

SQL (Deuxième partie)

Walter RUDAMETKIN

Bureau F011
Walter.Rudametkin@polytech-lille.fr

Les requêtes de consultation

Représente la majorité des requêtes

Encapsule complètement l'algèbre relationnel

Une seule commande !

2

Syntaxe partielle commande Select

```
SELECT [ ALL | DISTINCT [ ON ( expression
[, ...] ) ] ]
* | expression [ AS output_name ] [, ... ]
[ FROM from_item [, ...] ]
[ WHERE condition ]
[ GROUP BY expression [, ...] ]
[ HAVING condition [, ...] ]
[ { UNION | INTERSECT | EXCEPT [ ALL ] } select ]
[ ORDER BY expression [ ASC | DESC | USING
operator ] [, ...] ]
[ FOR UPDATE [ OF tablename [, ...] ] ]
[ LIMIT { count | ALL } [ { OFFSET | , } start ] ]
```

3

Ex. bibliothèque - état de la base

| auteur | num_a | nom |
|--------|-------|---------------|
| | 1 | Albert Uderzo |
| | 2 | Victor Hugo |
| | 3 | J.K. Rowling |

| livre | num_l | titre | auteur |
|-------|-------|--|--------|
| | 1 | Le fils d'Asterix | 1 |
| | 2 | Les misérables | 2 |
| | 3 | Notre dame de Paris | 2 |
| | 4 | Harry Potter à l'école des sorciers | 3 |
| | 5 | Harry Potter et la chambre des secrets | 3 |

| editeur | num_e | nom | ville |
|---------|-------|-------------|-----------|
| | 1 | Albert-René | Bruxelles |
| | 2 | Gallimard | Paris |
| | 3 | Folio | Paris |

4

Ex. bibliothèque - état de la base

| edite_par | num_l | num_e | date_edition | emprunt | num_l | num_u |
|-----------|-------|-------|--------------|---------|-------|-------|
| | 1 | 1 | 1998-03-24 | | 1 | 1 |
| | 2 | 3 | 1940-02-02 | | 2 | 4 |
| | 3 | 2 | 1967-06-12 | | 4 | 1 |
| | 4 | 2 | 1999-03-01 | | | |
| | 5 | 2 | 2000-02-01 | | | |

| reserve | num_l | num_u |
|---------|-------|-------|
| | 1 | 2 |
| | 4 | 2 |

| utilisateur | num_u | nom | premier |
|-------------|-------|------------|-------------|
| | 1 | Caron | Olivier |
| | 2 | Janot | Stéphane |
| | 3 | Seynhaeve | Franck |
| | 4 | Duthilleul | Jean-Michel |

5

Consultation simple d'une table

Syntaxe :

- select col1, col2, ..., coln from nomTable
- Variante usuelle : select * from nomTable

Exemple :

```
select * from utilisateur ;
```

| num_u | nom | premier |
|-------|------------|-------------|
| 1 | Caron | Olivier |
| 2 | Janot | Stephane |
| 3 | Seynhaeve | Franck |
| 4 | Duthilleul | Jean-michel |

6

Expression d'une projection

Définition : la projection d'une relation R de schéma R(A1,A2,...,An) sur les attributs A1i,A2i,...,Aip) ($p < n$) est une relation R0(A1i,A2i,...,Aip) dont les tuples sont obtenus par élimination des valeurs des attributs de R n'appartenant pas à R0.

Syntaxe :

```
select coli1, coli2, ..., colip from table_name
```

Exemple :

```
select nom, prenom from utilisateur ;
```

| nom | prenom |
|------------|-------------|
| Caron | Olivier |
| Janot | Stephane |
| Seynhaeve | Franck |
| Duthilleul | Jean-michel |

On peut inverser l'ordre de présentation (aucun impact sur le calcul)

La clause **distinct** (permet de supprimer les doublons)

7

Restriction

Définition : la restriction (ou sélection) de la relation R par une qualification Q est une relation R0 de même schéma dont les tuples sont ceux de R satisfaisant la qualification Q.

La qualification peut être exprimée à l'aide de constantes, comparateurs arithmétiques, opérateurs logiques

Prédicat :

- La qualification est de la forme **<attribut> <opérateur> <valeur>** avec opérateur $\in \{ =, \neq, <, \leq, >, \geq \}$
- Il est possible de composer plusieurs conditions de base à l'aide des opérateurs booléens de disjonction (OR), conjonction (AND), négation (NOT).

8

Expression d'une restriction

Introduction clause **WHERE**

Utilisation des opérateurs booléens : **and, or, not**

Comparaison de chaînes, dates, d'entiers, ...

Exemple :

```
select * from livre where auteur=2 ;
```

| num_l | titre | auteur |
|-------|---------------------|--------|
| 2 | Les misérables | 2 |
| 3 | Notre dame de Paris | 2 |

9

Traitement de chaînes (1/3)

Opérateur **LIKE**

- Caractère spéciaux : '%' (remplace de 0 à plusieurs caractères) et '?' (remplace exactement un caractère).

Exemple :

```
select distinct titre from livre where titre like 'H%' ;
```

| titre |
|--|
| Harry Potter à l'école des sorciers |
| Harry Potter et la chambre des secrets |

10

Traitement de chaînes (2/3)

Opérateur de comparaison =, <, >, <=, >=, <=, ... (ordre lexicographique) (aussi applicable à tout type INTEGER, DATE...)

Opérateur de concaténation ||, fonctions prédéfinies (ex : upper, lower)

Exemple :

```
select upper(nom || ' ' || prenom) as nom_prenom from utilisateur;
```

| nom_prenom |
|------------------------|
| CARON OLIVIER |
| JANOT STEPHANE |
| SEYNHAEVE FRANCK |
| DUTHILLEUL JEAN-MICHEL |

Renommage d'attribut

11

Traitement de chaînes (3/3)

Comparaison de chaînes : clause **BETWEEN**

permet de vérifier si la valeur d'un attribut est comprise entre deux constantes

Exemple :

```
select nom from utilisateur where nom between 'A%' and 'F%' ;
```

| nom |
|------------|
| Caron |
| Duthilleul |

Note : l'exemple suivant est identique :

```
select nom from utilisateur where nom >= 'A%' and nom <= 'F%' ;
```

Applicable à tout type (integer, chaîne, date, ...)

12

Présentation des données (1/2)

- Ordre d'affichage des colonnes
- Clause **distinct**, évite les doublons
- Ordre d'affichage des lignes, clause **Order By**
- Ordre des lignes multi-critères

- Aucun impact sur le traitement algébrique des requêtes

13

Présentation des données (2/2)

Syntaxe :

ORDER BY expression [ASC | DESC | USING [operator] [, ...]

Exemple :

```
select * from livre order by auteur DESC, titre ASC ;
```

| num_l | titre | auteur |
|-------|--|--------|
| 4 | Harry Potter à l'école des sorciers | 3 |
| 5 | Harry Potter et la chambre des secrets | 3 |
| 2 | Les misérables | 2 |
| 3 | Notre dame de Paris | 2 |
| 1 | Le fils d'Asterix | 1 |

14

Opérations de calcul

Opérateurs **arithmétiques** : +, -, ...

Exemple :

```
select now()-date_edition as duree, num_l from edite_par ;
```

| duree | num_l |
|---------------------|-------|
| 1423 days 17:30:01 | 1 |
| 22658 days 16:30:01 | 2 |
| 12666 days 17:30:01 | 3 |
| 1081 days 17:30:01 | 4 |
| 744 days 17:30:01 | 5 |

Expressions arithmétiques applicables dans la clause **where**

15

Fonctions de calcul

Une **fonction de calcul** est une fonction qui s'applique sur **un ensemble de tuples** et qui renvoie **une valeur unique**

Syntaxe :

nomFonction(nomColonne) ou nomFonction(*)

Résultat est stocké dans une colonne correspondant au nom de la fonction.

Toujours une ligne résultat.

Fonctions standards :

count, min, max, avg, sum

16

Fonctions de calcul - exemples

Exemple **avec renommage** :

```
select count(*) as nombre from livre ;
```

| nombre |
|--------|
| 5 |

Exemple **sans renommage** :

```
select min(num_l), max(num_l),  
avg(num_l), sum(num_l) from livre ;
```

| min | max | avg | sum |
|-----|-----|-------|-----|
| 1 | 5 | 3.000 | 15 |

17

Calcul sur des groupes de lignes (1/3)

- Sélectionner des lignes pour appliquer un calcul
- Introduction clause **Group By**
- permet de **partitionner la relation en sous-relations** ayant les mêmes valeurs sur les attributs précisés : on peut alors appliquer des fonctions à chaque sous-relation.

```
select ...  
from livre  
group by auteur ;
```

sous-relation ⇨

| auteur | titre | num_l |
|--------|-------|-------|
| a | x | 2 |
| a | y | 1 |
| b | x | 2 |
| b | t | 5 |
| b | u | 3 |
| c | y | 4 |

sous-relation ⇨

sous-relation ⇨

Note : On trouve dans le résultat une ligne par sous-relation.

18

Calcul sur des groupes de lignes (2/3)

Exemple :

```
select auteur, count(*) as nbre_par_auteur from
livre group by auteur ;
```

| auteur | nbre_par_auteur |
|--------|-----------------|
| 1 | 1 |
| 2 | 2 |
| 3 | 2 |

Note : Toutes les colonnes figurant dans un **group by** doivent apparaître dans la clause select.

19

Calcul sur des groupes de lignes (3/3)

- Imposer une condition aux groupes formés par la clause Group By
- Introduction clause **HAVING**
- Poser une condition portant sur chacune des sous-relations générées par le GROUP BY. Les sous-relations ne vérifiant pas la condition sont écartées du résultat.

Exemple :

```
select auteur, count(*) as nbre_par_auteur from livre group by
auteur having count(*)>1 ;
```

| auteur | nbre_par_auteur |
|--------|-----------------|
| 2 | 2 |
| 3 | 2 |

Note : Ne pas confondre avec la clause **where**

20

Produit cartésien

Définition : Le produit cartésien de deux relations R et S est une relation T ayant pour schéma la concaténation de celui de R avec celui de S et pour tuples **toutes les combinaisons des tuples** de R et S.

Opérateur commutatif

$$R \times S = S \times R$$

Opérateur intermédiaire (pas de sens en soi)

21

Expression d'un produit cartésien

Utilisation clause **FROM**

Déclaration de variables (ou utiliser le nom de la table)

Exemples :

```
select * from utilisateur, livre ; ... (20 lignes)
```

```
select distinct e.nom as nomEditeur , a.nom as
NomAuteur from editeur e, auteur a;
```

22

Jointure

Définition : La jointure de deux relations R et S selon une qualification multi-attributs Q est l'ensemble des tuples du produit cartésien R×S satisfaisant la qualification Q

Opérateur commutatif

$$R \llcorner_{\text{prédicat}} S = S \llcorner_{\text{prédicat}} R$$

Prédicat :

<attribut1> <opérateur> <attribut2> où attribut1 ∈ R et attribut2 ∈ S avec opérateur ∈ {=, ≠, <, ≤, >, ≥}

Il est possible de composer plusieurs conditions de base à l'aide des opérateurs booléens de disjonction (OR), conjonction (AND), négation (NOT).

23

Expression d'une jointure

Relier avec cohérence plusieurs tables.

Relier les **clés étrangères** avec les **clés primaires** correspondantes

Exemple :

```
select titre, nom from auteur, livre where
auteur.num_a=livre.auteur ;
```

Expression du produit cartésien

Prédicat de jointure

| titre | nom |
|--|---------------|
| Le fils d'Asterix | Albert Uderzo |
| Les misérables | Victor Hugo |
| Notre dame de Paris | Victor Hugo |
| Harry Potter à l'école des sorciers | J.K. Rowling |
| Harry Potter et la chambre des secrets | J.K. Rowling |

24

Jointures - Exemples

Exemple : liste de couples de livres ayant le même auteur

```
select l1.titre, l2.titre from livre l1, livre l2
where l1.auteur=l2.auteur and l1.titre > l2.titre ;
```

Renommage de table

Prédicat de jointure

| titre | titre |
|--------------------------------|----------------------------|
| Notre dame de Paris | Les misérables |
| Harry Potter et la chambre ... | Harry Potter à l'école ... |

25

Les jointures à la SQL/2

- Nouvelles possibilités d'expression de jointures
- Non encore implémenté par tous les SGBD (ex. Oracle)
- Les expressions de jointures sont exprimés dans la clause **from**
- Distinction de jointures : **inner join** (défaut), **left outer join**, **right outer join**, **full outer join**

- Pour les exemples suivants :

```
insert into livre values (6, 'Le livre inconnu', null) ;
insert into auteur values (4, 'Paltoquet') ;
```

26

Jointure SQL/2 classique

```
select titre, nom
from livre
inner join auteur
on livre.auteur=auteur.num_a ;
```

| titre | nom |
|--|---------------|
| Le fils d'Asterix | Albert Uderzo |
| Les misérables | Victor Hugo |
| Notre dame de Paris | Victor Hugo |
| Harry Potter à l'école des sorciers | J.K. Rowling |
| Harry Potter et la chambre des secrets | J.K. Rowling |

27

Jointure externe gauche

```
select titre, nom
from livre
left outer join auteur
on livre.auteur=auteur.num_a ;
```

| titre | nom |
|--|---------------|
| Le fils d'Asterix | Albert Uderzo |
| Les misérables | Victor Hugo |
| Notre dame de Paris | Victor Hugo |
| Harry Potter à l'école des sorciers | J.K. Rowling |
| Harry Potter et la chambre des secrets | J.K. Rowling |
| le livre inconnu | |

28

Jointure externe droite

```
select titre, nom
from livre
right outer join auteur
on livre.auteur=auteur.num_a ;
```

| titre | nom |
|--|---------------|
| Le fils d'Asterix | Albert Uderzo |
| Les misérables | Victor Hugo |
| Notre dame de Paris | Victor Hugo |
| Harry Potter à l'école des sorciers | J.K. Rowling |
| Harry Potter et la chambre des secrets | J.K. Rowling |
| | Paltoquet |

29

Jointure externe complète

```
select titre, nom
from livre
full outer join auteur
on livre.auteur=auteur.num_a ;
```

| titre | nom |
|--|---------------|
| Le fils d'Asterix | Albert Uderzo |
| Les misérables | Victor Hugo |
| Notre dame de Paris | Victor Hugo |
| Harry Potter à l'école des sorciers | J.K. Rowling |
| Harry Potter et la chambre des secrets | J.K. Rowling |
| le livre inconnu | |
| | Paltoquet |

30

Remarques : syntaxe des jointures

INNER et OUTER sont toujours **facultatifs**

LEFT, RIGHT et FULL impliquent une jointure externe

Les syntaxes :

- T1 { [INNER] | { LEFT | RIGHT | FULL } [OUTER] } join T2 on boolean_expression
- T1 { [INNER] | { LEFT | RIGHT | FULL } [OUTER] } join T2 using liste_nom_colonne
 - ↳ USING a équivalent à on t1.a = t2.a
 - ↳ USING (a,b) équivalent à on t1.a=t2.a and t1.b=t2.b
- T1 NATURAL { [INNER] | { LEFT | RIGHT | FULL } [OUTER] } join T2

NATURAL effectue une comparaison de toutes les colonnes de même nom dans les deux tables.

31

Expression d'une union

Définition : l'union de 2 relations R et S de même schéma est une relation T de même schéma contenant l'ensemble des tuples appartenant à R ou S

Syntaxe :

- < requête > UNION < requête >
- < requête > UNION ALL < requête >

Le résultats des deux requêtes doivent avoir **la même structure**

- même nombre de colonnes, mêmes types de données, même ordre.

Exemple :

```
select nom from auteur union select nom from editeur ;
```

| nom |
|---------------|
| Albert-René |
| Albert Uderzo |
| Folio |
| Gallimard |
| J.K. Rowling |
| Victor Hugo |

32

Expression d'une intersection

Définition : L'intersection de deux relations R et S de même schéma est une relation T de même schéma contenant les tuples appartenant à la fois à R et S

Syntaxe :

```
< requête > INTERSECT < requête >
```

| nom |
|-----|
| |

Exemple :

```
select nom from auteur  
intersect select nom from editeur ;
```

- est équivalent à :

```
select nom from auteur a where a.nom IN select nom from editeur ;  
select nom from auteur a where EXISTS select nom from editeur e where a.nom =  
e.nom;
```

- Le résultats des deux requêtes doivent avoir **la même structure**

- Note : Attention à la structure du **IN** et du **EXISTS**

33

Expression d'une différence

Définition : la différence de 2 relations R et S de même schéma est une relation T de même schéma contenant l'ensemble des tuples appartenant à R et n'appartenant pas à S

Syntaxe :

```
< requête > EXCEPT < requête >
```

Exemple :

```
select nom from auteur  
except select nom from editeur ;
```

- est équivalent à :

```
select nom from auteur a where a.nom NOT IN select nom from editeur  
select nom from auteur a where NOT EXISTS select nom from editeur e where a.nom =  
e.nom
```

Note : Attention à la structure du **NOT IN** et du **NOT EXISTS**

34

| nom |
|---------------|
| Albert Uderzo |
| J.K. Rowling |
| Victor Hugo |