Bases de Données TD4

Définition et Manipulation de Données en SQL et QBE

1. Définir (sur papier) les tables de la base S-P avec leurs clés. D'abord, en utilisant la commande Create Table du cours. Puis, essayez la syntaxe et les types réellement disponibles pour la commande Create Table de MsAccess. A découvrir dans l'aide de MsAccess en l'interrogeant avec les mots clés « Create Table », puis avec « Data type SQL ». Supposer que la base est créée, car l'instruction Create Database n'existe pas sous SQL standard (incroyable, mais vrai). Ensuite, restructurez S en y ajoutant l'attribut la photo disons du siège de la société et émail. Utilisez d'abord les types de l'interface graphique (QBE). Puis, tentez d'employer les types réellement disponibles sous MsAccess qui sont les mêmes que pour Create Table. Commentez le résultat. Enfin, supprimez toutes les tables.
2. Définir les tables de la base S-P en utilisant cette fois-ci l'interface graphique (ex. pratique). Définir les liens sémantiques dans S-P en utilisant la fenêtre Relations. Transformer chaque lien en contrainte d'intégrité référentielle, avec la propagation en cascade des mises à jour de la clé.
3. Faire les mêmes opérations pour la base Ecole de Conduite. Créer aussi un petit jeu d'essais. Pour rappel, on a les tables :

   Cl (Cl#, Nom, Prenom, Adr, D_Naiss)
   S_Code (Se#, D_sess, Heure, Cd#, S#)
   A (Cl#, Se#, N_Fautes)
   E (PC#, D_pref, Heure, Lieu)
   E-R (PC#, Cl#, N_Fautes)
   CD (Cd#, Edit)
   S (Cd#, S#)
   SRQ (Cd#it d, S#, Q#, P#)
   Q (Q#, Int, Rep, Diff, Theme)

A noter que le mot Date simplement pour tout nom d'attribut désignant une date dans les tables ci-dessus est possible sous MsAccess. Mais c'est aussi le nom réservé deMsAccess. Ce qui peut créer des problèmes.

4. Définir dans le langage de définition de l'interface graphique de MsAccess les contraintes d'intégrité suivantes, en sachant que les opérateurs de comparaison sont comme suit (< , >, <=, >=, =, Entre, <>) :

   a. Dans la définition de A : formuler de deux manières, au moins que le nombre de fautes est entre 0 et 40. Dire dans quel endroit on fait cette déclaration. Ajouter (où ?) le message « Nombre de fautes incorrect » si la contrainte est violée.

   b. Pour la table Q : formuler de deux manières la contrainte : Si la difficulté est plus que 10, alors le Thème est « Conduite de Rallye ». Avec un message « La difficulté ou le Thème erronés ». Les opérateurs logiques disponibles sont (Eqv, Et, Imp, Ou, Ou_x, Pas).

   c. Pour Q aussi formuler la contrainte: Si la difficulté est 5 ou plus, alors la réponse est nécessaire.
5. Définir pour la table Cl l'indexe ascendant sur le nom comme 1èr critère, suivi de prénom.
6. Exprimer en SQL standard ou en SQL de MsAccess et en QBE, si possible, les manipulations suivantes :
   1. Le numéro de client, le nom et le prénom en ordre ascendant sur le nom et le prénom, pour au plus 10 clients, nées avant 2002.
   2. Tous les éditeurs de CDs dont le nom d'éditeur commence par B, un tuple par éditeur.
   3. Noms de clients ayant le nombre de fautes inconnu pendant une de session, avec l'identification de cette session.
   4. Tout numéro et éditeur d'un CD, avec le numéro de toute session où ce CD a été appliqué, s'il y avait une telle session. Puis, une même requête avec la condition supplémentaire que S# ne doit pas être 10.
   5. Les couples de CD# ayant un même éditeur (sans doublons en tout genre)
   6. Noms de clients ayant suivi (i) la session '1' ou la session '2' ; (ii) la session '1' et la session '2' ; (iii) la session '2' sans avoir suivi la session '1'.
   7. Noms de clients ayant suivi toutes les sessions.
   8. Noms de clients n'ayant suivi aucune session.
   9. Tout nom de client ayant atteint le minimum de fautes connu, avec l'ID de la session de sa prouesse.
   10. Nombre total de fautes par client durant les examens, ainsi que sa variance, la partie entière de la racine carrée de celle-ci, et Cl#.
   11. La moyenne de fautes par client durant les sessions, avec le nom du client et la condition que cette moyenne est au-dessus de 3.
   12. Dates suivant de deux semaines celles des examens dans la table E, avec l'ID de chaque examen. Libellées comme dates d'affichage de résultats.
   13. Tout sur les exams dans un lieu donnée en paramètre.
   14. Ajouter l'info que le client 123, a fait 5 fautes dans la session 456.

15. Modifier l'heure de tout examen à Paris pour 9h, puis supprimer tout client à nombre de fautes maximal.
16. En supposant que dans la base Ecole de Conduite tout éditeur possède la page Web et que la table CD contient l'attribut Page_Web trouver sous MsAccess 2007 en SQL tous les CD# de http://www.dupont.fr/ .
17. Exprimer en SQL (Standard ou MsAcces) les requêtes en algèbre relationnelle à la base S-P vues en TDs.
18. Pour cette base, on cherche en SQL ou QBE le nom, la ville et le statut de tout fournisseur dont la ville est Paris ou Londres et dont le statut est 100.
19. Idem, mais la ville est soit Paris soit Londres.
20. Idem, mais la ville qui est soit Paris soit Londres implique que le statut est 100
21. Est-ce que les couples de requêtes suivantes sont équivalentes ? Sinon montrer et expliquer leurs résultats.
    Select * from S where status = 100 or status = 200 et Select * from S where status = 100 or 200
    SELECT * FROM S WHERE city = "london" xor city = "paris" imp status = 100 et
    SELECT * FROM S WHERE city = "london" xor (city = "paris" imp status = 100)
    SELECT * FROM S WHERE city = "london" or city = "paris" imp status = 100 et
    SELECT * FROM S WHERE city = "london" or (city = "paris" imp status = 100)
22. En supposant que le statut de Blake est inconnu, quels seraient les résultats de requêtes:
    Select * from S where status = True et Select * from S where status = False

7. Exprimer en <u>une</u> requête SQL de MsAccess ou en QBE, si possible, chaque manipulation *avancée* suivante:

1. Le nombre total d'éditeurs, en sachant qu'un éditeur peut éditer plusieurs CDs.
2. Nombre total de fautes par client durant les sessions, ensemble avec les fautes de chaque session.
3. La moyenne de fautes par client, pendant les sessions et les examens confondus.
4. Pour tout client, sa moyenne de fautes en sessions et la moyenne de fautes des autres clients.
5. La covariance/corrélation entre le nombre de fautes des exams au même lieu à des dates différentes.
6. Pour chaque fourniture dans la base S-P, indiquer le S#, P# comme Product ID, QTY, Quantité totale pour ce S# comme Total, enfin, l'Importance relative de cette quantité pour le fournisseur. Notez qu'il s'agit du risque de perte de revenu pour le fournisseur s'il perdait le client pour ce produit.
7. Calculez d'une manière similaire tous les risques pour le client s'il perdait chacun de fournisseurs d'un produit.
8. Le gestionnaire de S-P veut catégoriser chaque fournisseur selon le statut. Montrez dans chaque tuple S# et STATUS, puis OK si Statut < 30 et GOOD autrement. Ensuite, dans le même tuple, montrez une catégorisation plus détaillée: répétez Statut < 30 comme OK, mais seulement celui entre 30 et 100 devient GOOD, au-dessus c'est V-GOOD.
9. Le propriétaire de l'école emprunte de temps en temps. Formulez la requête qui lui permettrait d'indiquer la somme empruntée, p.ex. 1000€, le taux, p.ex. 0,04, le nombre d'années pour rembourser, p.ex. 7, et trouver l'annuité, la somme qu'il aurait payé et la surprime. Dans notre exemple il faudrait générer la table suivante:

Annuitée	taux_annuel	nbre_années	valeur présente	valeur_payée	surprime
-167	0,04	7	1000	-1169	-169

10. Un chargé de portefeuille garde pour chaque client l'intitulé du placement, la somme placée, le taux annuel fixe, et la date du placement. Aidez-le à calculer pour chaque client la valeur à jour du placement, avec la précision d'une année.
11. Aidez-le à calculer pour chaque client la valeur à jour du portefeuille.
12. Certains placements sont à taux variable. Pour un placement on a un tuple par l'année avec le taux de cette année. Calculez la valeur après *n* années, où *n* est un paramètre. Observer qu'il s'agit de réaliser ici la fonction agrégat que l'on **pourrait** appeler PRODUCT qui n'existe pas en standard sous les SGBDs actuels.
13. Dans la base S-P : la quantité moyenne de pièces fournies par fournisseur fournissant au moins une pièce.
14. la base S-P : la quantité moyenne de pièces fournies par fournisseur dans la base.
15. Les requêtes suivantes sont elles légales dans la base S-P? Si oui leur résultats svp.: Select 5+5 ; Select 5.0+5 ; Select 5-5 ; Select 5 – (-5) ; Select 5+nul ; select sum(qty), count(*), count (QTY) from SP ; en supposant que QTY pour la fourniture (S1,P1) est nul et seul à l'être. Explications et conclusions globales ?

14. Est-ce que Select * from S est équivalent à Select * from S where city = « Paris » union Select * from S where city <> « Paris » ?
15. Choisir un échantillon aléatoire de trois fournisseurs dans la table S de la base du cours.
16. Ecrire de mémoire toute requête du cours SQL Avancé. Ou mieux : les ré-exécuter sous MsAccess.
17. Une proposition pour une requête SQL calculant la valeur de X!, quel que soit le choix de X entre 1 et 30 ?

1. Définir (sur papier) les tables de la base S-P avec leurs clés. D'abord, en utilisant la commande Create Table du cours. Puis, essayez la syntaxe et les types réellement disponibles pour la commande Create Table de MsAccess. A découvrir dans l'aide de MsAccess en l'interrogeant avec les mots clés « Create Table », puis avec « Data type SQL ». Enfin, utilisez l'interface graphique (QBE). Supposer que la base est créée, car l'instruction Create Database n'existe pas sous SQL standard (incroyable, mais vrai). Ensuite, restructurez S en y ajoutant l'attribut la photo disons du siège de la société et émail. Utilisez d'abord les types de l'interface graphique (QBE). Puis, tentez d'employer les types réellement disponibles sous MsAccess qui sont les mêmes que pour Create Table. Commentez le résultat. Enfin, supprimez toutes les tables.

E- Création avec la syntaxe du cours :

```
CREATE TABLE S
(S#   CHAR (5)  NOT NULL,
SNAME        CHAR (20),
STATUS       INT,
CITY         CHAR (15),
PRIMARY KEY (S#) ) ;

CREATE TABLE P
(P#   CHAR (5)  NOT NULL,
PNAME  CHAR (20),
WEIGTH  REAL,
CITYCHAR (15),
PRIMARY KEY (S#) ) ;

CREATE TABLE SP
(S#   CHAR (5)  NOT NULL,
P#   CHAR (5)  NOT NULL,
QTY INT,
PRIMARY KEY ((S#), (P#)) ;
```

La syntaxe de MsAccess (par example):

```
create table S
([S#] text (5) constraint Cle primary key not null,
sname  memo,
status short,
city text (15)) ;

create table P
([P#] text (5) constraint Cle primary key not null,
pname  text (20),
color text (15),
weight  real,
city text (15)) ;

create table SP
([S#] text (5) not null,
[p#] text (5) not null,
Qty  short,
constraint  SPcle primary key ([s#], [p#]) );
```

b. Alteration en utilisant les types de données de l'interface graphique:

ALTER TABLE S ADD PHOTO Objet OLE ; ALTER TABLE S ADD Email Lien Hypertexte ;

En reel:

```
alter table S add  photo oleobject;  alter table S add  email binary;
```

Le type binaire n'est pas lien hypertexte le plus utile pour émail. Ce dernier ne semble pas possible pour Alter Table. Il faut changer en utilisant l'interface graphique. Chose amusante : on ne peut plus revenir au type binaire sous l'interface graphique, une fois le changement fait.

Suppression : comme dans le cours.

2. Définir les tables de la base S-P en utilisant cette fois-ci l'interface graphique (ex. pratique). Définir les liens sémantiques dans S-P en utilisant la fenêtre Relations. Transformer chaque lien en contrainte d'intégrité référentielle, avec la propagation en cascade des mises à jour de la clé.

A faire en pratique, en créant d'abord la base S-P comme on a montré en cours.

3. Faire les mêmes opérations pour la base Ecole de Conduite. Créer aussi un petit jeu d'essais. Pour rappel, on a les tables :

Cl (Cl#, Nom, Prenom, Adr, D_Naiss)
S_Code (Se#, Date, Heure, Cd#, S#)
A (Cl#, Se#, N_Fautes)
E (PC#, Date, Heure, Lieu)
E-R (PC#, Cl#, N_Fautes)
CD (Cd#, Edit)
S (Cd#, S#)
SRQ (Cd#, S#, Q#, P#)
Q (Q#, Int, Rep, Diff, Theme)

Même canevas que l'exercice 1. Avec le type NumeroAuto en plus à apprendre pour les clés. Les volontaires peuvent expérimenter aussi la définition de clés étrangères par l'interface SQL de MsAccess :

CONSTRAINT nom {PRIMARY KEY | UNIQUE | NOT NULL |
REFERENCES tableétrangère [(champétranger1, champétranger2)]
[ON UPDATE CASCADE | SET NULL]
[ON DELETE CASCADE | SET NULL]}

Contrainte sur plusieurs champs :

CONSTRAINT nom
{PRIMARY KEY (primaire1[, primaire2 [, …]]) |
UNIQUE (unique1[, unique2 [, …]]) |
NOT NULL (non_null1[, non_null2 [, …]]) |
FOREIGN KEY [NO INDEX] (réf1[, réf2 [, …]]) REFERENCES tableétrangère [(champétranger1 [, champétranger2 [, …]])]
[ON UPDATE CASCADE | SET NULL]
[ON DELETE CASCADE | SET NULL]}

4. Définir dans le langage de définition de l'interface graphique de MsAccess les contraintes d'intégrité suivantes, en sachant que les opérateurs de comparaison sont comme suit (< , >, <=, >=, =, Entre, <>) :

a. Dans la définition de A : formuler de deux manières, au moins que le nombre de fautes est entre 0 et 40. Dire dans quel endroit on fait cette déclaration. Ajouter (où ?) le message « Nombre de fautes incorrect » si la contrainte est violée.
Il faut pointé l'attribut N_Fautes. Puis, on écris dans la ligne « Valide si » l'une des expressions suivantes, par exemple :

>= 0 et <= 40 ; Entre 0 et 40 ;

Enfin on met le message dans la ligne « Message si erreur ».

b. Pour la table Q : formuler de deux manières la contrainte : Si la difficulté est plus que 10, alors le Thème est « Conduite de Rallye ». Avec un message « La difficulté ou le Thème erronés ». Les opérateurs logiques disponibles sont (Eqv, Et, Imp, Ou, Ou_x, Pas).

Il faut utiliser les lignes "Valide Si" et "Message si erreur" de la feuille de propriétés de la table Q.
Diff > 10 Imp Theme = "Conduite de Rallye"
Diff <= 10 Et Theme = "Conduite de Rallye"

c. Pour Q aussi formuler la contrainte: Si la difficulté est 5 ou plus, alors la réponse est nécessaire.
Diff >= 5 Imp (Rep Est Pas Null Et Rep <>"")

5. Définir pour la table Cl l'indexe ascendant sur le nom comme 1èr critère, suivi de prénom.

En SQL : Create Index CX on C1 (Nom, Prenom)
En QBE : exercice pratique

6. Exprimer en SQL standard ou en SQL de MsAccess et en QBE, si possible, les manipulations suivantes :

- Le numéro de client, le nom et le prénom en ordre ascendant sur le nom et le prénom, pour au plus 10 clients, nées avant 2002.

  Il faut utiliser TOP 10 de SQL MsAccess qui n'existe pas dans SQL standard. Dans Order By il faut mettre la clé pour éviter des ex-æquo. En QBE il faut aller dans la ligne 1ères valeurs de la feuille de propriétés de la requête.

  Select Top 10 nom, prenom From Cl where D_Naiss < 2002 Order By nom, prenom, Cl# ;

- Tous les éditeurs de CDs dont le nom d'éditeur commence par B, un tuple par éditeur.

  SQL standard: Select Distinct Edit From CD where Edit Like "B%" ;

  SQL MsAccess : Select Distinct Edit From CD where Edit Like "B*" ;

  QBE : Mettre la clause : comme "B*" dans la ligne critère de la colonne Edit de la grille de la requête. Utiliser aussi l'option Oui dans la ligne Valeurs distinctes de la feuille de propriétés de la requête.

- Noms de clients ayant le nombre de fautes inconnu pendant une de session, avec l'identification de cette session.

  SQL standard: Select distinct Nom From Cl, A Where N_Fautes is Null and A.Cl# = Cl.Cl# ;

  SQL MsAccess: Select distinct Nom From Cl, A Where N_Fautes is Null and A.[Cl#] = Cl.[Cl#] ;

  QBE: La jointure est implicite. Il suffit donc de mettre la clause: Est null (ou Is Nul) dans la ligne critère de la colonne N_Fautes de la grille de la requête. Cette colonne devrait être décochée. Utiliser aussi l'option Oui dans la ligne Valeurs distinctes de la feuille de propriétés de la requête.

- Tout numéro et éditeur d'un CD, avec le numéro de toute session où ce CD a été appliqué, s'il y avait une telle session. Puis, une même requête avec la condition supplémentaire que S# ne doit pas être 10.

  SQL MsAccess:

  Select [cd#], edit From CD Right Join S_Code On CD.[cd#] = S_Code.[cd#] ;
  Select [cd#], edit From CD Right Join S_Code On CD.[cd#] = S_Code.[cd#]
  Where [s#] <> 10 or [s#] is null ;

- Les couples de CD# ayant un même éditeur (sans doublons en tout genre). On utilise la Θ-jointure pour éliminer les doublons (voir le cours).

Dans toutes les requêtes qui suivent le dialecte SQL est celui de MsAccess.

```
Select * from CD X, CD Y where X.Edit = Y.Edit and X.[cd#] <= Y.[cd#];
```

- Noms de clients ayant suivi (i) la session '1' ou la session '2' ; (ii) la session '1' et la session '2' ; (iii) la session '2' sans avoir suivi la session '1'.

```
Select Nom from Cl, A where Cl.[cl#] = A.[cl#] and [se#] = 1 OR [se#] = 2;
Select Nom from Cl, A where Cl.[cl#] = A.[cl#] and [se#] = 1 and
    exists
        (select * from A X where Cl.[cl#] = X.[cl#] and X.[se#] = 2);

Select Nom from Cl, A where Cl.[cl#] = A.[cl#] and [se#] = 2 and
    Not exists
        (select * from A X where Cl.[cl#] = X.[cl#] and X.[se#] = 1);
```

- Noms de clients ayant suivi toutes les sessions.

```
Select Nom from Cl where not exists
    (select * from S_Code where not exist
        (select * from A where Cl.[cl#] = A.[cl#] and S_Code.[Se#] = A. [Se#]));
```

- Noms de clients n'ayant suivi aucune session.

```
Select Nom from Cl where not exists
        (select * from A where Cl.[cl#] = A.[cl#]);
```

- Tout nom de client ayant atteint le minimum de fautes connu, avec l'ID de la session de sa prouesse.

```
Select Nom, [SE#] from Cl, A where  Cl.[cl#] = A.[cl#] and
    N_Fautes = (select Min(N_Fautes) from A) ;
```

- Nombre total de fautes par client durant les examens, ainsi que sa variance, la partie entière de la racine carrée de celle-ci, et Cl#.

```
Select Sum(N_Fautes), Var(N_Fautes) as V, Int(V^(1/2)), [cl#]
    from [E-R]
    Group by [cl#];
```

A noter les crochés autour d'E-R et l'utilisation de l'attribut dynamique V.

- La moyenne de fautes par client durant les sessions, avec le nom du client et la condition que cette moyenne est au-dessus de 3.

```
Select Avg(N_Fautes) as [MFpC], Nom from Cl, A where Cl.[cl#] = A.[cl#]
Group by A.[cl#], Nom
Having  Avg(N_Fautes) > 3;
```

Notez la non-utilisation de l'attribut dynamique dans la clause Having. Demandez quel serait le résultat autrement ?

- Dates suivant de deux semaines celles des examens dans la table E, avec l'ID de chaque examen. Libellées comme dates d'affichage de résultats.

```
Select Date+14 as [Date d'affichage], [PC#] from E;
```

- Tout sur les exams dans un lieu donnée en paramètre.

```
Select * from E where Lieu = [Lieu SVP];

Select * from E where Lieu = [Lieu SVP]&"*";
```

La dernière formulation permet d'indiquer le lieu par préfixe.

- Ajouter l'info que le client 123, a fait 5 fautes dans la session 456

```
Insert  Into A Values ([Cl#], [N_Fautes], [Se#]) Values (123, 5, 456);
```

- Modifier l'heure de tout examen à Paris pour 9h, puis supprimer tout client à nombre de fautes maximal.

```
UPDATE E SET E.Heure = #9:00#;

Delete * from Cl where [Cl#] in (select [Cl#] from A where N_Fautes =
     (select max(N_Fautes) from A));
```

Notez que "=" au lieu de "in" est impossible. Pourquoi au juste ?

- En supposant que l'éditeur à la page Web et que la table CD contient l'attribut Page_Web trouver sous MsAccess 2007 en SQL tous les CD# de http://www.dupont.fr/ .

```
SELECT [CD#]
FROM CD where Page_Web = "#http://www.dupont.fr/#";
```

- Exprimer en SQL (Standard ou MsAcces) les requêtes en algèbre relationnelle à la base S-P vues en TDs.

Je vous laisse faire, Messieurs les Assistants.

7. Exprimer en SQL de MsAccess ou en QBE, si possible, les manipulations *avancées* suivantes :

- Le nombre total d'éditeurs, en sachant qu'un éditeur peut éditer plusieurs CDs.

  Select count (*) from (select distinct Edit from CD);

- Nombre total de fautes par client durant les sessions, ensemble avec les fautes de chaque session.

En QBE, il s'agit d'une requête de tabulation croisée. Je ne pense pas que l'on aura le temps de faire la formulation SQL (avec les clauses Transform et Pivot).

- La moyenne de fautes par client, pendant les sessions et les examens confondus.

  Select Avg (N_Fautes) From
  (Select [cl#], N_Fautes From A Union All Select [cl#], N_Fautes From E-R)
  Group By [cl#];

- Pour tout client, sa moyenne de fautes en sessions et la moyenne de fautes des autres clients.

  Voir le cours sur SQL avancé (T-Group By) ; peu probable que l'on fasse la technique de cette requête en cours (la clause select imbriquée).

- La covariance/corrélation entre le nombre de fautes des exams au même lieu à des dates différentes.

  Select Avg(X.N_Fautes*Y.N_Fautes)-Avg(X.N_Fautes)*Avg(Y.N_Fautes) AS Cov, V.Lieu
  FROM [E-R] as X, [E-R] as Y, E as V, E as Z
  Where V.Lieu = Z.Lieu and X.[PC#] = V.[PC#] and Y.[PC#] = Z.[PC#] and
  V.Date < Z.Date
  GROUP BY V.Lieu ;

  Ceux qui trouvent sont vraiment bons.

13 Les requêtes suivantes sont elles légales ? Si oui leur résultats svp.: Select 5+5 ; Select 5.0+5 ; Select 5-5 ; Select 5 – (-5) ; Select 5+nul ; select sum(qty), count(*), count (QTY) from SP ; en supposant que QTY pour la fourniture (S1,P1) est nul et seul à l'être. Explications et conclusions globales ?

55, 5, 0, nul, requête à essayer. On verra que le nul ne compte pas pour la somme sous forme d'agrégation, contrairement à la somme avec +. Vous avez dit bizarre ? Enfin, les deux comptages différons de 1.

14. Est-ce que **Select * from S** est équivalent à **Select * from S where city = « Paris » union Select * from S where city <> « Paris »** ? Non. Les résultats diffèrent pour les nuls dans **city**.