Mémento SQL

ISO / CEI 9075 / **2011**

Langage de Définition des Données ¤ LDD

Créer ou modifier la structure de la base de données

Création d'une table :

```
CREATE TABLE [base-de-données.[propriétaire].]nom-de-table
( nom-de-colonne  type-de-donnée | [contrainte],
  [nom-de-colonne2 type-de-donnée | [contrainte], …n])
```

Contrainte de clé primaire :

```
[CONSTRAINT nom-de-contrainte]
PRIMARY KEY [CLUSTERED | NONCLUSTERED]
(nom-de-colonne, [nom-de-colonne2][, …n])
```

Contrainte de clé étrangère :

```
[CONSTRAINT nom-de-contrainte]
FOREIGN KEY (nom-de-colonne, [nom-de-colonne2][, …n])
REFERENCES [propriétaire].] table-de-référence
(colonne-de-référence, [colonne-de-référence2][, …n])
```

Contrainte UNIQUE :

```
[CONSTRAINT nom-de-contrainte]
UNIQUE | [CLUSTERED | NONCLUSTERED]
(nom-de-colonne, [nom-de-colonne2][, …n])
```

Contrainte DEFAULT :

```
[CONSTRAINT nom-de-contrainte]
DEFAULT  {constante | fonction niladique | NULL }
[FOR nom-de-colonne]
```

Ajout de contraintes :

```
ALTER TABLE [base-de-données.[propriétaire].]nom-de-table
ADD CONSTRAINT nom-de-contrainte expression-de-la-contrainte | CHECK (nom-de-colonne ('Valeur1', 'Valeur2', 'Valeur3', 'Valeur4, …n))
```

Modification d'une table (Ajout de colonnes) :

```
ALTER TABLE [base-de-données.[propriétaire].]nom-de-table
ADD nom-de-colonne  type-de-donnée | [contrainte],
[nom-de-colonne2 type-de-donnée | [contrainte], …n]
```

Création d'index :

```
CREATE [UNIQUE] INDEX nom-d'index
ON [base-de-données.[propriétaire].]nom-de-table
(nom-de-colonne [ ASC | DESC ]
[, nom-de-colonne2 [ ASC | DESC ] [, …n] )
```

Création de vues :

```
CREATE VIEW [base-de-données.[propriétaire].]nom-de-vue
AS <ordre SELECT>
```

Suppressions de tables, Index, vues :

```
DROP TABLE [base-de-données.[propriétaire].]nom-de-table

DROP INDEX [base-de-données.[propriétaire].]nom-d'index

DROP VIEW [base-de-données.[propriétaire].]nom-de-vue
```

Langage de Manipulation des Données ¤ LMD

Consulter / modifier le contenu de la base de données

recherche de lignes dans une table

```
SELECT [ALL | DISTINCT] {colonne1, colonne2, …n | *}
FROM [base-de-données.[propriétaire].]nom-de-table
[ WHERE {prédicats} [AND | OR ] {prédicats} ]
[ GROUP BY {expressions} ]
[ HAVING {condition} ]
[ ORDER BY {expression1| position} [ASC | DESC]
{,expression2 | position} [ASC | DESC]},…n } ]
```

recherche de lignes dans plusieurs tables

Jointure interne (opérateur = <> > <) :

```
SELECT [ALL | DISTINCT] {colonne1, colonne1, …n| *}
FROM nom-de-table1 [AS Allias-table1]
INNER JOIN table2.colonne2 AS [AS Allias-table2]
ON A1.colonne1 = A2.colonne1
```

Jointure externe gauche :

```
SELECT Alias1.colonne1, Alias2.colonne2
FROM table1 [AS Allias1]
LEFT [OUTER] JOIN Allias2.colonne2
ON Alias1.colonne1= Allias2.colonne2
```

Jointure externe droite :

```
SELECT Alias1.colonne1, Alias2.colonne2
FROM table1 [AS Allias1]
RIGHT [OUTER] JOIN Allias2.colonne2
ON Alias1.colonne1= Allias2.colonne2
```

Jointure externe complète :

```
SELECT Alias1.colonne1, Alias2.colonne2
FROM table1 [AS Allias1]
FULL [OUTER] JOIN Allias2.colonne2
ON Alias1.colonne1= Allias2.colonne2
```

Ajout de données dans une table

```
INSERT INTO [base-de-données.[propriétaire].]nom-de-table
[(colonne1[, colonne2][,…n)]
VALUES (NULL | constante1 [,NULL | constante2,…n])
```

Ajout de données à partir d'une table

```
INSERT INTO [base-de-données.[propriétaire].]nom-de-table
[(nom-de-colonne1 [,nom-de-colonne2][,…n)]
<ordre select>
```

f Combine les données de plusieurs tables

```
MERGE INTO [base-de-données.[propriétaire].]nom-de-table
USING [base-de-données.[propriétaire].]nom-de-table_reference ON
WHEN MATCHED THEN
UPDATE SET nom-de-colonne1 = {NULL | constante1 }
[,nom-de-colonne2 = { NULL | constante2 },…n]
WHEN NOT MATCHED THEN
INSERT [(colonne1[, colonne2][,…n)]
VALUES (NULL | constante1 [,NULL | constante2,…n])
```

Modification de données d'une table

```
UPDATE [base-de-données.[propriétaire].]nom-de-table
SET nom-de-colonne1 = {NULL | constante1 }
[,nom-de-colonne2 = { NULL | constante2 },…n]
[ WHERE {prédicats} [AND | OR ] {prédicats} ]
```

 Modification de données à partir d'une table

UPDATE [base-de-données.[propriétaire].]nom-de-table
SET nom de colonne1 = { NULL | constante1 }
[,nom de colonne2 = { NULL | constante2 },…n]
= <ordre-select>

 Suppression de données (lignes) d'une table

DELETE FROM [base-de-données.[propriétaire].]nom-de-table
[WHERE {prédicats} [AND | OR] {prédicats}]

Langage de Contrôle des Données ¤ LCD

Gérer les privilèges et les actions des utilisateurs

 Octroi d'autorisations de serveurs

GRANT
[CONTROL SERVER |
,CREATE ANY BASEDONNÉES|
,CREATE DDL EVENT NOTIFICATION |
,CREATE ENDPOINT|
, CREATE TRACE EVENT NOTIFICATION |
, EXTERNAL ACCESS ASSEMBLY |
, SHUTDOWN |,UNSAFE ASSEMBLY |
,VIEW ANY BASEDONNÉES |
,VIEW ANY DEFINITION |
,VIEW SERVER STATE]
TO <grantee-principal>[**,** grantee2, **...**]
[WITH GRANT OPTION]
[AS <grantor-principal>]

 Octroi d'autorisations d'objet (tables, lignes, Colonnes)

GRANT [ALL | , ALTER | , CONTROL |, DELETE |
, EXECUTE |, INSERT |, RECEIVE |, REFERENCES |
, SELECT |, TAKE OWNERSHIP |, UPDATE |
, VIEW DEFINITION]
ON [OBJECT ::][nom-schéma]. object_name [
(column [,...n])]
TO <BaseDonnées_principal> [,...n]
[WITH GRANT OPTION]
[AS <BaseDonnées_principal>]
<BaseDonnées_principal> ::= **BaseDonnées-user**
| BaseDonnées_role | Application_role
| BaseDonnées_user_mapped_to_asymmetric_key
| BaseDonnées_user_with_no_login

 Suppression d'autorisations accordée ou refusée

REVOKE [GRANT OPTION FOR]
{ [ALL | , ALTER | , CONTROL |, DELETE |
, EXECUTE |, INSERT |, RECEIVE |, REFERENCES
|
, SELECT |, TAKE OWNERSHIP |, UPDATE |
, VIEW DEFINITION]]
permission [(column [,...n])] [,...n]
ON [class ::] securable]
{ TO | FROM } principal [,...n
] [CASCADE] [AS principal]

Langage de Contrôle des Transactions ¤ LCT

Gérer les transactions (instructions. enchaînés en séquence)

 Gestion des transactions :

 Début d'une transaction locale explicite

BEGIN { TRAN | **TRANSACTION** }
[{ nom-transaction| @nom-variable-transaction }
[WITH MARK ['description']]

 Fin d'une transaction implicite ou explicite réussie

COMMIT { TRAN | **TRANSACTION** }
[nom-transaction| @nom-variable-transaction]]

 Annulation d'une transaction implicite ou explicite

Jusqu'au début de la transaction ou jusqu'au dernier point d'enregistrement

ROLLBACK { TRAN | **TRANSACTION** }
[nom-transaction| @nom-variable-transaction
| nom-point-sauvegarde | @nom-variable-point]

Annulation d'une transaction spécifiée par l'utilisateur depuis le début de la transaction.

ROLLBACK [WORK]

SQL procedural : PSM (Persistent Stored Module)
Développer des procédures stockées, déclencheurs

 Création d'un déclencheur (triggers)

CREATE TRIGGER [nom-schéma .]**nom-trigger**
ON { table | view }
[WITH <dml_trigger_option> [,...n]]
{ FOR | AFTER | INSTEAD OF }
{ [INSERT] [,] [UPDATE] [,] [DELETE] } [
WITH APPEND] [NOT FOR REPLICATION]
AS { sql_statement [;] [,...n] | NOM
EXTERNE<method specifier [;] > }

 Création d'une procédure stockée

CREATE { PROC | **PROCEDURE** } [nom-schéma.]
nom-procédure [; nombre]
[{@parameter [type_nom-schéma.]data_type }
[VARYING] [= default] [OUT | OUTPUT]] [,...n]

[WITH [ENCRYPTION][RECOMPILE
] [EXECUTE_AS_Clause] [,...n]]
[FOR REPLICATION]
AS { [BEGIN] statements [END] ;...n]

 Modification d'un déclencheur (triggers)

ALTER TRIGGER [nom-schéma .]**nom-trigger**
ON { table | view }
[WITH <dml_trigger_option> [,...n]]
{ FOR | AFTER | INSTEAD OF }
{ [INSERT] [,] [UPDATE] [,] [DELETE] }
[WITH APPEND] [NOT FOR REPLICATION]
AS { sql_statement [;] [,...n] | NOM
EXTERNE<method specifier [;] > }

 Suppression d'un déclencheur (triggers)

DROP TRIGGER [nom-schéma .]nom-trigger

 Suppression d'une procédure stockée

DROP { PROC | **PROCEDURE** } { [nom-schéma.]
nom-procédure [; nombre]

 Fonctions agrégats (clause SELECT ou HAVING)

MAX ([ALL | DISTINCT] expression)
MIN ([ALL | DISTINCT] expression)
AVG ([ALL | DISTINCT] expression)
SUM ([ALL | DISTINCT] expression)
COUNT ({ [[ALL | DISTINCT] expression] | * }

Prédicats

ALL ANY BETWEEN BUT EXISTS LIKE IS NULL

Opérateurs de comparaison

= < > <> >= <=

Opérateurs arithmétiques

Addition : + Soustraction : - Multiplication : *
Division : / Exponentiation : ^

Type de Données

bit | bigint | int | smallint | tinyint | numeric | decimal | float | real | binary | varbinary | image char | varchar | text | money | smallmoney | time | timestamp | date | datetime | smalldatetime | nchar | nvarchar | ntext | timestamp | xml | sql variant

Fonctions de fenêtrage et analytique

Pagination

SELECT {colonne1, **ROW_NUMBER() OVER** ([**ORDER BY** {expression1| position} [ASC | DESC] {,expression2 | position} [ASC | DESC]},…n }], colonne2, …n | *}
FROM [base-de-données.[propriétaire].]nom-de-table

Division des lignes en groupes

SELECT {colonne1, **NTILE(**nombre entier de groupes**) OVER** ([**ORDER BY** {expression1| position} [ASC | DESC] {,expression2 | position} [ASC | DESC]},…n }], colonne2, …n | *}
FROM [base-de-données.[propriétaire].]nom-de-table

Comparaison de la ligne avec la ligne précédente

SELECT {colonne1,**LAG(**colonne2**) OVER** ([**ORDER BY** {expression1| position} [ASC | DESC] {,expression2 | position} [ASC | DESC]},…n }], colonne3, …n | *}
FROM [base-de-données.[propriétaire].]nom-de-table

Comparaison de la ligne avec la ligne suivante

SELECT {colonne1,**LEAD(**colonne2**) OVER** ([**ORDER BY** {expression1| position} [ASC | DESC] {,expression2 | position} [ASC | DESC]},…n }], colonne3, …n | *}
FROM [base-de-données.[propriétaire].]nom-de-table