Requêtes SQL

Nicolas Ribot

Plan

- Data Definition Language (DDL)
- Data Manipulation Language (DML)
- Data Control Language (DCL)
- Exemples de requêtes
- Sous-requêtes
- Common Table Expression
- Jointures

Data Definition Language

- Définition de la structure ou du schéma de la base:
- CREATE: creation d'objet dans la base (tables, schémas, fonctions)
- ALTER: modification de la structure de la base
- DROP: supprime des objets de la base
- TRUNCATE: supprime tous les enregistrements d'une table
- COMMENT: ajoute un commentaire sur un objet (base, schéma, table, colonne, etc.)
- RENAME: renomme un objet

Data Manipulation Language

- Manipulation des objets en base;
- SELECT: récupère des données depuis la base
- INSERT: insère des données dans une table
- UPDATE: Met à jour les données d'une table
- DELETE: supprime des données d'une table
- Autres (explain, lock, …)

Data Control Language

- Manipulation des droits en base;
- GRANT: attribue des droits à un utilisateur, un groupe
- REVOKE: enlève des droits à un utilisateur, un groupe

Exemple de requêtes

```
SELECT * from matable;
SELECT nom, id, st_transform(geom, 4326)
from communes;
SELECT * from matable
where (id between 12 and 2456)
OR NOM LIKE 'Dup%';
INSERT INTO matable (col1, col2, col3)
    VALUES (12,
            'un texte',
            st_geomFromText('POINT (1.24
42.23', 4326);
```

Exemple de requêtes

```
INSERT INTO matable
    SELECT col1, col2 FROM table;
UPDATE matable set col1 = col1 + 12
where col2 < 200;
UPDATE matable set col1 = t.col2
from table2 t
where t.col2 < 200;
DELETE from matable;
DELETE from matable WHERE col2 is null;
```

Sous-requêtes

- Permet de créer une table virtuelle dont on peut se servir dans le SELECT
- Imbrication infinie
- Exemple: Table (nom, age)
  - -> trouver les noms des personnes les plus vieilles

```
SELECT * from matable order by age DESC;
SELECT nom from matable where age = max(age) ?
SELECT nom
from
   (select max(age) as maxage from matable) as t
Where matable.age = t.maxage;
```

CTE

- Nouvelle construction permettant de réécrire les sous-requêtes
- Mot-clé WITH
- Plus lisible
- Exemple:

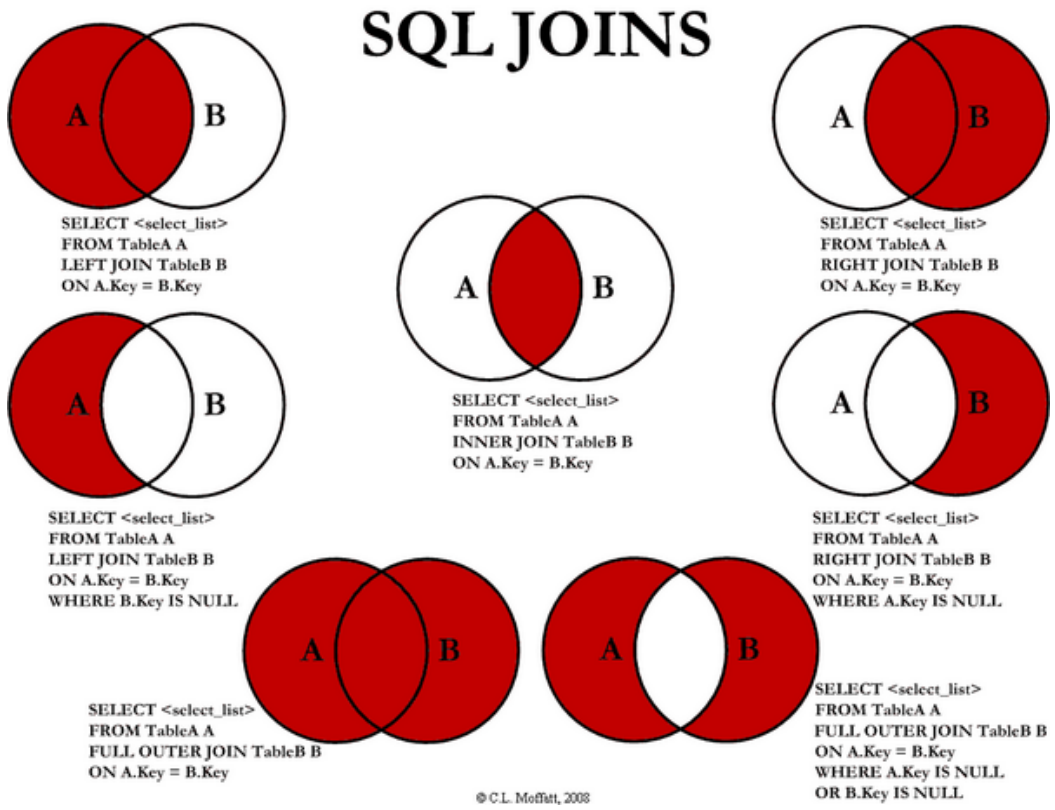
```
WITH tmp as (
   select max(age) as maxage from matable
) select nom
from matable, tmp
Where matable.age = tmp.maxage;
```

CTE

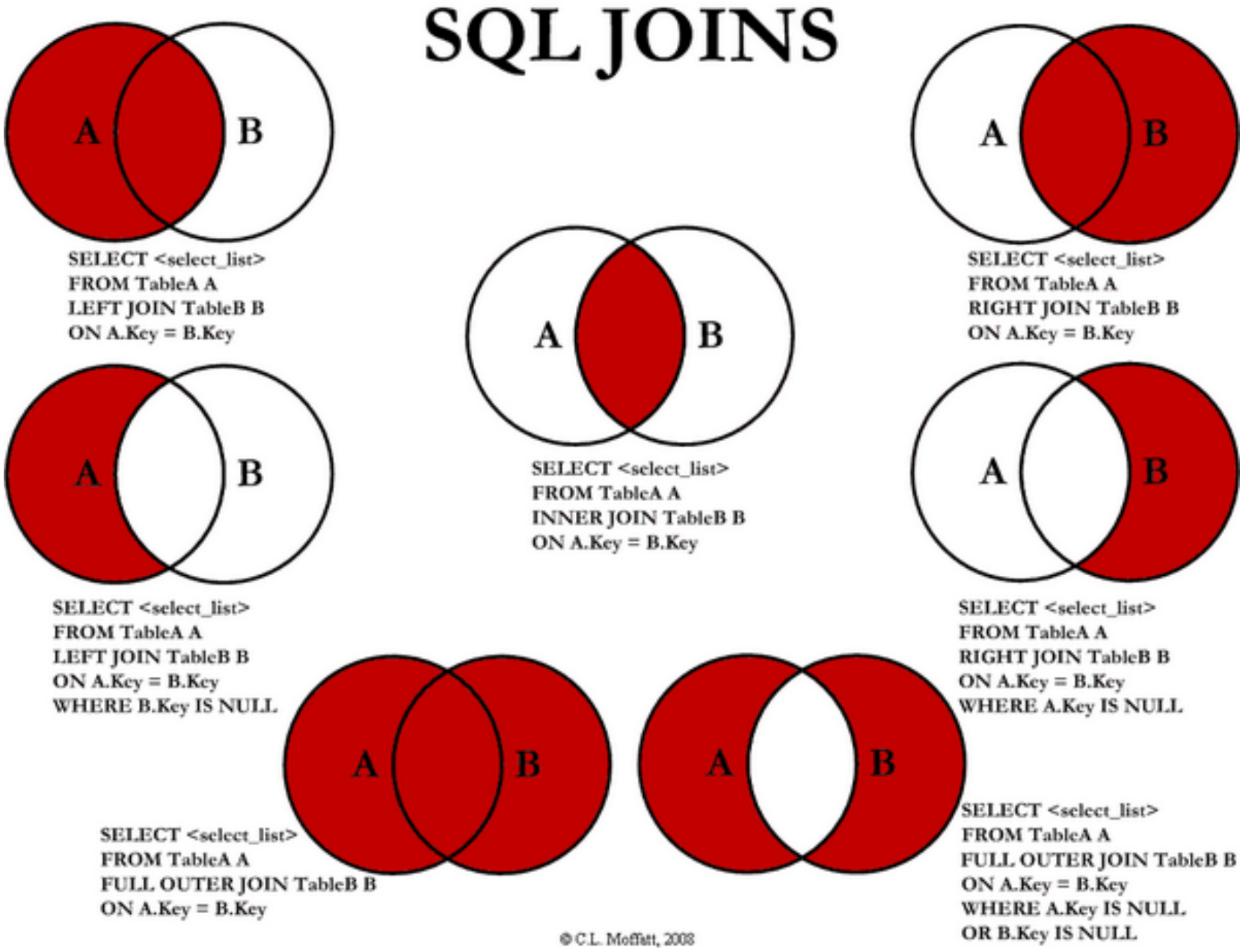
- Plusieurs tables intermédiaires peuvent être définies
- Utilisable avec SELECT, INSERT, UPDATE, DELETE

Jointures

- Permettent de relier plusieurs tables entre elles dans une requête
- Indispensables lors de requêtes complexes
- Implicites ou explicites



Jointures


SQL JOINS
A
B
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
© C.L. Moffatt, 2008

Jointures

- En l'absence de jointure entre 2 tables A et B, le produits cartésien des deux tables est retourné: Count(A) x Count(B)

```
select * from A, B;
```

id
1
2
3

X

nom
a
b
c

=

#	id	nom
1	1	a
2	1	b
3	1	c
4	2	a
5	2	b
6	2	c
7	3	a
8	3	b
9	3	c

Jointures implicites

- Lien entre champs d'une table sans le mot-clé JOIN

```
SELECT nom
from
   (select max(age) as maxage from matable) as t
Where matable.age = t.maxage;
```

Jointure implicite entre t et matable

Jointures explicites

- Lien entre champs d'une table avec le mot-clé JOIN

```
  table t1          table t2
 num | name      num | value
-----+------    -----+-------
   1 | a           1 | xxx
   2 | b           3 | yyy
   3 | c           5 | zzz
```

Jointures explicites

```
SELECT * FROM t1 CROSS JOIN t2;
  num | name |  num | value
 -----+------+------+-------
    1 | a    |    1 | xxx
    1 | a    |    3 | yyy
    1 | a    |    5 | zzz
    2 | b    |    1 | xxx
    2 | b    |    3 | yyy
    2 | b    |    5 | zzz
    3 | c    |    1 | xxx
    3 | c    |    3 | yyy
    3 | c    |    5 | zzz
(9 rows)
```

Jointures explicites

```
SELECT * FROM t1 INNER JOIN t2 ON t1.num = t2.num;
 num | name | num | value
-----+------+-----+-------
   1 | a    |   1 | xxx
   3 | c    |   3 | yyy
(2 rows)

SELECT * FROM t1 INNER JOIN t2 USING (num);
 num | name | value
-----+------+-------
   1 | a    | xxx
   3 | c    | yyy
(2 rows)

SELECT * FROM t1 NATURAL INNER JOIN t2;
 num | name | value
-----+------+-------
   1 | a    | xxx
   3 | c    | yyy
(2 rows)
```

Jointures explicites

```
SELECT * FROM t1 LEFT JOIN t2 ON t1.num = t2.num;
 num | name | num | value
-----+------+-----+-------
   1 | a    |   1 | xxx
   2 | b    |     |
   3 | c    |   3 | yyy
(3 rows)

SELECT * FROM t1 LEFT JOIN t2 USING (num);
 num | name | value
-----+------+-------
   1 | a    | xxx
   2 | b    |
   3 | c    | yyy
(3 rows)
```

Jointures explicites

```
SELECT * FROM t1 RIGHT JOIN t2 ON t1.num = t2.num;
 num | name | num | value
-----+------+-----+-------
   1 | a    |   1 | xxx
   3 | c    |   3 | yyy
     |      |   5 | zzz
(3 rows)
```

```
SELECT * FROM t1 FULL JOIN t2 ON t1.num = t2.num;
 num | name | num | value
-----+------+-----+-------
   1 | a    |   1 | xxx
   2 | b    |     |
   3 | c    |   3 | yyy
     |      |   5 | zzz
(4 rows)
```

Jointures spatiales

- Utilisation d'opérateurs spatiaux pour joindre les tables
- Implicite: forme souvent rencontrée:

```
select c.*
from zone_innondable i, commune c
where st_intersects(i.geom, c.geom);
```

- Explicite: JOIN

```
select c.*
from zone_innondable i
 join commune c
 on (st_intersects(i.geom, c.geom));
```