8. NUL: valeur non définie

Dans ce tout petit chapitre, nous discutons d'un détail, néanmoins important dans certains cas: ce que veut dire et ce qu'implique le fait qu'un attribut ne soit pas renseigné, qu'il possède ce qu'on appelle la valeur **NUL**.

NUL, nous l'écrivons exprès en majuscules et il reste toujours au masculin singulier, n'est pas l'équivalent du chiffre zéro. NUL signifie ici qu'aucune valeur ne figure dans la cellule correspondant à un certain attribut pour un certain objet.

Plusieurs difficultés sont liées à la notion de la valeur NUL.

Cette notion est ambigüe: NUL peut vouloir dire plusieurs choses (les exemples suivent):

a. Aucune valeur n'existe pour cet attribut dans le cas de cet objet, la cellule reste vide.

b. Cet attribut n'a pas de sens pour cet objet, la cellule doit rester vide.

c. La valeur de cet attribut pour cet objet est inconnue, elle manque.

Les SGBD existants ne nous laissent pas le choix: pour exprimer que la valeur d'un attribut est NUL, il faut laisser la cellule correspondante **vide**. Ce qui n'est ni équivalent à des espaces dans le cas de champs alphanumériques ni à 0 dans celui de champs numériques. On peut donc laisser un champ vide et tester ensuite avec un langage tel que SQL si ce champ "IS NUL" ou "IS NOT NUL". On peut aussi charger explicitement une valeur NUL dans une cellule: SET ADRESSE2 TO NUL.

Le langage de définition des données contenu dans SQL nous permet également de spécifier, lors de la définition d'une table, qu'un attribut ne doit pas être NUL. Ceci par exemple dans le cas d'un attribut clé primaire, une clé primaire devant toujours posséder une valeur définie univoque. Par contre, dans le cas de clés primaires composées, la valeur d'une partie (mais pas de l'ensemble!) des attributs composant cette clé peut être non-définie. Si la définition d'un attribut d'une table est suivie de la clause NOT NUL, le SGBD refusera d'enregistrer un objet pour lequel cet attribut précis ne reçoit pas de valeur.

Par contre la définition de requêtes sur des tables avec ce même langage produira en cas de présence de valeurs NUL des résultats incompréhensibles au premier abord lorsque l'on spécifie des conditions de sélection WHERE portant sur la valeur d'attributs restant vides. Pour se mettre à l'abri de surprises, il faut donc bien comprendre la manière dont SQL traite les valeurs NUL, mais cet aspect dépasse évidemment le cadre de ce livre.

Exemple portant sur le cas a) ci-dessus:

Personne (#-pers, nom, prénom, adresse 1, adresse 2, code postal, localité,
téléphone, fax, e-mail)

Personne

#-pers	Nom	Prénom	Adresse 1	Adresse 2	C.P.	Localité	T	F	Mail
100	Dupond	Fred	Gare 3		2233	Bourg	031…		fd@brg
101	Durand	Fanny	r. du Pont 5	Case 127	1715	Aix	022…		
102	Despland	Françoise	Ch. Port 17		1822	Bled	051…	051…	df@bld
103	Dutoit	Fernand	Lisière 2	Appart. 2b	3600	Fonds	047…	048…	fdu@yx
104	Dupont	Flavia	Forêt 3		7300	Crêt	082…		

Les cellules vides indiquent donc simplement qu'une personne ne possède pas de complément d'adresse, de fax ou d'adresse e-mail, ce qui est parfaitement légitime.

Concentrons-nous à présent sur le cas b) ci-dessus.

Personne (#-pers, nom, prénom, sexe, code-barbe, nombre-grossesses)

Personne

#-pers	Nom	Prénom	Sexe	Barbe	Grossesses
100	Dupond	Fred	m	oui	
101	Durand	Fanny	f		0
102	Despland	Françoise	f		0
103	Dutoit	Fernand	m	non	
104	Dupont	Flavia	f		2

Dans cet exemple, un peu tiré par les poils de la barbe, admettons-le, l'attribut code-*barbe* n'est pas défini pour les femmes et l'attribut *nombre-grossesses* ne l'est pas pour les hommes. Nous avons en fait deux sous-ensembles d'objets qui doivent être décrits de manière différente. Les puristes diront qu'une telle situation est inadmissible et qu'il faut créer deux ou même trois tables. Dans le premier cas, une table pour les hommes et une autre pour les femmes, mais cette solution ne satisfait pas d'un point de vue pratique si l'on veut dans la majorité des cas gérer l'ensemble des personnes sans distinction des sexes.

La seconde solution se présente comme suit:

Personne (#-pers, nom, prénom, sexe)

Barbe (*#-pers*, code-barbe)

Grossesses (*#-pers*, nombre-grossesses)

NB: dans la deuxième et la troisième table, l'attribut *#-pers* est à la fois clé primaire et clé étrangère.

La deuxième table ne contiendra évidemment que des hommes et la troisième que des femmes.

On pourrait d'ailleurs ne faire figurer dans la deuxième table que les hommes ayant une barbe, l'attribut *code-barbe* deviendrait alors inutile. De même on pourrait ne faire figurer dans la troisième table que les femmes ayant eu un enfant. Économie de place dans les deux cas. Par contre, la programmation devient plus complexe dans les cas où il faut effectivement gérer le nombre de grossesses d'une femme ou le fait qu'un homme soit barbu.