ;`
 - `select * from studio where NumStudio=1 ;`
 - `select * from studio where europeen=1 ;`
 - `select * from film where NumFilm=1 ;`
 - `select europeen from studio, film where studio.NumStudio=film.NumStudio ;`
 - `select europeen from studio, film where studio.NumStudio=film.NumStudio and film.NumFilm=4000 ;`
 - `select * from studio where tochar(NumStudio,9)='1' ;`

2 Utilisation d'index et de groupes

1. Pour créer des index primaires, il faut que la table soit ordonnée sur la clé de recherche. C'est le cas des clés primaires (si on l'a bien précisé au départ) et des attributs autour desquels sont organisés des groupes (clusters).
 - Effacez les tables précédentes et recréez les en précisant cette fois les clés primaires ; utilisez également l'ordre `ORGANIZATION INDEX` pour spécifier qu'elles doivent être triées sur leur clé primaire. Réévaluez les requêtes, comparez et commentez les différences.
 - Effacez les tables et créez un groupe (cluster) approprié. Recréez ensuite les tables en précisant leur appartenance au cluster. Enfin, créer un index sur la colonne commune de ce cluster. Réévaluez les requêtes, comparez et commentez les différences.
 - Les deux index précédent étaient des B-Arbres. Dans la deuxième solution on peut utiliser une table de hachage. Effacez le cluster précédent (`DROP CLUSTER nom_cluster INCLUDING TABLES`), et recréez-le avec une table de hachage. réévaluez les requêtes 5 et 6.
2. Un index est secondaire lorsque la table indexée n'est pas ordonnée sur la clé de recherche. Sans recréer les tables existantes :
 - Créez un index (par défaut, ce sera un B-Arbre) sur l'attribut *europeen* de studio et l'attribut *NumFilm* de film. Réévaluez les requêtes 3 et 4, commentez.
 - L'attribut *europeen* n'est pas assez sélectif pour un B-Arbre. Créer un index BitMap sur cet attribut, et observez le résultat.
3. Lorsqu'un attribut apparaît dans un appel de fonction, alors l'utilisateur ne peut utiliser un éventuel index. C'est le cas de la dernière requête. Il faut alors créer un index basé sur une fonction (function based index). Créez un index pour la dernière requête, et observez le résultat.