Bases de données
SQL

Opérateurs élémentaires

SQL_04
v400d
2020-09-12

Département d'informatique
Faculté des sciences

UNIVERSITÉ DE SHERBROOKE

Christina.Khnaisser@USherbrooke.ca
http://info.USherbrooke.ca/ckhnaisser
Luc.Lavoie@USherbrooke.ca
http://info.USherbrooke.ca/llavoie

PLAN

- Expressions, conditions et contraintes simples
- Opérateurs prédéfinis élémentaires
  - Nombres
  - Textes
- *Les nuls de nuls pour les nuls par les nuls! [sic]*
- Exercice
- Les colles du prof

CONTRAINTES SIMPLES

- CHECK
- condition (et expression logique)
- expression

2020-09-12 BD103-SQLLDD 03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

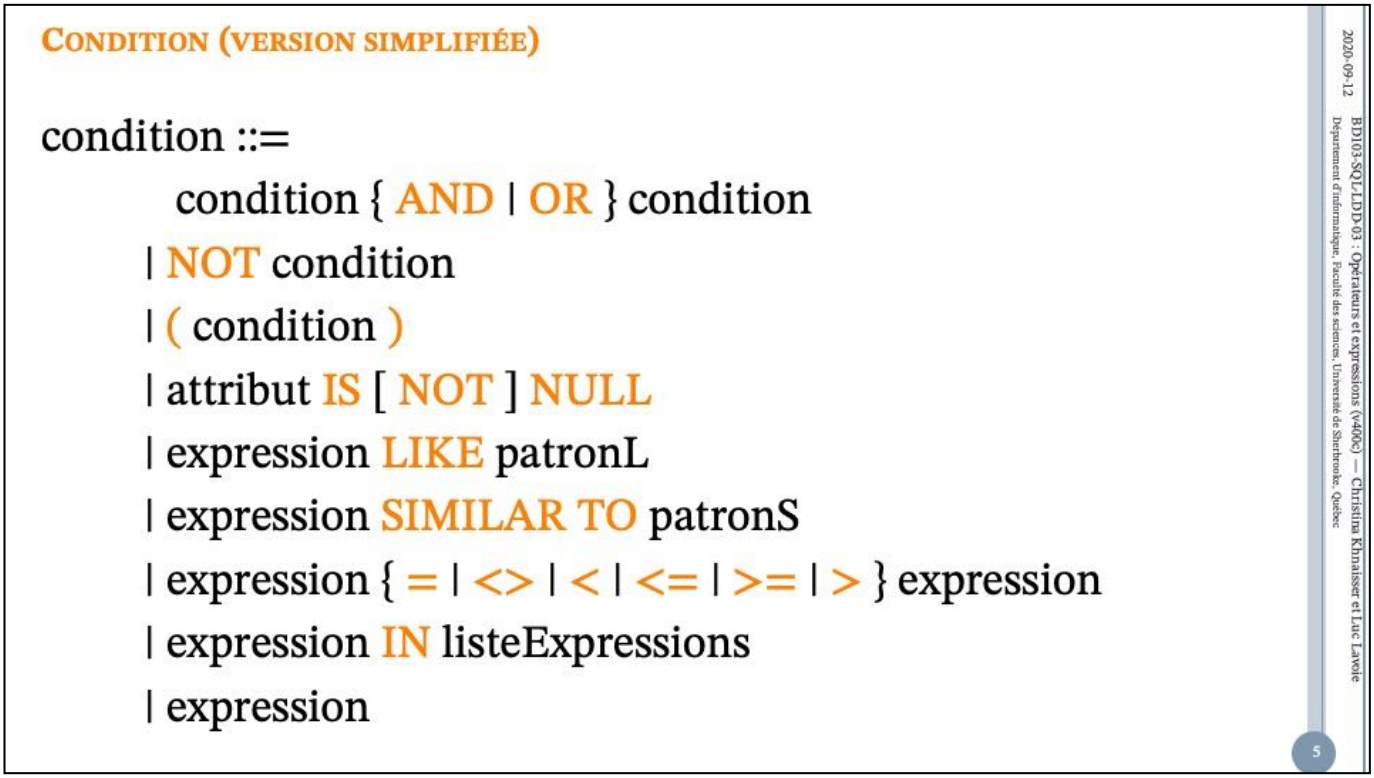
3

CONTRAINTES SIMPLES

- Contraintes simples :
  - Clés candidates -- BD101-SQL-LDD-01,
  - **CHECK (condition)**
- Qu'est donc une condition ?
  - Tout simplement une expression dont l'évaluation résulte en une valeur booléenne.
- Qu'est donc une expression ?
  - La dénotation d'une suite d'opérations portant sur des variables dont l'évaluation résulte en une valeur.
- Dans une contrainte simple (aussi appelée contrainte de table), les variables admises sont les attributs de la table au sein de laquelle elle est définie.

2020-09-12 BD103-SQL-LDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

4




```
CREATE TABLE Activite (
  sigle         CHAR(6) NOT NULL,
  titre         VARCHAR(46) NOT NULL,
  CONSTRAINT Activite_cc0 PRIMARY KEY (sigle),
  CONSTRAINT Activite_sigle0 CHECK
      ( sigle SIMILAR TO '[A-Z]{3}[0-9]{3}' )
  CONSTRAINT Activite_sigle1 CHECK
      ( SUBSTR(sigle,1,3) IN ('IFT', 'IMN', 'IGL', 'GMQ', 'MAT') )
  );
```

Remarques

- les deux contraintes peuvent être combinées en une seule ;
- la contrainte Activite_sigle1 est une fausse bonne idée, il est préférable de scinder le sigle en deux parties (la discipline et le numéro de cours) et de recenser les disciplines indépendamment dans une table propre. Pouvez-vous dire pourquoi ?

LE LANGAGE SQL
EXPRESSION (VERSION SIMPLIFIÉE)

```
expression ::=
        expression  opBinaire expression
      | opUnaire expression
      | nomFonction listeExpressions
      | ( expression )
      | attribut
      | constante
opBinaire ::=
      + | - | * | / | ||        -- et quelques autres
opUnaire ::=
      + | -                     -- et quelques autres
listeExpressions ::=
      ( { expression ... , } )
```

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

6

LE LANGAGE SQL
FONCTION

- Une fonction est représentée par un identificateur (de fonction) suivi de ses paramètres.
- Deux catégories de fonctions
  - prédéfinies
    - *par la norme SQL*
      - les principales (mais pas la plupart) sont présentées ci-après
    - *par le SGBD*
      - se reporter à la documentation de l'éditeur du SGBD
      - faire attention à la non-transportabilité
  - définies par l'utilisateur
    - *autonomes*
    - *associées* au mécanisme de définition de type

2020-09-12 BD103-SQLLDD 03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

7

OPÉRATEURS ET FONCTIONS

- Une différence de syntaxe, une identité de rôle.
- Opérateurs et fonctions : même combat !

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

8

FONCTIONS ET OPÉRATEURS PRÉDÉFINIS

- Fonctions et opérateurs logiques
- Fonctions et opérateurs numériques
- Fonctions et opérateurs textuels
- Fonctions et opérateurs temporels
- Fonctions opérateurs des langages rationnels
  - reconnaissance (similar)
  - extraction (substr)
  - forme historique simplifiée (like)

2020-09-12 BD103-SQLLDD 03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

9

FONCTIONS ET OPÉRATEURS LOGIQUES

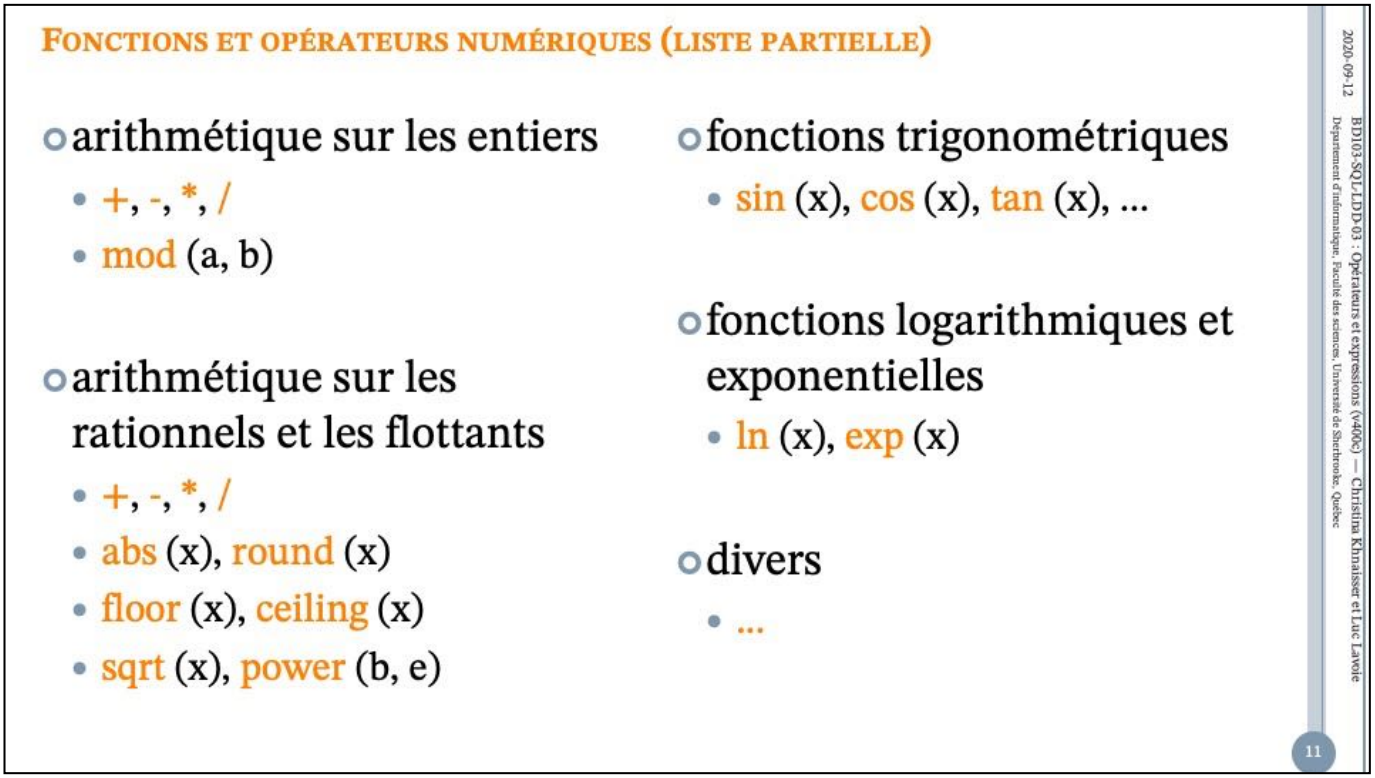
OR	true	unknown	false
true	true	true	true
unknown	true	unknown	unknown
false	true	unknown	false

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

P	true	unknown	false
NOT P	false	unknown	true
P IS TRUE	true	false	false
P IS UNKNOWN	false	true	false
P IS FALSE	false	false	true

2020-09-12 BD103-SQLLDD 03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

10



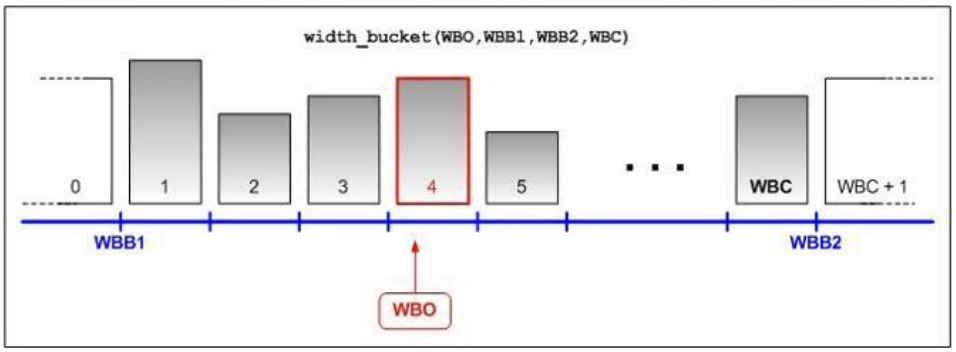
Il existe une bonne centaine de fonctions et d'opérateurs supplémentaires.

Pour la différence entre mod et rem, voir https://fr.wikipedia.org/wiki/Modulo_(opération)
Vérifier ce que dit la norme SQL.

Sauriez-vous comment obtenir le log10 ?

OPÉRATEURS ET FONCTIONS
UN EXEMPLE D'OPÉRATEUR « DIVERS » PARMI D'AUTRES

WIDTH_BUCKET(<value>, <min_value>, <max_value>, <number_of_buckets>)



ISO/IEC 9075-2:2003 (E) p. 29

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

Création d'un histogramme, voir
http://docs.oracle.com/cd/B19306_01/server.102/b14200/functions214.htm

```
SELECT customer_id, cust_last_name, credit_limit,
  WIDTH_BUCKET(credit_limit, 100, 5000, 10) "Credit Group"
  FROM customers WHERE nls_territory = 'SWITZERLAND'
  ORDER BY "Credit Group";

CUSTOMER_ID CUST_LAST_NAME      CREDIT_LIMIT Credit Group
----------- -------------------- ------------ ------------
        825 Dreyfuss                  500            1
        826 Barkin                    500            1
        853 Palin                     400            1
        827 Siegel                    500            1
        843 Oates                     700            2
        844 Julius                    700            2
        835 Eastwood                 1200            3
        840 Elliott                  1400            3
        842 Stern                    1400            3
        841 Boyer                    1400            3
        837 Stanton                  1200            3
        836 Berenger                 1200            3
        848 Olmos                    1800            4
        849 Kaurusmdki               1800            4
```

828	Minnelli	2300	5
829	Hunter	2300	5
852	Tanner	2300	5
851	Brown	2300	5
850	Finney	2300	5
830	Dutt	3500	7
831	Bel Geddes	3500	7
832	Spacek	3500	7
838	Nicholson	3500	7
839	Johnson	3500	7
833	Moranis	3500	7
834	Idle	3500	7
845	Fawcett	5000	11
846	Brando	5000	11
847	Streep	5000	11

https://www.postgresql.org/docs/11/functions-string.html

FONCTIONS ET OPÉRATEURS TEMPORELS PRÉDÉFINIS (LISTE PARTIELLE)

- current_timestamp
- current_date
- current_time

- age(ts0, ts1) -- différence entre ts0 et ts1
- age(ts) -- équivalent à age(current_timestamp, ts)

- extract(<field> from timestamp)
  - **<field> := YEAR | MONTH | DAY | HOUR | MINUTE | SECOND**

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

14

FONCTIONS OPÉRATEURS DES LANGAGES RATIONNELS

- Introduction
- Syntaxe
- Conformité (similar)
- Extraction (substr)
- Forme historique simplifiée (like)
- Différences entre SQL et PostgreSQL
- Différences entre SQL et POSIX
- Exemples
- Exercices

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

15

FONCTIONS OPÉRATEURS DES LANGAGES RATIONNELS

- Les langages rationnels (LR) permettent de décrire et d'analyser les suites de symboles à l'aide d'un nombre réduits d'opérateurs.
- Les LR sont équivalents aux automates d'états finis.
- Leur étude sera approfondie au cours des activités
  - MAT115,
  - IFT313.
- Nous nous contenterons ici d'en décrire l'utilisation au sein du langage SQL.

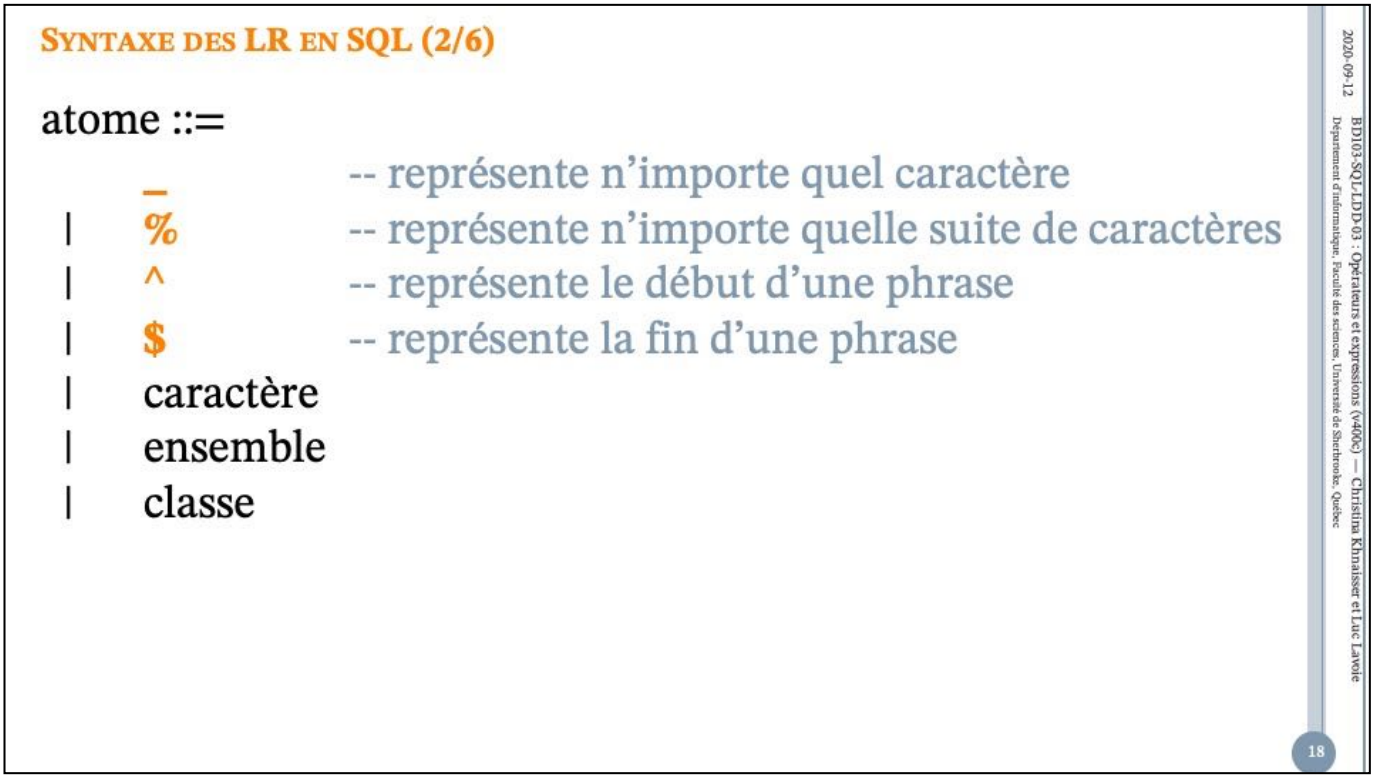
Note
Les langages rationnels sont aussi parfois appelés langages réguliers, voire expressions régulières (par calque de l'anglais).

SYNTAXE DES LR EN SQL (1/6)

```
expR ::=
        atome
  |     expR fermeture        -- la fermeture de l'expression expR
  |     expR1 expR2           -- expR1 suivi de expR2
  |     expR1 | expR2         -- expR1 ou expR2
  |     ( expR )              -- l'expression expR elle-même
```

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

17



'_' : représente n'importe quel caractère (alors que c'est le point dans le standard POSIX)
'%' : représente n'importe quelle suite de caractères (alors que c'est, naturellement, .* dans le standard POSIX).

SYNTAXE DES LR EN SQL (3/6)

fermeture ::=

?	-- 0 ou 1 fois
| +	-- 1 ou plusieurs fois
| *	-- 0, 1 ou plusieurs fois
| { entier }	-- exactement entier fois
| { entier , }	-- au moins entier fois
| { $entier_1$, $entier_2$ }	-- entre $entier_1$ et $entier_2$ fois

2020-09-12 BD103-SQLLDD 03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

19

SYNTAXE DES LR EN SQL (4/6)

caractère ::=

caractère-simple | caractère-codé

caractère-simple ::=

« tout caractère sauf ceux-ci _%^$[+*{\|() »

caractère-codé ::=

\ code

hex ::=

« une suite de chiffres hexadécimaux »

entier ::=

« une suite de chiffres décimaux »

2020-09-12 BD103-SQLLDD 03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

20

SYNTAXE DES LR EN SQL (5/6)

```
code ::=
        t           -- la commande HT (tabulation horizontale)
    |   r           -- la commande CR (retour à la ligne)
    |   n           -- la commande LF (ligne suivante)
    |   v           -- la commande VT (tabulation verticale)
    |   f           -- la commande FF (page suivante)
    |   x hex       -- le car. Unicode de point de code hex
    |   entier      -- le car. Unicode de point de code entier
    |   « tout autre caractère sauf le : » -- ce caractère lui-même
```

2020-09-12 BD103-SQLLDD 03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

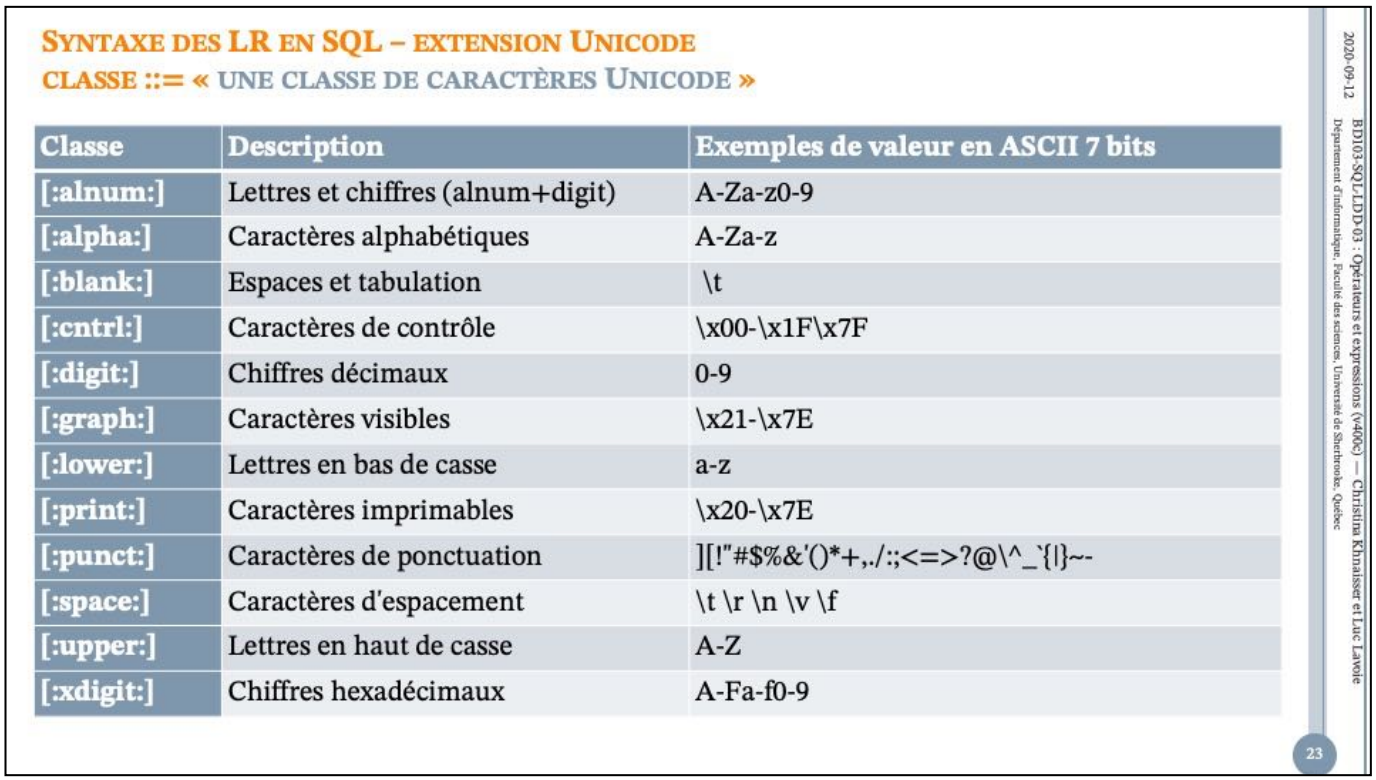
21

SYNTAXE DES LR EN SQL (6/6)

ensemble ::=
    ensemble-simple | ensemble-complémentaire
ensemble-simple ::=
    [suite-sans-crochet]
ensemble-complémentaire ::=
    [^ suite-sans-crochet]
suite-sans-crochet ::=
    « toute suite de caractères sans crochet],
    mais pouvant comprendre des intervalles
    représentés par deux caractères réunis
    par un tiret »

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

22



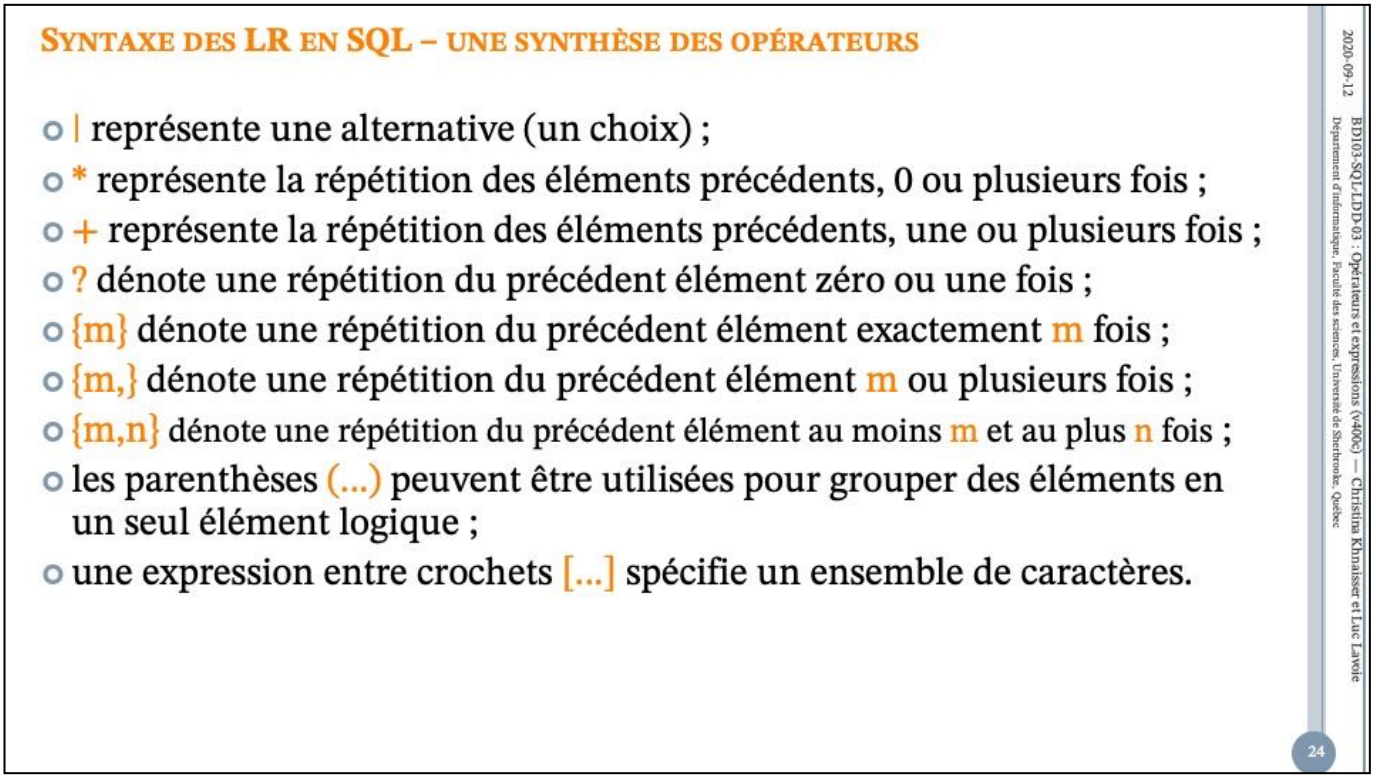
SYNTAXE DES LR EN SQL – EXTENSION UNICODE

CLASSE ::= « UNE CLASSE DE CARACTÈRES UNICODE »

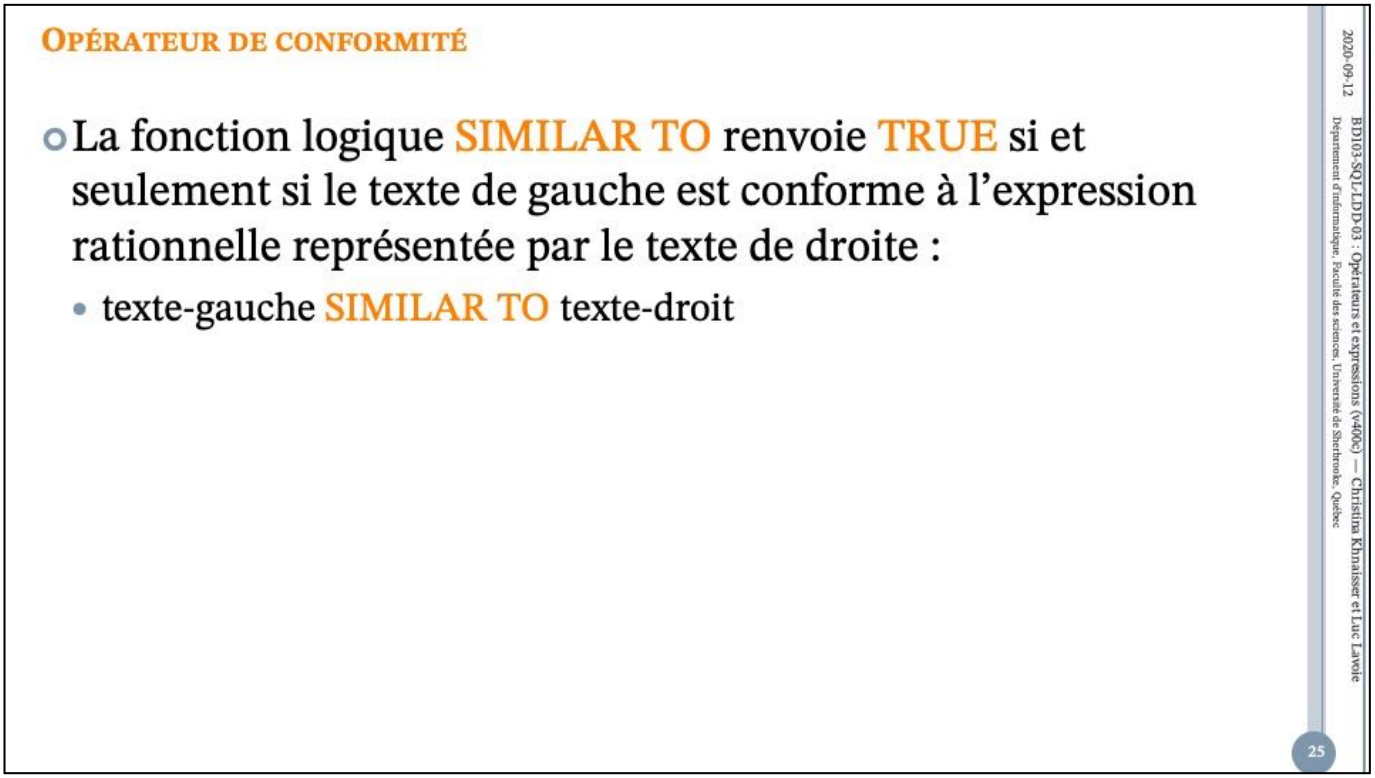
Classe	Description	Exemples de valeur en ASCII 7 bits
[:alnum:]	Lettres et chiffres (alnum+digit)	A-Za-z0-9
[:alpha:]	Caractères alphabétiques	A-Za-z
[:blank:]	Espaces et tabulation	\t
[:cntrl:]	Caractères de contrôle	\x00-\x1F\x7F
[:digit:]	Chiffres décimaux	0-9
[:graph:]	Caractères visibles	\x21-\x7E
[:lower:]	Lettres en bas de casse	a-z
[:print:]	Caractères imprimables	\x20-\x7E
[:punct:]	Caractères de ponctuation	][!"#$%&'()*+,./:;<=>?@\^_`{|}~-
[:space:]	Caractères d'espacement	\t \r \n \v \f
[:upper:]	Lettres en haut de casse	A-Z
[:xdigit:]	Chiffres hexadécimaux	A-Fa-f0-9

2020-09-12 BD103-SQLLDD-03 - Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec
23

À l'intérieur d'une expression entre crochets, le nom d'une classe de caractères entouré entre [: et :] signifie la liste de tous les caractères appartenant à cette classe. Une classe de caractères ne peut pas être utilisée comme point final d'un intervalle. Le standard POSIX définit les noms de ces classes de caractères : alnum (lettres et chiffres), alpha (lettres), blank (espace et tabulation), cntrl (caractères de contrôle), digit (chiffres), graph (caractères affichages, sauf l'espace), lower (lettres minuscules), print (caractères affichages, incluant l'espace), punct (ponctuation), space (tout espace blanc), upper (lettres majuscules), et xdigit (chiffres hexadécimaux). Le comportement de ces classes de caractères standards est généralement cohérent sur les plateformes pour les caractères de l'ensemble ASCII 7 bits. Qu'un caractère non ASCII donné appartienne ou non à une de ces classes dépend de la *collation* utilisée par la fonction ou l'opérateur de l'expression rationnelle, (voir **Section 24.2**), ou par défaut de la locale indiquée par LC_CTYPE pour cette base (voir **Section 24.1**). La classification de caractères non ASCII peut varier entre les plateformes même pour des locales de nom similaire. (Mais la locale C ne considère jamais tout caractère non ASCII comme appartenant à une de ces classes.) En plus de ces classes de caractères standard, PostgreSQL définit the word character class, which is the same as alnum plus the underscore (_) character, and la classe de caractères ascii, qui contient l'ensemble ASCII 7 bits exactement.



LIKE et SIMILAR : http://www.postgresql.org/docs/9.6./static/functions-matching.html



Les expressions rationnelles SQL sont un curieux mélange de la notation LIKE (introduite dans les versions initiales de la norme ANSI) et de la notation POSIX (communes aux systèmes de type Unix).

LIKE et SIMILAR : https://www.postgresql.org/docs/current/static/functions-matching.html

OPÉRATEUR D'EXTRACTION

- substr (*s* similar to *p* [escape *c*])

- Il est possible de segmenter une ER en trois parties : préfixe, infixe et suffixe de façon à isoler la seule partie infixe (voir diapositive suivante) à des fins d'extraction ou de remplacement.
- Les parties sont séparées par le caractère d'échappement c (*escape c*).

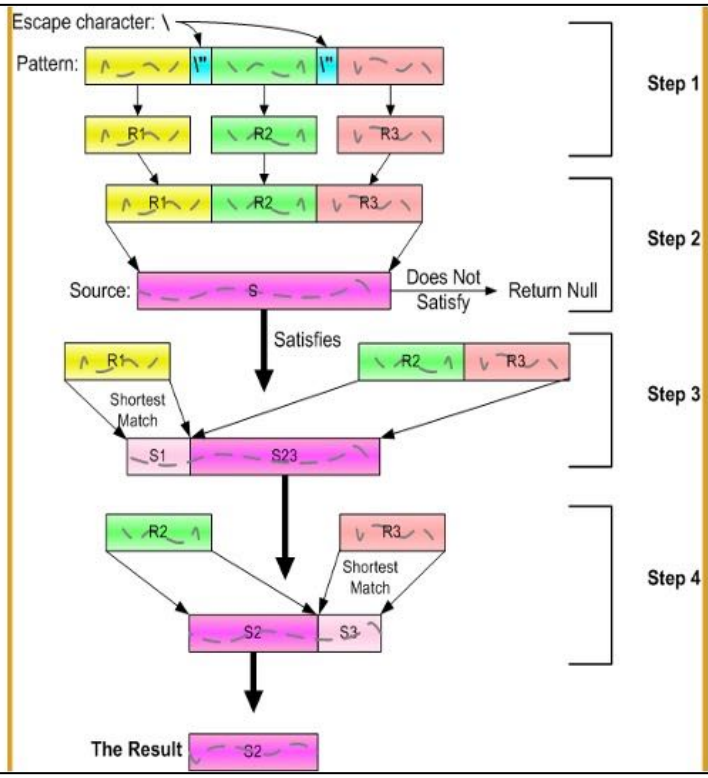
2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

26

ALGORITHME UTILISÉ PAR L'OPÉRATEUR D'EXTRACTION

ISO/IEC 9075-2:2003 (E)

p. 18



FORME HISTORIQUE SIMPLIFIÉE

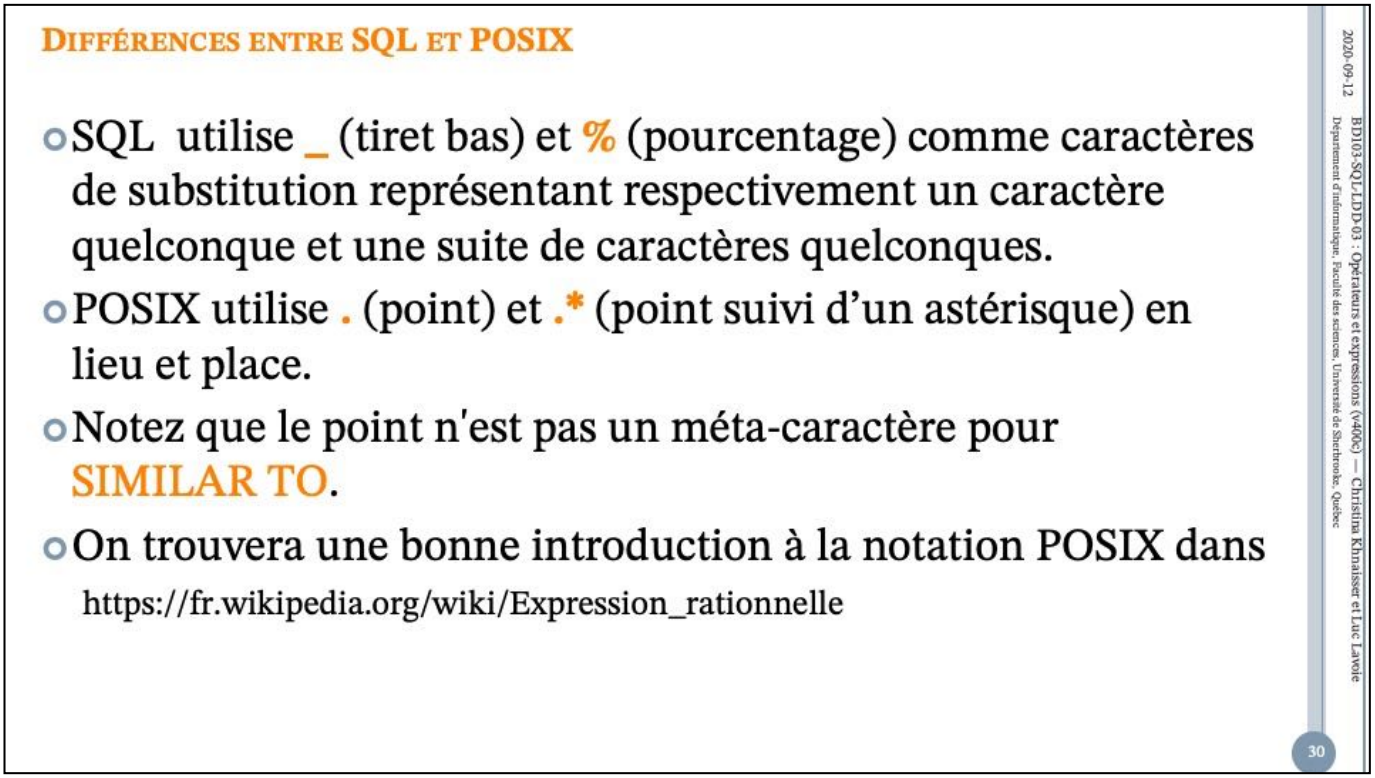
- *s* [not] like *p* [escape *c*]
- ...
- bof!

DIFFÉRENCES ENTRE SQL ET POSTGRESQL

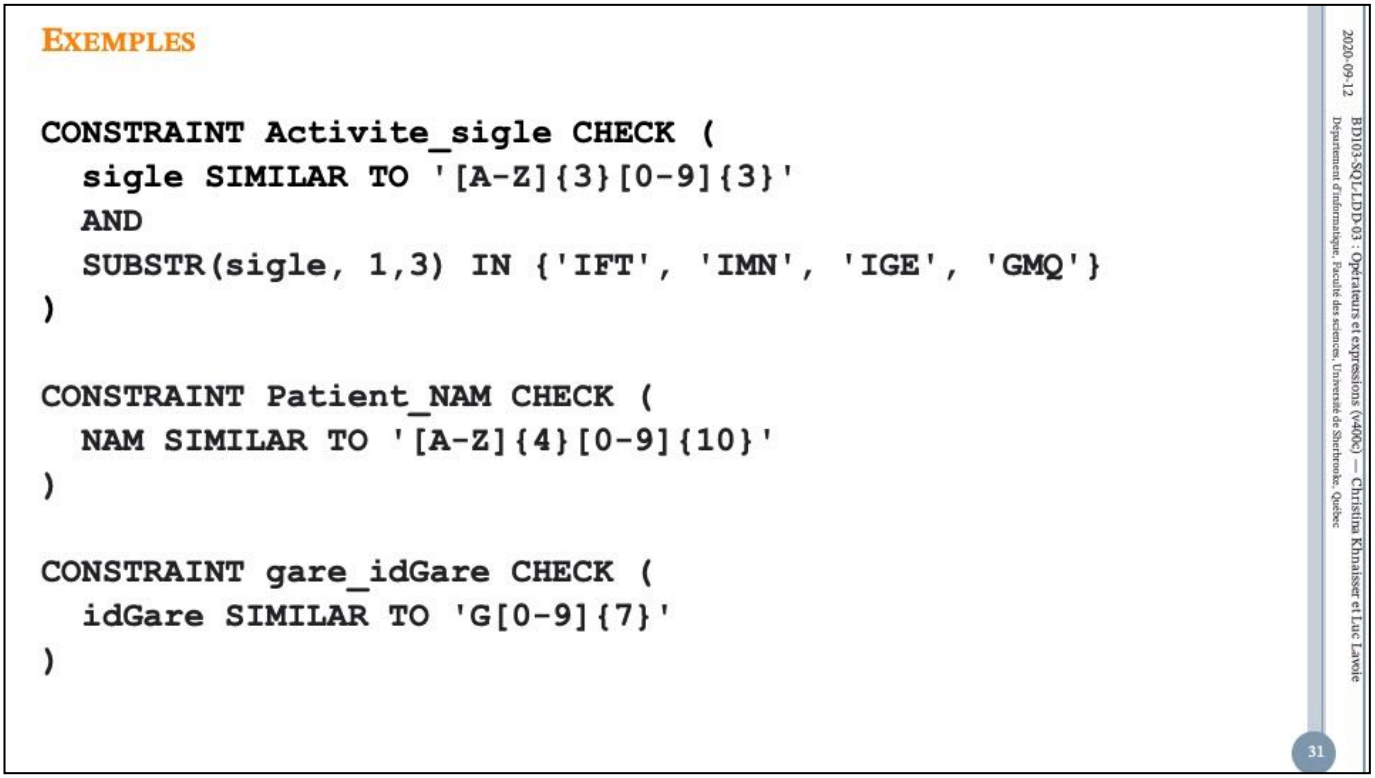
- Le dialecte PostgreSQL met à disposition de nombreuses abréviations, comme
  - ~ pour SIMILAR TO
- et de nombreuses fonctions supplémentaires
  - regexp_replace
  - regexp_matches
  - regexp_split_to_table
  - etc.
- Voir
  - https://docs.postgresql.fr/9.6/functions-matching.html

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

29



LIKE et SIMILAR : http://www.postgresql.org/docs/9.6./static/functions-matching.html



EXEMPLES

```
CONSTRAINT Activite_sigle CHECK (
  sigle SIMILAR TO '[A-Z]{3}[0-9]{3}'
  AND
  SUBSTR(sigle, 1,3) IN {'IFT', 'IMN', 'IGE', 'GMQ'}
)

CONSTRAINT Patient_NAM CHECK (
  NAM SIMILAR TO '[A-Z]{4}[0-9]{10}'
)

CONSTRAINT gare_idGare CHECK (
  idGare SIMILAR TO 'G[0-9]{7}'
)
```

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

31

-- Restreindre le catalogue des activité à celles d'informatique et de géomatique

EXERCICES

- Récrire la contrainte **`Activite_sigle`** en utilisant qu'une expression régulière (ne pas utiliser SUBSTR).
- Écrire une contrainte pour valider des numéros de téléphone canadiens.
- Écrire une contrainte pour valider des numéros de téléphone internationaux.

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

32

NULS : MARQUEURS ET VALEURS

- Opérations
- Marqueur ou valeur ?
- Cohérence
- Annulabilité
- Conséquences

OPÉRATIONS AVEC LES NULS (1/3)

- En général, en SQL, NULL doit être considéré comme un marqueur d'attribut et non comme une valeur.
- Quelques conséquences
  - dans une expression, NULL marque l'absence de valeur
    - INSERT INTO R (*a*, *b*) VALUES (12, NULL)
    - *a* prend pour valeur 12; *b* n'a pas de valeur
  - soit *a* un attribut quelconque, on ne peut pas écrire
    - *a* = NULL
  - on doit écrire
    - *a* IS NULL

2020-09-12 BD103-SQLLDD 03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

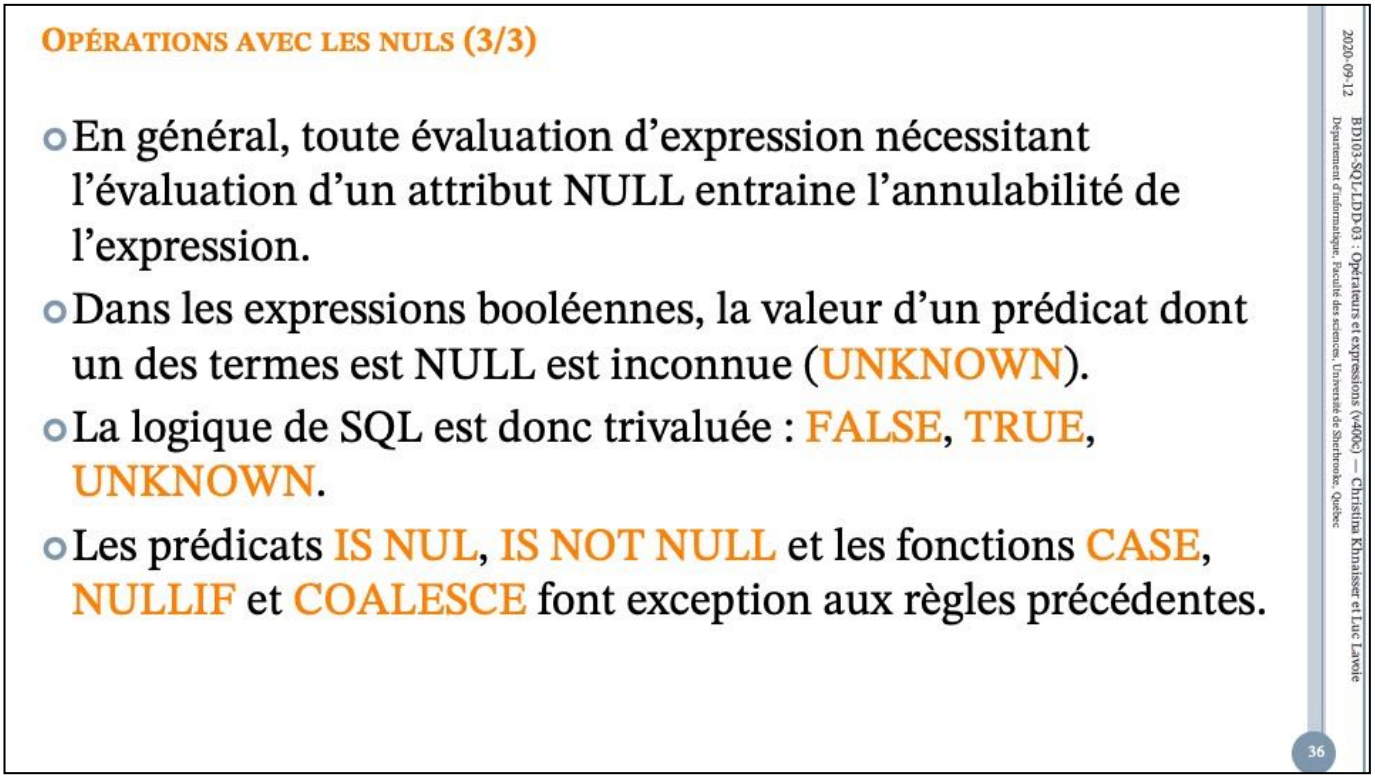
34

OPÉRATIONS AVEC LES NULS (2/3)

- Par ailleurs, soit *a* et *b* deux attributs nuls,
  - *a* = *b* est inconnu
  - *a* <> *b* est inconnu aussi
- En particulier,
  - *a* = *a* est inconnu
  - *a* <> *a* est inconnu aussi

2020-09-12 BD103-SQLLDD 03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

35



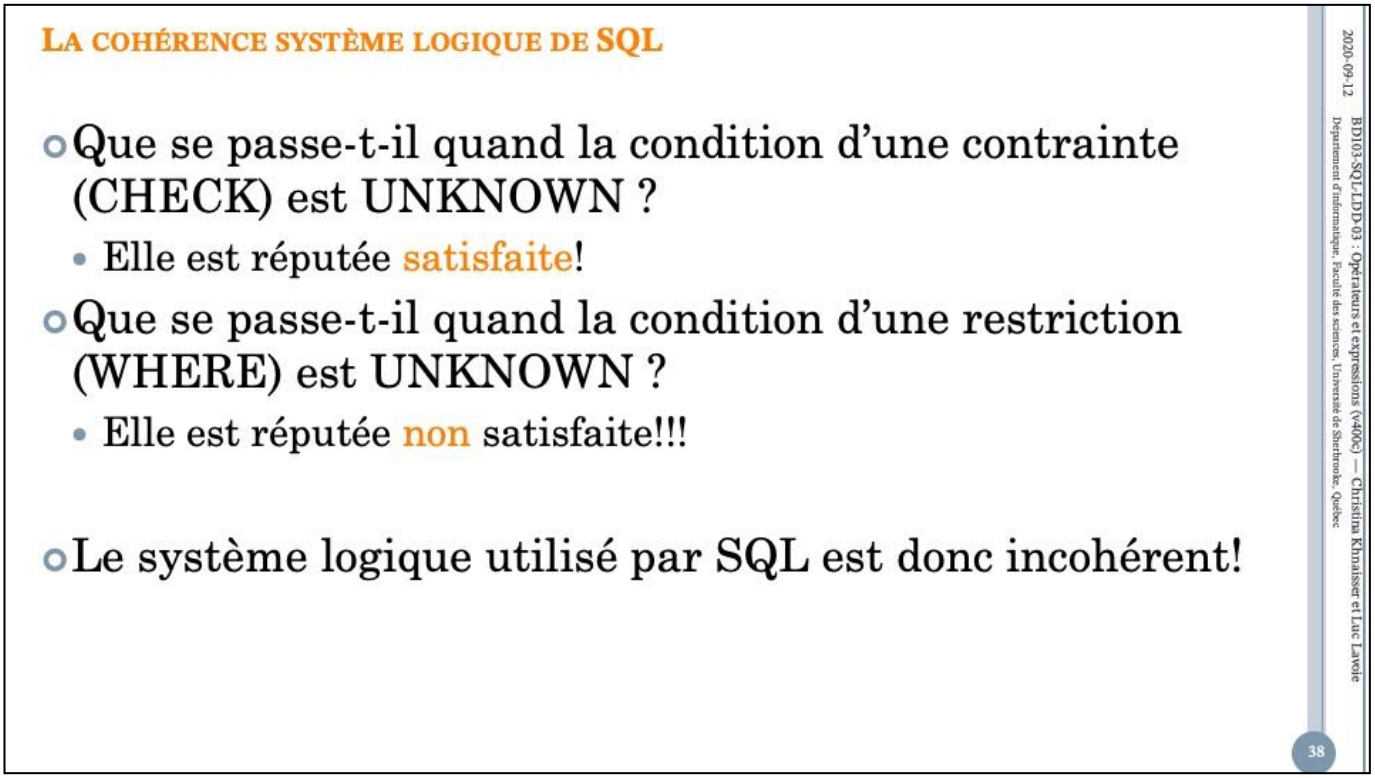
Nous verrons plus tard que les nuls appellent aussi un traitement particulier lors de l'agrégation.

NULL : MARQUEUR OU VALEUR ?

- Notons l'ambigüité fréquente entre
  - marqueur d'attribut : c'est-à-dire une propriété d'un attribut (en extension au modèle relationnel)
  - valeur d'une expression : ce qui nécessite d'ajouter cette valeur à tous les domaines (en extension au modèle de typage).
- Malheureusement, la norme et les dialectes ont recours aux deux mécanismes!

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

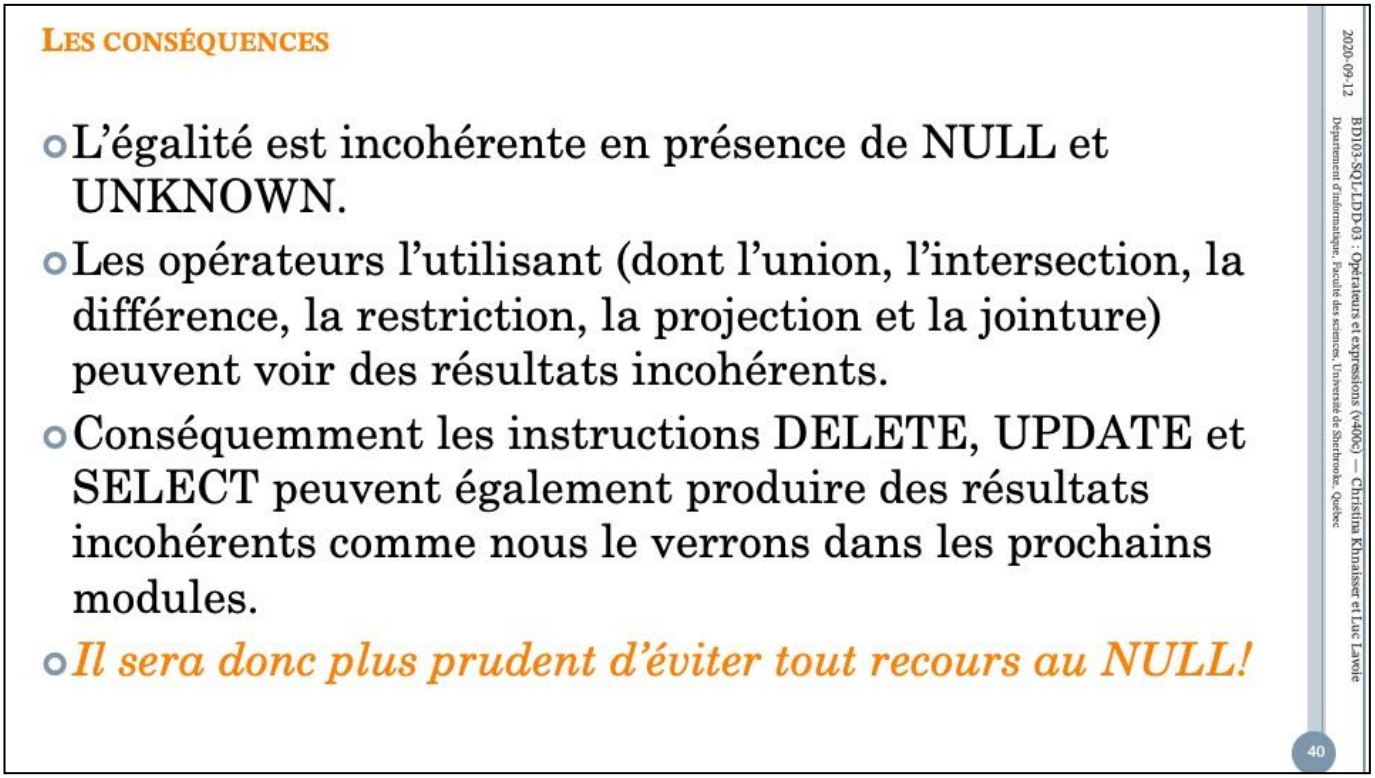
37



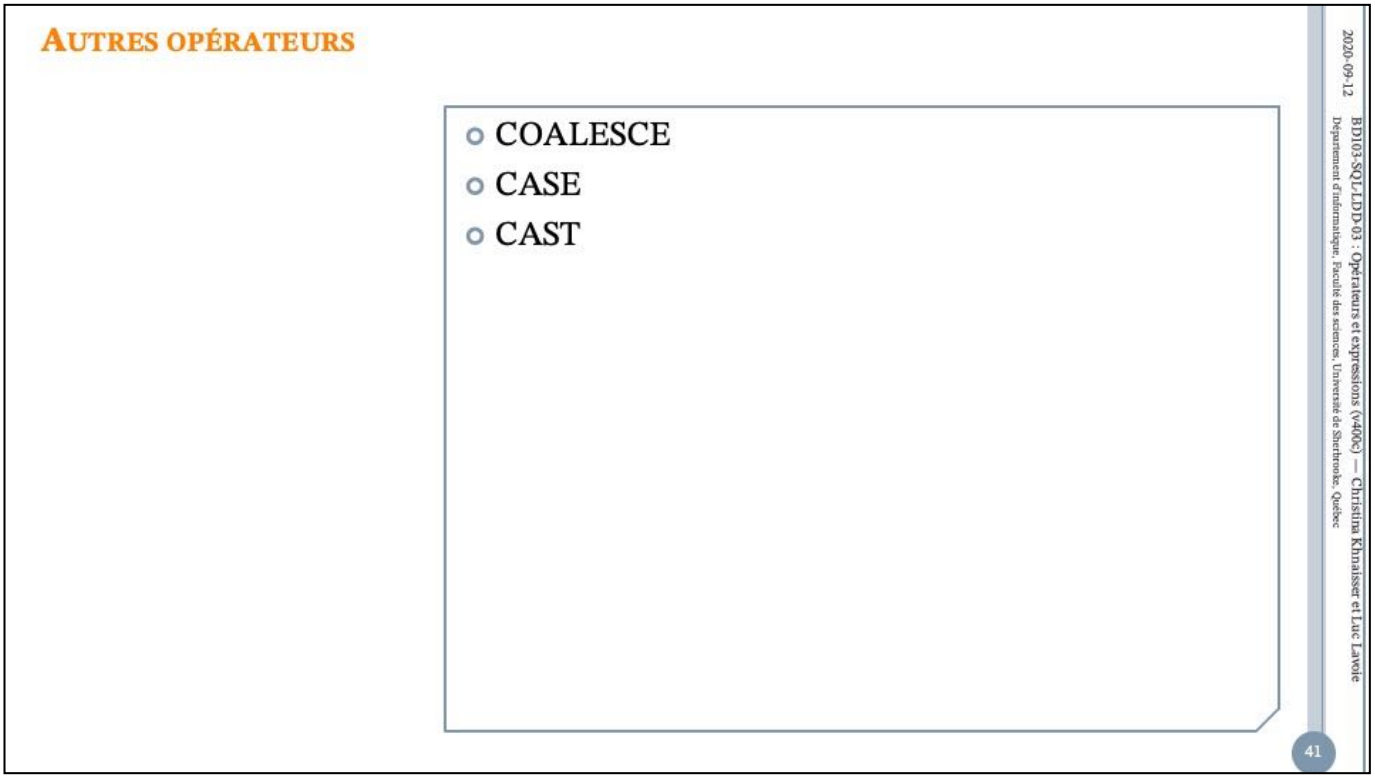
On ne peut donc s'assurer simplement du respect d'une contrainte avec une clause WHERE équivalente, il faut s'assurer de traiter adéquatement les cas UNKNOWN explicitement

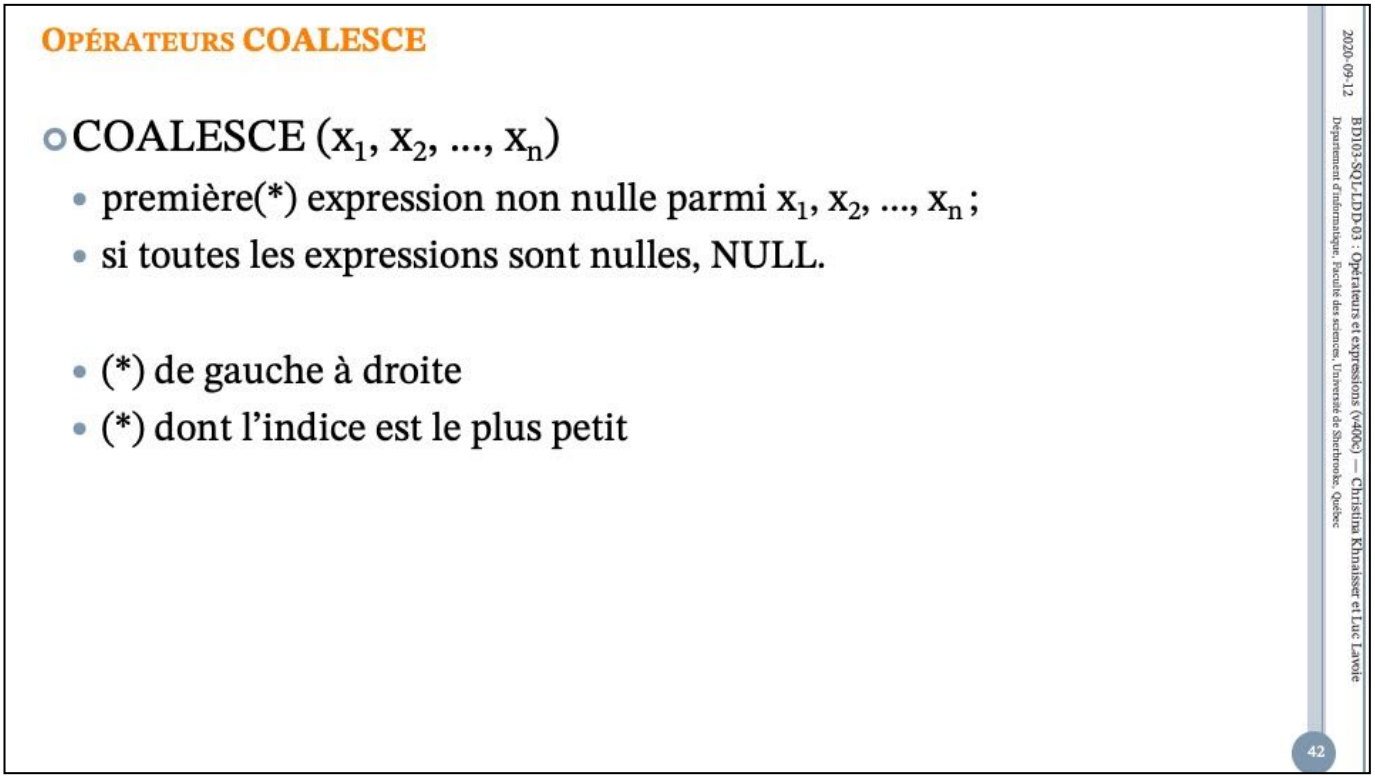
L'ANNULABILITÉ LOGIQUE (UNKNOWN)

- Qu'arrive-t-il si un attribut logique (BOOLEAN) est NULL ?
- Il est considéré comme UNKNOWN.



Seul le renommage semble à l'abrisi de l'incohérence induite!


AUTRES OPÉRATEURS
COALESCE
CASE
CAST
2020-09-12
BD103-SQLLDD 03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec
41



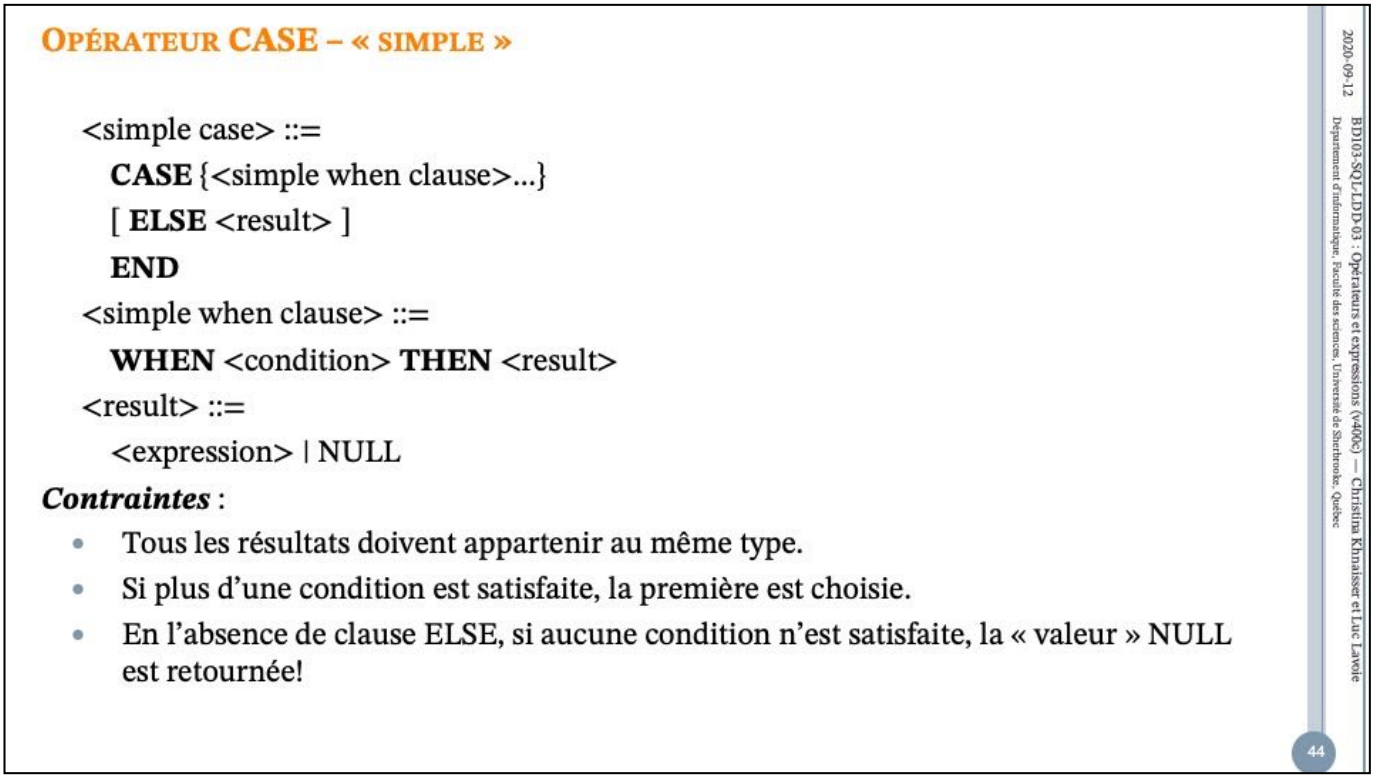
COALESCE : http://www.postgresql.org/docs/9.6./static/functions-conditional.html

OPÉRATEUR CASE

- C'est un *opérateur* de choix (*pas une instruction*).
- C'est-à-dire qu'il permet de choisir une valeur sur la base d'une catégorisation établie à l'aide d'une condition (*simple case*) ou d'une énumération de valeurs (*searched case*).
- Il en existe donc deux formes

  <case expression> ::=

  <simple case> | <searched case>
- Dans tous les cas, l'opérateur retourne une valeur.



OPÉRATEUR CASE – « SIMPLE »

<simple case> ::=
CASE {<simple when clause>...}
[**ELSE** <result>]
END
<simple when clause> ::=
WHEN <condition> **THEN** <result>
<result> ::=
<expression> | NULL

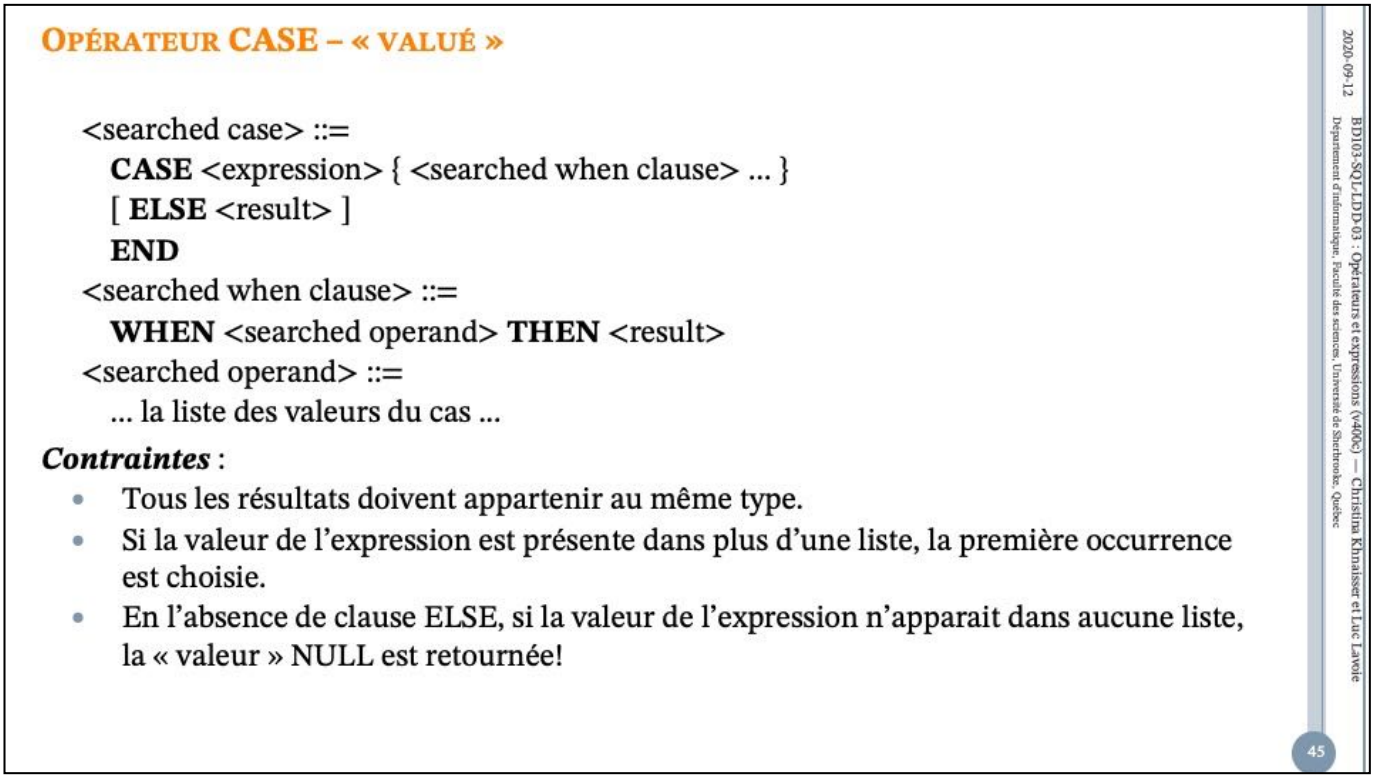
Contraintes :

- Tous les résultats doivent appartenir au même type.
- Si plus d'une condition est satisfaite, la première est choisie.
- En l'absence de clause ELSE, si aucune condition n'est satisfaite, la « valeur » NULL est retournée!

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec
44

```
-- valider la réussite au premier cycle
-- « somme » est la somme des cotes des cours réussis et « n » le nombre de cours réussis

CHECK
 (
  CASE
   WHEN n <> 0 THEN somme/n >= 2.0
   ELSE false
  END
 )
```



OPÉRATEUR CASE – « VALUÉ »

<searched case> ::=
CASE <expression> { <searched when clause> ... }
[**ELSE** <result>]
END
<searched when clause> ::=
WHEN <searched operand> **THEN** <result>
<searched operand> ::=
... la liste des valeurs du cas ...

Contraintes :

- Tous les résultats doivent appartenir au même type.
- Si la valeur de l'expression est présente dans plus d'une liste, la première occurrence est choisie.
- En l'absence de clause ELSE, si la valeur de l'expression n'apparait dans aucune liste, la « valeur » NULL est retournée!

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec
45

-- valider la réussite selon le cycle
-- « somme » est la somme des cotes des cours réussis et « n » le nombre de cours réussis

```
CHECK
 (
  CASE
   WHEN n <> 0 THEN
    somme/n >= CASE cycle
               WHEN 1 THEN 2.0
               WHEN 2, 3 THEN 2.7
               ELSE 3.0
              END
   ELSE
    false
  END
 )
```

FORMES ABRÉGÉES DU CASE

- Pour terminer, la norme précise ceci :
  - NULLIF (*V*1, *V*2) est équivalent à :

```
CASE
WHEN V1=V2 THEN NULL
ELSE V1
END
```

  - COALESCE (*V*1, *V*2) est équivalent à :

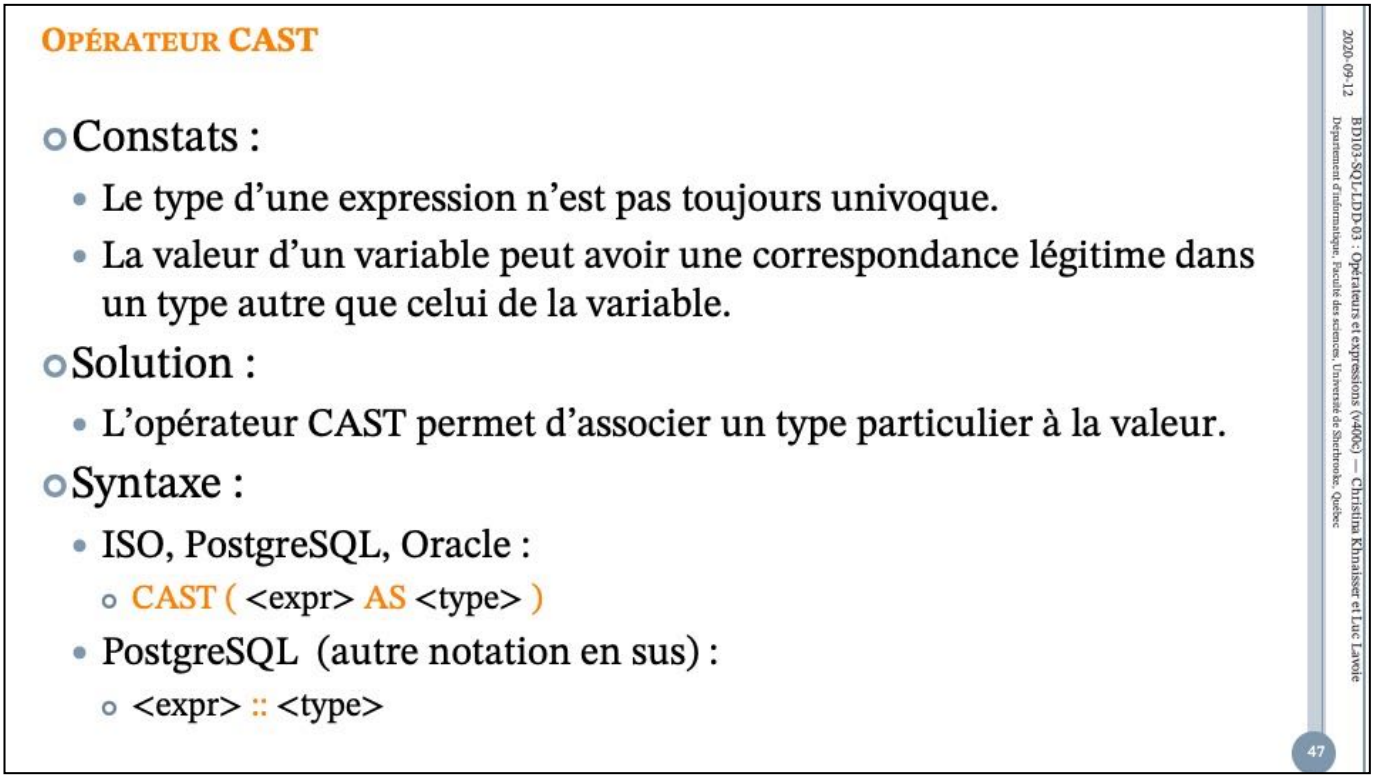
```
CASE
WHEN V1 IS NOT NULL THEN V1
WHEN V2 IS NOT NULL THEN V2
ELSE NULL
END
```

  - 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
```

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

46



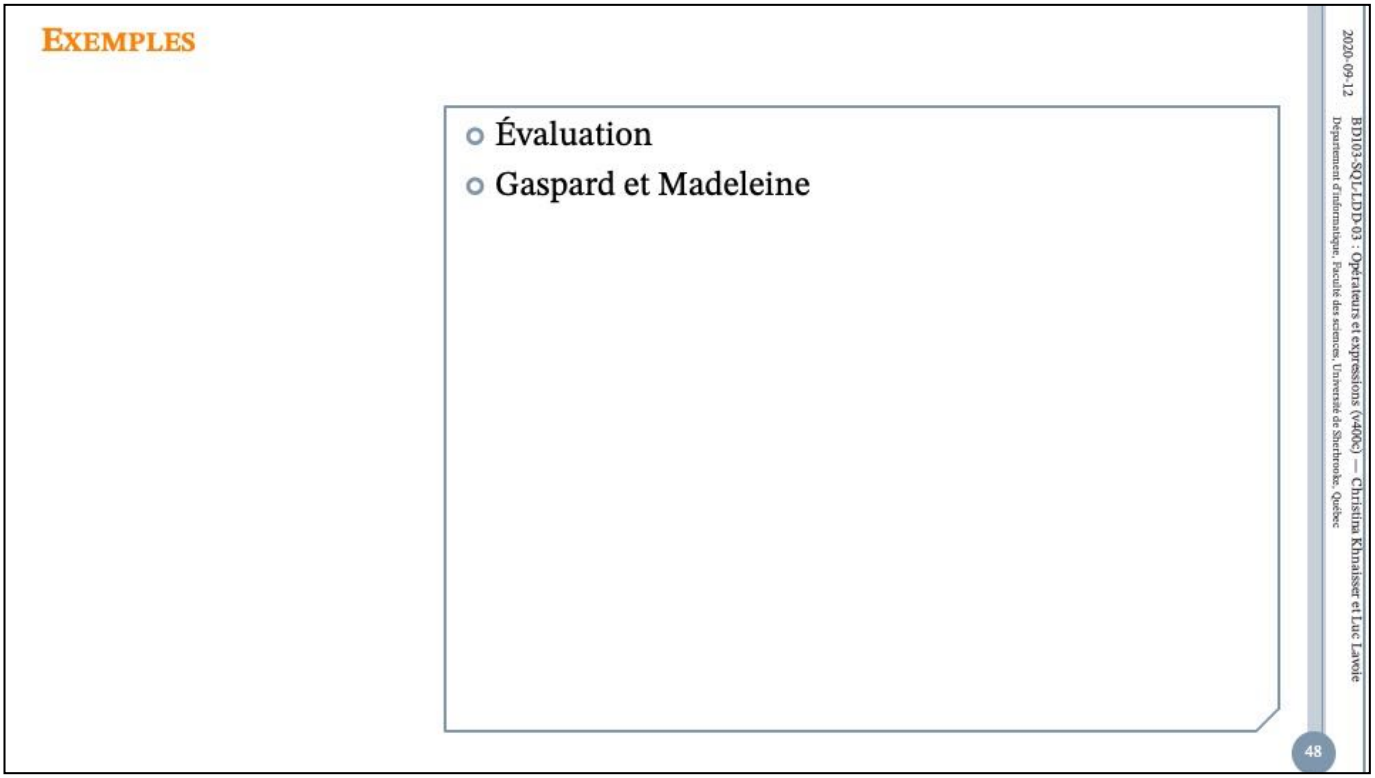
Un code de produit est représenté par 12 chiffres et la représentation choisie par le concepteur est NUMERIC(12)
Les produits congelés sont reconnaissables par le fait que les 4 derniers chiffres forment un nombre plus grand que 5000

```
CHECK
 (
  substr(CAST(code AS TEXT), 9, 4) > '5000'
 )
```

ou

```
CHECK
 (
  CAST (substr(CAST(code AS TEXT), 9, 4) AS INTEGER) > 5000
 )
```

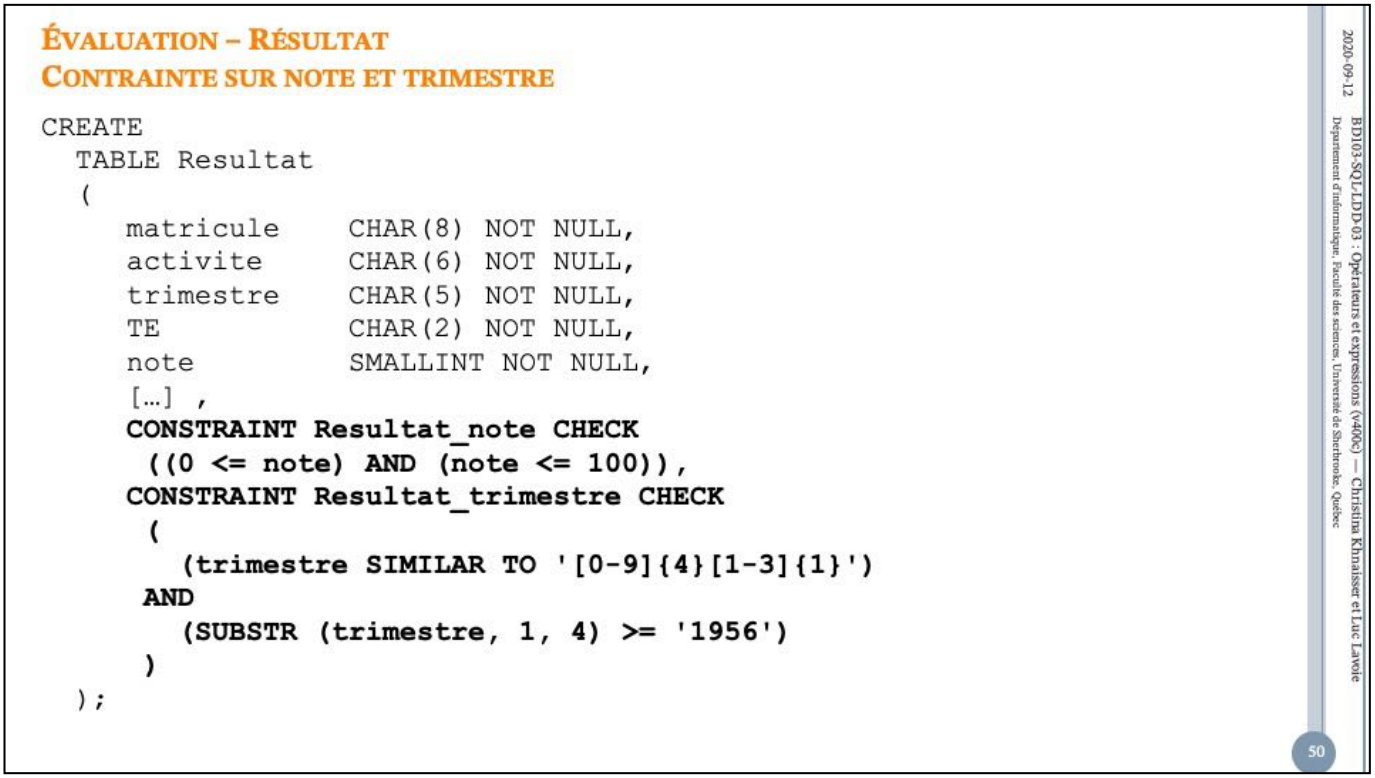
Existe-t-il un autre moyen n'utilisant que des opérations arithmétiques ?


EXEMPLES
Évaluation
Gaspard et Madeleine
2020-09-12
BD103-SQLLDD 03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec
48

ÉVALUATION – ACTIVITÉ
CONTRAINTE SUR SIGLE

```
CREATE TABLE Activite (
  sigle        CHAR(6) NOT NULL,
  titre        VARCHAR(46) NOT NULL,
  CONSTRAINT Activite_cc0 PRIMARY KEY (sigle),
  CONSTRAINT Activite_sigle0 CHECK
      (sigle SIMILAR TO '[A-Z]{3}[0-9]{3}')
  );

ALTER TABLE Activite
  ADD CONSTRAINT Activite_sigle1 CHECK
    (
      SUBSTR(sigle, 1,3)
        IN ('IFT', 'IMN', 'IGE', 'GMQ')
    );
```



ÉVALUATION – RÉSULTAT
CONTRAINTE SUR NOTE ET TRIMESTRE

```
CREATE
  TABLE Resultat
  (
     matricule     CHAR(8) NOT NULL,
     activite      CHAR(6) NOT NULL,
     trimestre     CHAR(5) NOT NULL,
     TE            CHAR(2) NOT NULL,
     note          SMALLINT NOT NULL,
     […] ,
     CONSTRAINT Resultat_note CHECK
      ((0 <= note) AND (note <= 100)),
     CONSTRAINT Resultat_trimestre CHECK
      (
        (trimestre SIMILAR TO '[0-9]{4}[1-3]{1}')
      AND
        (SUBSTR (trimestre, 1, 4) >= '1956')
      )
  );
```

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec
50

```
-- Les trimestres sont encodés en suffixant le chiffre du trimestre
-- à l'année. L'Université de Samarcande a été fondée en 1956.
```

UN PETIT EXERCICE ?
GASPARD ET MADELEINE – CE N'ÉTAIT PAS FINI!

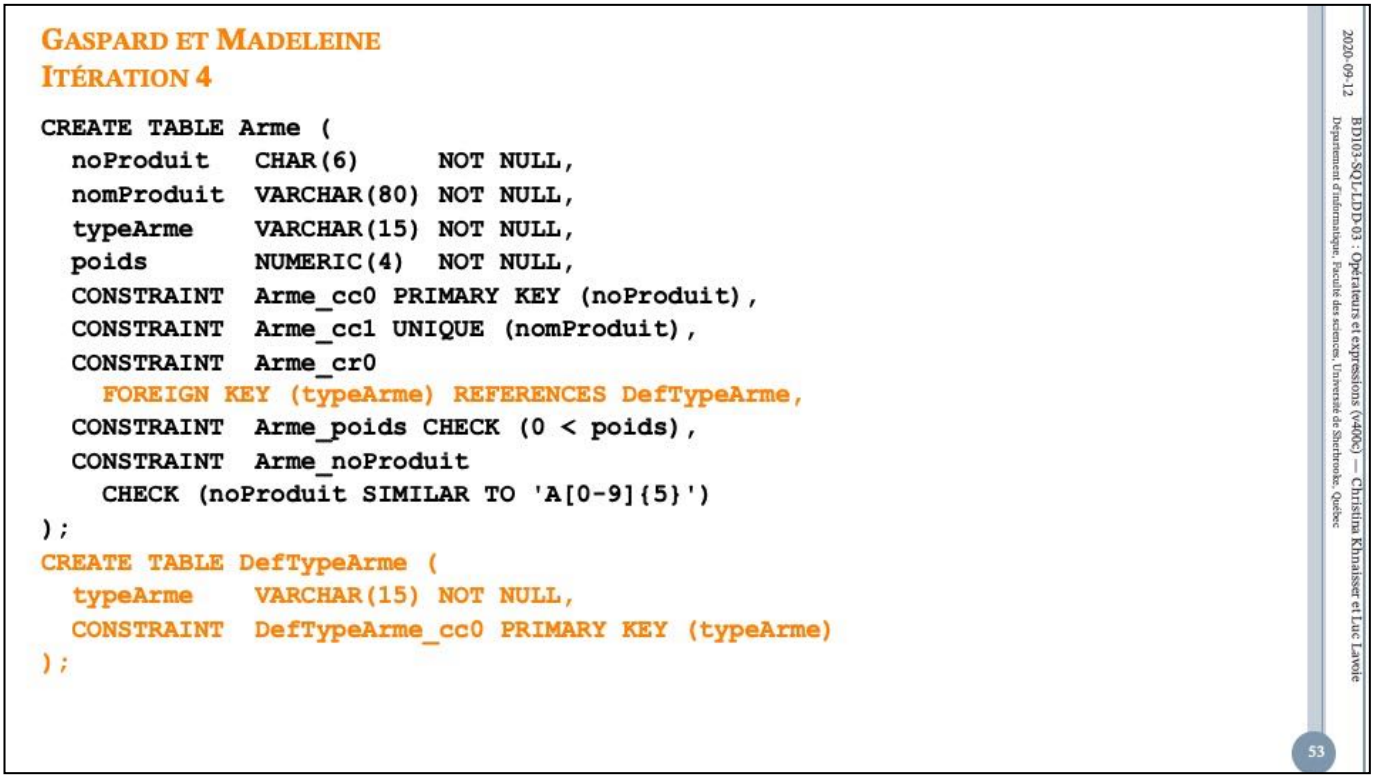
```
CREATE TABLE Arme (
    noProduit    CHAR(6)      NOT NULL,
    nomProduit   VARCHAR(80)  NOT NULL,
    typeArme     VARCHAR(15)  NOT NULL,
    poids        NUMERIC(4)   NOT NULL,
    CONSTRAINT Arme_cc0 PRIMARY KEY (noProduit),
    CONSTRAINT Arme_cc1 UNIQUE (nomProduit),
    CONSTRAINT Arme_poids CHECK (0 < poids),
    CONSTRAINT Arme_noProduit CHECK
      (noProduit SIMILAR TO 'A[0-9]{5}')
);
```

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

51

GASPARD ET MADELEINE
DISCUSSION

- Pouvons-nous mettre n'importe quel type d'armes ou ne faut-il pas s'assurer qu'il fait partie d'un ensemble déterminé ?
  - Voir version 4



GASPARD ET MADELEINE
ITÉRATION 4

```
CREATE TABLE Arme (
  noProduit    CHAR(6)      NOT NULL,
  nomProduit   VARCHAR(80)  NOT NULL,
  typeArme     VARCHAR(15)  NOT NULL,
  poids        NUMERIC(4)   NOT NULL,
  CONSTRAINT   Arme_cc0 PRIMARY KEY (noProduit),
  CONSTRAINT   Arme_cc1 UNIQUE (nomProduit),
  CONSTRAINT   Arme_cr0
    FOREIGN KEY (typeArme) REFERENCES DefTypeArme,
  CONSTRAINT   Arme_poids CHECK (0 < poids),
  CONSTRAINT   Arme_noProduit
    CHECK (noProduit SIMILAR TO 'A[0-9]{5}')
);
CREATE TABLE DefTypeArme (
  typeArme     VARCHAR(15) NOT NULL,
  CONSTRAINT   DefTypeArme_cc0 PRIMARY KEY (typeArme)
);
```

2020-09-12 BD103-SQLLDD-03 : Opérateurs et expressions (v400c) — Christina Khnaisser et Luc Lavoie
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

53

Faire remarquer
* l'identité de définition de typeArme
* l'impact qu'aurait l'annulabilité de typeArme dans un sens et dans l'autres

GASPARD ET MADELEINE
EST-CE FINI ?

- Nous avons trouvé une solution technique, mais n'aurions-nous pas dû revenir à la modélisation au préalable ?
  - Voir le module BD010 - Gaspard et Madeleine !

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

LES COLLES DU PROF

- SQL comporte un opérateur mod (pour modulo), est-ce la même chose que le reste de division entière ?
- En mathématiques ?
- En SQL ?