Le langage SQL pour Oracle - partie 2 : SQL comme langage de requêtes

Exemple jouet

- Dans la suite, on considère la base composée des tables suivantes :
  - CLIENT(**numCl**,nomCl,télCl)
  - COMMANDE(**numCmde**,dateCmde,numCl))
  - LigneCommande(**numCmde,numProd**,Quantité)
  - PRODUIT(**numProd**,désiProd,prixUnitaire,stockProd)

LMD
- Requêtes SQL (SELECT) -

- Syntaxe de *requêteSQL*

  selectSQL |

  (requêteSQL) {UNION|INTERSECT|EXCEPT} (requêteSQL)

- Syntaxe du *selectSQL*

  SELECT {[ALL|DISTINCT] expression [AS nomColonne][,expression [AS nomColonne]]…}|*
  FROM table [AS nomTable [(nomColonne[,nomColonne])]]]
  [,table [AS nomTable [(nomColonne[,nomColonne])]]]…
  [WHERE conditionSQL]
  [GROUP BY nomColonne [,nomColonne]…
  [HAVING conditionSQL]
  [ORDER BY nomColonne [ASC|DESC] [,nomColonne[ASC|DESC]]…]

Projection d'une table et la clause DISTINCT

- Afficher les *numCl* et *dateCmde* de toutes les *Commandes*

```
SELECT numCl, dateCmde
FROM Commande
```

Produit un multi-ensemble …

```
SELECT DISTINCT numCl, dateCmde
FROM Commande
```

Élimine les doublons

Sélection sur une table (WHERE)

- Sélectionner les *Produits* dont le prix est inférieur à 20.00€ et le numéro est supérieur à 30

```
SELECT *
FROM Produit
WHERE prixUnitaire < 20 AND numProd > 30
```

Syntaxe de *conditionSQL*

- (conditionSQL)|
  NOT(conditionSQL)|
  conditionSQL AND conditionSQL |
  conditionSQL OR conditionSQL}
- Syntaxe (incomplète) de la *conditionSQL* :

  {expression {=|<|>|<=|>=|<>} expression|
  expression BETWEEN expression AND expression|
  expression {IS NULL |IS NOT NULL}|
  expression {IN |NOT IN} listeConstantes|
  expression {LIKE |NOT LIKE} patron}

ConditionSQL - BETWEEN

- Sélectionner les Commandes du mois de juin de l'année 2000

```
SELECT *
FROM Commande
WHERE dateCmde BETWEEN '01/06/2000' AND '30/06/2000 ‘

SELECT *
FROM Commande
WHERE dateCmde >= '01/06/2000' AND dateCmde <='30/06/2000'
```

ConditionSQL - IN

- Sélectionner les Commandes du Client dont le numCl est 10 ou 40 ou 80

```
SELECT *
FROM Commande
WHERE numCl IN (10, 40, 80)
```

Ou encore :

```
SELECT *
FROM Commande
WHERE numCl = 10 OR numCl = 40 OR numCl = 80
```

ConditionSQL - LIKE

- Sélectionner les Clients dont le nomCl contient le mot Le :

```
SELECT *
FROM Client
WHERE nomCl LIKE '%Le% ‘
```

- 2ième lettre du nomCl = o et dernière lettre est un k

```
SELECT *
FROM Client
WHERE nomCl LIKE '_o%k'
```

ConditionSQL - IS NOT NULL

- Sélectionner les *Produits* dont le stock est renseigné

```
SELECT *
FROM Produit
WHERE stockProd IS NOT NULL
```

Sélection-projection sur une table

- Afficher les *numCl* et *dateCmde* des *Commandes* dont la date est supérieure au *05/07/2000*

```
SELECT numCl, dateCmde
FROM Commande
WHERE dateCmde > '05/07/2000'
```

Produit cartésien avec SELECT-FROM

- Afficher toutes les combinaisons possibles de lignes de Client et de Commande :

```
SELECT *
FROM Client, Commande
```

Jointure naturelle avec SELECT-FROM-WHERE

- Afficher les informations au sujet des *Clients* et de leurs *Commandes*

```
SELECT Client.numCl, nomCl, télCl, numCmde,dateCmde
FROM Client, Commande
WHERE Client.numCl = Commande.numCl
```

Jointure de plusieurs tables

- Sélectionner les *nomCl* des *Clients* qui ont commandé au moins un plant d'herbe à puce

```
SELECT nomCl
FROM Client, Commande, LigneCommande, Article
WHERE description = 'Herbe à puce' AND
        Client.numCl = Commande.numCl AND
        Commande.numCmde =    LigneCommande.numCmde AND
        LigneCommande.numProd = Produit.numProd
```

Définition d'un alias (clause AS)

- renommer ( )

```
SELECT Client.numCl, nomCl, télCl, numCmde,dateCmde
FROM Client, Commande
WHERE Client.numCl = Commande.numCl
```

Peut s'écrire :

```
SELECT Cl. .numCl, nomCl, télCl, numCmde,dateCmde
FROM Client  Cl, Commande  Co
WHERE Cl.numCl = Co.numCl
```

Auto-jointure

- Quels sont les *Clients* qui ont le même numéro de téléphone?

```
SELECT C1.numCl, C2.numCl
FROM Client C1, Client C2
WHERE C1.télCl = C2.télCl
```

Opérations ensemblistes (UNION, INTERSECT, EXCEPT)

- Afficher les noms et numéros de téléphone des *Employés* qui sont aussi des *Clients* de la pépinière

```
(SELECT nomCl AS Nom, télCl As Téléphone
FROM Client)

INTERSECT

(SELECT nomEmployé AS Nom, télEmployé AS Téléphone
FROM Employé)
```

Expressions générales sur les colonnes

- La liste des *numProd* avec le *prixUnitaire* avant et après inclusion de la taxe de 15%

```
SELECT numProd, prixUnitaire, prixUnitaire*1.15 AS prixPlusTaxe
FROM Article
```

- Afficher le détail de chacun des Produits commandés dans la *Commande numéro* 1 incluant le prix total avant et après la taxe de 15% pour chacun des Produits commandés

```
SELECT L.numProd, quantité, prixUnitaire, prixUnitaire*quantité AS total,
prixUnitaire*quantité*1.15 AS totalPlusTaxe
FROM LigneCommande L, Produit P
WHERE L.numProd = P.numProd AND L.numCmde = 1
```

Expression sur colonne du WHERE

- Les Produits dont le prixUnitaire incluant la taxe de 15% est inférieur à 16.00€

```
SELECT numProd, prixUnitaire, prixUnitaire*1.15 AS prixPlusTaxe
FROM Produit
WHERE prixUnitaire*1.15 < 16
```

Pseudo-colonnes

- Les *Commandes* de la journée

```
SELECT *
FROM Commande
WHERE dateCmde = CURRENT_DATE
```

Quelques fonctions SQL

- POSITION(patron IN chaîne)
- CHARACTER_LENGTH(chaîne)
- OCTET_LENGTH (chaîne)
- BIT_LENGTH(chaîne)
- EXTRACT(champ FROM dateOuTime)
- SUBSTRING (chaîne FROM indiceDébut FOR nombreCaractères)
- UPPER | LOWER (chaîne)
- TRIM ([LEADING|TRAILING|BOTH] caractère FROM chaîne)
- CAST(expression AS type)
- …
- Voir documentation du SGBD

Fonctions d'agrégat

- Le nombre de produits différents à vendre ainsi que le prix unitaire moyen des produits

```
SELECT COUNT(*) AS nombreProd,AVG(prixUnitaire) AS prixMoyen
FROM Article
```

Partition d'une table avec la clause GROUP BY

- Produire le nombre de Commandes passées par chacun des Clients qui ont passé au moins une Commande

```
SELECT numCl, COUNT(*) AS nombreCommandes
FROM Commande
GROUP BY numCl
```

Clause HAVING

- Afficher le nombre de Commandes passées par chacun des Clients qui ont passé deux Commandes ou plus

```
SELECT numCl, COUNT(*) AS nombreCommandes
FROM Commande
GROUP BY numCl
HAVING COUNT(*) >= 2
```

- Afficher le nombre de Commandes passées par chacun des Clients qui ont passé deux Commandes ou plus après le 02/06/2000

```
SELECT numCl, COUNT(*) AS nombreCommandes
FROM Commande
WHERE dateCmde > '02/06/2000'
GROUP BY numCl
HAVING COUNT(*) >= 2
```

Tri du résultat (ORDER BY)

- Les Clients en ordre alphabétique du nom

```
SELECT *
FROM Client
ORDER BY nomCl
```

```
SELECT *
FROM Client
ORDER BY nomCl DESC, télCl ASC
```

SELECT imbriqué

- Les Clients qui ont passé au moins une Commande

```
SELECT *
FROM Client
WHERE numCl IN
      (SELECT numCl
       FROM Commande)
```

- Ou encore :

```
SELECT DISTINCT Client.numCl, nomCl, télCl
FROM Client, Commande
WHERE Client.numCl = Commande.numCl
```

Ligne à plusieurs colonnes

- Chercher les *LigneCommandes* pour lesquelles au moins une *Livraison* a été effectuée

```
SELECT *
FROM LigneCommande
WHERE (numCmde,numProd) IN
    (SELECT numCmde, numProd
    FROM DétailLivraison)
```

Test d'ensemble vide (EXISTS)

- Produire les informations au sujet des Clients qui ont passé au moins une Commande

```
SELECT *
FROM Client
WHERE EXISTS
      (SELECT *
       FROM Commande
       WHERE numCl = Client.numCl)
```

Test de double (UNIQUE)

- Vérifier s'il y a plus d'un Client qui porte le même nom

```
NOT UNIQUE
(SELECT nomCl FROM Client)
```

- Clients qui ont passé au moins deux Commandes

```
SELECT *
FROM Client
WHERE NOT UNIQUE
(SELECT numCmde
FROM Commande
WHERE numCl = Client.numCl)
```

Quantificateurs (ALL, SOME/ ANY)

- Commandes passées après la dernière Livraison (date ultérieure)

```
SELECT * FROM Commande
WHERE dateCmde > ALL
(SELECT dateLiv
FROM Livraison)
```

- Commandes passées après au moins une des Livraisons

```
SELECT * FROM Commande
WHERE dateCmde > ANY
(SELECT dateLiv
FROM Livraison)
```

Niveau externe du schéma en SQL

- Gestion de la sécurité
  - GRANT
- Tables virtuelles
  - VIEWS

Sécurité en SQL (GRANT)
Identification et authentification

- Identification des utilisateurs
  - authorizationID
    - PUBLIC : tous les utilisateurs
- Authentification
  - mot de passe
  - …
- Oracle
  - SYS, SYSTEM
    - CREATE USER *authorizationID*...

Privilèges

- privilège :

  GRANT listePrivilèges ON objet TO listeAuthorizationIDs
  [WITH GRANT OPTION]

- objet :

  SELECT |
  DELETE |
  INSERT [listeColonnes] |
  UPDATE [listeColonnes] |
  REFERENCES listeColonnes |
  USAGE
  [TABLE] nomTable |
  DOMAIN nomDomaine |
  CHARACTER SET nomCharacterSet
  COLLATION nomCollation
  TRANSLATION nomTranslation

Exemples

- GRANT SELECT ON Commande TO commisLivraison
- GRANT SELECT ON LigneCommande TO commisLivraison

- GRANT SELECT, DELETE, INSERT, UPDATE ON Commande TO commisAchat
- GRANT GRANT SELECT, DELETE, INSERT, UPDATE ON LigneCommande TO commisAchat

- GRANT SELECT ON Produit TO PUBLIC

- GRANT UPDATE (stockProd) ON Produit TO commisLivraison

Suppression de privilèges

- REVOKE [GRANT OPTION FOR] listePrivilèges ON objet FROM listeIdUtilisateurs [RESTRICT | CASCADE]

Table virtuelle ou Vue (VIEW)

```
CREATE VIEW ProdPrixModique AS
SELECT numProd, désiProd, prixUnitaire
FROM Produit
WHERE prixUnitaire < 15
```

On peut alors l'utiliser « comme une table » :

```
SELECT *
FROM ProdPrixModique
```

Gestion des utilisateurs

- Création d'un utilisateur :

```
CREATE USER nom
IDENTIFIED BY mot_de_passe
DEFAULT TABLESPACE nom_tablespace
TEMPORARY TABLESPACE nom_tablespace
QUOTA entier K | M | UNLIMITED
ON nom_tablespace
[QUOTA entier K | M | UNLIMITED
ON nom_tablespace ];
```

- Modification d'un utilisateur
  - L'administrateur ou l'utilisateur, de la base peut modifier à tout moment :
    - le mot de passe (utilisateur),
    - le tablespace par défaut et/ou le quota
    - le tablespace temporaire
    - les droit d'accès sur un tablespace
    - les rôles affectés.
- Syntaxe :

```
ALTER USER nom
[IDENTIFIED BY mot_de_passe
[DEFAULT TABLESPACE nom_tablespace
[TEMPORARY TABLESPACE nom_tablespace
[QUOTA entier K | M | UNLIMITED
ON nom_tablespace ]
[QUOTA entier K | M | UNLIMITED
ON nom_tablespace ]
[DEFAULT ROLE nom_rôle];
```

- Suppression d'un utilisateur :
  - DROP USER nom [CASCADE]
  - L'option CASCADE supprime tous les objets appartenant à cet utilisateur.