# TP N°3 : Requêtes

**Document à rendre :**
**un rapport de TP qui contient**
**les commandes utilisées,**
**avec description et références à l'énoncé,**
***ainsi que les résultats* et commentaires éventuels**

Soit la base de données du TP N°1

Service (**NumService**, LibelleS, Hierarchie#)

Employe (**NumEmploye**, Nom, Prenom, NumService#)

NatureConge (**NumNatureConge**, LibelleNat)

DroitConge (**NumEmploye#, NumNatureConge#,** NbJours)

CongePris (**NumConge**, DateDebConge, DateFinConge, NumNatureConge#, NumEmploye#)

Rappels :

- pour désactiver une clé étrangère :

  **ALTER TABLE <Nom_Table> DISABLE CONSTRAINT <Nom_cléEtrangère> ;**

- **Les contraintes de « clé primaire » et dô « unicité » génèrent automatiquement des index (pk_.... ou uné ..)**

Attention : pour accélérer les recherches, lorsquôil y a des jointures avec des clés étrangères, il serait souhaitable dôindexer les clés étrangères. Côest-à-dire créer des index sur ces clés étrangères.

Syntaxe : create index <idx_cleEtrangere> ON <Nom_Table>(clé étrangère)

Exemple : soit la table Produit (**CodeProduit**, Designation, é , NumFournisseur)

On crée un index sur la clé étrangère NumFournisseur soit

**CREATE INDEX idx_numfournisseur ON Produit (NumFournisseur) TABLESPACE <nom_tablespace>;**

Pour effacer un index : DROP INDEX <nom_index> ;

Exemple : DROP INDEX idx_numfournisseur ;

Vérifier toujours dans la vue appropriée, la création des index.

Les résultats des différentes requêtes seront stockés dans un fichier intitulé RequetesBdConge.txt à lôaide de la commande spool

1) Nom et prénom de tous les employés par ordre alphabétique du nom.
2) Nom et prénom des employés dont le nom commence par un C.
3) Nom et prénom des employés dont le nom est terminé par un T.
4) Nombre total dôemployés.

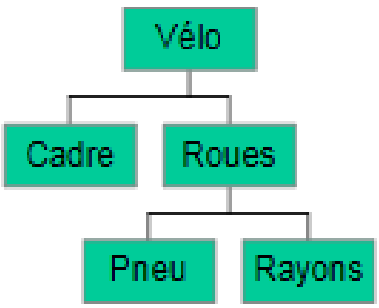
5) Nombre total de services.
6) Nom et prénom des employés travaillant dans lâatelier montage.
7) Service, nom et prénom des employés travaillant dans le service numéro 1100 ou dans un des services dépendant de 1100 (2 méthodes).
8) Numéro de services dans lesquels travaillent strictement plus dâune personne.
9) Libellé des services où aucun employé nâa été affecté.
10) Libellé des services où au moins un employé travaille (2 méthodes en utilisant la requête précédente).
11) Nombre total de jours de congé déjà pris par employé.
12) Nombre total de jours de congé auxquels chaque employé a droit.
13) Nom et prénom des employés ayant un nombre de jours de congé supérieur à un nombre saisi au clavier (paramètre)
14) Nombre dâemployés par service.
15) Nom et prénom des employés qui prennent au moins un jour de congé entre le 18/12/17 et le 26/12/17
16) Nom et prénom des employés qui prennent au moins un jour de congé entre 2 dates saisies au clavier.
17) Détecter le nom et prénom des employés non référencés dans la table DroitConge.
18) Employés ayant droit à des congés pour motif de récupération.
19) Nom et prénom des employés ayant posé des congés (éliminer les doublons)
20) Structure hiérarchique avec les niveaux des services (NumService, LibelleS), sachant que le service qui dirige lâensemble de lâentreprise est la direction générale.
21) Structure hiérarchique avec les niveaux des employés (Nom, Prénom et le libellé du service) en fonction de leur service dâappartenance.
22) Nombre dâemployés pour chaque niveau de service.

Remarque : requête hiérarchique ?

Les requêtes hiérarchiques extraient des données provenant dâune structure arborescente. Les enregistrements dâune structure arborescente appartiennent, en général, à la même table et sont reliés entre eux par des associations auto-récursives à plusieurs niveaux.

Câest le cas de la table Service.

Exemple de hiérarchie :



La table correspondante :

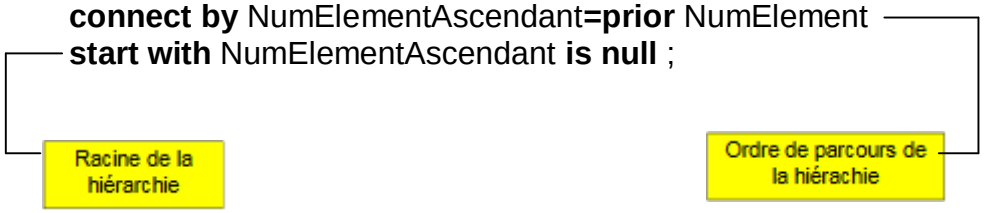
Element (**<u>NumElement</u>**, Designation, NumElementAscendant#)

**Element**

| NumElement | Designation | NumElementAscendant |
|---|---|---|
| 1 | Velo | NULL |
| 2 | Cadre | 1 |
| 3 | Roue1 | 1 |
| 4 | Roue2 | 1 |
| 5 | Pneu1 | 3 |
| 6 | Pneu2 | 4 |
| 7 | Rayon11 | 3 |
| 8 | Rayon12 | 3 |
| 9 | Rayon13 | 3 |
| 10 | Rayon21 | 4 |

Structure hiérarchique des éléments à partir de la racine :

```
select Designation from  Element
  connect by NumElementAscendant=prior NumElement
  start with NumElementAscendant is null ;
```




Résultat : Velo
Cadre
Roue1
Pneu1
Rayon11
Rayon12
Rayon13
Roue2
Pneu2
Rayon21

Structure hiérarchique des éléments à partir de la racine avec indication du niveau dans la hiérarchie :

```
select level, Designation from  Element
  connect by NumElementAscendant=prior NumElement
  start with NumElementAscendant is null ;
```

Résultat : 1 Velo
2 Cadre
2 Roue1
3 Pneu1
3 Rayon11
3 Rayon12
3 Rayon13
2 Roue2
3 Pneu2
3 Rayon21

Structure hiérarchique des éléments à partir de la racine avec indication du niveau dans la hiérarchie avec élagage d'une branche

```
select level, Designation from Element
  connect by NumElementAscendant=prior NumElement
  and Designation<>'Roue2'
  start with NumElementAscendant is null ;
```

Résultat :
```
1 Velo
2 Cadre
2 Roue1
3 Pneu1
3 Rayon11
3 Rayon12
3 Rayon13
```

Nombre d'éléments dans chaque niveau :

```
select level, count(NumElement) from Element
  connect by NumElementAscendant=prior NumElement
  start with NumElementAscendant is null
  group by level ;
```

Résultat :
```
1  1
2  3

3  6
```

# Les vues

C'est une table **virtuelle calculée** à partir d'autres tables grâce à une requête.
La création d'une vue est stocké dans le dictionnaire des données dans la vue **dba_views** ou **all_views** ou **user_views**

Exemple : pour user_views

| Colonne | Type de données | NUL | La description |
|---|---|---|---|
| VIEW_NAME | VARCHAR2(30) | NOT NULL | Nom de la vue |
| TEXT_LENGTH | NUMBER | | Longueur du texte de la vue |
| TEXT | LONG | | Afficher le texte |
| TYPE_TEXT_LENGTH | NUMBER | | Longueur de la clause de type de la vue dactylographiée |
| TYPE_TEXT | VARCHAR2(4000) | | Tapez la clause de la vue dactylographiée |
| OID_TEXT_LENGTH | NUMBER | | Longueur de la clause WITH OID de la vue dactylographiée |
| OID_TEXT | VARCHAR2(4000) | | WITH OID clause de la vue dactylographiée |
| VIEW_TYPE_OWNER | VARCHAR2(30) | | Propriétaire du type de la vue si la vue est une vue dactylographiée |
| VIEW_TYPE | VARCHAR2(30) | | Type de vue si la vue est une vue dactylographiée |
| SUPERVIEW_NAME | VARCHAR2(30) | | Nom de la supervision |

## Définition d'une vue :

**create view** <nom_vue> **as** requête

Exemple :

```
create view ClientsLillois as
select Nom, Prenom from Client
where Ville='Lille';
```

## Intérêt des vues :

→ Simplification de l'accès aux données en masquant les opérations de jointure

```
create view ProduitCommande as
select P.NumProd, Designation, PrixUnitaire, Date, Quantite
from Produit P, Commande C
where P.NumProd=C.NumProd

select NumProd, Designation from ProduitCommande
where Quantite>10;
```

→ Sauvegarde indirecte de requêtes complexes

→ Présentation de mêmes données sous différentes formes adaptées aux différents usagers particuliers

→ Support de l'indépendance logique

ex. Si la table Produit est remaniée, la vue ProduitCommande doit être refaite, mais les requêtes qui utilisent cette vue n'ont pas à être remaniées.

→ Renforcement de la sécurité des données par masquage des lignes et des colonnes sensibles aux usagers non habilités

## Problèmes de mise à jour, restrictions

La mise à jour de données via une vue pose des problèmes et la plupart des systèmes impose d'importantes restrictions.

- Le mot clé **distinct** doit être absent.
- La clause **from** doit faire référence à une seule table.
- La clause **select** doit faire référence directement aux attributs de la table concernée (pas d'attribut dérivé).
- Les clauses **group by** et **having** sont interdites.

23) Créer la vue VEmploye (NumEmploye, Nom, Prenom) à partir de la table Employe et vérifier son contenu.

24) A travers la vue VEmploye, modifier le nom de l'employé 11 et consulter le contenu de la vue VEmploye de la table Employe.

25) Créer la vue VAffectation (NumEmploye, Nom, Prenom, LibelleS) qui regroupe les nom et prénom des employés avec le libellé de leur service d'affectation.

26) A travers la vue VAffectation, modifier le liellé du service de l'employé 6 et consulter le contenu de la vue VAffectation et de la table Employe. Que se passe-t-il ?

27) Lister toutes les contraintes définies sur les tables de la base de données. Quelle vue utilise-t-on ?

Afficher pour chaque contrainte son nom, la table à laquelle elle s'applique ainsi que son type.

## Privilèges

1) Connectez vous **en étant system**, créer lôutilisateur titi avec comme mot de passe titi ;

2) Donner lui le droit de se connecter ;

3) Connectez vous en tant que lôutilisateur toto. Donner lui les droits de consulter la table Employe de titi

4) Connectez vous en tant que titi et modifiez le nom de lœemployé 2 dans la table Employé.. Que se passe-t-il ?

5) Donner lui le droit de mise à jour dans la table Employe, modifier le nom de lœemployé 2