

Bases de données SQL

Expressions

Retour sur le «modèle» SQL et synthèse

SQL_03d

v320e

2025-02-11

Christina.Khnaisser@USherbrooke.ca

Luc.Lavoie@USherbrooke.ca

© 2018-2021, Myrius (<http://info.usherbrooke.ca/lavoie>)

CC BY-NC-SA 4.0 (<https://creativecommons.org/licenses/by-nc-sa/4.0/>)

Plan

- **Retour sur le « modèle » SQL**
- **Expression relationnelle ou SELECT ?**
- **Références**

Retour sur le « modèle » SQL

- La complexité apparente des expressions relationnelles
- La multiplicité des jointures
- L'annulabilité
- Les doublons (ALL et DISTINCT)
- L'ordre des attributs
- Le nom des attributs
- Le moment de la vérification d'une contrainte

Retour sur le « modèle » SQL

La complexité apparente des expressions relationnelles (début...)

Requête ::=

[*Contexte*]
SELECT *OpMode* { * | *Projection-extension* }
FROM *ListeDeJointures*
 [*Restriction*]
 [**Groupement**]
 [*OpComplémentaire*]
 [**Ordonnancement**]
 [**Divers**]

OpComplémentaire ::=

INTERSECT *OpMode* *Requête*
 | **UNION** *OpMode* *Requête*
 | **EXCEPT** *OpMode* *Requête*
 | *Jointure*

ListeDeJointures ::=

DenotationTable [[**AS**] *alias*] [*Jointure ...*]

DenotationTable ::=

nomTable | (*Requête*) | *autresDénotations*

OpMode ::=

[**DISTINCT** | **ALL**]

Retour sur le « modèle » SQL**La complexité apparente des expressions relationnelles (... presque la fin)***Jointure ::=**Jointure_naturelle* | *Produit* | *Jointure_qualifiée* | *Jointure_externe**Jointure_naturelle ::=***NATURAL** [**INNER**] **JOIN** *DenotationTable* [[**AS**] *alias*]*Produit ::=**OpProduit* *DenotationTable* [[**AS**] *alias*]*OpProduit ::=***CROSS JOIN** | ,*Jointure_qualifiée ::=*[**INNER**] **JOIN** *DenotationTable* [[**AS**] *alias*] *QualificationJ**QualificationJ ::=***ON** *conditionJ*| **USING** (*listeNomCol*)*Jointure_externe ::=**OpJointExterne* **JOIN** *DenotationTable* [[**AS**] *alias*] *qualificationJ**OpJointExterne ::=***LEFT** [**OUTER**] | **RIGHT** [**OUTER**] | **FULL** [**OUTER**]

Le langage SQL

La multiplicité des jointures

- Jointure naturelle
 - Égalité de tous les attributs de mêmes noms.
 - Risqué relativement à l'évolution des tables.
- Le CROSS JOIN
 - C'est le produit cartésien!
 - Source d'explosion quadratique.
- Jointures internes
 - USING : explicitation des attributs de jointures (sans renommage)
 - ON (equi-join) : explicitation des attributs de jointures (avec renommage)
 - ON (θ -join) : combinaison d'une jointure et d'une restriction (inutile)
- Jointures externes
 - Utilisation des NULL pour ne pas « perdre » la perte d'information!
 - Discuté et discutable.
- Les mots INNER et OUTER
 - Superfétatoires!

Retour sur le « modèle » SQL

L'annulabilité

- Que se passe-t-il quand la condition d'une contrainte (CHECK) est UNKNOWN ?
 - Elle est réputée **satisfaite!**
- Que se passe-t-il quand la condition d'une restriction (WHERE) est UNKNOWN ?
 - Elle est réputée **non** satisfaite!!!

On ne peut donc pas s'assurer du respect d'une contrainte avec une clause where équivalente,

il faut s'assurer de traiter adéquatement les cas unknown explicitement (et différemment).

Retour sur le « modèle » SQL

Les doublons

- Quand sont-ils insérés ou préservés ?
 - *Implicitement*, par une projection lors d'un SELECT (par élimination d'un des attributs permettant de discriminer uniquement les tuples issus de la jointure FROM)
 - *Explicitement*, lors d'une opération ensembliste (union, intersection, différence)
- Comment peut-on les enlever ?
 - En spécifiant (*généralement*) DISTINCT après SELECT, UNION, INTERSECT et EXCEPT
 - En ne spécifiant pas (*jamais*) ALL après SELECT, UNION, INTERSECT et EXCEPT

Retour sur le « modèle » SQL

L'ordre des attributs

- Comment est-il déterminé ?
 - Par l'ordre des champs du CREATE TABLE.
 - Par l'ordre des expressions projection-extension du SELECT.
 - Par l'ordre des opérandes de jointure (FROM) (la jointure n'est donc plus commutative).
 - Selon des règles propres aux attributs de jointure (règles à voir ultérieurement).
- Quand importe-t-il ?
 - Dans les INSERT.
 - Dans toutes les instructions portant la mention CORRESPONDING (comme l'union).

Retour sur le « modèle » SQL

L'ordre des attributs, les solutions

○ INSERT :

- Toujours indiquer explicitement la liste des attributs.
- En général, ne pas utiliser la clause DEFAULT (pour cela et pour d'autres raisons).

○ CORRESPONDING :

- Ne jamais utiliser cette mention (qui n'est d'ailleurs pas présentée dans ce cours).

Retour sur le « modèle » SQL

Les noms d'attribut

- Comment peuvent-ils **ne pas** être présents ?
 - Expression non atomique dans la projection-extension du SELECT
 - Redondance dans les noms des opérandes des jointures
- Que se passe-t-il quand ils ne sont pas présents?
 - **On ne sait pas !!!**
- Qu'en est-il de la capitalisation?
 - Variable selon les SGBD

Retour sur le « modèle » SQL

Les noms d'attribut, les solutions (1/2)

○ Présence :

- Toujours nommer explicitement les attributs anonymes (*AS alias*).
- Toujours renommer explicitement les noms d'attributs homonymes (*AS alias*)

○ Capitalisation et autres caractères « spéciaux » (y compris les accents) :

- Soit, toujours utiliser les guillemets
- Soit, ne jamais utiliser les guillemets
et s'en tenir à [A-Za-z]([A-Za-z0-9_]{29})

Pourquoi 29 ? Parce que la longueur maximale est 30. Pourquoi 30 ?

Retour sur le « modèle » SQL

Les noms d'attribut, les solutions (2/2)

dialecte	longueur	start	extend
DB2	30	a, @, #, \$	a, n, _, \$, #, @
MS-SQL	116	a, _, @, #	a, n, _, \$, #, @
Oracle	30	a	a, n, _, \$, #
PostgreSQL	31	a, _, \$	a, n, _, \$
MySQL	64	a, n?, _, \$	a, n, _, \$
MariaDB	64	a, n?, _, \$	a, n, _, \$

- Notes :
 - Le \$ est souvent réservé aux seules tables du catalogue.
 - En MySQL et MariaDB, les chiffres en position initiale ne sont permis que s'ils ne sont pas ambigus avec la dénotation d'un nombre flottant.
- Conclusion : `[A-Za-z]([A-Za-z0-9_]{29})`

- *DB2 : aussi peu que 8, selon le SE et la nature de la table!!!

Retour sur la notion d'intégrité

Le moment de la vérification d'une contrainte

- Quand une contrainte est-elle vérifiée ?
 - À la fin de l'instruction ou à la fin de la transaction, selon le « mode de contrainte »
- Qu'est-ce qu'une transaction ?
 - Une suite d'instructions indivisible terminée par un COMMIT (plus de détails ultérieurement)
- Comment modifier le mode ?
 - SET CONSTRAINTS [MODE] IMMEDIATE -- fin d'instruction
 - SET CONSTRAINTS [MODE] DEFERRED -- fin de transaction

En guise de bilan du SELECT



Le langage SQL

SELECT – regrouper les opérateurs et leur imposer un ordre

- A posteriori, cela apparaît comme une fausse bonne idée en regard de la souplesse et de la clarté des expressions de l'algèbre relationnelle.
- De plus, l'optimisation des requêtes nécessite de les « reconvertir » en expressions relationnelles de toute façon!
- Pour en savoir plus, voir IGE487.

Références

- Elmasri et Navathe (4^e ed.), chapitre 7
- Elmasri et Navathe (6^e ed.), chapitre 4
- [Loney2008]
Loney, Kevin ;
Oracle Database 11g: The Complete Reference.
Oracle Press/McGraw-Hill/Osborne, 2008.
ISBN 978-0071598750.
- [Date2012]
Date, Chris J. ;
SQL and Relational Theory : How to Write Accurate SQL Code.
2nd edition, O'Reilly, 2012.
ISBN 978-1-449-31640-2.
- Le site d'Oracle (en anglais)
 - http://docs.oracle.com/cd/E11882_01/index.htm
- Le site de PostgreSQL (en français)
 - <http://docs.postgresqlfr.org>

