BDD Tuto 9 : Requêtes imbriquées et calcul entre champs

Diapo 1

Dans un tableur, on peut faire des calculs et utiliser des fonctions avec les valeurs qui sont dans les tables.

Diapo 2

Est-ce qu'on peut faire pareil avec un logiciel de base de donnée ? Eh bien oui, et pour cela il faut rentrer les calculs dans les requêtes.
On peut écrire des calculs avec les signes + - * /, et aussi utiliser des fonctions, un peu comme on a vu en tableur, c'est-à-dire des fonctions de date, de chaîne de caractère, des fonctions conditionnelles etc

Diapo 3

Dans cette vidéo et les précédentes, j'ai fait le choix d'utiliser LibreOffice pour aborder la base de donnée, mais ce logiciel a le défaut de ne pas être très facile pour les requêtes. Access serait plus simple d'utilisation car il comporte des assistants pour écrire les formules, mais quoiqu'il arrive les interfaces graphiques ont leur limite. Quel que soit le logiciel choisi, les requêtes sont traduites dans un langage qu'on appelle le SQL, c'est ce langage qui est utilisé pour une utilisation plus professionnelle. Les requêtes que je vais construire seraient plus facile à écrire directement avec ce langage, mais ceci nécessiterait quelques vidéos supplémentaires.

Diapo 4

Dans notre exemple, on avait laissé le tableau des animateurs avec les dates de naissances. Comme les âges seraient plus faciles à lire pour eux, on va mettre en place le calcul.
On va devoir pour cela mettre en œuvre deux fonctions : Une première qui va donner la date du jour de l'ordinateur. Elle s'écrit CURDATE et elle n'a pas d'argument, donc, comme en tableur, on écrit des parenthèses vides.
La deuxième fonction à utiliser sera DATEDIFF. Elle permet de faire la différence entre deux dates. On écrit son nom, et après la parenthèse, on écrit le mode de la fonction. Comme on veut qu'elle calcule en se basant sur des années, on va écrire 'yy'. Ensuite on place le premier opérateur de notre soustraction, ici c'est la date de naissance de l'enfant, et j'ai nommé la colonne Date_naissance. Il faut l'écrire entre crochets, ou entre guillemets double.
En dernier on place le deuxième opérateur, c'est la date courante qui est renvoyé par la fonction CURDATE vue plus haut.

Manip [2:14]

Sur notre base de donnée on édite la requête pour écrire cette formule dans une nouvelle colonne. On pourra ensuite masquer le champ date de naissance

Et voilà le résultat.

Diapo 5 [2:45]

Une dernière modification est demandée: imaginons qu'on doit faire payer chaque semaine aux parents, en plus du coût des activités, des frais liés à la présence de leur enfant dans le centre. Le montant de ces frais de pension peut varier d'un parent à l'autre, en fonction des revenus, notamment.
On aimerait donc avoir deux colonnes en plus dans la requête qu'on a créée dans la vidéo précédente, qui présente pour chaque parent les frais de pension, les coûts des activités et le total.

Comme le montant des frais varient selon les parents, on va devoir ajouter une colonne à la table parent. Attention ! Dans ce type de situation ou on modifie une table, il faut bien veiller à faire toutes les modifications qui en découlent. Il faudra mettre à jour le formulaire pour qu'il corresponde.

Diapo 5 [3:34]

Habituellement les requêtes sont conçues pour aller chercher des données qui sont dans les tables. Le résultat d'une requête peut parfaitement être exploité par une autre requête, comme si c'était un simple table. On parle alors de requête imbriqué. Ce procédé nous permet de faciliter le découpage en étape d'une requête qui serait compliquée à construire d'emblée.
Le résultat de la requête qu'on a laissée ne présente que les parents dont les enfants ont participé à des activités

Diapo 6 [3:57]

Or, il nous faut un tableau avec tous les parents. Ce genre de problème, on l'a déjà vu et on a pu le résoudre en modifiant le type de jointure, mais ici la complexité liée au regroupement dans notre requête ne le permet pas directement. On va donc utiliser une requête « principale » qui ira chercher des données dans la table parents, mais aussi dans notre première requête. On pourra alors ajouter et configurer la jointure, à condition d'avoir des colonnes clés pour lier la table et la requête, il faudra donc la colonne des matricules dans notre première requête
Le reste du travail va consister à faire le calcul du montant total en ajoutant le coût d'activité et le coût d'inscription. Comme rien n'est simple, le logiciel ne voudra pas faire le calcul avec des cellules qui sont vides, on doit écrire des 0 aux endroits ou il n'y a pas de montant.

Diapo 6 [4:52]

Pour cela, il faut utiliser la fonction conditionnelle CASE WHEN. Elle permet d'évaluer une condition et d'effectuer une instruction si elle est vraie. Si la condition est fausse, c'est l'instruction 2 qui est exécutée. Ici, si le contenu de la colonne coût activité est vide (on écrit IS NULL), alors (on écrit THEN) il faut écrire 0, sinon (on écrit ELSE) il faut écrire le montant qui est dans la colonne coût activité et on termine par un END
Il nous restera à écrire le calcul du total en additionnant les colonnes.

Manip – modif table et formulaire [5:36]

Première étape, on modifie la table parent pour ajouter le coût de la pension. Une fois que la colonne a été ajoutée, il faut mettre à jour le formulaire pour qu'il corresponde. Il faut faire un

copier coller de l’un des champs, puis modifier l’étiquette et la source de donnée. On doit dissocier le groupe, puis changer le texte de l’étiquette et la source de données du champ de texte
J’enregistre mon formulaire et j’en profite pour saisir des données d’essai dans la nouvelle colonne.

Manip – ajout de deux colonnes à la requête des montants [6:43]

À présent appliquons les modifications nécessaires à la requête de la dernière fois.
On doit y ajouter le coût de la pension et le matricule. Dans cette requête, on a fait un regroupement des NomPrenom de parent, c’est donc qu’on doit utiliser aussi une fonction pour les coûts de pension et les matricules. Il ne faut pas en faire la somme, ni les compter, mais bien les regrouper.

Manip – création d’une requête qui utilise la requête précédente [7:17]

Enfin, on crée notre nouvelle requête, en ajoutant la table parent, puis la requête qu’on vient de modifier. On établi ensuite le lien d’intégrité entre le matricule du parent de la table et celui qui est dans la requête. On configure ensuite le type de jointure pour inclure tous les parents, et on vérifie le résultat avec l’aperçu. On enregistre la requête que je vais appeler "tableau parent avec total coût"

Manip – utilisation de la fonction CASE WHEN pour ajouter des zéros à la place des cases vides [7:58]

Pour ajouter des zéros à la place des cases vides, on écrit notre formule ici, attention aux erreurs de syntaxe, (on ne le dira jamais assez.) Je vérifie mon résultat avec l’aperçu
Le calcul de la somme peut se faire directement dans cette colonne, il suffit d’ajouter coût_pension comme ceci. J’écris enfin l’alias «total activité et pension »

Diapo 7 [8 :35]

Et voilà, c’est terminé. Je ne vous cache pas que les requêtes sont parfois compliquées à construire, et souvent, elles mettent en évidence des défauts de conception d’une base de donnée. Pour s’exercer, il faut se limiter à des exemples simples, car assez vite on se retrouve dans la situation ou il faut faire appelle au langage SQL ou à de la programmation pour faire ce qu’on veut. Les exemples sont multiples, mais tout n’est pas réalisable à votre niveau.

Diapo 9 [9:00]

En guise d’entraînement, essayer de reproduire les requêtes que j’ai faites ici, et par la suite essayer de créer les requêtes que j’ai placées dans l’espace des exercices. Certaines sont très simples et d’autres vont demander plus de réflexion. Bref il y en a pour tous les goûts
Bonne chance et, à bientôt