DB2, intégrité référentielle & compagnie

En juin 1988 j'assistai, dans le grand amphi de la tour Descartes à la Défense, à la présentation de la version 2 de DB2 (pour MVS[*]). A défaut d'une introduction aux domaines (ne rêvons tout de même pas…), nous eûmes droit à un exposé présentant (enfin) la mise en œuvre des clés primaires et étrangères, donc de l'intégrité référentielle : autant dire que je suivis l'exposé avec une extrême attention, afin de m'assurer qu'IBM avait suivi scrupuleusement les règles et les recommandations énoncées par le père du Modèle Relationnel de Données et par l'indéfectible Gardien du Phare (je veux dire Ted Codd et Chris Date). Je connaissais pas mal le sujet, pour avoir épluché et mis en pratique les travaux de ces géants, et j'avais complètement adhéré à leurs thèses, selon lesquelles :

a. Ces objets que les concepteurs de DB2 prétendaient être des tables n'étaient jusque là— du fait de l'absence de clés primaires— que des sacs à tuples (*tuple bags*)[†] .
b. La non prise en compte de l'intégrité référentielle conduisait à faire très fortement douter de la cohérence des bases de données, car très difficile à garantir de façon totale par des programmes ad-hoc [Date86a].

Lors de cette présentation du mois de juin, j'ai pu constater que, d'un point de vue relationnel, DB2 traitait correctement des clés primaires (à ceci près que ses pères ne voyaient pas d'inconvénient à ce que l'on ne les utilise pas…), mais ignorait le cas plus général des clés candidates (oubli réparé cinq ans plus tard, avec la version 3 du produit). J'étais quand même en l'occurrence content.

En revanche, je toussai quand l'orateur qui présentait les règles d'intégrité référentielle nous apprit qu'il était interdit de modifier la valeur d'une clé primaire d'une table T1, si cette valeur était en même temps une valeur de clé étrangère d'une table T2 connectée à T1 (RESTRICT autorisé, CASCADE et SET NULL interdits). Me souvenant alors que Ted Codd et Chris Date avaient prévu toutes les possibilités, je ne pus m'empêcher d'interpeller l'orateur : « Pourquoi n'autorise-t-on que la seule règle RESTRICT ? » et il me fut répondu de façon péremptoire : « Parce que les théoriciens en ont décidé ainsi »… L'amphi étant bondé, je ne voulus pas faire de scandale et préférai garder pour moi mes états d'âme (au fait, pensions-nous aux mêmes théoriciens ?).

Ce n'est pas que je sois forcément d'accord avec le fait que l'on puisse modifier à tout bout de champ des clés primaires, car il y a de quoi mettre sur les genoux un ordinateur, aussi puissant soit-il (imaginez que vous ayez affaire à une table E décrivant deux millions d'entreprises et ayant pour clé primaire le numéro de SIREN de ces entreprises : sachant que tous les mois l'INSEE vous envoie des monceaux de correctifs concernant ces numéros, sachant encore que cette table E est une référence pour une quinzaine d'autres tables dont la clé primaire peut aussi comporter ce même numéro de SIREN et qu'à leur tour ces tables peuvent être des références, on peut facilement douter de la performance d'un traitement de correction des numéros de SIREN…). Mais quand même ! En passant, j'ai lu plus tard chez Ted Codd que, pour lui, le SGBD est en meilleure position que quiconque pour assurer correctement la modification des clés étrangères [Codd90].

Dix ans plus tard, je me défoule à l'occasion sur mon micro quand je mets en œuvre ma base de données de mes chansons préférées, en faisant faire à ACCESS ce que DB2 ne sait pas faire (ils ne jouent pas dans la même cour il est vrai). Ceci dit, je ne sais toujours pas pourquoi DB2 a été bridé. Mon imagination me pousse à penser qu'IBM a agi en fonction de critères tels que ceux-ci :

- Reprise vraisemblablement de ce qui avait été déjà été fait pour les tables du catalogue relationnel et pour lesquelles il me semble que la seule règle de modification soit RESTRICT.

[*] Et pour cause, DB2 n'existant pas alors sur les autres plates-formes IBM.

[†] C'est-à-dire capables de contenir des doublons (la recommandation d'IBM étant, dans les versions précédentes, de résoudre au niveau physique le problème de l'unicité des clés, au moyen d'index *uniques*, quelle salade !)

- Coût de l'opération : modifier une clé primaire est plus coûteux que la supprimer (multiplication par deux des opérations : supprimer puis insérer, tant au niveau table qu'au niveau index). A l'utilisateur de se débrouiller.

- Complexité de l'opération : si, par exemple, le système commence par modifier les valeurs des clés étrangères, alors l'intégrité référentielle est enfreinte, ce qui signifie qu'il doit mettre en œuvre, pour son propre compte un mécanisme de contrôle non pas immédiat mais différé (au COMMIT). Voir à ce sujet [Date86b], page 56). A l'utilisateur de se débrouiller.

Si quelqu'un a des informations objectives, je suis preneur. En tout état de cause, j'ai intégré à mon bêtisier personnel ce qui suit, concernant l'interdiction qui nous est faite de modifier en cascade.

Dans [IBM88] page 48, il est écrit :	Que l'on peut traduire ainsi :
« The restriction is that the CASCADE and SET NULL options on the UPDATE rule are not supported… The reason for this restriction is philosophical. Some argue that by its very definition, a primary key does not change. Once a primary key value is assigned, it remains the same for the lifetime of the entity occurrence. Therefore, the application of the UPDATE CASCADE rule or UPDATE SET NULL rule does not make sense. However, if the application requires an update of a primary key value, then the recommendation is to delete the entity occurrence and all its dependants, then re-insert them with the new value of the primary key. DB2 permits UPDATE RESTRICT so that data entry type of mistakes can be corrected. »	« La restriction consiste en ce que les options CASCADE et SET NULL de la règle UPDATE ne sont pas admises… La raison en est philosophique. D'aucuns avancent que, de par sa définition même, une clé primaire ne varie pas. Une fois attribuée, une valeur de clé primaire reste la même durant la vie de l'occurrence d'entité. En conséquence, l'utilisation des règles CASCADE et SET NULL est sans objet. Cela dit, si l'on a besoin de changer la valeur d'une clé primaire, alors on supprimera l'occurrence d'entité ainsi que celles qui en dépendent, puis on les réinsérera avec la nouvelle valeur de clé primaire. DB2 autorise l'UPDATE RESTRICT pour que l'on puisse corriger les erreurs de saisie. »
Page 50 :	
« …The value of a primary key may be changed using the UPDATE statement as long as it is not a parent row… »	« …Tant qu'elle ne constitue pas une ligne parente, on peut modifier la valeur d'une clé primaire à l'aide de l'instruction UPDATE… » (A noter la confusion entre ce qu'est une valeur de clé et une ligne…)

Avec les recommandations d'usage pour modifier une clé primaire référencée par une clé étrangère :

a. « Change the foreign keys of all dependant rows of this primary key to NULL (or another value). »	(« Changer en NULL (ou en une autre valeur) les clés étrangères dans toutes les lignes dépendant de cette clé primaire. »
b. « Update the primary key of this row. »	« Modifier la clé primaire de cette ligne. »
c. « Reconnect the foreign keys that where changed to NULL to the new value of the primary key… »	« Reconnecter sur la nouvelle valeur de clé primaire les clés étrangères qui avaient été changées en NULL… »

Je ferai en l'occurrence quelques commentaires. Supposons qu'une table T1 joue le rôle de table « mère » (référencée) et T2 celui de table « fille » (référençante).

- ✓ Espérons que, pour T2, l'attribut (ou les attributs) composant la clé étrangère dont la valeur doit aussi changer, n'entre(nt) pas dans la composition de la clé primaire de T2.
- ✓ Pour quitter le monde de la philosophie, il est dommage de doubler la charge de travail de mise à jour du SGBD si l'on « nullifie » pour revaloriser par la suite (temps CPU, or *Time is money*…).

- ✓ Pourquoi n'autoriser la « correction des erreurs » que dans le seul cas où —au moment de la modification— la valeur de la clé primaire de T1 n'est effectivement présente dans aucune clé étrangère ?
- ✓ Bref, pourquoi invoquer des **raisons philosophiques** —hors de propos— à l'occasion de la conception de DB2 ? La possibilité de la modification d'une valeur de clé primaire devrait toujours être autorisée ; c'est à nous les concepteurs et administrateurs des bases de données d'apprécier le bien fondé d'une stratégie de mise à jour. Nous ne sommes quand même pas inconscients et nous savons prendre nos responsabilités…

Il est intéressant de noter ce qui est écrit (en 1997) de façon laconique, sans explication quelconque, en ce qui concerne DB2 V5 ([DB2V5] paragraphe 2.3.1.4.4)

> « *UPDATE Rules*
>
> > *For Parent Tables: You cannot change a parent key column of a row that has a dependent row. If you do, the dependent row no longer satisfies the referential constraint, so the operation is prohibited…* »

Références

[Codd90] E. F. Codd. *The Relational Model for Database Management: Version 2* (Reading, Mass.: Addison-Wesley, 1990).

On peut lire, à l'occasion de la description de la règle RB-33 relative à la modification des clés primaires avec modification en cascade des clés étrangères (page 91) : *« …the DBMS is in a better position than any user to handle correctly the updating of* all *the matching foreign-key values…* »

[Date86a] C.J. Date. *An Introduction to Database Systems: Volume I, 4th edition*. (Reading, Mass.: Addison-Wesley. 1986).

Pour la petite histoire, Chris Date avait un jour fait le commentaire qui suit, concernant les tables du catalogue de DB2 (page 325) : « *And if (for example) the user deletes a record from SYSTABLES (by issuing a DROP TABLE operation), then the system will automatically delete the matching records in SYSCOLUMNS also (in other words, the system supports a cascade delete rule with respect to the foreign key SYSCOLUMNS.TBNAME). Given the fact that such support must exists at the internal level of the system, it is hard to understand why the system designers did not see fit to support the same function at the external level also…* »

Je traduis : « *…Et si (par exemple) l'utilisateur supprime un enregistrement de la SYSTABLES (du fait d'une commande DROP TABLE), alors le système va automatiquement supprimer les enregistrements correspondants dans la SYSCOLUMNS (autrement dit, le système met en œuvre une règle de suppression en cascade en conformité avec la clé étrangère SYSCOLUMNS.TBNAME). Étant donné que cette fonctionnalité existe nécessairement au niveau interne du système, il est difficile de comprendre pourquoi les concepteurs du système n'ont pas cru bon d'en faire autant au niveau externe…* »

[Date86b] C.J. Date. *Relational Database, Selected Writings First Edition* (Reading, Mass.: Addison Wesley, 1986).

[DB2V5] DB2 for OS/390 Version 5 - Administration Guide Document Number SC26-8957-00 ; First Edition (June 1997)

[IBM88] IBM Database 2 Referential Integrity Usage Guide – Document number GG24-3312-0 – June 1988.