Université Abou Bakr Belkaid Tlemcen
Faculté des Sciences
Département d'informatique

Administration des Bases de Données

L3 2017-2018

Mr H.MATALLAH

Administration des Bases de Données

1. Notions fondamentales
2. SQL Avancé
3. Gestion d'intégrité et de cohérence
4. Vues et Index
5. Optimisation des requêtes
6. Gestion des transactions : Gestion des accès concurrents

CHAPITRE 2

SQL Avancé

SQL Avancé

Actions sur une BD

- **Définition de la structure de données** (Contenant)
- **Interrogation des données** (Opérations de lecture du contenu)
- **Mise à jour des données** (Opérations d'écriture du contenu)
  - Insertion
  - Modification
  - Suppression
- **Contrôle des données**

SQL Avancé

SQL « Structured Query Language »

- Langage d'interrogation structuré qui consiste à définir , manipuler, contrôler les BDs
- Composé de 3 Sous langages :

LDD, DDL	Create, Drop, Alter, Rename
LMD, DML	Select, Insert, Update, Delete
LCD, DCL	Grant, Revoke, Commit, Rollback

- Une commande s'appelle un ordre, une instruction s'appelle une requête
- Une requête peut être utilisée de manière interactive ou incluse dans 1programme
- SQL est un standard des BD-Rel, Chaque éditeur tend de développer son propre dialecte

SQL Avancé

- ***SQL « Structured Query Language »***

Date de première version	1974
Auteur	Donald Chamberlin et Raymond Boyce
Développeur	IBM
Dernière version stable	SQL 2011
Paradigme	Déclaratif
Dialectes	SQL 86, SQL 89(SQL1), SQL 92(SQL2), SQL 99(SQL3), SQL 2003, SQL 2008, SQL 2011
Système d'exploitation	Multi-Plateforme

SQL Avancé

Langage de Définition de données

- **Création de table**
  - **CREATE TABLE** (Attribut1 Domaine [Contrainte_integrité],
    Attribut2 Domaine [Contrainte_integrité],
    ……
    ……
    [Contrainte_integrité], …
    [AS SELECT]) ;
  - **AS requête-SQL** : Création de table avec insertion de données issues d'une autre table (Importation de données)
  - Le schéma de la table (Nbre d'attributs, Domaine) correspond à celui retourné par SELECT
  - La définition de la clé primaire peut être reportée
  - « **Describe** » ou « **Desc** » permet d'afficher la structure de la table

SQL Avancé

Langage de Définition de Données

- **Modification de la structure de la table**
  - **ALTER TABLE** Nom_Table
    RENAME TO Nouveau_Nom_Table |
    RENAME COLUMN Old_Name TO New_Name |
    ADD | MODIFY Nom_Col Domaine|
    DROP COLUMN Nom-Col ;
  - **COLUMN** : Facultatif dans plusieurs SGBD
  - **MODIFY** ou **CHANGE** dans quelques SGBD permet aussi de renommer les colonnes
- **Suppression de la table**
  - **DROP TABLE** Nom_Table ;

Langage de Manipulation de Données

- **Ajout de données**

1. **INSERT INTO** Nom_Table (Col1, Col2, Col3, ….)
   VALUES (Val1, Val2, Val3,……) ;

   - Les attributs non spécifiés seront NULL ou à la valeur par défaut
   - Les noms de colonnes sont facultatifs si on respecte l'ordre de définition et toutes les valeurs de colonnes sont fournies

2. **INSERT INTO** Nom_Table (Col1, Col2, Col3, ….)
   SELECT……..….;

   - Insertion de données issues d'une autre table (Migration de données)
   - Le schéma de la table (Nbre d'attributs, Domaine) correspond à celui retourné par SELECT
   - On peut pas ajouter des données dans une table issues d'une sous-sélection de la même table
   - Une constante placée dans SELECT permet d'insérer des lignes avec une valeur par défaut
   - Toutes les clauses du SELECT sont acceptées mise à part ORDER BY

SQL Avancé

Langage de Manipulation de Données

- **Modification des données**

1. **UPDATE** Nom_Table
   SET Col=Exp1, Col2=Exp2, …..
   [WHERE Condition] ;

   - « Exp » peut être une valeur, une fonction ou une formule (250, Max(Qte), Moyenne+2)

2. **UPDATE** Nom_Table
   SET (Col1, Col2, …..) = (SELECT ……)
   [WHERE Condition] ;

   - « Exp » peut être un SELECT renvoyant des valeurs attribuées aux colonnes
   - La condition peut contenir des sous interrogations
   - Il n'est pas conseillé de faire des UPDATE sur des colonnes utilisées dans 1 clé primaire

SQL Avancé

Langage de Manipulation de Données

- **Suppression des données**

1. **DELETE FROM** Nom_Table

   [WHERE Condition] ;

   - Elle est appliquée pour supprimer une ou plusieurs lignes « complétement »
   - Suppression d'une valeur d'une colonne c'est un UPDATE (SET Col=NULL)
   - Si la clause WHERE n'est pas précisée, toutes les lignes de la table sont supprimées
   - La condition peut être plus complexe comme le test d'existence dans une autre table [WHERE Col IN (SELECT Col FROM Table)]

2. **Conseils**

   - Tester d'abord, la clause WHERE dans un simple SELECT
   - Sauvegarder les tuples dans une table temporaire (CREATE …. AS SELECT……)

SQL Avancé

Langage de Manipulation de Données

- **Interrogation**

1. **SELECT** [DISTINCT] {* | Liste_Exp} **Quoi ?** Noms des colonnes à afficher

   FROM {Liste_Tables | Requête AS Nom} **Où ?** Noms des tables ou se trouvent les colonnes

   [WHERE Condition] **Quelle condition ?** Conditions à remplir par les lignes

   [GROUP BY Liste_Exp] Regroupements des lignes en sous tables

   [HAVING Condition] Condition à remplir par le groupe

   [ORDER BY Liste Exp] (ASC, DESC) ; **Comment ?** Ordre d'affichage

- « Exp » est une **colonne,** sinon une **constante** ou une **fonction** ou même une **formule**
- La condition peut employer le résultat d'une autre requête (Sous requête)
- Le nom de la colonne peut ou doit être préfixé par le nom de la table (Table.Col)
- Seules les colonnes du Group By ou les fonctions sont acceptées dans le Select de la requête

SQL Avancé

Langage de Manipulation de Données

- **Alias ou Synonymes**

1. Permettent de renommer des colonnes à l'affichage

```
SELECT  NIN AS Numéro d'Identifiant National, Nom
FROM Etudiant ;
```

NIN	NOM
1763954797	BRAHMI
3984934939	KHEDIM
3274736487	CHEKKAF

- Utile pour un affichage compréhensible de colonnes

1. Permettent de renommer des tables dans la requête

```
SELECT Et.NIN, Et.Nom
FROM Etudiant AS Et ;
```

Numéro d'Identifiant National	NOM
1763954797	BRAHMI
3984934939	KHEDIM
3274736487	CHEKKAF

- Utile pour abréger les noms des tables ou donner un nom au résultat d'une requête
- Utile dans les auto-jointures
- AS est optionnel

SQL Avancé

Langage de Manipulation de Données

- **Fonctions d'agrégation**

1. Moyenne : **AVG** (Col)
2. Somme : **SUM** (Col)
3. Minimum : **MIN** (Col)
4. Maximum : **MAX** (Col)
5. Cardinalité : **COUNT**

COUNT (*)	Nombre d'enregistrements de la table
COUNT (PK)	Nombre de valeurs de la clé (= COUNT(*))
COUNT (Col)	Nombre de valeurs renseignées de la colonne
COUNT (DISTINCT Col)	Nombre de valeurs distinctes et renseignées

SQL Avancé

Langage de Manipulation de Données

- **Jointures**

1. **Produit catésien** (Jointure sans condition)
2. **Thêta-jointure** (Jointure avec une condition quelconque de comparaison entre les attributs)
3. **Equi-jointure** (Jointure avec la condition d'égalité entre les attributs des deux tables)
4. **Jointure naturelle** (Equi-jointure avec test d'égalité entre 2 attributs équivalents en gardant un seul)
5. **Auto-jointure** (Jointure naturelle d'une table avec elle même)
6. **Jointure externe**

Langage de Manipulation de Données

- **Jointures internes et externes**

1. **INNER JOIN** : Jointure interne ou fermée, les données doivent être à la fois dans les 2 tables (*Choix par défaut*)

2. **OUTER JOIN** : Jointure externe ou ouverte, on lit les données d'une table en y associant éventuellement celle de l'autre table (*Remplir avec des valeurs NULL si la condition n'est pas respectée*)

Langage de Manipulation de Données

- **Jointures externes**

3 types d'associations :

1. **LEFT [OUTER] JOIN** : On lit les données de la table de gauche en y associant éventuellement celle de la table de droite

2. **RIGHT [OUTER] JOIN** : On lit les données de la table de droite en y associant éventuellement celle de la table de gauche

3. **FULL [OUTER] JOIN** : On lit les données de la table gauche en y associant celle de la table droite ainsi que les données la table droite en y associant celle de la table gauche

Ex : Liste de tous les étudiants avec leurs notes éventuelles

- SELECT Nom, Note FROM Etudiant E LEFT JOIN Note N ON E.Code = N.Code ;
- Les étudiants qui n'ont pas de notes auront les champs de la table Note à NULL

Langage de Manipulation de Données

- **Jointure interne (INNER)**

NOTE-ETUD

CODE	NOM	NOTE
I45	MOHAMMEDI	17
M64	MOUMENI	12
S12	BOUARFA	8
M27	BELARABI	10

AGE-ETUD

CODE	AGE
S12	20
M44	21
I45	19

INNER JOIN (NOTE-ETUD, AGE-ETUD)

CODE	NOM	NOTE	AGE
I45	MOHAMMEDI	17	19
S12	BOUARFA	8	20

Langage de Manipulation de Données

- **Jointure externe (OUTER)**

NOTE-ETUD

CODE	NOM	NOTE
I45	MOHAMMEDI	17
M64	MOUMENI	12
S12	BOUARFA	8
M27	BELARABI	10

AGE-ETUD

CODE	AGE
S12	20
M44	21
I45	19

LEFT JOIN (NOTE-ETUD, AGE-ETUD)

CODE	NOM	NOTE	AGE
I45	MOHAMMEDI	17	19
M64	MOUMENI	12	
S12	BOUARFA	8	20
M27	BELARABI	10	

Langage de Manipulation de Données

- **Jointure externe (OUTER)**

NOTE-ETUD

CODE	NOM	NOTE
I45	MOHAMMEDI	17
M64	MOUMENI	12
S12	BOUARFA	8
M27	BELARABI	10

AGE-ETUD

CODE	AGE
S12	20
M44	21
I45	19

RIGHT JOIN (NOTE-ETUD, AGE-ETUD)

CODE	NOM	NOTE	AGE
S12	BOUARFA	8	20
M44			21
I45	MOHAMMEDI	17	19

Langage de Manipulation de Données

- **Jointure externe (OUTER)**

NOTE-ETUD

CODE	NOM	NOTE
I45	MOHAMMEDI	17
M64	MOUMENI	12
S12	BOUARFA	8
M27	BELARABI	10

AGE-ETUD

CODE	AGE
S12	20
M44	21
I45	19

FULL JOIN (NOTE-ETUD, AGE-ETUD)

CODE	NOM	NOTE	AGE
I45	MOHAMMEDI	17	19
M64	MOUMENI	12	
S12	BOUARFA	8	20
M27	BELARABI	10	
M44			21

SQL Avancé

Langage de Manipulation de Données

- **Sous requêtes**

La sous-requête dans WHERE de la requête peut retourner une constante ou une relation :

- SELECT…... WHERE Prix = (SELECT Prix FROM…....); *(Retourne une seule valeur)*
- SELECT…... WHERE Prix IN (SELECT Prix FROM…...); *(Retourne une ou plusieurs valeurs)*
- SELECT…... WHERE Prix NOT IN (SELECT Prix FROM…...); *(Retourne une ou plusieurs valeurs)*
- SELECT…..... WHERE Prix < ALL (SELECT Prix FROM…....); *(Retourne une ou plusieurs valeurs)*
- SELECT……. WHERE Prix < ANY (SELECT Prix FROM…...); *(Retourne une ou plusieurs valeurs)*
- SELECT…..... WHERE EXISTS (SELECT... WHERE Prix=5); *(Teste si l'ensemble n'est pas vide)*
- SELECT…..... WHERE NOT EXISTS (SELECT... WHERE Prix=5); *(Teste si l'ensemble est vide)*

SQL Avancé

Langage de Manipulation de Données

- **Sous requêtes**

Etudiant (<u>Code</u>, Nom, Age, Note)

1. Noms des étudiants ayant le même âge que « Benaissa »

```
SELECT Nom  FROM Etudiant
WHERE Age = (SELECT Age FROM Etudiant WHERE Nom='Benaissa')
```

Cette sous interrogation doit renvoyer une et une seule ligne

2. Noms des étudiants ayant au moins une note supérieure à toutes les notes de « Mokri »

```
SELECT Nom  FROM Etudiant
WHERE Note > ALL (SELECT Note FROM Etudiant WHERE Nom='Mokri')
```

Cette sous interrogation peut renvoyer une ou plusieurs lignes