Entrepôts de Données TP3

Objectif : Maîtrise des vues matérialisées

Syntaxe

CREATE MATERIALIZED VIEW <nomvue>
BUILD { IMMEDIATE|DEFERRED }
REFRESH { COMPLETE|FAST|FORCE|NEVER } { ON DEMAND|ON COMMIT }
ENABLE QUERY REWRITE
AS SELECT ... ;

Options

IMMEDIATE : Création de la vue matérialisée et population de la vue

DEFERRED : Création de la vue matérialisée sans être alimentée en données.
DBMS_MVIEW.REFRESH(<liste_vues>) alimente la vue

ON COMMIT : Rafraîchissement à chaque fin de transaction modifiant les tables sources

ON DEMAND : Rafraîchissement avec DBMS_MVIEW.REFRESH

COMPLETE : Recalcul complet de la vue

FAST : Application d'un rafraîchissement incrémental

FORCE : FAST si possible, COMPLETE sinon NEVER : pas de rafraîchissement

L'option **ENABLE QUERY REWRITE**, dans la commande de création d'une vue matérialisée permet l'exploitation de celle-ci dans la réécriture des requêtes sur les tables sources dans le but d'optimiser les temps d'accès.

Travail à faire :

A partir du schéma du TP1 :

Attention dans toutes les requêtes évitez les jointures imbriquées.

1. Activer les options autotrace, et timing de oracle. (set autotrace on, set timing on)
2. Ecrire une requête R1 pour obtenir la liste des vols (NumVol, datevol) entre l'Algérie et la Suède. Vous pouvez modifier deux lignes quelconque de la table pays pour avoir les valeurs demandées.
3. Examiner le temps et le plan d'exécution.
4. Créer une vue matérialisée VM1 en utilisant les options (IMMEDIATE, COMPLETE, ON DEMAND, ENABLE QUERY REWRITE) contenant une jointure entre les tables vol, aéroport, ville et pays.
5. Ré exécuter la requête R1. Examiner le temps et le plan d'exécution et comparer avec (3).
6. Ecrire une requête R2 pour obtenir le nombre de vol par Année (Année, NBVols)
7. Examiner le temps et le plan d'exécution.
8. Créer une vue matérialisée VM2 (Année, NbVols) en utilisant les options (IMMEDIATE, COMPLETE, ON DEMAND, ENABLE QUERY REWRITE)
9. Ré exécuter la requête R2. Examiner le temps et le plan d'exécution et comparer avec (9).
10. Augmenter le nombre d'instances de vol à 1200000, puis à 2000000 et retester la requête R2 avec et sans la vue matérialisée VM2 à chaque fois. (N'oubliez pas de rafraichir la vue après chaque augmentation du nombre d'instances : commande : Execute DBMS_MVIEW.Refersh('VM2') ;).

11. Donner un tableau comparatif des temps d'exécution (avec et sans la vue matérialisée) en fonction du nombre d'instances.
12. Quelles Conclusions tirez-vous de ce TP?

Indications :

Avant toute exécution des **requêtes** vider tous les buffers à l'aide des commandes :
alter system flush shared_pool;
alter system flush buffer_cache;

Entrepôts de Données TP4

Objectif : Création d'un magasin de données

Schéma du magasin de données :

Dimensions :

DAéroportDép(NumAerD, NomAerD, CodeVille, NomVille, CodePays, NomPays, CodeTypAerD, TypeAerD)

DAéroportArr(NumAerA, NomAerA, CodeVille, NomVille, CodePays, NomPays, CodeTypAerA, TypeAerA)

DCompagnie(CodeComp, NomComp, CodeTypeComp, TypeComp)

DModèle(CodeModèle, LibModèle, CodeCons, NomConst)

DTemps(CodeTemps, Jour, LibJour, Mois, Libmois, Année) // Format jour 'JJ/MM/AAAA', Mois 'MM/AAAA'

Fait :

FTraffic(NumAerD , NumAerA ,CodeComp, CodeModèle, CodeTemps, NbVol, NbVolRetard)

Travail demandé :

- Créer le schéma (tables + contraintes) et le remplir à partir des sources dans un nouvel utilisateur

Indications :

- Ci dessous un squelette de script pour créer remplir la table DAéroportDép:

```
Create table DAéroportDép  (……..);
begin
for i in
( SELECT  NumAer,…
FROM  master.Aeroport, mater.ville,….
WHERE ….) loop
insert into DAéroportDép values(i.Numaer, i.nomaer….);
end loop;
commit ;
end ;
/
```

- La clé de la dimension Dtemps est une valeur séquentielle, il est possible de créer une séquence via le script :

```
CREATE SEQUENCE seq
MINVALUE 1
MAXVALUE 100000
 START WITH 1
 INCREMENT BY 1 ;
```

  Et de l'invoquer dans une insertion comme suit :

```
INSERT INTO Dtemps VALUES (seq.NEXTVAL, …..);
```

- L'attribut jour correspond au format (DD/MM/YYYY), mois (MM/YYYY), année(YYYY)
- L'affichage d'une date peut être modifié en utilisant la fonction `TO_CHAR` : `TO_CHAR(arg1,arg2)` : `arg1` colonne (ou valeur) de type `DATE`, `arg2` masque de sortie.

- **Quelques masques au format numérique :**

YYYY	Année
Q	Trimestre
MM	Mois
DD	Jour dans le mois
MONTH	Nom du mois en toutes lettres
WW	Semaine dans l'année
DAY	Nom du jour en toutes lettres

Exemple :

```
For I in (SELECT distinct
TO_CHAR(DateVol,'DD/MM/YYYY') as jour ,
TO_CHAR(DateVol,'DAY') as libjour ,....
FROM vol)LOOP
Insert Dtemps values (seq.nextval, jour,
libjour,…) END LOOP
Commit ;…
```