Bases de données
SQL

Opérateurs pour les nuls

SQL_08
v410a
2025-02-18

Christina.Khnaisser@USherbrooke.ca
Luc.Lavoie@USherbrooke.ca

Plan

- NULL : marqueur ou valeur ?
- Impacts sur la logique
- Impacts sur les opérateurs relationnels
- Impacts sur les clés
- Opérateurs spéciaux
- Jointures externes

NULL : marqueur ou valeur ?

- NULL considéré comme marqueur
- NULL considéré comme valeur

3

NULL considéré comme un marqueur d'attribut (1/3)

- NULL marque l'absence de valeur, ainsi
  - INSERT INTO R (*a*, *b*) VALUES (12, NULL)

  a pour effet
  - d'affecter la valeur 12 à l'attribut *a* ;
  - de marquer l'attribut *b* comme étant NULL (sans valeur associée) ;
  - si *b* n'est pas annulable, c'est une erreur.

NULL considéré comme un marqueur d'attribut (2/3)

- Soit x un attribut quelconque, on **ne peut pas** évaluer
  - $x = \text{NULL}$

  on doit évaluer le statut du marqueur ainsi
  - x IS NULL

NULL considéré comme un marqueur d'attribut (3/3)

- Soit a et b deux attributs dont au moins un est marqué NULL,
  - $a = b$ est inconnu
  - $a <> b$ est inconnu aussi
- En particulier, si a est marqué NULL
  - $a = a$ est inconnu
  - $a <> a$ est inconnu aussi

NULL considéré comme une valeur (1/2)

- En général, toute évaluation d'expression nécessitant l'évaluation d'un attribut marqué NULL entraine « l'annulabilité de l'expression ».
- Or comment une expression pourrait-elle être « nulle » si NULL est un marqueur d'attribut (*seulement*) ?
- Conséquemment, **il y aura aussi** des valeurs nulles afin de permettre l'évaluation des expressions.
- Ceci s'applique donc aux expressions logiques aussi.

7

Nous verrons plus tard que les nuls appellent aussi un traitement particulier lors de l'agrégation.

NULL considéré comme une valeur (2/2)

- Ainsi, la *valeur* d'un prédicat dont un des termes a pour valeur NULL est inconnue (UNKNOWN).
- La logique de SQL est donc trivaluée :
  FALSE, TRUE et UNKNOWN
  et les opérateurs logiques usuels doivent donc été « étendus » en conséquence, mais de façon différente de celle utilisée pour les valeurs non logiques (un terme UNKNOWN ne forcera l'expression à être nécessairement UNKNOWN également).
- Finalement, l'évaluation des opérateurs IS NUL, IS NOT NULL, NULLIF et COALESCE relèveront de règles différentes encore.

8

Il y a donc deux exceptions à l'exception, de l'exception de la règle...
Nous verrons plus tard que d'autres règles président à l'évaluation des fonctions l'agrégation...

Est-ce que NULL et UNKNOWN sont identifiques ? NON!
Sont-ils traités de façons différentes ? Parfois.
Sont-ils traités de façons équivalentes ? Parfois.
Ce traitement est-il identique d'un dialecte à un autre ? NON.

Impacts sur la logique

- Opérateurs (« tables de vérité »)
- Assertions et contraintes

9

Logique non classique utilisée par SQL

OR	true	false	unknown
true	true	true	true
false	true	false	unknown
unknown	true	unknown	unknown

AND	true	false	unknown
true	true	false	unknown
false	false	false	false
unknown	unknown	false	unknown

P	NOT P
true	fase
false	true
unknown	unknown

IS	true	false	unknown
true	true	false	false
false	false	true	false
unknown	false	false	true

Assertions et contraintes
L'incohérence du système logique de SQL

- 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 !!!
- Le système logique utilisé par SQL est donc incohérent!

11

On ne peut donc s'assurer simplement du respect d'une contrainte avec une clause WHERE utilisant la même condition,
il faut s'assurer de traiter adéquatement les cas UNKNOWN explicitement.

Conséquences

- L'égalité est également incohérente en présence de NULL et UNKNOWN.
- Les opérateurs l'utilisant implicitement (dont l'union, l'intersection, la différence, la restriction et la jointure) peuvent avoir des comportements incohérents.
- Conséquemment les instructions DELETE, UPDATE et SELECT peuvent également produire des résultats incohérents comme nous le verrons dans les prochains modules.

12

Impacts sur opérateurs relationnels

- Sauf la projection et le renommage, tous les opérateurs sont touchés et « étendus » :
  - par l'extension de la logique (UNKNOWN -> FALSE)
    - WHERE
  - par l'extension de l'égalité (UNKNOWN -> FALSE)
    - JOIN,
    - UNION
    - INTERSECT
    - EXCEPT

13

Impacts sur les clés

- Clés candidates
  - primaires
  - secondaires
- Clés référentielles

Impact sur les clés candidates primaires
PRIMARY KEY

- Aucun impact, les attributs d'une clé candidate primaire ne peuvent être annulables
  - SQL impose d'ailleurs automatiquement la contrainte **NOT NULL** aux attributs participant à une clé candidate primaire.

15

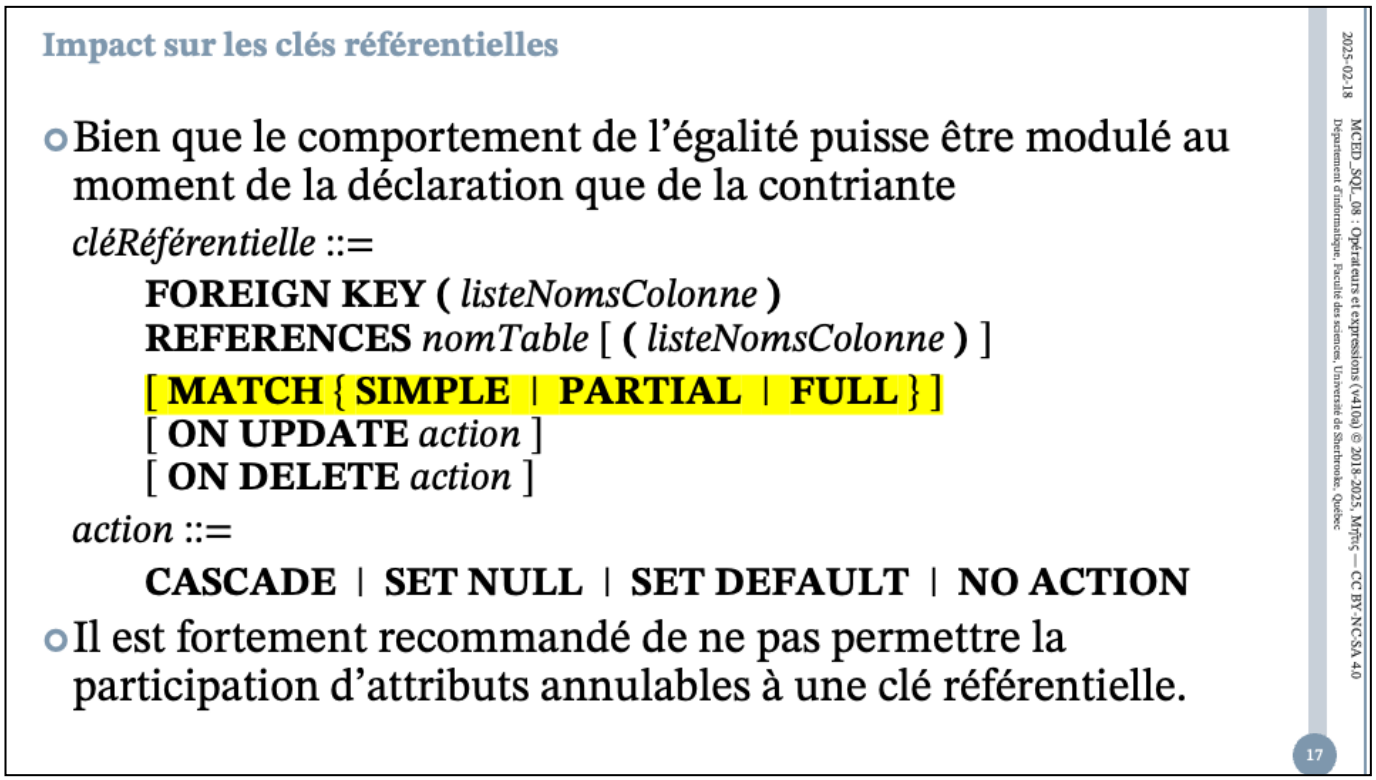
Impact sur les clés candidates secondaires
UNIQUE

- Bien que le comportement de l'égalité puisse être modulé au moment de la déclaration que de la contrainte

  cléSecondaire ::=
  UNIQUE [NULLS [NOT] DISTINCT] (*listeNomsColonne* **)**

- Il est fortement recommandé de ne pas permettre la participation d'attributs annulables à une clé candidate, fut-elle secondaire.

16



Extraits de la norme ISO 9075:2011

The checking of a constraint depends on its constraint mode within the current SQL-transaction. If the constraint mode is immediate, then the constraint is effectively checked at the end of each SQL-statement.

NOTE 29 — This includes SQL-statements that are executed as a direct result or an indirect result of executing a different SQL-statement.

If the constraint mode is deferred, then the constraint is effectively checked when the constraint mode is changed to immediate either explicitly by execution of a <set constraints mode statement>, or implicitly at the end of the current SQL-transaction.

A referential constraint is satisfied if one of the following conditions is true, depending on the <match type> specified in the <referential constraint definition>:

— If no <match type> was specified then, for each row R1 of the referencing table, either at least one of the values of the referencing columns in R1 shall be a null value, or the value of each referencing column in R1 shall be equal to the value of the corresponding referenced column in some row of the referenced table.

— If MATCH FULL was specified then, for each row R1 of the referencing table, either the value of every referencing column in R1 shall be a null value, or the value of every referencing column in R1 shall not be null and there shall be some row R2 of the referenced table such that the value of each referencing column in R1 is equal to the value of the corresponding referenced column in R2.

— If MATCH PARTIAL was specified then, for each row R1 of the referencing table, there shall be some row R2 of the referenced table such that the value of each referencing column in R1 is either null or is equal to the value of the corresponding referenced column in R2.

NOTE 30 — If MATCH FULL or MATCH PARTIAL is specified for a referential constraint and if the referencing table has only one column specified in <referential constraint definition> for that referential constraint, or if the referencing table has more than one specified column for that <referential constraint definition>, but none of those columns is nullable, then the effect is the same as if no <match type> were specified.

Opérateurs spéciaux

- COALESCE
- NULLIF
- Jointures externes

Formes abrégées du CASE, l'opérateur COALESCE

- COALESCE (*V*1, *V*2) est équivalent à :
  - ```
    CASE
    WHEN V1 IS NOT NULL THEN V1
    WHEN V2 IS NOT NULL THEN V2
    ELSE NULL
    END
    ```
- COALESCE (*V*1, *V*2, ..., *Vn*), pour $n \geq 3$, est équivalent à :
  - ```
    CASE
    WHEN V1 IS NOT NULL THEN V1
    ELSE COALESCE (V2, ..., Vn)
    END
    ```


19

Opérateur COALESCE

Autrement dit...

- COALESCE (x_1, x_2, ..., x_n)
  - première(*) expression non nulle parmi x_1, x_2, ..., x_n ;
  - si toutes les expressions sont nulles, NULL.

  - (*) de gauche à droite
  - (*) dont l'indice est le plus petit

20

COALESCE : http://www.postgresql.org/docs/9.6./static/functions-conditional.html

Formes abrégées du CASE - NULLIF

- NULLIF (*V*1, *V*2) est équivalent à :
  - ```
    CASE
    WHEN V1=V2 THEN NULL
    ELSE V1
    END
    ```
- Mais de quelle égalité s'agit-il ?
  - La clause `WHEN` considère qu'un résultat `UNKNOWN` est **faux**
    ```

Opérateur NULLIF
Autrement dit...

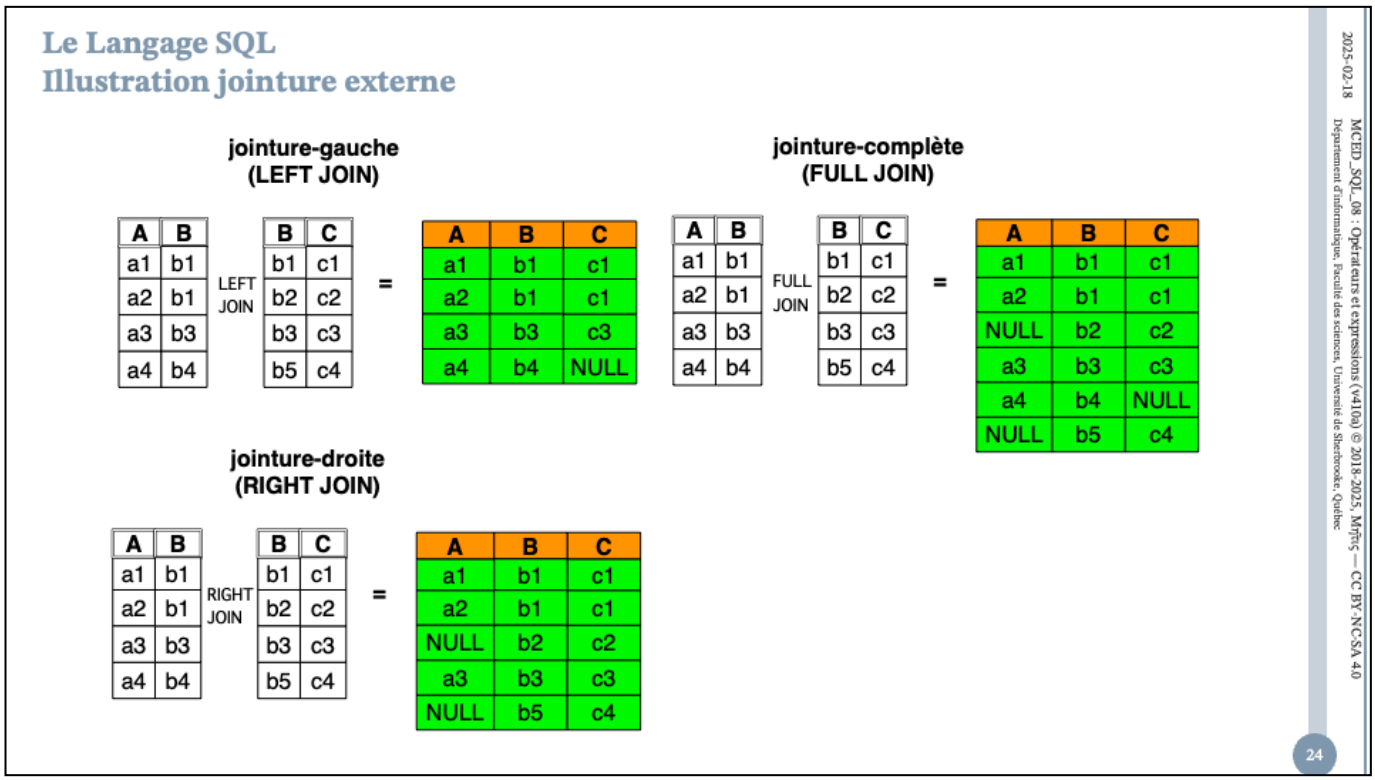
- NULLIF (x_1, x_2)
  - si x_1 est nul alors NULL
  - sinon si x_2 est NULL alors x_1
  - sinon si $x_1 = x_2$ alors NULL
  - sinon x_1

COALESCE : https://www.postgresql.org/docs/current/functions-conditional.html

Jointures externes

jointure_externe ::=
  jExterne JOIN *denotationTable* [[AS] *alias*] *qualificationJ*
jExterne ::=
  LEFT [OUTER] | RIGHT [OUTER] | FULL [OUTER]

- Une jointure externe permet de ne pas « perdre » les données des tuples sans correspondant en affectant des NULL aux attributs de valeur inconnue.
- Le mot OUTER est superfétatoire (sic).
- Le mot AS permet de (re)nommer la *dénotationTable* en même temps.




```
CREATE TABLE R (A CHAR(2) PRIMARY KEY, B CHAR(2));
CREATE TABLE S (B CHAR(2) PRIMARY KEY, C CHAR(2));
INSERT INTO R VALUES
  ('a1', 'b1'),
  ('a2', 'b1'),
  ('a3', 'b3'),
  ('a4', 'b4');
INSERT INTO S VALUES
  ('b1', 'c1'),
  ('b2', 'c2'),
  ('b3', 'c3'),
  ('b5', 'c4');
SELECT * FROM R LEFT JOIN S USING(B);
SELECT * FROM R RIGHT JOIN S USING(B);
SELECT * FROM R FULL JOIN S USING(B);
```

Le langage SQL
Jointures externes simplifiées

```
SELECT C.nom, T.tel
FROM
  C RIGHT JOIN T ON C.id = T.id
```

est équivalent à :

```
SELECT C.nom, T.tel
FROM
  T LEFT JOIN C ON C.id = T.id
```

```
SELECT C.nom, T.tel
FROM
  C FULL JOIN T ON C.id = T.id
```

est équivalent à :

```
SELECT C.nom, T.tel
FROM
  C RIGHT JOIN T ON C.id = T.id
UNION
SELECT C.nom, T.tel
FROM
  C LEFT JOIN T  ON C.id = T.id
```

Références

- Elmasri et Navathe (4e ed.), chapitre 7
- Elmasri et Navathe (6e ed.), chapitre 4
- [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.
- La documentation de PostgreSQL (en français)
  - https://docs.postgresql.fr
- La documentation de MariaDB (en anglais)
  - https://mariadb.com/kb/en/library/documentation/
- La documentation d'Oracle (en anglais)
  - http://docs.oracle.com/cd/E11882_01/index.htm