Centre de Réadaptation de Mulhouse

DI

DEVELOPPEUR INFORMATIQUE

SQL

Langage de définition de données

Dossier d'informations

Module 1

Séquence 5

Capacité 1

1 PRESENTATION GENERALE 3
2 DEFINITION D'UNE TABLE 4
2.1 Créer une table 4
2.1.1 Création simple 4
2.1.2 Création avec insertion de données 5
2.2 Contraintes d'intégrité 5
2.3 Modifier une table 7
2.3.1 Modification du type de colonnes : 8
2.3.2 Ajout de colonnes 8
2.3.3 Ajout d'une contrainte 8
2.3.4 Suppression d'une contrainte 8
2.4 Supprimer une table 9
3 DEFINITION D'UNE VUE 10
3.1 Qu'est-ce qu'une vue 10
3.2 Créer une vue 10
3.3 Supprimer une vue 10
4 DEFINITION D'UN INDEX 11
4.1 Qu'est-ce qu'un index 11
4.2 Créer in index 11
4.3 Supprimer un index 11
5 Contrôle d'accès aux données 12
5.1 Attribution de permissions 12
5.2 Révocation de permissions 12

1 PRESENTATION GENERALE

Les systèmes de gestion de bases de données relationnelles (SGBDR) offrent un langage de requêtes puissant : SQL=Structured Query Language.

SQL comporte un nombre limité de mots-clés que l'on peut classer en trois groupes :

- DDL (Data Definition Language) pour la description de la structure de la base de données.
  Exemples: CREATE, DROP, ALTER
- DML (Data Manipulation Language) pour la manipulation des tables et des vues.
  Exemples: SELECT, INSERT, UPDATE, …
- DCL (data Control Language) pour la gestion des transactions et le contrôle des accès aux données
  Exemples: GRANT, REVOKE, COMMIT, ROLLBACK,…

SQL est disponible sous deux formes :

- Un modèle interactif (exécution immédiate de la requête)
- La possibilité d'intégrer des requêtes SQL dans un langage hôte tel que C, PASCAL, COBOL, Visual Basic…

L'objectif de ce dossier est l'acquisition du langage de définition de données pour gérer les objets principaux d'une base de données :

- Les tables
- Les vues
- Les index

D'autre part sont présentées les instructions permettant le contrôle d'accès aux données

Les exemples de ce dossier s'appliquent au SGBD SQL SERVER.

2 DEFINITION D'UNE TABLE

La table est l'objet de la base contenant les données des utilisateurs.
Chaque table possède un nom unique dans la base de données
Une table est composée d'une ou plusieurs colonnes.
Chaque colonne possède un nom unique à l'intérieur de la table

2.1 Créer une table

2.1.1 Création simple

CREATE TABLE *Nom_de_la_table* (*nom_colonne1 Type_de_donnée*,
nom_colonne2 Type_de_donnée,
…)

Quelques types de données sous SQL Server :

CHAR(n)	Chaîne de caractères de longueur fixe *n*
VARCHAR(n)	Chaîne de caractères de *n* caractères maximum
NUMERIC(n,[d]) ou DECIMAL(n,[d])	Nombre de *n* chiffres [optionnellement *d* après la virgule]
INTEGER (ou INT), SMALLINT	Entier et entier court
DATETIME	Données constituées d'une date et d'une heure
FLOAT	Nombre à virgule flottante

Les types de données changent suivant les SGBD

Exemple :

```
CREATE TABLE défauts
                (def_num INTEGER,
                 def_lib CHAR(20),
                 def_freq INTEGER,
                 def_création DATETIME)
```

2.1.2 Création avec insertion de données

(possible avec certains SGBD; pas avec SQL Server)

CREATE TABLE *Nom_de_la_table* (*nom_colonne1 Type_de_donnée*,
nom_colonne2, *Type_de_donnée*
…)
AS SELECT …

On peut ainsi en un seul ordre SQL créer une table et la remplir avec des données provenant du résultat d'un SELECT.
Avec certains SGBD, le type de données peut être omis; les types de données sont ceux provenant du SELECT

Exemple :
CREATE TABLE défauts_fréquents (def_num Integer, def_lib varchar(20))
AS SELECT def_num, def_freq
FROM défauts
WHERE def_freq > 50

2.2 Contraintes d'intégrité

NOT NULL
La colonne ne peut pas contenir de valeur NULL (attribut obligatoire)
UNIQUE
Chaque ligne de la table doit avoir une valeur différente ou NULL pour cette colonne.
PRIMARY KEY
Identifie la clé primaire
FOREIGN KEY
Identifie une clé étrangère
CHECK
Permet de spécifier des valeurs acceptables pour une colonne
DEFAULT
Définit une valeur par défaut

Il est possible de donner un nom à une contrainte avec le mot-clé CONSTRAINT. Ce nom s'affiche en cas de non-respect de l'intégrité. Il est également possible de supprimer une contrainte nommée par l'instruction DROP CONSTRAINT.

Exemple 1 :

```
CREATE TABLE clients (Nom        char(30) NOT NULL,
                      Prenom     char(30) NOT NULL,
                      Age        integer CHECK (age < 100),
                      Dept       integer CHECK (dept BETWEEN 1 and 95),
                      Email      char(50) NOT NULL, CHECK (Email LIKE
                      '%@%'))
```

- Contrainte NOT NULL: Les champs "Nom", "Prenom" et "Email' sont obligatoires.
- Contrainte CHECK: limite les valeurs d'un champ par une comparaison (age < 100) ou un format (un @ dans email).

Exemple 2 :

```
CREATE TABLE Station (nomStation VARCHAR (30),
                      capacite NUMERIC (10) NOT NULL,
                      lieu VARCHAR(30) NOT NULL,
                      region VARCHAR (30),
                      tarif NUMERIC (10,2) DEFAULT 0,
                      CONSTRAINT cle_station PRIMARY KEY (nomStation),
                      CONSTRAINT cle_lieu_region UNIQUE (lieu, region),
                      CONSTRAINT nom_region CHECK (region IN ('Océan
                                                  Indien',
                                                  'Antilles',
                                                  'Europe',
                                                  'Amériques',
                                                  'Extrême
                                                  Orient')))
```

- Un nom a été affecté à certaines contraintes
- Contrainte DEFAULT: si aucune valeur n'est affectée au champ "Tarif", il prendra la valeur 0
- Contrainte PRIMARY_KEY: désigne la clé primaire ("nomStation")
- Contrainte UNIQUE: impose l'unicité d'un champ ou d'un ensemble de champs (le couple "lieu-région")
- Contrainte CHECK: limite les valeurs d'un champ par une liste ("région")

```
CREATE TABLE Activite (nomStation VARCHAR (30),
                       libelle VARCHAR(30),
                       prix NUMERIC (10,2) DEFAULT 0,
                       PRIMARY KEY (nomStation, libelle),
                       FOREIGN KEY (nomStation) REFERENCES Station
                                   (nomStation))
```

- La contrainte PRIMARY_KEY s'applique ici à 2 champs ("nomStation" et "libelle")
- Contrainte FOREIGN_KEY: désigne une clé étrangère ("nomStation") en liaison avec la clé primaire d'une autre table (champ "nomStation" de la table "Station"). La clé primaire de la table de référence peut être. On aurait pu écrire:
  FOREIGN KEY (nomStation) REFERENCES Station

```
CREATE TABLE Client (idClient NUMERIC (10),
                     nom VARCHAR(30) NOT NULL,
                     prenom VARCHAR (30),
                     ville VARCHAR (30) NOT NULL,
                     region VARCHAR(30),
                     solde NUMERIC (10,2) DEFAULT 0 NOT NULL,
                     PRIMARY KEY (idClient))

CREATE TABLE Sejour (idClient NUMERIC (10),
                      nomStation VARCHAR (30),
                      debut NUMERIC (10),
                      nbPlaces NUMERIC (4) NOT NULL,
                      PRIMARY KEY (idClient, nomStation, debut),
                      FOREIGN KEY (idClient) REFERENCES Client(idClient),
                      FOREIGN KEY (nomStation) REFERENCES
                                    station(nomStation))
```

2.3 Modifier une table

Quatre types de modifications sont possibles avec SQL Server:
- Modification du type d'une colonne (avec des restrictions)
- Ajout d'une colonne
- Ajout d'une contrainte
- Suppression d'une contrainte

Dans la plupart des SGBD, on ne peut supprimer ou rétrécir un attribut qu'en copiant la table dans une autre table.

2.3.1 Modification du type de colonnes :

ALTER TABLE *Nom_de_la_table*
ALTER COLUMN *Nom_colonne nouveau type*

Exemple :
ALTER TABLE défauts
ALTER COLUMN def_num SmallInt

2.3.2 Ajout de colonnes

ALTER TABLE *Nom_de_la_table* **ADD**
nom_colonne1 Type_de_donnée,
nom_colonne2 Type_de_donnée,
…

Exemple :
ALTER TABLE défauts ADD
def_remarque CHAR(30)

2.3.3 Ajout d'une contrainte

ALTER TABLE *Nom_de_la_table* **ADD**
CONSTRAINT …..

Exemple :
ALTER TABLE défauts
ADD
CONSTRAINT CK_def_freq CHECK (def_freq < 100)

2.3.4 Suppression d'une contrainte

ALTER TABLE *Nom_de_la_table* **DROP**
CONSTRAINT *Nom_de_la_contrainte*

Exemple :
ALTER TABLE défauts
DROP
CONSTRAINT CK_def_freq

2.4 Supprimer une table

DROP TABLE *Nom_de_la_table*

Renommer une table

RENAME TABLE Nom_ancien TO Nouveau_Nom

3 DEFINITION D'UNE VUE

3.1 Qu'est-ce qu'une vue

Une vue est une table virtuelle , c'est-à-dire qui ne contient pas de données et qui correspond à un filtre appliqué sur une ou plusieurs tables. Une vue permet à un utilisateur d'avoir sa vision propre des données.

3.2 Créer une vue

CREATE VIEW *Nom_de_la_vue [* (*nom_colonne1*,
nom_colonne2,
…)]
AS SELECT …

Exemple :
CREATE VIEW liste
AS SELECT def_num, def_lib
FROM défauts

CREATE VIEW liste (numéro, libellé_défaut)
AS SELECT def_num, def_lib
FROM défauts

3.3 Supprimer une vue

DROP VIEW *Nom_de_la_vue*

4 DEFINITION D'UN INDEX

4.1 Qu'est-ce qu'un index

Selon le modèle relationnel, les sélections peuvent être faites en utilisant le contenu de n'importe quelle colonne et les lignes sont stockées dans n'importe quel ordre.
Un moyen de retrouver la ou les lignes concernée(s) par une sélection est de balayer toute la table. Les temps de réponse peuvent alors être importants. Une solution offerte par tous les SGBD est la création d'index

Mais attention, l'adjonction d'index ralentit les mises à jour, par conséquent sa création doit être justifiée. Certains SGBD crée automatiquement un index correspondant à la clé primaire. (C'est le cas de SQL Server)

4.2 Créer in index

CREATE [UNIQUE] INDEX *Nom_index*
ON *Nom_table* (*nom_colonne1*, *nom_colonne2*,
…)

Exemple :
CREATE INDEX idxlib
ON défauts (def_lib)

4.3 Supprimer un index

DROP INDEX *Nom_index*

5 Contrôle d'accès aux données

Plusieurs personnes peuvent travailler sur une même base de données, toutefois ces personnes n'ont pas forcément les mêmes besoins. Certaines peuvent par exemple modifier des données alors que d'autres ne feront que les consulter.

5.1 Attribution de permissions

GRANT Liste_de_privilèges **ON** Liste_d_objets **TO** Liste_d_utilisateurs

Les privilèges peuvent être DELETE, UPDATE, INSERT, SELECT

Exemple :
GRANT UPDATE (Nom, Prenom)
ON Etudiants
TO Jerome, Françoise, Georges

5.2 Révocation de permissions

REVOKE Liste_de_privilèges **ON** Liste_d_objets **FROM** Liste_d_utilisateurs

Les privilèges peuvent être DELETE, UPDATE, INSERT, SELECT

Exemple :
REVOKE UPDATE (Nom, Prenom)
ON Etudiants
FROM PUBLIC