

# Bases de données *SQL*

**Clés primaires et secondaires**  
**Clés relatives et absolues**  
**Clés externes ou internes**

SQL\_07  
v130c  
2025-04-01

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

© 2018-2021, **Matus** (<http://info.usherbrooke.ca/llavoie>)  
CC BY-NC-SA 4.0 (<https://creativecommons.org/licenses/by-nc-sa/4.0/>)

1

## Plan

- Clés primaires et secondaires
- Clés relatives et absolues
- Clés externes et internes
- Générateurs de clés internes
  - IDENTITY
  - SEQUENCE
  - SERIAL et les autres
- Quelques utilisations remarquables des clés

2025-04-01  
MCEP\_SQL\_07 — Clés (v130c) © 2018-2025, Matus — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

2

2

## Clés

### Pourquoi ?

- «Pour identifier (et donc distinguer) ce qui doit l'être.»
- Les clés contribuent à intégrer à faible coût d'importantes contraintes d'intégrité.
- Une clé n'en remplace pas une autre, elle s'y ajoute.
  - La raison d'un tel ajout doit donc être explicite et motivée.

2025-04-01

MCHD\_SQL\_07 — Clés (v1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

3

3

## Clés primaires et secondaires

- Définition
- Origines
- Utilité
- Critères de choix
- Représentation sous SQL

2025-04-01

MCHD\_SQL\_07 — Clés (v1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

4

4

## Clés primaires et secondaires

### Définition et origine

#### ○ Définition

- Parmi les clés candidates d'une relation, une d'elle est désignée comme clé primaire; les autres (s'il en est) sont désignées comme clés secondaires.

#### ○ Origine de la différenciation

- Les limites des fichiers séquentiels indexés... en 1966!

2025-04-01

MCEP SQL 07 — Clés (V1302) © 2018-2025, M. H. S. — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

5

5

## Clés primaires et secondaires

### Utilité

#### ○ Pertinence du concept

- Ce concept est inutile du strict point de vue de la théorie relationnelle.

#### ○ Toutefois

- Le choix d'une clé candidate de référence (clé primaire) peut contribuer à la clarté et à la modifiabilité du schéma, particulièrement lors de la définition de clés référentielles, ce qui importe du point de vue du génie logiciel.
- En dernier ressort, le choix d'une clé plutôt qu'une autre peut avoir une incidence sur la performance en fonction des techniques d'indexation utilisées.

2025-04-01

MCEP SQL 07 — Clés (V1302) © 2018-2025, M. H. S. — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

6

6

## Clés primaires et secondaires

### Critère de choix

#### ○ Flou

- Nombre d'attributs constitutifs ?
- Taille totale ?
- Flexibilité au moment des jointures ?
- Flexibilité au moment des projections (donc des groupements) ?

- En pratique, le choix dépend tout autant de considérations d'utilisation (évoluant dans le temps) que de considérations de représentation (variant d'un SGBD à l'autre).

2025-04-01

MCHD\_SQL\_07 — Clés (v1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

7

7

## Clés primaires

### Représentation sous SQL

#### ○ Clé primaire

- PRIMARY KEY
- les attributs la composant sont automatiquement marqués  
« not null »

#### ○ Clés secondaires

- UNIQUE
- les attributs les composant **ne sont pas** automatiquement marqués  
« not null »
  - ce qui est une erreur, aucun attribut d'une clé ne peut être annulable!

2025-04-01

MCHD\_SQL\_07 — Clés (v1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

8

8

## Clés relatives et absolues

- Rappels
- Définitions
- Pourquoi ?
- Quand ?
- Avantages (et inconvénients)
- Coexistence ?

2025-04-01

MCHD\_SQL\_07 — Clés (Y1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

9

9

### Rappel

- Ci-après, sauf indication contraire, le terme *clé* désigne une clé irréductible (*candidate key*). Au besoin, lorsqu'il sera nécessaire de désigner tout sous-ensemble d'attributs identifiant les tuples d'une relation, nous utiliserons l'expression *clé irréductible ou non* (ou l'équivalent, *clé stricte ou non*).
- Une relation possède toujours une clé (à la limite l'ensemble de ses attributs à défaut de tout autre).

2025-04-01

MCHD\_SQL\_07 — Clés (Y1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

10

10

### Clé propre Définition

- Une clé est dite *propre* si et seulement si :
  - aucun sous-ensemble de ses attributs n'est une clé référentielle.
- Une clé propre est dite *absolue* si et seulement si :
  - elle ne comporte qu'un seul attribut ;
- Une clé absolue représente le fait qu'une relation comporte un attribut qui lui est propre et qui identifie la totalité de ses tuples.

2025-04-01

MCHD\_SQL\_07 — Clés (V1302) © 2018-2025, M. H. S. — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

11

11

### Clé relative Définition

- Une clé est dite relative si et seulement si :
  - au moins un sous-ensemble de ses attributs est une clé référentielle ;
- Une clé relative est dite
  - *partielle*, si au moins un attribut lui est propre (c'est-à-dire ne participant à aucune clé référentielle) ;
  - *totale*, sinon.
- Corolaires :
  - toute clé est soit propre, soit relative.
  - une clé relative partielle comporte au moins deux attributs.

2025-04-01

MCHD\_SQL\_07 — Clés (V1302) © 2018-2025, M. H. S. — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

12

12

### Clés relatives

#### Pourquoi ? Quand ?

- Les clés relatives sont nécessaires lorsqu'une dépendance fonctionnelle doit être modélisée entre relations.
- Les clés relatives sont donc utilisées lors de la traduction du schéma conceptuel vers le schéma logique.
- En particulier, pour la représentation d'entités faibles en regard des schémas EA.
- Voir module MCD\_04.

2025-04-01

MCD SQL 07 — Clés (V1302) © 2018-2025, M. H. S. — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

13

13

### Coexistence des clés absolues et relatives

- Certains auteurs recommandent préférentiellement les clés absolues comme cible des clés référentielles, voire même d'en créer artificiellement une s'il n'y a pas de clé absolue externe.
- Cette tactique est rarement payante en pratique attendu les algorithmes contemporains d'indexation.
- Au contraire, ceci concourt souvent à augmenter le nombre de jointures dans les requêtes.

2025-04-01

MCD SQL 07 — Clés (V1302) © 2018-2025, M. H. S. — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

14

14

## Clés internes et externes

- Définition
- Pourquoi ?
- Quand ?
- Comment ?

**internes**  
**artificielles**  
**subsidiaries**

**externes**  
**naturelles**

### Clé interne

- Clé absolue n'ayant aucune sémantique associée en regard du domaine d'application.
- En pratique, une clé interne doit donc être maintenue par le SGBD en regard du schéma et de la table ciblée, puisqu'elle ne provient pas de la réalité modélisée.
- Complément
  - Une clé est dite externe, si et seulement si elle n'est pas interne !



## Clé interne

Quand **doit-on** l'utiliser ?

- Pour distinguer ce qui doit l'être en l'absence de clé externe (éviter les amalgames accidentels)
  - *exemple* : les humains, les animaux, les plantes ...
- Pour éviter la propagation d'informations sensibles contenues dans les clés externes
  - *exemple* : le NAS (numéro d'assurance sociale)
- Pour éviter que la clé d'une entité retirée soit réutilisée
  - *exemple* : l'unicité et la pérennité des clés externes sont rarement garanties à long terme

2025-04-01

MCHD\_SQL\_07 — Clés (V1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

17

17

## Clé interne

Quand **peut-on** l'utiliser ?

- Pour faciliter que la clé d'une entité retirée soit réutilisée
  - *exemple* : l'unicité et l'immutabilité des clés externes sont rarement garanties à long terme
- Pour pallier les contingences externes relativement à la définition et la modification d'une clé externe
  - *exemple* : l'unicité et la pérennité des clés externes sont rarement garanties à long terme
- Pour optimiser le temps de calcul
  - *exemple* : clé externe TRÈS volumineuse (enfin, suffisamment volumineuse pour avoir un impact réel sur les ressources).

2025-04-01

MCHD\_SQL\_07 — Clés (V1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

18

18

### Clé interne

#### Quand **ne doit-on pas** l'utiliser ?

- On ne doit pas remplacer une clé relative par une clé externe. Tout au plus est-il possible de l'ajouter, et encore avec une solide justification.
- En pratique, il y a peu d'avantages concrets à cette ajout. Dans certains cas, il y a même une perte notable de performance et un encombrement supérieur.
- Pour plus de détails, voir [Sainte-Marie].

### Clé interne et code

#### Un **code n'est pas une clé interne** !

- Un code est une clé externe dotée sémantique en regard du domaine d'application.
- Le code peut résulter d'une convention connue ou être généré à l'interne comme à l'externe.
  - *exemples* : les symboles atomiques du tableau périodique, les abréviations, les acronymes.
- Un code a souvent une valeur mnémotechnique afin
  - de servir à fins d'affichage, par souci de compacité dans des tableaux, par exemple;
  - de pouvoir servir dans la programmation des requêtes.
- Un code est donc contingent à des règles externes et ne peut être modifié sans accord (obtenu explicitement ou par une règle).
- Un code n'est pas une clé interne!

### Clés internes

#### Exemples de codes qui sont de mauvaises clés internes

- Le matricule de la Régie d'assurance-maladie du Québec RAMQ
  - référence au nom, prénom et date de naissance
- Le numéro d'assurance sociale
  - référence au statut de résidence
- Le code d'identification personnel (CIP) de l'Université de Sherbrooke
  - référence au nom et prénom
- Le matricule étudiant de l'Université de Sherbrooke
  - référence au nom, prénom, date de naissance et date d'admission
- Le matricule employé de l'Université de Sherbrooke
  - référence à la date d'embauche
- Constat
  - En pratique, il n'est de bonne clé interne que celles qui *ne proviennent pas* du domaine d'application. La tentation de permettre d'en déduire des informations, souvent sensibles, semble trop forte.

### Clés internes

#### Comment les générer ?

- En SQL
  - SEQUENCE
- Dans de nombreux dialectes SQL
  - SMALLSERIAL
  - SERIAL
  - BIGSERIAL
  - UID
  - GUID
  - UUID

## Génération de clés internes en SQL

- Caractéristiques souhaitées
- Les solutions
  - IDENTITY
  - SEQUENCE
    - générateurs
    - fonctions
  - SERIAL, SMALLSERIAL, BIGSERIAL
  - UID, GUID, UUID

2025-04-01

MCHD\_SQL\_07 — Clés (v1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

23

23

### Génération de clés internes en SQL Caractéristiques souhaitées

- Artificialité garantie lors de la génération
- Unicité garantie lors de la génération
- Unicité garantie dans un contexte transactionnel
- Automatisation de la génération lors de l'insertion

2025-04-01

MCHD\_SQL\_07 — Clés (v1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

24

24

## Clés internes en SQL ISO

## Solution 1 : Par annotation de type (IDENTITY)

*type\_identitaire* ::=

*type* GENERATED [ALWAYS | AS DEFAULT]

AS IDENTITY *séquence*

*séquence* ::=

*nom\_de\_séquence* | *générateur*

## Notes relatives à PostgreSQL

- Le *type* est limité aux seuls types numériques entiers (SMALLINT, INTEGER, BIGINT).
- La mise à disposition du mécanisme ISO a été graduelle, depuis la version 9.6 jusqu'à la 12.4.

2025-04-01

MCHD\_SQL\_07 — Clés (v1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'Informatique, Faculté des sciences, Université de Sherbrooke, Québec

25

25

## Clés internes en SQL ISO

## Solution 2 : Création d'un générateur (SEQUENCE)

*création\_de\_séquence* ::=

CREATE [ TEMPORARY | TEMP ]

SEQUENCE *nom\_de\_séquence* [ AS *type* ]

*générateur*

[ OWNED BY { *nom\_table.nom\_colonne* | NONE } ]

*générateur* ::=

[ INCREMENT [ BY ] *incrément* ]

[ MINVALUE *valeur\_min* | NO MINVALUE ]

[ MAXVALUE *valeur\_max* | NO MAXVALUE ]

[ START [ WITH ] *début* ]

[ CACHE *cache* ]

[ [ NO ] CYCLE ]

La clause OWNED BY est une extension PostgreSQL

Remarquer qu'un générateur peut se réduire à... rien du tout! Dans ce cas, un générateur implicite est utilisé.

2025-04-01

MCHD\_SQL\_07 — Clés (v1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'Informatique, Faculté des sciences, Université de Sherbrooke, Québec

26

26

## Clés internes en SQL ISO

## Solution 2 : Utilisation d'un générateur (SEQUENCE)

- Pour obtenir *implicitement* la prochaine valeur, lors d'une insertion, ne pas donner de valeur à l'attribut !
- Pour obtenir *explicitement* la prochaine valeur, utiliser la fonction standard **NEXT VALUE FOR**.

## Notes relatives à PostgreSQL

- Le mécanisme *implicite* est le même.
- Pour le mécanisme *explicite*, voire la diapositive suivante.

2025-04-01

MCHD\_SQL\_07 — Clés (V1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

27

27

## Clés internes en SQL ISO

## Solution 2 : Utilisation d'un générateur (SEQUENCE)... PostgreSQL

setval (regclass, bigint)	bigint	Définit la valeur courante de la séquence
setval (regclass, bigint, boolean)	bigint	Définit la valeur courante de la séquence et son statut (booléen)
currval (regclass)	bigint	Renvoie la valeur la plus récemment obtenue avec nextval pour la séquence indiquée
lastval ( )	bigint	Renvoie la valeur la plus récemment obtenue avec nextval (quelle que soit la séquence)
nextval (regclass)	bigint	Incrémente la séquence et renvoie la nouvelle valeur

Les objets sequence sont de type regclass  
voir <https://docs.postgresql.fr/15/functions-sequence.html>

2025-04-01

MCHD\_SQL\_07 — Clés (V1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

28

28

## SEQUENCE

### Remarque importante

- Pour éviter le blocage de transactions concurrentes qui obtiennent des nombres de la même séquence, une opération *nextval* n'est jamais annulée ; c'est-à-dire qu'une fois la valeur récupérée, elle est considérée utilisée, même si la transaction qui exécute *nextval* avorte par la suite. Cela signifie que les transactions annulées peuvent laisser des « trous » inutilisés dans la séquence des valeurs assignées.

2025-04-01

MCHD\_SQL\_07 — C16 (V1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

29

29

## SEQUENCE

### SERIAL - définition

- Le type SERIAL n'est pas un vrai type, mais plutôt un raccourci pour créer des valeurs uniques de type INTEGER :  

```
CREATE TABLE nom_de_table (
    nom_de_colonne SERIAL,
    ...
);
```
- est équivalent à écrire :  

```
CREATE SEQUENCE nom_de_table_nom_de_colonne_seq;
CREATE TABLE nom_de_table (
    nom_de_colonne INTEGER
    GENERATED AS DEFAULT AS IDENTITY nom_de_table_nom_de_colonne_seq,
    ...
);
ALTER SEQUENCE nom_de_table_nom_de_colonne_seq
OWNED BY nom_de_table.nom_de_colonne;
```
- Un attribut de type entier est ainsi créé dont la valeur par défaut est obtenue par un générateur de séquence propre à l'attribut.
- Enfin, la séquence est marquée OWNED BY (possédée par) l'attribut pour qu'elle soit automatiquement supprimée si l'attribut ou la table est supprimé.

2025-04-01

MCHD\_SQL\_07 — C16 (V1302) © 2018-2025, MCHS — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

30

30

## SEQUENCE SERIAL – un exemple en SQL ISO

Ceci

```
CREATE TABLE T (
    C SERIAL,
    ...
);
```

est équivalent à écrire :

```
CREATE SEQUENCE T_C_seq;
CREATE TABLE T (
    C INTEGER
      GENERATED AS DEFAULT AS IDENTITY T_C_seq,
    ...
);
ALTER SEQUENCE T_C_seq
  OWNED BY T.C;
```

2025-04-01

MCHD\_SQL\_07 – CIs (v1302) © 2018-2025, MCHS – CC BY-NC-SA 4.0  
Département d'Informatique, Faculté des sciences, Université de Sherbrooke, Québec

31

31

## SEQUENCE SERIAL – un exemple en PostgreSQL

Ceci

```
CREATE TABLE T (
    C SERIAL,
    ...
);
```

est équivalent à écrire :

```
CREATE SEQUENCE T_C_seq;
CREATE TABLE T (
    C INTEGER NOT NULL
      DEFAULT nextval('T_C_seq'),
    ...
);
ALTER SEQUENCE T_C_seq
  OWNED BY T.C;
```

2025-04-01

MCHD\_SQL\_07 – CIs (v1302) © 2018-2025, MCHS – CC BY-NC-SA 4.0  
Département d'Informatique, Faculté des sciences, Université de Sherbrooke, Québec

32

32



## SEQUENCE SERIAL – notes

- SMALLSERIAL, SERIAL et BIGSERIAL sont mis en oeuvre en utilisant des séquences correspondant respectivement à SMALLINT, INTEGER et BIGINT; ils en ont donc les caractéristiques.
- Dans la plupart des cas, une contrainte UNIQUE ou PRIMARY KEY est ajoutée pour interdire que des doublons soient créés par accident, mais ce n'est pas automatique.
- Pour affecter la valeur suivante de la séquence à un attribut, il faut préciser que la valeur par défaut de l'attribut doit être utilisée. Cela peut notamment se faire en excluant cet attribut de la liste des attributs de la commande INSERT.

2025-04-01

MCHD\_SQL\_07 – CIs (V1302) © 2018-2025, MCHS – CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

33

33

## SEQUENCE SERIAL – un exemple en PostgreSQL

Ceci est le plus souvent préférable

```
CREATE TABLE T (
  C SERIAL,
  ...
  PRIMARY KEY (C)
);
```

Ce qui est équivalent à :

```
CREATE SEQUENCE T_C_seq;
CREATE TABLE T (
  C INTEGER NOT NULL
    DEFAULT nextval('T_C_seq'),
  ...
  PRIMARY KEY (C)
);
ALTER SEQUENCE T_C_seq
  OWNED BY T.C;
```

2025-04-01

MCHD\_SQL\_07 – CIs (V1302) © 2018-2025, MCHS – CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

34

34

## SEQUENCE

### Les UID, GUID et autres UUID

- Hors norme SQL.
- Spécifiés par l'ISO, la CEI et l'IETF.
- Bien que le type UUID soit parfois disponible (comme PostgreSQL), les fonctions sont plus souvent disponibles par l'entremise de « packages » (ou « extensions »)
  - <https://docs.postgresql.fr/17/datatype-uuid.html>
  - <https://docs.postgresql.fr/17/functions-uuid.html>
  - <https://docs.postgresql.fr/17/uuid-openssl.html>

2025-04-01

MCPD\_SQL\_07 — C16 (V1302) © 2018-2025, MfTis — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

35

35

## Références

- Loney, Kevin ;  
*Oracle Database 11g: The Complete Reference*.  
Oracle Press/McGraw-Hill/Osborne, 2008.  
ISBN 978-0071598750.
- 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](http://docs.oracle.com/cd/E11882_01/index.htm)
- Le site de PostgreSQL (en français)
  - <http://docs.postgresqlfr.org>
  - <https://docs.postgresql.fr/17/uuid-openssl.html>

2025-04-01

MCPD\_SQL\_07 — C16 (V1302) © 2018-2025, MfTis — CC BY-NC-SA 4.0  
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec


40

40

2025-04-01

MCHD\_SQL\_07 — CIs (v1.30) © 2018-2025, M<sup>CH</sup>S — CC BY-NC-SA 4.0

Département d'Informatique, Faculté des sciences, Université de Sherbrooke, Québec



41