SQL : syntaxe et exemples

Pour décrire la syntaxe des commandes SQL, on utilise les conventions suivantes :

Les crochets [ ] indiquent un élément facultatif (optionnel).
Les accolades { } indiquent plusieurs choix possibles séparés par des |.
[. . .] indique une répétition de l'élément précédent.
[, . . .] indique une répétition de l'élément précédent(avec une virgule pour séparer).

Exemple :

xxx { a | b | c } [yyy [, yyy]]

signifie que après **xxx** figure soit **a** soit **b** soit **c** suivi éventuellement de **yyy** répété plusieurs fois et séparés alors par des virgules.

Par exemple :

xxx a
xxx b yyy
xxx c yyy, yyy
xxx c yyy, yyy, yyy

Vous trouverez, dans les pages suivantes, les syntaxes des commandes SQL vues dans le module M1104 :

- SELECT
- CREATE, ALTER, DROP TABLE
- INSERT, UPDATE, DELETE
- CREATE, DROP VIEW

1) SELECT

```
SELECT [ ALL | DISTINCT ] { * | expression [ AS nom ] [, ...] }
INTO [ TEMPORARY | TEMP ] [ TABLE ] nouv_nom_relation
[ FROM clause_from [, ...] ]
[ WHERE condition ]
[ GROUP BY expression [, ...] ]
[ HAVING condition [, ...] ]
[ { UNION | INTERSECT | EXCEPT } [ ALL ] requete ]
[ ORDER BY expression [ ASC | DESC ] [, ...] ]
[ LIMIT { nombre | ALL } ]
[ OFFSET début ]]
```

où clause_from est :

```
      nom_relation [ [ AS ] alias [ ( liste_alias_attributs ) ] ]
|     ( select ) [ AS ] alias [ (liste_alias_attributs) ]
|     clause_from [NATURAL] JOIN clause_from [ON condition | USING (liste_attributs)]
```

Exemples :

```
Select titre, contenu from cours where numcours = 1;

Select distinct c.numetu
      from cours c, resultat r
      where c.numetu = r.numetu and note < 8;

Select numetu, note/2 as note_sur_10
      from resultat
      order by note desc;

Select numcours, max(note)
      from resultat
      group by numcours;

Select an_naiss, avg(note)
      from resultat
      where an_naiss > 1984
      group by an_naiss
      having avg(note) >= 13;

Select numcours, count(numetu) into petits_effectifs (numcours, nbetu)
      from resultat
      group by numcours
      having count(numetu) < 15;
```

Jointures :

```
        Select * from cours, resultat where cours.numetu = resultat.numetu;
      = Select * from cours join resultat on cours.numcours = resultat.numcours;
      = Select * from cours join resultat using (numcours);
      = Select * from cours natural join resultat;
```

Les 2 premières jointures dupliquent l'attribut de jointure numcours.

2) CREATE, ALTER, DROP TABLE

```
CREATE [ { TEMPORARY | TEMP } ] TABLE nom_relation (
{ nom_attribut type_attribut [ DEFAULT val_par_defaut ] [ contrainte_att [ ... ] ]
| contrainte_relation
} [, ... ]
)
```

où contrainte_att est :

```
[ CONSTRAINT nom_contrainte ]
{ NOT NULL | NULL | UNIQUE | PRIMARY KEY |
  CHECK (expression) | REFERENCES relation_ref [ ( att_ref ) ] }
```

et contrainte_relation est:

```
[ CONSTRAINT nom_contrainte ]
{ UNIQUE ( nom_attribut [, ... ] )
| PRIMARY KEY ( nom_attribut [, ... ] )
| CHECK ( expression )
| FOREIGN KEY ( nom_attribut [ ,... ] ) REFERENCES relation_ref ( att_ref [ ,... ] )
}
```

```
ALTER TABLE nom_relation
{     ADD [ COLUMN ] nom_attribut type_attribut [ contrainte_att [ , ... ] ]
|     DROP [ COLUMN ] nom_attribut
|     ALTER [ COLUMN ] nom_attribut { SET DEFAULT val_par_defaut | DROP DEFAULT }
|     ALTER [ COLUMN ] nom_attribut { SET | DROP } NOT NULL
|     RENAME [ COLUMN ] nom_attribut TO nouv_nom_attribut
|     RENAME TO nouv_nom_relation
|     ADD contrainte_relation
|     DROP CONSTRAINT nom_contrainte
}
```

```
DROP TABLE nom_relation
```

Exemples :

```
CREATE TABLE etudiant
(     numetu      numeric(2)  primary key,
      nom         varchar(15) not null,
      an_naiss    numeric(4)  DEFAULT 1984 CONSTRAINT c_naiss CHECK (an_naiss≥1975));

CREATE TABLE cours
(     numcours numeric (3),
      titre varchar (30) not null ,
      contenu text,
      primary key (numcours));

CREATE TABLE resultat
(     numetu numeric (2) references etudiant(numetu),
      numcours numeric (3),
      note numeric(2),
      primary key (numetu, numcours),
      foreign key (numcours) references cours (numcours));
```

Remarque : Dans une clé étrangère, on peut référencer des attributs qui forment la clé primaire de l'autre relation (ou qui ont la contrainte UNIQUE).

3) INSERT, UPDATE, DELETE

```
INSERT INTO nom_relation [ ( nom_attribut [, ...] ) ]
{     DEFAULT VALUES
|     VALUES ( { expression | DEFAULT } [, ...] )
|     SELECT requete }

UPDATE nom_relation SET nom_attribut = expression [, ...] [
 WHERE condition ]

DELETE FROM table [ WHERE condition ]
```

Exemples :

```
Insert into etudiant (numetu, nom) values (3, 'Martin') ;
Insert into cours values (2, 'BD', 'Principes des BD') ;
Insert into resultat select numetu, 4 , NULL from etudiant where nom = 'Dupont' ;

Update cours set titre = 'programmation' where numcours = 2 ;
Update resultat set note = note + 1 ;
Update resultat set note = (Select avg(note) from resultat) where numetu = 2 ;

Delete from etudiant where numetu = 2 ;
Delete from cours ;                -- Supprime tous les n-uplets !
```

4) CREATE, DROP VIEW

```
CREATE VIEW nom_vue [ ( liste_attributs ) ] AS SELECT requete

DROP VIEW nom_vue [, ...]
```

Exemples :

```
   CREATE VIEW moyenne (numetu, moy)
   AS SELECT numetu, avg(note) FROM resultat GROUP BY numetu
ou
   CREATE VIEW moyenne
   AS SELECT numetu, avg(note) as moy FROM resultat GROUP BY numetu
```