

Mini-projet Bases de données reparties



Sujet du projet :

On considère la base de données « Usine » qui permet de stocker et gérer les données d'une usine de fabrication de pièces automobiles. La base de données « Usine » contient les tables suivantes :

- ♣ **Employe** : Contenant la liste des employés de l'usine
- ♣ **Magasin** : Contenant un inventaire du stock des pièces fabriquées
- ♣ **Atelier** : Contenant les informations sur les différents ateliers fabriquant les pièces
- ♣ **Piece** : contient les informations sur les pièces fabriquées par l'usine
- ♣ **Commande** : contient les informations des commandes d'achat des pièces automobiles. Pour simplifier, on considère qu'une commande concerne un seul modèle de pièce.

Partie I : Création et configuration :

En tant qu'Administrateur de BD, créer les structures nécessaires pour la BD « Usine ».

Créer les tablespaces nécessaires :

- ♣ Les tables « Employe », « Piece » et « Atelier » ont une fréquence d'évolution assez faible ;
- ♣ Les tables « Commande » et « Magasin » évoluent rapidement (le volume de données augmente rapidement et/ou est modifié fréquemment) ;
- ♣ Disposez d'un espace de stockage total de 100G ;
- ♣ Pour des raisons de limitation de stockage, on souhaite désactiver la journalisation pour les tablespaces dont les données sont plutôt stables ;

```
SQL> connect system
Enter password:
Connected.
SQL> create bigfile tablespace u_hr
2 datafile 'C:\Users\ASUS\Desktop\tp AdminBD\employe.dbf' size 100M autoextend on next 20M Maxsize 10G nologging;
Tablespace created.
```

```
SQL> create smallfile tablespace u_workshop
2 datafile 'C:\Users\ASUS\Desktop\tp AdminBD\piece.dbf' size 100m autoextend on next 500m maxsize 20g,
3 'C:\Users\ASUS\Desktop\tp AdminBD\atelier.dbf' size 100m autoextend on next 20m maxsize 10g nologging;
Tablespace created.

SQL> create smallfile tablespace u_catalog
2 datafile 'C:\Users\ASUS\Desktop\tp AdminBD\magasin.dbf' size 500m autoextend on next 100m maxsize 30g,
3 'C:\Users\ASUS\Desktop\tp AdminBD\commande.dbf' size 500m autoextend on next 100m maxsize 30g logging;
Tablespace created.
```

Mini-projet

Bases de données reparties

Il est recommandé de vérifier que les tablespaces ont été créés une fois le travail est achevé.

```
SQL> select tablespace_name from dba_tablespaces;

TABLESPACE_NAME
-----
SYSTEM
SYSAUX
UNDOTBS1
TEMP
USERS
TBS_COMPTA
TBS_COMMERCES
U_HR
U_WORKSHOP
U_CATALOG

10 rows selected.
```

Créer les tables constituant la BD « Usine » puis alimenter les tables en y insérant au moins trois lignes dans chacune, en respectant les contraintes d'intégrité :

```
SQL> create table employe (num_emp varchar(3) primary key, nom_emp varchar(50), grade varchar(15) check(
  2 ('Agent_Admin', 'Ingénieur', 'Technicien', 'Ouvrier')), address varchar(100), salaire decimal) table
space u_hr;

Table created.

SQL>
SQL> create table atelier (code_at varchar(3) primary key, location varchar(30), responsable varchar(3), p
  2 productivite number, CONSTRAINT FK_resp FOREIGN KEY (responsable)
  2 REFERENCES employe(num_emp)) tablespace u_workshop;

Table created.

SQL>
SQL> create table piece (id_p varchar(8) primary key, libelle_p varchar(50), atelier varchar(3), prix_unit
  2 aire decimal, CONSTRAINT FK_atl FOREIGN KEY (atelier)
  2 REFERENCES atelier(code_at)) tablespace u_workshop;

Table created.

SQL>
SQL> create table magasin (piece varchar(8) primary key, qte number, date_appro date, CONSTRAINT FK_p FOREI
  2 GN KEY (piece)
  2 REFERENCES piece(id_p)) tablespace u_catalog;

Table created.

SQL>
SQL> create table commande (id_client varchar(8), piece varchar(8), date_cmd date, qte_cmd number, montant
  2 decimal, CONSTRAINT FK_pcmd FOREIGN KEY (piece)
  2 REFERENCES piece(id_p), CONSTRAINT PK_cmd PRIMARY KEY (id_client, piece, date_cmd)) tablespace u_ca
  2 talog;

Table created.
```

Mini-projet

Bases de données reparties

```
SQL> insert into employe values('ig2', 'mohammed', 'Agent_Admin', 'rue liberte', 200.000);
1 row created.

SQL>
SQL> insert into employe values('5jh', 'ahmed', 'Agent_Admin', 'rue jeunesse', 80.000);
1 row created.

SQL>
SQL> insert into employe values('kj6', 'Salleh', 'Technicien', 'rue liberte', 130.000);
1 row created.
```

```
SQL>
SQL> insert into atelier values('qb7', 'BlockA', 'ig2', 20);
1 row created.

SQL>
SQL> insert into atelier values('k5r', 'BlockA', '5jh', 25);
1 row created.

SQL>
SQL> insert into atelier values('001', 'BlockB', '5jh', 37);
1 row created.
```

```
SQL>
SQL> insert into piece values ('M25130', 'screw', '001' , 3.000);
1 row created.

SQL>
SQL> insert into piece values ('A00560', 'engine', '001' , 50.000);
1 row created.

SQL>
SQL> insert into piece values ('B12569', 'chain', 'qb7' , 62.000);
1 row created.
```

Mini-projet
Bases de données reparties

```
SQL>
SQL> insert into magasin values('M25130', 300, sysdate);

1 row created.

SQL>
SQL> insert into magasin values('A00560', 100, sysdate);

1 row created.

SQL>
SQL> insert into magasin values('B12569', 230, sysdate);

1 row created.
```

```
SQL>
SQL> insert into commande values('hfkeh4j6', 'A00560', sysdate, 20, 1000.000);

1 row created.

SQL>
SQL> insert into commande values('qdfs2307', 'B12569', sysdate, 10, 620.000);

1 row created.

SQL>
```

Mini-projet Bases de données reparties

Soit les trois schémas suivants qui sont utilisés par les employés de l'usine travaillant sur la BD :

Gest Personnel : c'est un agent administratif chargé de la gestion des employés en ajout, modification et suppression. Il doit avoir une visibilité sur l'état du stock dans le magasin, ainsi que la possibilité de gérer les responsables des ateliers.

```
SQL> create user gest_personnel identified by gestp;  
  
User created.  
  
SQL>  
SQL> alter user gest_personnel default tablespace u_hr quota 100m on u_hr;  
  
User altered.  
  
SQL> grant create session to gest_personnel;  
  
Grant succeeded.  
  
SQL>  
SQL> grant insert,select,update,delete on employe to gest_personnel;  
  
Grant succeeded.  
  
SQL>  
SQL> grant select on magasin to gest_personnel;  
  
Grant succeeded.  
  
SQL>  
SQL> grant update on atelier to gest_personnel;  
  
Grant succeeded.
```

Mini-projet Bases de données reparties

Magasinier : c'est un agent administratif chargé du contrôle du magasin en termes d'approvisionnement en pièces automobiles. Il gère les opérations d'achat ainsi que les commandes correspondantes.

```
SQL> create user magasinier identified by mag;

User created.

SQL>
SQL> alter user magasinier default tablespace u_catalog quota 100m on u_catalog;

User altered.

SQL>
SQL> grant create session to magasinier;

Grant succeeded.

SQL>
SQL> grant insert, update, select, delete on magasin to magasinier;

Grant succeeded.

SQL>
SQL> grant update on commande to magasinier;

Grant succeeded.
```

Mini-projet Bases de données reparties

Resp. Atelier : c'est un ingénieur chargé de gérer les pièces créées dans l'atelier dont il est responsable.

```
SQL> create user resp_atelier identified by resp;

User created.

SQL>
SQL> alter user resp_atelier default tablespace u_workshop quota 100m on u_workshop;

User altered.

SQL>
SQL> grant create session to resp_atelier;

Grant succeeded.

SQL>
SQL> grant insert, update, select, delete on atelier to resp_atelier;

Grant succeeded.

SQL>
SQL> grant insert, update, select, delete on piece to resp_atelier;

Grant succeeded.
```

Il est recommandé de vérifier l'existence de ces schémas après leur création.

```
SQL> select username from dba_users;

USERNAME
-----
MAGASINIER
RESP_ATELIER
GEST_PERSONNEL
U_DRH
U_DAF
AMIRA
HR
DRH
DAF
ANONYMOUS
FLOWS_FILES
```

Partie II : Administration :

L'exploitation de la base de données a montré que pour garantir plus de fluidité à l'évolution des prix unitaires des pièces automobiles et à l'évaluation de la productivité des ateliers, le Magasinier devrait pouvoir consulter et mettre à jour les informations du prix d'une pièce ainsi que de la productivité d'un atelier.

```
SQL> grant select, update on piece to magasinier;
Grant succeeded.

SQL>
SQL> grant select, update on atelier to magasinier;
Grant succeeded.

SQL>
```

Le magasinier a réclamé une erreur lors de l'insertion d'une nouvelle commande qui semble être liée à l'espace de stockage.

```
SQL> alter user magasinier quota 200m on u_catalog;
User altered.

SQL>
```

Le gestionnaire du personnel vous a contacté car il n'arrive plus à accéder à son compte après avoir saisi son mot de passe d'une façon erronée plusieurs reprises.

```
SQL> alter user magasinier account unlock ;
User altered.

SQL>
```


Mini-projet Bases de données reparties



Le magasinier a réclamé que l'opération de recherche sur l'historique des commandes par date ou par pièce devient de plus en plus lente.


```
SQL> create index ind_date on commande(date_cmd);  
  
Index created.  
  
SQL>  
SQL> create index ind_piece on commande(piece);  
  
Index created.  
  
SQL>
```

Partie III: Transactions :

Deux magasiniers ont ouvert deux sessions simultanées dans la BD « Usine », le scénario ci-dessous s'est déroulé :

T1 : les deux sessions vont recevoir les valeurs initiales des champs :

Session 1 :

 Run SQL Command Line

```
SQL> connect magasinier  
Enter password:  
Connected.  
SQL> Select * from system.magasin where piece='M25130' ;
```

PIECE	QTE	DATE_APPR
M25130	300	28-APR-22

Mini-projet Bases de données reparties

Session 2 :

```
SQL> connect magasinier
Enter password:
Connected.
SQL> Select * from system.magasin where
  2 piece='A00560' ;

PIECE          QTE  DATE_APPR
-----
A00560         100  28-APR-22
```

T2 :

Session 1 :

```
SQL> Update system.magasin set Qte=120 where piece='M25130' ;

1 row updated.
```

⇒ La modification s'applique mais ne sera pas enregistrée, elle est seulement vue par session1.

T3 :

Session 2 :

```
SQL> Select * from system.magasin where piece='M25130' ;

PIECE          QTE  DATE_APPR
-----
M25130         300  28-APR-22
```

⇒ La session 2 reçoit l'ancienne valeur car la modification s'applique mais ne sera pas enregistrée, elle est seulement vue par session 1.

T4 :

Session 1 :

```
SQL> Commit ;

Commit complete.
```

⇒ La modification est enregistrée.

Mini-projet Bases de données reparties

T5 :

Session 2 :

```
SQL> Select * from system.magasin where piece='M25130' ;
```

PIECE	QTE	DATE_APPR
M25130	120	28-APR-22

⇒ La session 2 peut avoir la nouvelle valeur car la modification est enregistrée.

T6 :

Session 2 :

```
SQL> Update system.magasin set Qte=85 where piece= 'A00560' ;
```

1 row updated.

⇒ La session 2 modifier la table mais n'enregistre pas sa modification.

T7 :

Session 1 :

```
SQL> Update system.magasin set Qte=30 where piece='B12569' ;
```

1 row updated.

⇒ La session 1 essaie d'effectuer une modification mais il reste en attente session 2 qui ne finit pas ses modifications sur la même ligne.

T8 :

Session 1 :

```
SQL> Update system.magasin set Qte=87 where piece='A00560' ;
```

1 row updated.

⇒ La session 1 essaie d'effectuer une modification mais il reste en attente de la session 2 qui ne finit pas ses modifications sur la même ligne.

Mini-projet Bases de données reparties



T9 :

Session 2 :

```
SQL> commit;  
Commit complete.
```

⇒ Enregistrer les modifications.

Session 1 : => la modification a été effectuée.

T10 :

Session 2 :

```
SQL> Select * from system.magasin where piece='A00560' ;  
  
PIECE          QTE DATE_APPR  
-----  
A00560          85 28-APR-22
```

⇒ Après la sauvegarde de part de session 2 (T9), la modification effectuée par session 1 s'exécute alors que session 2 ne peut pas la voir.

T11:

Session 1 :

```
SQL> Commit ;  
Commit complete.
```

Session 2 :

```
SQL> Select * from system.magasin where piece='A00560' ;  
  
PIECE          QTE DATE_APPR  
-----  
A00560          87 28-APR-22
```

⇒ Session 1 enregistre ses modification (commit) alors session 2 peut voir ces modifications.

Mini-projet
Bases de données reparties

Quel serait le résultat de la commande issue par la session 2 à T12 si la session 1 aurait exécuté un ROLLBACK à T11 ?

- ⇒ Si session 1 a exécuté un ROLLBACK en T11, le résultat vu par session 2 est celle de T10 car cette commande fait supprimer toutes les modifications précédentes non enregistrées.

Mini-projet
Bases de données reparties



Mini-projet
Bases de données reparties



Mini-projet
Bases de données reparties



Mini-projet
Bases de données reparties

