Langage de requêtes S.Q.L

DEUXIEME PARTIE

Regroupement de données

Les requête de regroupement sont utilisées pour :

1. **totaliser des données**
2. **Calculer des statistiques**
3. **Localiser des données erronées**
4. **Recherche de tendances**

Trois type de regroupement :

- Simple requête, sans la clause GROUP BY
- Requête utilisant la clause GROUP BY
- Requête analyse croisée dynamique
- Utilisant la clause TRANSFORM

Langage SQL!!!

Fonction prédéfinies

Type	Description
AVERAGE	Moyenne d'un champs des tuples
COUNT	Le nombre des valeurs non vide de la colonne
COUNT(*)	Le nombres des tuples
MINIMUM	La valeur minimale
MAXIMUM	La valeur maximale
SUM	La somme des valeur d'un attribut
Fiste([colonne])	Valeur de la première ligne non vide de la colonne
Laste([Colonne])	Valeur de la dernière ligne non vide de la colonne
StDev[(Colonne)]	Calcule l'écart type pour les valeur non vide de la colonne
StDeP[(Colonne)]	Calcule l'écart type de population pour les valeur non vide

S.Q.L

Exemple d'utilisation des fonctions prédéfinies

- **COUNT** : Compter le nombre des professeurs :

  SELECT Count(*) AS Nbr

  FROM Professeur;

- **SUM** : Calculer la somme des coefficients de toutes les matières.

  SELECT SUM(Coef_Matiere) AS Somme

  FROM Matiere.

- **AVG** : Affiche la note maximal de CC1 pour la matière de base de données :

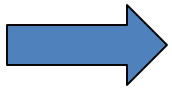
  SELECT AVG(CC1) AS Moyenne

  FROM Notes

  WHERE Code_Matière='M1'

Exemple 1 : Afficher tous les étudiants regroupés par ville

```
SELECT *
FROM Etudiants
GROUP BY Ville;
```



E1	Alaoui	Karim	Tanger
E4	Kalouch	Ouard	Tanger
E2	Chakour	Rachide	Tetouan
E3	Daoud	Noufal	Fes

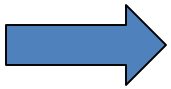
Exemple 2 : Afficher la somme des notes pour chaque étudiant

```
SELECT Code_Etudiant,
SUM(CC1+CC2) Som
FROM Notes
GROUP BY Code_Etudiant;
```

E1	12
E2	6
E3	14
E4	2
E5	14

Exemple 1 : Afficher le nombre des étudiants qui ont passé l'examen pour chaque matière

```
SELECT Code_Matiere,
COUNT(*) Nbr
FROM Notes
GROUP BY Code_Matiere
```



M1	5
M2	4
M3	4

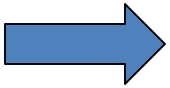
Exemple 2 : Afficher la somme des notes pour les étudiants qui ont passer trois examens.

```
SELECT Code_Etudiant,
Sum(CC1+CC2) Som
FROM Notes
GROUP BY Code_Etudiant
HAVING count(*)=3;
```

E1	12
E2	6
E3	14

Exemple 2 : Afficher la moyenne des notes pour chaque étudiants selon un ordre décroissante

```
SELECT Code_Etudiant,
AVG((CC1+CC2)*Coef) Moy FROM
Notes N, Matiere M
WHERE  N.Code_Matiere =
            M.Code_Matiere
GROUP  BY  Code_Etudiant
ORDER BY Moy
```



E1	12
E2	6
E3	14

Remarque :

- Pour utiliser le champs '**Coef'** il faut introduire la table **'Matiere'** dans la clause **FROM.**

Les prédicats :

1. Prédicats de Simulation

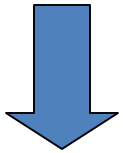
Opérateur	Signification
IN	Il comporte une liste de valeurs et test si une valeur spécifiée apparaît dans une liste ou bien dans une sous- interrogation
EXIST	Il contient une sous-interrogation qui, associée à la condition de EXIST, peut être évaluée comme vrais ou fausse.
NULL	Spécifie un test pour détecter les valeurs NULL d'un champs.
LIKE	Spécifie une comparaison dans laquelle on va confronter un modèle générique avec : - (_) : Pour représenter un blanc. - (?) : Pour représenter un caractère quel quant. - (%) : pour représenter une chaîne de caractères.

Exemple : IN

Affiche tous les étudiants dont la note de CC1 de base de données soit 12,14 ou 16.

```
SELECT   Code_Etudiant, Nom_etudiant, CC1
FROM     Notes N, Etudiants E
WHERE    N.Cod_etudiant = E.Cod_Etudiant     AND
         Nom_Matiere =  'BDD'     AND
CC1 IN (12,14,16);
```

Résultat



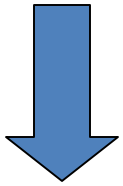
E1	**Alaoui**	**12**
E2	**Chakour**	**14**
E4	**Kalouch**	**16**

Exemple : NOT IN

Lister tous les professeurs qui n'enseigne aucune matière.

```
SELECT   DISTINT Code_professeur, Nom_Professeur
FROM     Professeur
WHERE    Code_pro NOT IN (SELECT   DISTINT Code_prof
                          FROM    Matiere);
```

Résultat



P5	Daouda

Exemple : EXISTS

Lister tous les professeurs qui enseignent au mois une matière.

```
SELECT DISTINT Code_professeur, Nom_Professeur, Pre_Professeur
FROM   Professeur P
WHERE  EXISTS (SELECT *
                    FROM   Matiere M
                    WHERE  P.Code-Prof=M.Code_prof);
```

Résultat

P1	Chaoui	Adnan
P2	Banani	Adil
P3	Samoud	Koutar
P4	Zine	Monir

Exemple : IS NULL

Lister tous les étudiants qui n'ont pas passé Le CC2 de matière '**BDD**'

```
SELECT    DISTINT Code_Etudiant, Nom_Etudiant
FROM      Etudiants E, Notes N
WHERE     E.Code_Etudiant = N.Code_Etudiant AND
          Cod_Matier = 'M1' AND
          Note_CC1 IS NUUL;
```

Résultat

Code_Etudiant	Nom_Etudiant
E1	Alaoui
E3	Daoud
E5	

Exemple : IS NOT NULL

Lister tous les professeurs qui enseigne au mois une matière.

```
SELECT DISTINT Code_professeur, Nom_Professeur, Pre_Professeur
FROM   Professeur P, Matières M
WHERE  P.code_Professeur = M.CodeProfesseur AND
           Note_CC1 IS NOT NUUL;
```

Résultat

Code_Etudiant	Nom_Etudiant
E1	Alaoui
E3	Daoud
E5	

Remarques sur l'utilisation de NULL

1. Rien n'est égale à NULL, même la valeur NULL
   - WHERE Note_CC1 = NULL ne donne pas les matières qui n'ont pas la notes de CC1

2. La clause SELECT n'autorise pas l'utilisation de NULL
   - SELECT P.Code_Professeur, **NULL**
     FROM Professeur P, Matieres M

   Cette requête n'est pas valide.

3. On ne peut pas trouver la valeur NULL par exclusion

Prédicats de quantité

1. Nécessite ‘utilisation d’un prédicat de comparaison appliqué aux résultats d’un sous-interrogation, qui contient un opérateur de comparaison (= , >, < …)
2. Ils permettent de tester une valeur unique face à toutes celles qui figurent dans l’ensemble de valeurs de sous-interrogation.
3. Quand l’opérateur utilisé est égal ‘=‘ le terme ANY est interchangeable avec IN
4. Dans de nombreux cas, ANY a le même sens que celui de SOME.

Les prédicats :

1. Prédicats de quantité

Opérateur	Signification
ALL	Permet de comparer une valeur X avec un ensemble des valeurs de même type de celui de X d'une sous interrogation SELECT. Le test de comparaison ALL est valide si le test de comparaison entre la valeur X et chaque élément de l'ensemble est valide.
ANY	Permet de comparer une valeur X avec un ensemble des valeurs de même type de celui de la valeur X d'une sous interrogation SELECT. Le test de comparaison ANNY est valide si le test est valide, au mois entre la valeur X et un élément de l'ensemble.
SOME	Identique a celui de ANY

Exemple : ALL

Afficher tous les étudiants qui ont la plus grande moyenne dans la matière de base de données.

```
SELECT   Code_Etudiant, Nom_etudiant, SUM(Coef*(CC1+CC2))
FROM      Etudiants E, Notes N, Matiers M,
WHERE   N.Cod_etudiant = E.Cod_Etudiant     AND
            N.Code_Matiere = M.Code_Matiere AND
            Code_Matiere  = 'M1'
GROUP BY  1,2
HAVING SUM(Coef*(CC1+CC2)) >=
            ALL ( SELECT  SUM(Coef*(CC1+CC2))
                        FROM     Notes N, Etudiants E, Matiers M,
                        WHERE   N.Cod_etudiant = E.Cod_Etudiant
                                    AND N.Code_Matiere = M.Code_Matiere AND
Code_Matiere ='M1'
                        GROUP BY  1,2);
```

Exemple : ANY/SOME

Afficher les étudiants dont les moyennes de la matière de base de données supérieur à au mois de la moyenne d'un étudiant de la ville de 'Tanger'

```
SELECT   Code_Etudiant, Nom_etudiant, SUM(Coef*(CC1+CC2))
FROM     Notes N, Etudiants E, Matiers M,
WHERE    N.Cod_etudiant = E.Cod_Etudiant     AND
         N.Code_Matiere = M.Code_Matiere AND
         Code_Matiere ='M1'
GROUP BY  1,2
HAVING SUM(Coef*(CC1+CC2)) >=
            ANY/SOME( SELECT   SUM(Coef*(CC1+CC2))
                      FROM     Notes N, Etudiants E, Matiers M,
                      WHERE    N.Cod_etudiant = E.Cod_Etudiant
                     AND N.Code_Matiere = M.Code_Matiere AND
                      );
```

Exemple : MINUS

Affiche tous les étudiants dont la note de CC1 de base de données soit 12,14 ou 16.

```
SELECT   Code_Etudiant, Nom_etudiant, CC1
FROM     Notes N, Etudiants E
WHERE    N.Cod_etudiant = E.Cod_Etudiant    AND
         Nom_Matiere =  'BDD'
MINUS
SELECT   Code_Etudiant, Nom_etudiant, CC1
FROM     Notes N, Etudiants E
WHERE    N.Cod_etudiant = E.Cod_Etudiant    AND
         CC1 NOT IN (12,14,16);
```

Résultat

E1	**Alaoui**	**12**
E2	**Chakour**	**14**
E4	**Kalouch**	**16**

Exemple : UNION

Affiche tous les étudiants de la base de données et dont les notes de CC1 d'une matière de base de données différente de 12,14 et 16.

```
SELECT   Code_Etudiant, Nom_etudiant, CC1
FROM     Notes N, Etudiants E
WHERE    N.Cod_etudiant = E.Cod_Etudiant    AND
         Nom_Matiere = 'BDD'
UNION
SELECT   Code_Etudiant, Nom_etudiant, CC1
FROM     Notes N, Etudiants E
WHERE    N.Cod_etudiant = E.Cod_Etudiant    AND
         CC1 NOT IN (12,14,16);
```

Résultat

Code_Etud	Nom_Etud	Note_CC1
E1	Alaoui	12
E2	Chakour	14
E3	Daoud	2
E4	Kalouch	16

E1	Alaoui	6
E2	Chakour	2
E2	Chakour	3
E3	Daoud	2
E3	Daoud	1
E4	Kalouch	1

Langage de contrôle des données

- permet aux administrateurs et à certains utilisateur autorisés, dans un environnement multi-utilisateur :
  - de contrôler les accès à la base de données ;
  - de supprimer les autorisations d'accès ;
  - d'établir des procédures pour préserver l'intégrité de données.
- Le contrôle des identifications des utilisateurs s'effectue aux niveau de système.

Extensions JET 4 au standard ANSI SQL-92

Catégorie	Instruction	Extension Jet 4
Tables	ALTER Table	Permet de modifier les définitions des colonnes
	CREATE TABLE	Ajouter un support pour les valeurs par défaut, les restrictions, l'intégrité référentielle en cascade, les clés étrangères rapide et la personnalisation des numéro auto.
Vues et procédures	CREATE PROCEDURE	Création d'un procédure enregistrée
	CREATE VIEW	Création d'une Vue
	DROP PROCEDURE	Suppression d'une procédure existante
	DROP VIEW	Suppression d'une vue existante
	EXECUTE	Exécution d'une procédure

Extensions JET 4 au standard ANSI SQL-92

Catégorie	Instruction	Extension Jet 4
Transaction	BEGIN TRANSACTION	Initiation d'une transaction
	COMMIT [TRANSACTION]	Met fin à la transaction en cours et enregistrer les modification
	ROLLBACK [TRANSACTION]	Met fin à la transaction et restaure les modification
Sécurité	ADD USER	Ajouter un utilisateur
	ALTER DATABASE	Changer le mot de passe de la base de données
	ALTER USER	Changer le mot de passe d'un utilisateur
	CREATE GROUPE	Ajouter d'un nouveau groupe
	CREATE USER	Ajouter d'un nouveau compte utilisateur au groupe de travail
	DROP GROUP	Supprimer le compte d'un groupe
	DROP USER	Supprimer le compte d'un utilisateur
	GARANT	Accorde des privilèges à un utilisateur ou à un groupe
	REVOKE	Supprimer les privilèges d'un utilisateur ou d'un groupe

Langage de contrôle des données

- Contrôle d'accès
  - La commande GRANT : permet d'autoriser les utilisateurs de la baes de données, soit pour un accès total à la base de données, soit de degré limités d'accès sur les tables ou bien sur certains champs d'une table. Les utilisateur peuvent avoir le droit de donner les autorisation aux autres client.
  - Syntaxe :
    - GRANT [ALL PRIVILIGES//Liste_Privilèges][Liste_Champs] ON [Listes_Tables]
      TO Liste_utilisateurs//PUBLIC
      [WITH GRANT OPTION]

Langage de contrôle de données

- Exemples :
  1. Donner l'autorisation à XX pour effectuer toutes les opérations sur les données de base de données 'GestionNotes'. Et qu'il aura la possibilité de donner les autorisations pour autres utilisateurs :
     - **GARANT ALL PRIVILIGES**
       TO XX
       WITH GRANT OPTION;
  - Donner l'autorisation à YY et ZZ pour effectuer les opérations de recherche sur la table 'Notes'.
    - **GARANT SELECT**
      ON Notes
      TO XX, ZZ;

Langage de contrôle de données

- Exemples :
  1. Donner l'autorisation à TT pour effectuer les opérations de modification et de suppression des notes de CC1 et CC2 de la table 'Notes' :
     - **GARANT UPDATE, DELETE (CC1,CC2) ON Notes TO TT;**
  2. Donner l'autorisation à tous les utilisateurs pour effectuer les opérations de recherche et de modification sur la table 'Etudiants'.
     - **GARANT SELECT, UPDATE ON Etudiants TO PUBLIC;**

Langage de contrôle de données

- Contrôle d'accès
  - La commande REVOKE : permet aux administrateurs de la bases de données, pour enlever les autorisations accordées aux utilisateurs. Les utilisateurs autorisés avec l'option WITH GRANT OPTION, peuvent également retirer les droits d'accès accordés aux utilisateurs.
  - Syntaxe :
    - REVOKE [ALL PRIVILIGES**//**Liste_Privilèges][Liste_Champs] ON [Listes_Tables]
      FROM Liste_utilisateurs//PUBLIC;

Langage de contrôle de données

- Exemples :
  1. Retirer toutes les autorisation accordées a l'utilisateur XX :
     - **REVOKE ALL PRIVILIGES**
       FROM XX
  - Retirer l'autorisation de recherche accordée aux l'utilisateurs XX et ZZ sur la table 'Notes'.
    - **REVOKE SELECT**
      ON Notes
      FROM XX, ZZ;

- Private Sub AutoIncTest()
- Dim cnx As ADODB.Connection
- Dim cmd As ADODB.Command
- Dim rst As ADODB.Recordset
- Set cnx = CurrentProject.Connection
- Set cmd = New ADODB.Command
- Set cmd.ActiveConnection = cnx
- cmd.CommandType = adCmdText
- Set rst = New ADODB.Recordset
- rst.Open "select code_client, age FROM client", cnx, Options:=adCmdText

- cmd.CommandText = "DELETE * FROM CLIENT where AGE > 20"
- cmd.Execute
- cmd.CommandText = "BEGIN TRANSACTION"
- cmd.Execute
- If (rst.RecordCount > 5) Then
- cmd.CommandText = "ROLLBACK"
- Else
- cmd.CommandText = "COMMIT"
- End If
- cmd.Execute

- cmd.CommandText = " create user toto titi tata "
- cmd.Execute'
- cmd.CommandText = " create group fofo fifi "
- cmd.Execute
- cmd.CommandText = " ADD USER toto TO fofo »
- cmd.Execute
- Set cmd = Nothing