Le SQL de A à Z

Exercices et travaux pratiques

Par SQLPro

Problèmes et exercices corrigés sur le langage SQL.

Voici une série de travaux pratiques sur le langage SQL appelant des réponses devant être écrites uniquement à partir de requêtes, en utilisant le plus souvent possible le jeu de commande du SQL 2 normalisé (1999 ou 2003). Les données de départ, comme la structure des tables en jeu et la réponse attendue - sous la forme d'un jeu de résultat (table) - sont données dans chacun des énoncés.

Pour répondre à ces, questions, rien de plus simple : envoyez-moi vos solutions en utilisant le lien hypertexte ouvrant votre messagerie. Bien entendu, je vous répond personnellement et vous propose une correction didactique et pédagogique afin que vous compreniez le mécanisme de construction de la requête.

I - Les règles du jeu

Pour répondre à la question posée, il vous faut écrire une seule requête SQL.

Vous avez le droit d'utiliser toutes les techniques disponibles dans SQL:2003 :

- Prédicats : **BETWEEN, IN, IS NULL, IS TRUE, IS FALSE, IS UNKNOWN, ALL, ANY, SOME, EXISTS, UNIQUE, MATCH, LIKE, OVERLAPS...**
- Mots clef : **ALL, DISTINCT...**
- Comparaisons : **=, <=, >=, <, >, <>...**
- Connecteurs logiques : **AND, OR, NOT...**
- Constructions : **Row Value Constructor, CASE...**
- Opérateurs : **UNION, EXCEPT, INTERSECT...**
- Fonctions : **CAST, OCTET_LENGTH, BIT_LENGTH, COALESCE, NULLIF, CHARACTER_LENGTH, SUBSTRING, POSITION, TRIM, UPPER, LOWER, ABS, MOD, LN, EXP, POWER, FLOOR, CEILING, EXTRACT, CURRENT_DATE,CURRENT_TIME,CURRENT_TIMESTAMP...**
- Agrégats statistiques : **COUNT(*), COUNT(...), MAX(...), MIN(...), SUM(...), AVG(...)...**
- Clauses : **WHERE, HAVING, GROUP BY, COLLATE, ORDER BY...**
- Les jointures : **INNER JOIN, LEFT, RIGHT ou FULL OUTER JOIN, CROSS JOIN, UNION JOIN**
- Les agrégats cumulatifs **CUBE, ROLLUP, GROUPING SETS** et la fonction **GROUPING** (SQL:1999)
- Les fonctions de fenêtrage (**RANK, DENSE_RANK, PERCENT_RANK, ROW_NUMBER, CUM_DIST** et leurs clauses spécifiques : ORDER BY, PARTITION BY (SQL:2003)
- Les agrégats partiels obtenur avec les clauses **ORDER BY, PARTITION BY** (SQL:2003)
- la technique de la **CTE** (Common table expression - expression de table) dont vous trouverez une étude ici : **CTE ET récursivité des requêtes**

Vous pouvez utiliser toutes les opérations disponibles dans SQL, y compris les sous requêtes dans les clauses SELECT, FROM, WHERE et HAVING, et bien entendu les sous requêtes corrélées.

Vous pouvez ajoutez dans la base une ou plusieurs nouvelles tables et leurs données ou encore définir autant de vues que vous le voulez,pour concourrir à la solution. En revanche vous n'avez pas le droit d'utiliser une UDF (fonction utilisateur), ni une procédure stockée, ni un trigger, ni bien entendu modifier la ou les tables et les données à l'origine du problème.

Tous ces exercices ont une solution et souvent plusieurs !

Pour vous aider à répondre, nous vous donnons :

1 la structure de la ou les tables sous la forme d'un ordre SQL CREATE TABLE...;
2 les données à insérer sous la forme d'un jeu d'ordre SQL INSERT INTO...;
3 le résultat attendu (lignes résultant de l'exécution de la requête solution);
4 le niveau de difficulté noté avec des étoiles de 1 (facile) à 5 (très difficile);
5 parfois une ébauche du résultat pour vous mettre sur la piste.

Pour savoir si votre solution est bonne, comme pour obtenir la solution, envoyez votre réponse ou votre demande par mail en précisant le titre ou le n° du problème.

II - Exercices - 1° partie

II-A - Problème n° 1 - un dans dix

Notre premier problème est intitulé '**un dans dix**' et il nous est posé par Joe CELKO...

Une table est composée de 11 colonnes comme suit :

Création de la table

```
CREATE TABLE T_CELKO_TEN_IN_ON_TIO
(TIO_ID INTEGER NOT NULL,
 TIO_1  INTEGER NOT NULL,
 TIO_2  INTEGER NOT NULL,
 TIO_3  INTEGER NOT NULL,
 TIO_4  INTEGER NOT NULL,
 TIO_5  INTEGER NOT NULL,
 TIO_6  INTEGER NOT NULL,
 TIO_7  INTEGER NOT NULL,
 TIO_8  INTEGER NOT NULL,
 TIO_9  INTEGER NOT NULL,
 TIO_10 INTEGER NOT NULL);
```

Insertion des données

```
INSERT INTO T_CELKO_TEN_IN_ON_TIO VALUES (1,   0,  1,  1,  0,  0,  0,  0,  0,  0,  0);
INSERT INTO T_CELKO_TEN_IN_ON_TIO VALUES (2,   0,  0,  0,  0,  0,  0,  0,  1,  0,  0);
INSERT INTO T_CELKO_TEN_IN_ON_TIO VALUES (3,   0,  1, -2,  3, -4,  5, -6,  7, -8,  5);
INSERT INTO T_CELKO_TEN_IN_ON_TIO VALUES (4,   0,  0,  0,  0,  0,  0,  0,  0,  9,  0);
INSERT INTO T_CELKO_TEN_IN_ON_TIO VALUES (5,   0,  0,  0,  0,  0, -1,  1,  0,  0,  0);
INSERT INTO T_CELKO_TEN_IN_ON_TIO VALUES (6,   0,  0,  0,  0,  0, -1,  1,  1,  0,  0);
INSERT INTO T_CELKO_TEN_IN_ON_TIO VALUES (7,   0,  1,  0,  1,  0,  0,  0,  0,  0,  1);
INSERT INTO T_CELKO_TEN_IN_ON_TIO VALUES (8,   1,  0,  0,  0,  0,  0,  0,  0,  0,  0);
INSERT INTO T_CELKO_TEN_IN_ON_TIO VALUES (9,   0,  0,  0, -1,  0,  0,  0,  0,  0,  0);
INSERT INTO T_CELKO_TEN_IN_ON_TIO VALUES (10,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0);
INSERT INTO T_CELKO_TEN_IN_ON_TIO VALUES (11,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1);
INSERT INTO T_CELKO_TEN_IN_ON_TIO VALUES (12,  1,  0,  0,  0,  0,  0,  0,  0,  0, -1);
```

Données de la table

TIO_ID	TIO_1	TIO_2	TIO_3	TIO_4	TIO_5	TIO_6	TIO_7	TIO_8	TIO_9	TIO_10
1	0	1	1	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	1	0	0
3	0	1	-2	3	-4	5	-6	7	-8	5
4	0	0	0	0	0	0	0	0	9	0
5	0	0	0	0	0	-1	1	0	0	0
6	0	0	0	0	0	-1	1	1	0	0
7	0	1	0	1	0	0	0	0	0	1
8	1	0	0	0	0	0	0	0	0	0
9	0	0	0	-1	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0
11	1	1	1	1	1	1	1	1	1	1
12	1	0	0	0	0	0	0	0	0	-1

Il semble que Joe, le concepteur de cette base de données veuille stocker des tableaux d'entiers à 10 cellules. Le problème qui est soulevé, c'est que notre quidam veut obtenir en une seule requête :

1 les lignes de la table dont toutes les cellules f1 à f10 sont à zéro sauf une,
2 les lignes de la table dont toutes les cellules f1 à f10 sont à zéro sauf une valuée à un.

Les résultats attendus :

Les lignes de la table dont toutes les cellules f1 à f10 sont à zéro sauf une :

TIO_ID	TIO_1	TIO_2	TIO_3	TIO_4	TIO_5	TIO_6	TIO_7	TIO_8	TIO_9	TIO_10
2	0	0	0	0	0	0	0	1	0	0
4	0	0	0	0	0	0	0	0	9	0
8	1	0	0	0	0	0	0	0	0	0
9	0	0	0	-1	0	0	0	0	0	0

Les lignes de la table dont toutes les cellules f1 à f10 sont à zéro sauf une valuée à un :

TIO_ID	TIO_1	TIO_2	TIO_3	TIO_4	TIO_5	TIO_6	TIO_7	TIO_8	TIO_9	TIO_10
2	0	0	0	0	0	0	0	1	0	0
8	1	0	0	0	0	0	0	0	0	0

Il existe de multiples solutions à ce problème et j'en découvre une nouvelle tous les mois... A vous de jouer !

cliquez pour répondre et titrez votre mail "**Problème n° 1 - un dans dix, ma solution**"

II-B - Problème n°2 - le publipostage

Notre second problème est intitulé 'le publipostage' et il m'a été suggéré par Larry DiGiovanni...

Nous disposons d'une table contenant les noms et adresses de personnes (table PERSON) doté d'une colonne NOMBRE. Larry à besoin de reproduire certaines lignes plusieurs fois afin de faire des étiquettes de publipostage. Ainsi pour Monsieur MARTIN il désire 2 étiquettes, tandis que pour M. DUPOND il en faut trois. Ce nombre est saisie dans la table T_PERSONNE_PRS qui a la structure suivante :

```
CREATE TABLE T_PERSONNE_PRS
(PRS_NOM    CHAR(16) NOT NULL,
 PRS_VILLE  CHAR(16),
 PRS_NOMBRE INTEGER);
```

```
INSERT INTO T_PERSONNE_PRS VALUES ('MARTIN', 'PARIS', 3);
INSERT INTO T_PERSONNE_PRS VALUES ('DUPOND', 'STRASBOURG', 2);
```

```
PRS_NOM    PRS_VILLE    PRS_NOMBRE
---------- ------------ -----------
MARTIN     PARIS        3
DUPOND     STRASBOURG   2
```

Le but étant de fournir une réponse dans laquelle chaque ligne de la table PERSON est recopié autant de fois que spécifié par la valeur de la colonne NOMBRE.

Comment faire, en une requête et une seule, pour obtenir la réponse :

```
PRS_NOM          PRS_VILLE
---------------- ----------------
MARTIN           PARIS
MARTIN           PARIS
MARTIN           PARIS
DUPOND           STRASBOURG
DUPOND           STRASBOURG
```

Vous ne pouvez pas modifier la structure de la table ni modifier les données, mais vous avez le droit de vous aider en utilisant ou en créant d'autres éléments dans la base de données...
A vous de jouer !

cliquez pour répondre et titrez votre mail "**Problème n°2 - le publipostage**"

II-C - Problème n° 3 - la date an 2000

Ce troisième problème "intitulé date an 2000" est inspiré des petits tracas que l'arrivée de l'an 2000 a laissé à nos développeurs.

La table comptable ci dessous comporte une colonne AMT_FIN qui indique la date de fin d'amortissement de certaines références. Mais cette colonne est de type CHAR(6) et contient des données formatée de la façon suivante : AAMMJJ (années sur deux chiffres, mois sur deux chiffres, jour sur deux chiffres) afin de pouvoir trier facilement sur l'ordre alpha correspondant dans ce cas à l'ordre calendaire.

Notre développeur a décidé d'ajouter une nouvelle colonne 'AMT_FIN_Y2K' de type CHAR(10) cette fois, et aimerais votre aide pour rétablir toutes les anciennes dates au bon format. Aucune date n'est antérieure au 1/1/1960. Sauriez-vous l'aider ???
Je vous précise que la date doit être stockée dans cette base de données sous la forme AAAA-MM-JJ (format ISO).

Voici un extrait des données de cette table intitulé T_AMORTISSEMENT_AMT

```
CREATE TABLE T_AMORTISSEMENT_AMT
(AMT_FIN     CHAR(6),
 AMT_FIN_Y2K CHAR(10));
```

```
INSERT INTO T_AMORTISSEMENT_AMT (AMT_FIN) VALUES ('990601');
INSERT INTO T_AMORTISSEMENT_AMT (AMT_FIN) VALUES ('970201');
INSERT INTO T_AMORTISSEMENT_AMT (AMT_FIN) VALUES ('021201');
INSERT INTO T_AMORTISSEMENT_AMT (AMT_FIN) VALUES ('941101');
INSERT INTO T_AMORTISSEMENT_AMT (AMT_FIN) VALUES ('920715');
```

```
AMT_FIN AMT_FIN_Y2K
------- ----------
990601  NULL
970201  NULL
021201  NULL
941101  NULL
920715  NULL
```

Et voici le résultat attendu :

```
AMT_FIN     AMT_FIN_Y2K
---------   -------------
990601      06-01-1999
970201      02-01-1997
021201      12-01-2002
941101      11-01-1994
920715      07-15-1992
```

cliquez pour répondre et titrez votre mail "**Problème n°3 - la date an 2000**"

II-D - Problème n° 4 - les chambres libres

Ce problème "**chambres libres**" m'a été inspiré par un internaute qui se posait la question suivante : "*il me faut gerer les 350 jours de l'année pour des réservations hotelières. Par exemple, il faudrait pouvoir afficher les chambres libre du 22/03/2000 au 15/04/2000*"

Notre internaute avait modélisé cela dans deux tables : la table T_CHAMBRE_CHB, contenant, entre autres, le n° des différentes chambres existantes et une table T_PLANNING_PLN contenant 3 colonnes, le numero de la chambre, la date et une colonne contenant une représentation booléenne pour indiquer que la chambre est libre ou non. L'hôtel de notre exemple comporte 4 chambres numérotés de 1 à 4, et la table planning ne contient que des lignes pour les chambres réservées ou occupées. En principe, les chambres libres à une date données ne sont pas représentées dans la table T_PLANNING_PLN, sauf si la colonne PLN_LIBRE est valuée à True.

```
CREATE TABLE T_CHAMBRE_CHB
(CHB_NUM INTEGER);

CREATE TABLE T_PLANNING_PLN
(PLN_JOUR  DATE,
```

```
 CHB_NUM   INTEGER,
 PLN_LIBRE CHAR(5));
```

```
INSERT INTO T_CHAMBRE_CHB VALUES (1) ;
INSERT INTO T_CHAMBRE_CHB VALUES (2) ;
INSERT INTO T_CHAMBRE_CHB VALUES (3) ;
INSERT INTO T_CHAMBRE_CHB VALUES (4) ;

INSERT INTO T_PLANNING_PLN VALUES ('2000-01-12', 1, 'False') ;
INSERT INTO T_PLANNING_PLN VALUES ('2000-01-12', 2, 'False') ;
INSERT INTO T_PLANNING_PLN VALUES ('2000-01-13', 1, 'False') ;
INSERT INTO T_PLANNING_PLN VALUES ('2000-01-13', 2, 'False') ;
INSERT INTO T_PLANNING_PLN VALUES ('2000-01-13', 4, 'True' ) ;
```

Sauriez-vous retrouver, à l'aide d'une requête SQL de votre cru,
1) les chambres qui sont libre pendant toute la période allant du 11 au 14 janvier 2000 ?
2) l'occupation des chambres pour la journée du 13 janvier 2000 ?
3) le planning des occupations pour toutes les chambres et toutes les dates du 11 au 14 janvier 2000 ?

Les réponses attendues sont les suivantes :

1) Chambres libres période du 11 au 14 janvier 2000

```
CHB_NUM
-----------
3
4
```

2) Occupation au 13 janvier 2000

```
CHB_NUM     PLN_LIBRE
----------- ---------
1           False
2           False
3           True
4           True
```

3) Planning occupation des chambres du 11 au 14 janvier 2000 ?

```
CLD_JOUR    CHB_NUM     PLN_LIBRE
----------- ----------- ---------
2000-01-11  1           True
2000-01-11  2           True
2000-01-11  3           True
2000-01-11  4           True
2000-01-12  1           False
2000-01-12  2           False
2000-01-12  3           True
2000-01-12  4           True
2000-01-13  1           False
2000-01-13  2           False
2000-01-13  3           True
2000-01-13  4           True
2000-01-14  1           True
2000-01-14  2           True
2000-01-14  3           True
2000-01-14  4           True
```

cliquez pour répondre et titrez votre mail "**Problème n°4 - les chambres libres**"**http://cerbermail.com/?98a8lH4SOe**

II-E - Problème n° 5 - dates d'anniversaire

Ce problème "**date d'anniversaire**" est venu de mon patron, qui m'a dit un jour, comme ça, tout de go, "*je voudrais envoyer une carte pour chacun de mes clients afin de lui souhaiter un bon anniversaire...*"
Fastoche, me suis-je dis. Hélas, j'ai du piannoter un bon moment avant d'arriver à trouver une solution honnête.

Sauriez vous trouver aussi bien, sinon mieux encore que ce que j'ai fait ???

Voici un exemple de la table de nos clients :

```
CREATE TABLE T_CLIENT_CLI
(CLI_ID             INTEGER,
 CLI_NOM            VARCHAR(16),
 CLI_DATE_NAISSANCE DATE);
```

```
INSERT INTO T_CLIENT_CLI VALUES (24, 'HESS',       '1984-08-30') ;
INSERT INTO T_CLIENT_CLI VALUES (25, 'CHATON',     '1938-12-31') ;
INSERT INTO T_CLIENT_CLI VALUES (26, 'PLATONOFF',  '1960-03-29') ;
INSERT INTO T_CLIENT_CLI VALUES (27, 'LETERRIER',  '1945-08-26') ;
INSERT INTO T_CLIENT_CLI VALUES (28, 'MONTEIL',    '1985-03-23') ;
INSERT INTO T_CLIENT_CLI VALUES (29, 'SPITHAKIS',  '1956-03-31') ;
INSERT INTO T_CLIENT_CLI VALUES (30, 'ORELL',      '1996-01-27') ;
INSERT INTO T_CLIENT_CLI VALUES (31, 'MARTINET',   '1946-10-24') ;
INSERT INTO T_CLIENT_CLI VALUES (32, 'RAY',        '1979-12-20') ;
INSERT INTO T_CLIENT_CLI VALUES (33, 'TARSAC',     '1992-08-14') ;
INSERT INTO T_CLIENT_CLI VALUES (34, 'COULOMB',    '1963-01-01') ;
INSERT INTO T_CLIENT_CLI VALUES (35, 'SAVY',       '1942-10-03') ;
INSERT INTO T_CLIENT_CLI VALUES (36, 'DAVID',      '1922-09-19') ;
INSERT INTO T_CLIENT_CLI VALUES (37, 'FORGEOT',    '1989-09-13') ;
INSERT INTO T_CLIENT_CLI VALUES (38, 'BERGER',     '1946-10-16') ;
INSERT INTO T_CLIENT_CLI VALUES (39, 'DOUBLET',    '1961-03-02') ;
INSERT INTO T_CLIENT_CLI VALUES (40, 'MATHIEU',    '1990-02-24') ;
INSERT INTO T_CLIENT_CLI VALUES (41, 'MOURGUES',   '1953-01-14') ;
INSERT INTO T_CLIENT_CLI VALUES (42, 'PIERROT',    '1933-02-11') ;
INSERT INTO T_CLIENT_CLI VALUES (44, 'ZAMPIERO',   '1985-01-19') ;
INSERT INTO T_CLIENT_CLI VALUES (45, 'PASCOT',     Null        ) ;
INSERT INTO T_CLIENT_CLI VALUES (46, 'MECHRI',     '1950-08-11');
```

```
CLI_ID      CLI_NOM          CLI_DATE_NAISSANCE
----------- ---------------- ------------------
24          HESS             1934-08-30
25          CHATON           1938-12-31
26          PLATONOFF        1960-03-29
27          LETERRIER        1945-08-26
28          MONTEIL          1935-03-23
29          SPITHAKIS        1956-03-31
30          ORELL            1956-01-27
31          MARTINET         1946-10-24
32          RAY              1939-12-20
33          TARSAC           1942-08-14
34          COULOMB          1963-01-01
35          SAVY             1942-10-03
36          DAVID            1922-09-19
37          FORGEOT          1949-09-13
38          BERGER           1946-10-16
39          DOUBLET          1961-03-02
40          MATHIEU          1940-02-24
41          MOURGUES         1953-01-14
42          PIERROT          1933-02-11
44          ZAMPIERO         1945-01-19
45          PASCOT           NULL
46          MECHRI           1950-08-11
```

Le problème est, par exemple, d'extraire les clients qui vont avoir leur anniversaire entre :
1) le 21 février et le 20 mars ;
2) le 21 décembre au 20 janvier ;
3) n'importe quelle fourchette de date (même à cheval sur deux années).

Les réponses attendues sont les suivantes :

1) Anniversaires entre le 21 février et le 20 mars

```
CLI_ID      CLI_NOM          CLI_DATE_NAISSANCE
----------- ---------------- -----------
```

1) Anniversaires entre le 21 février et le 20 mars

```
39          DOUBLET          1961-03-02
40          MATHIEU          1940-02-24
```

2) Anniversaires entre le 21 décembre et le 20 janvier

```
CLI_ID      CLI_NOM          CLI_DATE_NAISSANCE
----------- ---------------- ------------------
25          CHATON           1938-12-31
34          COULOMB          1963-01-01
41          MOURGUES         1953-01-14
44          ZAMPIERO         1945-01-19
```

A vos requêtes...

cliquez pour répondre et titrez votre mail "**Problème n° 5 - dates d'anniversaire**"

II-F - Problème n° 6 - énumération

Pour cet exercice, il s'agit d'alimenter les nombres de 0 à 9999 dans une table de nom T_ENTIER_ENT contenant un seul champ de type entier et de nom ENT_N. Au départ, cette table contient les nombres de 0 à 9.
J'ai intitulé ce problème, ' **énumération'** ...

La table T_ENTIER_NET est ainsi constituée :

```
CREATE TABLE T_ENTIER_ENT
(ENT_N INTEGER);
```

```
INSERT INTO T_ENTIER_ENT VALUES (0) ;
INSERT INTO T_ENTIER_ENT VALUES (1) ;
INSERT INTO T_ENTIER_ENT VALUES (2) ;
INSERT INTO T_ENTIER_ENT VALUES (3) ;
INSERT INTO T_ENTIER_ENT VALUES (4) ;
INSERT INTO T_ENTIER_ENT VALUES (5) ;
INSERT INTO T_ENTIER_ENT VALUES (6) ;
INSERT INTO T_ENTIER_ENT VALUES (7) ;
INSERT INTO T_ENTIER_ENT VALUES (8) ;
INSERT INTO T_ENTIER_ENT VALUES (9) ;
```

```
ENT_N
------
0
1
2
3
4
5
6
7
8
9
```

Sauriez vous faire une telle requête avec un seul update ?

Voici un extrait de la table des données à insérer :

```
NOMBRE
-----------
10
11
12
13
14
...
```

```
9995
9996
9997
9998
9999
```

cliquez pour répondre et titrez votre mail " **Problème n° 6 - énumération** "

II-G - Problème n° 7 - le comptage

Voici un problème qui m'a été soumis par un collègue. Je l'ai intitulé **le comptage...**
Comment faire en sorte que l'on puisse générer une colonne de comptage allant de 1 à n pour compter chaque occurence de la réponse ? C'est un problème récurent. Beaucoup de développeurs souhaitent numéroter les lignes d'une table réponse.

Dans notre exemple, mon collègue possède une table "T_PROSPECT_PSP", dans laquelle il y a des noms de personne (colonne PSP_NOM) et il voudrait faire en sorte que le résultat de sa requête numérote les lignes dans l'aordre alphabétique des noms :

```
CREATE TABLE T_PROSPECT_PSP
(PSP_NOM VARCHAR(16));
```

```
INSERT INTO T_PROSPECT_PSP VALUES ('ARMAND');
INSERT INTO T_PROSPECT_PSP VALUES ('DUPONT');
INSERT INTO T_PROSPECT_PSP VALUES ('BAILLE');
INSERT INTO T_PROSPECT_PSP VALUES ('GAUTIER');
INSERT INTO T_PROSPECT_PSP VALUES ('MARTIN');
INSERT INTO T_PROSPECT_PSP VALUES ('CLAUDE');
INSERT INTO T_PROSPECT_PSP VALUES ('DUPONT');
```

Avez-vous une idée pour traiter ce problème ?

Voici le résultat attendu :

```
PSP_NOM          N
---------------- -----------
BAILLE           1
CLAUDE           2
DUPONT           3
DUPONT           4
GAUTIER          5
MARTIN           6
```

A vos requêtes !

cliquez pour répondre et titrez votre mail " **Problème n° 7 - comptage** "

II-H - Problème n° 8 - linéarisation

Nous apellerons ce problème " **linéarisation** "...

Opaz, un internaute me demande : " *J'ai besoin d'un affichage en ligne plutot qu'en colonne. J'ai trois tables différentes et je souhaite simplement afficher le nombre de lignes de chacunes... Voici mes tables et les données ...* "

```
CREATE TABLE TBL1
(TBL_ID INTEGER);

CREATE TABLE TBL2
(TBL_ID INTEGER);
```

```
CREATE TABLE TBL3
(TBL_ID INTEGER);
```

```
INSERT INTO TBL1 VALUES (1);
INSERT INTO TBL1 VALUES (3);
INSERT INTO TBL1 VALUES (5);
INSERT INTO TBL1 VALUES (7);
INSERT INTO TBL1 VALUES (9);

INSERT INTO TBL2 VALUES (2);
INSERT INTO TBL2 VALUES (4);
INSERT INTO TBL2 VALUES (6);
INSERT INTO TBL2 VALUES (8);

INSERT INTO TBL3 VALUES (1);
INSERT INTO TBL3 VALUES (2);
INSERT INTO TBL3 VALUES (3);
INSERT INTO TBL3 VALUES (5);
INSERT INTO TBL3 VALUES (7);
```

Cette façon de présenter l'information ne lui va pas :

```
SELECT 'TBL1' AS "TABLE",  COUNT(*) AS NB
FROM TBL1
UNION ALL
SELECT 'TBL2' AS "TABLE",  COUNT(*) AS NB
FROM TBL2
UNION ALL
SELECT 'TBL3' AS "TABLE",  COUNT(*) AS NB
FROM TBL3
```

```
TABLE NB
----- -----------
TBL1  5
TBL2  4
TBL3  5
```

Il veut le résultat sous la forme suivante :

```
TBL1  TBL2   TBL3
----- ------ ------
5     4      5
```

Sauriez vous aider Opaz et présenter ce sésulta en une seule requête ???

cliquez pour répondre et titrez votre mail " **Problème n°8 - linéarisation** "

II-I - Problème n° 9 - les trous

Voici un problème envoyé par Guillaume, un internaute, je l'ai apellé " **Les Trous** " !

" *J'ai une table dont la clef primaire est un entier (non auto incrémenté). Je cherche une requête qui me donnerai le premier entier disponible dans la séquence parmi ces clefs. Par exemple si la table contient les enregistrements dont les clefs sont 1, 2, 3, 5, 6, 8, 9, 10, 11, 12, 14, 15 suite à divers ajouts et suppressions d'enregistrements, je veux que la requête me donne 4... .*"

Pour simplifier la demande de Guillaume, voici la table qui constitue le coeur de cet exercice :

```
CREATE TABLE T_NUMERO_NMR
(NMR INTEGER);
```

```
INSERT INTO T_NUMERO_NMR VALUES (1);
```

```
INSERT INTO T_NUMERO_NMR VALUES (2);
INSERT INTO T_NUMERO_NMR VALUES (3);
INSERT INTO T_NUMERO_NMR VALUES (5);
INSERT INTO T_NUMERO_NMR VALUES (6);
INSERT INTO T_NUMERO_NMR VALUES (8);
INSERT INTO T_NUMERO_NMR VALUES (9);
INSERT INTO T_NUMERO_NMR VALUES (10);
INSERT INTO T_NUMERO_NMR VALUES (11);
INSERT INTO T_NUMERO_NMR VALUES (12);
INSERT INTO T_NUMERO_NMR VALUES (14);
INSERT INTO T_NUMERO_NMR VALUES (15);
```

Je suppose que Guillaume veut gérer les trous de clefs générés par les suppressions d'enregistrements afin d'en récupérer la place... C'est en général une fort mauvaise idée. Mais sauriez vous quand même dépanner Guillaume ? Donc récupérer tous les trous ? En une seule requête évidemment !

Voici une image du résultat :

```
TROU
-----------
4
7
13
```

cliquez pour répondre et titrez votre mail " **Problème n°9 - les trous** "

II-J - Problème n° 10 - symétrie négative

C'est une collègue qui m'a posé ce problème ... que j'ai intitulé **symétrie négative** !

Dans une application comptable ma collègue avait besoin d'annuler logiquement des lignes de détail liée à une ligne maître, c'est à dire contrepasser une écriture. Comme un exemple vaut mieux qu'on long discours, nous prendrons une facture et les lignes afférentes à la facture.

```
CREATE TABLE T_FACTURE_FAC
(FAC_NUM  INTEGER,
 FAC_DATE DATE);

CREATE TABLE T_LIGNE_FACTURE_LIF
(LIF_NUM     INTEGER,
 FAC_NUM     INTEGER,
 LIF_ARTICLE VARCHAR(16),
 LIF_MONTANT DECIMAL(16,2));
```

```
INSERT INTO T_FACTURE_FAC VALUES(56, '2001-08-11') ;
INSERT INTO T_FACTURE_FAC VALUES(79, '2001-08-17') ;
INSERT INTO T_FACTURE_FAC VALUES(101,'2001-09-01');

INSERT INTO T_LIGNE_FACTURE_LIF VALUES (11, 56, 'Cartable', 123.50) ;
INSERT INTO T_LIGNE_FACTURE_LIF VALUES (13, 56, 'Crayons', 17.52) ;
INSERT INTO T_LIGNE_FACTURE_LIF VALUES (15, 56, 'Trousse', 29.99) ;
INSERT INTO T_LIGNE_FACTURE_LIF VALUES (16, 56, 'Feutres', 11.25) ;
INSERT INTO T_LIGNE_FACTURE_LIF VALUES (18, 56, 'Ardoise', 44.21) ;
INSERT INTO T_LIGNE_FACTURE_LIF VALUES (19, 79, 'Cigarettes', 21.55) ;
INSERT INTO T_LIGNE_FACTURE_LIF VALUES (21, 79, 'Whisky', 147.65) ;
INSERT INTO T_LIGNE_FACTURE_LIF VALUES (22, 79, 'Petite pépé', 12857.59) ;
INSERT INTO T_LIGNE_FACTURE_LIF VALUES (27, 101, 'Savon', 16.24) ;
INSERT INTO T_LIGNE_FACTURE_LIF VALUES (28, 101, 'Serviette', 45.00) ;
INSERT INTO T_LIGNE_FACTURE_LIF VALUES (32, 101, 'Shampoing', 44.22) ;
INSERT INTO T_LIGNE_FACTURE_LIF VALUES (33, 101, 'Bain moussant', 118.40);
```

```
FAC_NUM     FAC_DATE
----------- ------------
56          2001-08-11
```

```
79          2001-08-17
101         2001-09-01
```

```
LIF_NUM     FAC_NUM     LIF_ARTICLE      LIF_MONTANT
----------- ----------- ---------------- ------------------
11          56          Cartable         123.50
13          56          Crayons          17.52
15          56          Trousse          29.99
16          56          Feutres          11.25
18          56          Ardoise          44.21
19          79          Cigarettes       21.55
21          79          Whisky           147.65
22          79          Petite pépé      12857.59
27          101         Savon            16.24
28          101         Serviette        45.00
32          101         Shampoing        44.22
33          101         Bain moussant    118.40
```

Mais voilà... La facture n°79 (lignes en gras) doit être annulée et pour ce faire, on doit recopier les lignes de la table T_LIGNE_FACTURE_LIF correspondant à cette facture, tout en répondant à deux critères bien précis :
- la clef de ces nouvelles lignes doit être négative
- l'ordre de ces nouvelles lignes doit être le même que les lignes d'origine...
- les montants doivent être négatifs

Sauriez vous alimenter ces nouvelles lignes et extraire le résultat global comme ceci ?

```
FAC_NUM     LIF_ARTICLE      LIF_MONTANT
----------- ---------------- ------------------
56          Cartable         123.50
56          Crayons          17.52
56          Trousse          29.99
56          Feutres          11.25
56          Ardoise          44.21
79          Cigarettes       21.55
79          Cigarettes       -21.55
79          Whisky           -147.65
79          Whisky           147.65
79          Petite pépé      12857.59
79          Petite pépé      -12857.59
101         Savon            16.24
101         Serviette        45.00
101         Shampoing        44.22
101         Bain moussant    118.40
```

On constate que la clause de tri provoque l'apparition de la ligne de contrepassation d'écriture juste après la ligne de l'écriture comptable originale...

A vous de jouer !

cliquez pour répondre et titrez votre mail " **Problème n°10 - symétrie négative** "

III - Exercices - 2° partie

III-A - Problème n° 11 - premiers

En partant d'une simple table contenant les nombre de 0 à 10000, sauriez vous en une seule requête obtenir tous les nombres premiers entre 1 et 10000 ?

Pour cela repartons de la table générée à l'exercice 6 :

```
CREATE TABLE T_ENTIER_ENT
```

```
(ENT_N INTEGER);
```

```
INSERT INTO T_ENTIER_ENT VALUES (0);
INSERT INTO T_ENTIER_ENT VALUES (1);
INSERT INTO T_ENTIER_ENT VALUES (2);
INSERT INTO T_ENTIER_ENT VALUES (3);
INSERT INTO T_ENTIER_ENT VALUES (4);
INSERT INTO T_ENTIER_ENT VALUES (5);
INSERT INTO T_ENTIER_ENT VALUES (6);
INSERT INTO T_ENTIER_ENT VALUES (7);
INSERT INTO T_ENTIER_ENT VALUES (8);
INSERT INTO T_ENTIER_ENT VALUES (9);
...
```

```
ENT_N
------
0
1
2
3
4
5
6
7
8
9
...
```

C'est plus facile que cela en à l'air, si l'on se souvient qu'un nombre premier est un nombre qui n'est divisible que par lui même et 1. Ou plutôt qu'un nombre qui n'est pas premier est divisible par un nombre compris entre 2 et lui-même moins un...

En l'occurrence, le résultat doit être le suivant :

```
ENT_N
-----------
0
1
2
3
5
7
11
13
17
19
23
29
31
...
```

On pourra bien entendu disserter éternellement sur la fait que 0 est premier ou non ! Mais sauriez vous exprimer la solution en une seule requête ?

cliquez pour répondre et titrez votre mail " **Problème n°11 - premiers** "

III-B - Problème n° 12 - la traduction

Voici un probléme intéressant pour les utilisateurs d'applications multilingues. C'est à nouveau un internaute, Sylvain qui m'a posé une colle ! Voici ce que Sylvain à posté :
"Bonjour, j'ai une table avec trois colonnes : id, langue, titre avec comme clé (id, langue). Je voudrais faire une requête du genre :

```
SELECT titre, id, langue
```

```
FROM   matable
WHERE  "(langue='francais' ou alors langue='anglais' si n'existe pas en francais)"
```

En gros, je veux les résultats, de préférence en francais, mais si il n'y a pas de ligne pour le français, alors je voudrais la ligne pour la langue anglaise."
Est-ce possible ?

Voici le jeu d'essais avec lequel j'ai travaillé :

```
CREATE TABLE T_TRADUCTION_TDR
(TDR_ID      INTEGER,
 TDR_LANGUE  VARCHAR(8),
 TDR_LIBELLE VARCHAR(256));
```

```
INSERT INTO T_TRADUCTION_TDR VALUES (1, 'Français', 'Erreur irrécupérable');
INSERT INTO T_TRADUCTION_TDR VALUES (1, 'Anglais',  'Fatal error');
INSERT INTO T_TRADUCTION_TDR VALUES (3, 'Français', 'Disque saturé');
INSERT INTO T_TRADUCTION_TDR VALUES (4, 'Anglais',  'Memory fault');
```

Quelques-uns de mes collègues ont trouvé d'élégantes solutions... Sauriez vous trouver au moins une façon de faire ?

Voici en tout cas le résultat attendu :

```
TDR_LIBELLE                TDR_ID      TDR_LANGUE
-------------------------- ----------- ----------
Erreur irrécupérable       1           Français
Disque saturé              3           Français
Memory fault               4           Anglais
```

A vos claviers !

cliquez pour répondre et titrez votre mail " **Problème n°12 - traduction** "

III-C - Problème n° 13 - scores

La question vient d'un forum. Étant anonyme, je vous la donne telle quelle :
"Je fait un petit QCM sur internet et je crée 2 tables dans ma base de données. L'une de ces tables contient les informations sur les sondés (nom,prénom,etc..) ainsi que leurs réponses à mes questions (table T_PANEL_PNL). L'autre table contient, elle, les réponses justes à mes questions (T_REPONSES_RPS). J'aimerais savoir si il est possible à l'aide de requètes de comparer ces 2 tables, c'est à dire que j'aimerais comparer les réponses de mes sondès aux réponses justes afin de déterminer les scores..."

Voici le jeu d'essais utilisé :

```
CREATE TABLE T_PANEL_PNL
(PNL_NOM VARCHAR(16),
 PNL_REPONSE1 INTEGER,
 PNL_REPONSE2 INTEGER,
 PNL_REPONSE3 INTEGER,
 PNL_REPONSE4 INTEGER,
 PNL_REPONSE5 INTEGER);

CREATE TABLE T_REPONSES_RPS
(RPS_REPONSE1 INTEGER,
 RPS_REPONSE2 INTEGER,
 RPS_REPONSE3 INTEGER,
 RPS_REPONSE4 INTEGER,
 RPS_REPONSE5 INTEGER);
```

```
INSERT INTO T_PANEL_PNL VALUES ('Pierre',  1, 2, 3, 0, 0);
INSERT INTO T_PANEL_PNL VALUES ('Paul',    1, 5, 3, 2, 1);
```

```
INSERT INTO T_PANEL_PNL VALUES ('Jacques', 0, 3, 3, 2, 2);

INSERT INTO T_REPONSES_RPS VALUES (1, 4, 3, 2, 1);
```

Sauriez vous trouver une réponse pour qu'en une seule requête apparaisse la réponse suivante :

```
PNL_NOM          BONNES_REPONSES
---------------- ---------------
Paul             4
Pierre           2
Jacques          2
```

A vos claviers !

cliquez pour répondre et titrez votre mail " **Problème n°13 - scores** "

III-D - Problème n° 14 - tranches d'âge

C'est encore un anonyme qui me pose une question sur le forum SQLpro...

Bonjour,
Sur une table comprenant une date de naissance, je cherche à obtenir le nombre d'enregistrements dont les personnes ont de 1 jour à 18 ans, de 18 à 40 ans et les plus de 40 ans. Qui peut m'indiquer la requête à effectuer ?

Reprenons les données de l'exercice 5 :

```
CREATE TABLE T_CLIENT_CLI
(CLI_ID             INTEGER,
 CLI_NOM            VARCHAR(16),
 CLI_DATE_NAISSANCE DATE);
```

```
INSERT INTO T_CLIENT_CLI VALUES (24, 'HESS',      '1984-08-30');
INSERT INTO T_CLIENT_CLI VALUES (25, 'CHATON',    '1938-12-31');
INSERT INTO T_CLIENT_CLI VALUES (26, 'PLATONOFF', '1960-03-29');
INSERT INTO T_CLIENT_CLI VALUES (27, 'LETERRIER', '1945-08-26');
INSERT INTO T_CLIENT_CLI VALUES (28, 'MONTEIL',   '1985-03-23');
INSERT INTO T_CLIENT_CLI VALUES (29, 'SPITHAKIS', '1956-03-31');
INSERT INTO T_CLIENT_CLI VALUES (30, 'ORELL',     '1996-01-27');
INSERT INTO T_CLIENT_CLI VALUES (31, 'MARTINET',  '1946-10-24');
INSERT INTO T_CLIENT_CLI VALUES (32, 'RAY',       '1979-12-20');
INSERT INTO T_CLIENT_CLI VALUES (33, 'TARSAC',    '1992-08-14');
INSERT INTO T_CLIENT_CLI VALUES (34, 'COULOMB',   '1963-01-01');
INSERT INTO T_CLIENT_CLI VALUES (35, 'SAVY',      '1942-10-03');
INSERT INTO T_CLIENT_CLI VALUES (36, 'DAVID',     '1922-09-19');
INSERT INTO T_CLIENT_CLI VALUES (37, 'FORGEOT',   '1989-09-13');
INSERT INTO T_CLIENT_CLI VALUES (38, 'BERGER',    '1946-10-16');
INSERT INTO T_CLIENT_CLI VALUES (39, 'DOUBLET',   '1961-03-02');
INSERT INTO T_CLIENT_CLI VALUES (40, 'MATHIEU',   '1990-02-24');
INSERT INTO T_CLIENT_CLI VALUES (41, 'MOURGUES',  '1953-01-14');
INSERT INTO T_CLIENT_CLI VALUES (42, 'PIERROT',   '1933-02-11');
INSERT INTO T_CLIENT_CLI VALUES (44, 'ZAMPIERO',  '1985-01-19');
INSERT INTO T_CLIENT_CLI VALUES (45, 'PASCOT',    Null        );
INSERT INTO T_CLIENT_CLI VALUES (46, 'MECHRI',    '1950-08-11');
```

```
CLI_ID      CLI_NOM          CLI_DATE_NAISSANCE
----------- ---------------- ------------------
24          HESS             1934-08-30
25          CHATON           1938-12-31
26          PLATONOFF        1960-03-29
27          LETERRIER        1945-08-26
28          MONTEIL          1935-03-23
29          SPITHAKIS        1956-03-31
30          ORELL            1956-01-27
31          MARTINET         1946-10-24
```

```
32          RAY             1939-12-20
33          TARSAC          1942-08-14
34          COULOMB         1963-01-01
35          SAVY            1942-10-03
36          DAVID           1922-09-19
37          FORGEOT         1949-09-13
38          BERGER          1946-10-16
39          DOUBLET         1961-03-02
40          MATHIEU         1940-02-24
41          MOURGUES        1953-01-14
42          PIERROT         1933-02-11
44          ZAMPIERO        1945-01-19
45          PASCOT          NULL
46          MECHRI          1950-08-11
```

Le résultat de votre requête doit apparaître comme suit :

```
TRANCHE_MIN TRANCHE_MAX NOMBRE
----------- ----------- -----------
0           18          6
18          40          4
40          9999        13
```

Sauriez vous répondre à notre internaute et de façon générique, c'est à dire quelle que soit les futures tranches d'âge à gérer ?

cliquez pour répondre et titrez votre mail " **Problème n°14 - tranches d'age** "

III-E - Problème n° 15 - titres sans article

Dans les titres des livres (comme dans les titres des films) il y a souvent un article (le, la les, un, une, des...) situé en tête qui parasite l'apparition des oeuvres dans l'ordre alphabétique. D'ou la question de notre internaute, "Bouchon", son nom de code sur le forum SQL de www.developpez.com, qui demande : "*J'aimerais, s'il y a un article au début du titre ("LE","LA","L'"), que celui-ci soit retiré de la chaine et se retrouve à la fin de celle-ci et entre parenthèses.*"
Voici notre jeu d'essais :

```
CREATE TABLE T_OUVRAGES_OVG
(OVG_TITRE VARCHAR(64));
```

```
INSERT INTO T_OUVRAGES_OVG VALUES ('Le journal de Raymond');
INSERT INTO T_OUVRAGES_OVG VALUES ('Le vélo d''Alphonse');
INSERT INTO T_OUVRAGES_OVG VALUES ('Découvrir l''escalade');
INSERT INTO T_OUVRAGES_OVG VALUES ('L''âge du capitaine');
INSERT INTO T_OUVRAGES_OVG VALUES ('Comment dévisser la vis');
INSERT INTO T_OUVRAGES_OVG VALUES ('D''amour et d''eau fraiche');
```

Contenu de la table	Résultat attendu
`SELECT TITRES FROM OUVRAGES`	`???`

`TITRE`
`----------`
`Le journal de Raymond`
`Le vélo d'Alphonse`
`Découvrir l'escalade`
`L'âge du capitaine`
`Comment dévisser la vis`
`D'amour et d'eau fraiche`
	`TITRE`
	`----------`
	`journal de Raymond (Le)`
	`vélo d'Alphonse (Le)`
	`Découvrir l'escalade`
	`âge du capitaine (L')`
	`Comment dévisser la vis`
	`amour et d'eau fraiche (D')`

Une colle finalement assez facile. Je l'ai résolue en 10 minutes... Sauriez vous être aussi performant que moi ???
Sauriez vous répondre à notre internaute ?

cliquez pour répondre et titrez votre mail " **Problème n°15 - titres sans article** "

III-F - Problème n° 16 - Tri des titres

Voici encore un problème de tri... Il m'a été posé par un internaute sur le forum SQL de www.developpez.com. Il pourrait être le suite de notre exercice précédent.
Notre internaute voulait trier les titres de ses livres par ordre aplhabétique et laisser tous ceux qui ne commençent pas par une lettre à la fin. Mais en plus il exige que les titres commençant par un chiffre soient positionnés par rapport à leur écriture en lettre... Autrement dit "2001 l'odyssée de l'espace', doit figurer après "Ben Hur" et avant "Double assassinat dans la rue morgue.
Sauriez vous lui répondre ?

Exemple :

```
CREATE TABLE T_LIVRE_LVR
(LVR_TITRE  VARCHAR(64));
```

```
INSERT INTO T_LIVRE_LVR VALUES ('À la recherche du temps perdu');
INSERT INTO T_LIVRE_LVR VALUES ('La bible') ;
INSERT INTO T_LIVRE_LVR VALUES ('2001 l''odyssée de l''espace');
INSERT INTO T_LIVRE_LVR VALUES ('Ben Hur');
INSERT INTO T_LIVRE_LVR VALUES ('Double assassinat dans la rue morgue');
INSERT INTO T_LIVRE_LVR VALUES ('Les trois mousquetaires');
```

Si vous avez brillament répondu à notre précédent puzzle, alors vous devriez solutionner celui là !

Voici le résultat attendu :

```
LVR_TITRE
----------------------------------------------------------------
À la recherche du temps perdu
Ben Hur
2001 l'odyssée de l'espace
Double assassinat dans la rue morgue
La bible
Les trois mousquetaires
```

cliquez pour répondre et titrez votre mail " **Problème n°16 - tri des titres** "

III-G - Problème n° 17 - Appariement

Le terme appariement vient de mettre en paires, c'est à dire apparier. On dit appariement et et non apparition comme s'il s'agissait de fantôme par exemple...
Cette demande, qui m'a donné du fil à retordre, provient du forum SQL du site www.developpez.com que j'anime.
Il s'agit de récupérer les valeurs de la colonne A pour laquelle les données de la colonne B sont exactement les mêmes qu'une autre valeur de la colonne A.

```
CREATE TABLE T_PAIRE_PER
(PER_1 CHAR(2),
 PER_2 CHAR(2));
```

```
INSERT INTO T_PAIRE_PER VALUES ('A1', 'B1');
INSERT INTO T_PAIRE_PER VALUES ('A1', 'B2');
INSERT INTO T_PAIRE_PER VALUES ('A2', 'B1');
INSERT INTO T_PAIRE_PER VALUES ('A2', 'B3');
INSERT INTO T_PAIRE_PER VALUES ('A3', 'B1');
INSERT INTO T_PAIRE_PER VALUES ('A3', 'B3');
INSERT INTO T_PAIRE_PER VALUES ('A3', 'B7');
INSERT INTO T_PAIRE_PER VALUES ('A3', 'B4');
INSERT INTO T_PAIRE_PER VALUES ('A4', 'B1');
INSERT INTO T_PAIRE_PER VALUES ('A4', 'B3');
INSERT INTO T_PAIRE_PER VALUES ('A5', 'B2');
```

```
INSERT INTO T_PAIRE_PER VALUES ('A6', 'B1');
INSERT INTO T_PAIRE_PER VALUES ('A6', 'B4');
INSERT INTO T_PAIRE_PER VALUES ('A6', 'B7');
INSERT INTO T_PAIRE_PER VALUES ('A6', 'B3');
INSERT INTO T_PAIRE_PER VALUES ('A8', 'B1');
INSERT INTO T_PAIRE_PER VALUES ('A8', 'B4');
INSERT INTO T_PAIRE_PER VALUES ('A8', 'B7');
INSERT INTO T_PAIRE_PER VALUES ('A8', 'B3');
INSERT INTO T_PAIRE_PER VALUES ('A9', 'B1');
INSERT INTO T_PAIRE_PER VALUES ('A9', 'B4');
INSERT INTO T_PAIRE_PER VALUES ('A9', 'B8');
INSERT INTO T_PAIRE_PER VALUES ('A9', 'B3');
```

Pour bien comprendre le problème, nous allons analyser ensemble les deux premières valeurs de la colonne A. Démarrons avec la valeur 'A1' de la colonne PER_1 :

```
PER_1 PER_2
----- -----
A1    B1
A1    B2

A2    B1
A2    B3
A3    B1
A3    B3
A3    B7
A3    B4
A4    B1
A4    B3
A5    B2
A6    B1
A6    B4
A6    B7
A6    B3
A8    B1
A8    B4
A8    B7
A8    B3
A9    B1
A9    B4
A9    B8
A9    B3

Pour la valeur 'A1' de la
 colonne PER_1,
la colonne PER_2 possède les
 valeurs :
'B1' et 'B2'.

Quelles sont les autres valeurs
 de PER_1
qui ont en regard les
 valeurs 'B1' et 'B2'
dans la colonne PER_2 ?
```

```
PER_1 PER_2
----- -----
A1    B1
A1    B2

A2    B1
A2    B3

A3    B1
A3    B3
A3    B7
A3    B4

A4    B1
A4    B3

A5    B2

A6    B1
A6    B4
A6    B7
A6    B3

A8    B1
A8    B4
A8    B7
A8    B3

A9    B1
A9    B4
A9    B8
A9    B3

Aucune autre valeur de PER_1
n'a simultanément les valeurs
'B1' et 'B2' dans la colonne PER_2
```

Analysons maintenant la valeur 'A2' de la colonne PER_1 :

```
PER_1 PER_2
----- -----
A1    B1
A1    B2

A2    B1
A2    B3

A3    B1
A3    B3
A3    B7
A3    B4
```

```
PER_1 PER_2
----- -----
A1    B1
A1    B2

A2    B1
A2    B3

A3    B1
A3    B3
A3    B7
A3    B4
```

```
A4    B1
A4    B3
A5    B2
A6    B1
A6    B4
A6    B7
A6    B3
A8    B1
A8    B4
A8    B7
A8    B3
A9    B1
A9    B4
A9    B8
A9    B3

Pour la valeur 'A2' de la
 colonne PER_1,
la colonne PER_2 possède les
 valeurs :
'B1' et 'B3'.

Quelles sont les autres valeurs
 de PER_1
qui ont en regard les
 valeurs 'B1' et 'B3'
dans la colonne PER_2 ?
```

```
A4    B1
A4    B3

A5    B2

A6    B1
A6    B4
A6    B7
A6    B3

A8    B1
A8    B4
A8    B7
A8    B3

A9    B1
A9    B4
A9    B8
A9    B3

'A3', 'A4', 'A6', 'A8' et 'A9'
 ont simultanément
les valeurs 'B1' et 'B3' dans
 la colonne PER_2
mais certaines
 occurrences ('A3', 'A6', 'A8'
 et 'A9')
ont en sus d'autres valeurs...
Seule la valeur A4 est donc à retenir.
```

Ici les seuls résulats valables sont 'A2' avec 'A4', puis 'A3' avec 'A6' et 'A8'. En effet les données de la colonne PER_2 sont identiques pour 'A2' et 'A4' ('B1' et 'B3'). Les données de la colonne B sont aussi identiques pour 'A3', 'A6' et 'A8', (' B1', 'B3', 'B4', 'B7').
Le résultat doit donc être :

```
PER_1
-----
A2
A3
A4
A6
A8
```

Simple en apparence, ce problème s'avère en fin de compte assez complexe. Mais j'ai déjà commencé le travail de décortication pour vous mettre sur la voie...

cliquez pour répondre et titrez votre mail " **Problème n°17 - appariement** "

III-H - Problème n° 18 - Meilleure correspondance

Voici une question qui m'a été posée sur le forum SQL de www.developpez.com par neness...

Je possède une table T_ROUTE_RTE ainsi remplie :

```
CREATE TABLE T_ROUTE_RTE
(RTE_DESTINATION VARCHAR(32),
 RTE_CODE        CHAR(16));
```

```
INSERT INTO T_ROUTE_RTE VALUES ('albania-Mobile', '0035538');
INSERT INTO T_ROUTE_RTE VALUES ('albania-Mobile', '0035569');
INSERT INTO T_ROUTE_RTE VALUES ('albania-Mobile', '0035560');
INSERT INTO T_ROUTE_RTE VALUES ('SFR-Mobile',     '0077280');
```

```
INSERT INTO T_ROUTE_RTE VALUES ('SFR-Mobile',     '0077390');
INSERT INTO T_ROUTE_RTE VALUES ('BOUYGE_TEL',     '0078452');
```

Pour chaque RTE_DESTINATION, je recherche la partie de la colonne RTE_CODE dont les données sont communes dans le sens de lecture. Notre quidam essaye donc de trouver les meilleures correspondances partielles de RTE_CODE communes à une même famille de RTE_DESTINATION. Notez qu'il s'agit d'ailleurs d'une opération proche d'un calcul d'agrégat...
Dans l'exemple donné, la requête doit renvoyer :

```
RTE_DESTINATION      RTE_CODE
-------------------- ---------------
albania-Mobile       00355
BOUYGE_TEL           0078452
SFR-Mobile           0077
```

En effet, '00355' est la plus grande partie commune commençant chaque RTE_CODE pour 'albania-Mobile', de même '0078452' pour 'BOUYGE_TEL' et de même '0077' pour 'SFR-Mobile'.
Autrementy dit, quelle est la partie de RTE_DESTINATION correspondant aux premiers caractères communs à chaque occurence de la colonne RTE_CODE.

Simple, mais comment faire ?

cliquez pour répondre et titrez votre mail " **Problème n°18 meilleure correspondance** "

III-I - Problème n° 19 - La médiane

Dans un forum consacré à SQL Server de Microsoft, j'ai trouvé cette question forte intéressante :
Bonjour,Je doit calculer des données statistiques et la médiane est absente...Comment cela peut il être réalisé en une requête ?.

Comme ce problème s'avère en définitif assez complexe, le voici exprimés pour différents cas...

1) les données à calculées sont toutes différentes (il n'y a pas de doublon) et en nombre impaires :
Voici le jeu de données que j'ai utilisé pour ce cas :

```
CREATE TABLE T_STATISTIQUES_STT
(STT_ID     INT,
 STT_VALEUR FLOAT NOT NULL);
```

```
INSERT INTO T_STATISTIQUES_STT VALUES (1, 22.0);
INSERT INTO T_STATISTIQUES_STT VALUES (2, 27.5);
INSERT INTO T_STATISTIQUES_STT VALUES (3, 22.5);
INSERT INTO T_STATISTIQUES_STT VALUES (4, 24.0);
INSERT INTO T_STATISTIQUES_STT VALUES (5, 23.0);
```

La solution étant :

```
STT_VALEUR
----------
23.0
```

2) les données à calculées sont toutes différentes (il n'y a pas de doublon) et en nombre paires :
Rajoutons au jeu précédent la ligne :

```
INSERT INTO T_STATISTIQUES_STT VALUES (6, 23.5);
```

La solution devient :

```
STT_VALEUR
----------
23.25
```

3) voyons ce qu'il se passe avec des valeurs identiques multiples et un nombre de lignes impair :
Rajoutons au jeu précédent les lignes :

```
INSERT INTO T_STATISTIQUES_STT VALUES (7, 22.0);
INSERT INTO T_STATISTIQUES_STT VALUES (8, 22.0);
INSERT INTO T_STATISTIQUES_STT VALUES (9, 22.0);
INSERT INTO T_STATISTIQUES_STT VALUES (10, 22.0);
INSERT INTO T_STATISTIQUES_STT VALUES (11, 22.0);
```

La solution devient :

```
STT_VALEUR
----------
22.0
```

C'est le genre de problème qui a passionné les spécialistes de SQL que sont Joe Celko et Chris Date. Sauriez vous au moins traiter le cas n° 1 ?

cliquez pour répondre et titrez votre mail " **Problème n°19 - mediane** "

III-J - Problème n° 20 - Insertion en bloc

Voici un problème intéressant pour s'affanchir de certaines procédures stockées bêtes et méchante...Comment insérer de multiple ligne, par exemple dans une table de jointure, sans faire appel à une boucle et à l'aide d'une seule et unique requête ?

Notre problème est le suivant. Soit une table d'objets définie comme suit :

```
CREATE TABLE T_OBJET_OBJ
(OBJ_OBJET VARCHAR(16));
```

```
INSERT INTO T_OBJET_OBJ VALUES ('feu tricolore');
INSERT INTO T_OBJET_OBJ VALUES ('drapeau français');
```

... une table listant des couleurs :

```
CREATE TABLE T_COULEUR_CLR
(CLR_COULEUR   VARCHAR(8));
```

```
INSERT INTO T_COULEUR_CLR VALUES ('blanc');
INSERT INTO T_COULEUR_CLR VALUES ('bleu');
INSERT INTO T_COULEUR_CLR VALUES ('vert');
INSERT INTO T_COULEUR_CLR VALUES ('rouge');
INSERT INTO T_COULEUR_CLR VALUES ('orange');
```

... et une table de jointure associant les deux, table actuellement vide :

```
CREATE TABLE T_OBJET_COULEUR_OBC
(OBJ_OBJET   VARCHAR(16),
 CLR_COULEUR VARCHAR(8));
```

Nous aimerions parvenir à remplir de la sorte la table d'association T_OBJET_COULEUR_OBC :

```
OBJ_OBJET        CLR_COULEUR
---------------- -----------
feu tricolore    vert
feu tricolore    rouge
feu tricolore    orange
drapeau français blanc
drapeau français bleu
drapeau français rouge
```

Sauriez-vous trouver en une seule requête comment procéder à l'insertion massive de ces six lignes ?

cliquez pour répondre et titrez votre mail " **Problème n°20 - insertion en bloc** "

IV - Exercices - 3° partie

IV-A - Problème n° 21 - Ordonner, réordonner !

Un problème d'ordonnancement avec SQL m'a été proposé par Mouse sur le forum SQL de developpez. Il s'agit de modifier une colonne d'une table assurant l'ordonnancement des données de la table.
Par exemple, la colonne position est numérotée de 1 à 5 et l'on désire remplacer la valeur 4
par la valeur 2 tout en préservant la continuité de l'ordre de 1 à 5...
Un petit exemple permettra de mieux comprendre la chose... Voici une table de pays. La colonne position indique la position que le pays obtiendra dans la liste une fois quelle sera triée.

```
CREATE TABLE T_PAYS_PAY
(PAY_NOM      VARCHAR(16),
 PAY_POSITION INTEGER);
```

```
INSERT INTO T_PAYS_PAY VALUES ('Allemagne', 1);
INSERT INTO T_PAYS_PAY VALUES ('Belgique', 2);
INSERT INTO T_PAYS_PAY VALUES ('Croatie', 3);
INSERT INTO T_PAYS_PAY VALUES ('Espagne', 4);
INSERT INTO T_PAYS_PAY VALUES ('France', 5);
INSERT INTO T_PAYS_PAY VALUES ('Grèce', 6);
```

```
SELECT *
FROM   T_PAYS_PAY;
```

```
PAY_NOM          PAY_POSITION
---------------- ------------
Allemagne        1
Belgique         2
Croatie          3
Espagne          4
France           5
Grèce            6
```

Notre utilisateur cherche à recombiner l'ordre de la liste en faisant passer la France en tête, sans pour autant bousculer l'ordre des autres pays. Tant est si bien que finalement, après cette requête de mise à jour, la table doit apparaître comme ceci :

```
PAY_NOM          PAY_POSITION
---------------- ------------
Allemagne        2
Belgique         3
Croatie          4
Espagne          5
France           1
Grèce            6
```

1) Sauriez-vous faire cela en une seule requête ?

2) Pourriez-vous proposer une solution générique pour cette permutation d'ordre de tri ?
Nous allons corser le problème en admettant que la colonne PAY_POSITION puisse prendre des valeurs non continues :
Voici donc les nouvelles conditions du jeu d'essai :

```
DELETE FROM T_PAYS_PAY;
```

```
INSERT INTO T_PAYS_PAY VALUES ('Allemagne', 11);
INSERT INTO T_PAYS_PAY VALUES ('Belgique', 8);
INSERT INTO T_PAYS_PAY VALUES ('Croatie', 9);
INSERT INTO T_PAYS_PAY VALUES ('Espagne', 5);
INSERT INTO T_PAYS_PAY VALUES ('France', 12);
INSERT INTO T_PAYS_PAY VALUES ('Grèce', 7);
```

```
PAY_NOM          PAY_POSITION
---------------- ------------
Espagne          5
Grèce            7
Belgique         8
Croatie          9
Allemagne        11
France           12
```

3) Sauriez-vous renuméroter les positions en continuité de 1 à n (n étant le nombre de lignes dans la table) quelque soit les données de la colonne position, mais en gardant toujours le même ordre... ? Et, en partant des données ci dessus, obtenir le résultat suivant :

```
PAY_NOM          PAY_POSITION
---------------- ------------
Espagne          1
Grèce            2
Belgique         3
Croatie          4
Allemagne        5
France           6
```

A vos claviers...

cliquez pour répondre et titrez votre mail "**Problème n°21 - Ordonner, réordonner**"

IV-B - Problème n° 22 - Jointure hétérogène

Voici un internaute qui a hérité d'un modèle particulièrement mal modélisé. En effet, dans l'une des colonnes de la première table ("public") on trouve toutes les données relatives aux références de l'autres tables sous la forme suivante : [clef1].[clef2].[clef3]...

Problème... comment réaliser une requête qui fait la jointure entre ces deux tables ?
Voici le jeu d'essais de notre internaute :

```
CREATE TABLE T_PUBLIC_PBL
(PBL_ID  INTEGER,
 PBL_NOM VARCHAR(16));

CREATE TABLE T_PRESTATION_PST
(PST_ID      INTEGER,
 PST_LIBELLE VARCHAR(25),
 PST_PUBLIC  VARCHAR(32));
```

```
INSERT INTO T_PUBLIC_PBL VALUES (1, 'particulier');
INSERT INTO T_PUBLIC_PBL VALUES (3, 'entreprise');
INSERT INTO T_PUBLIC_PBL VALUES (2, 'groupe');
```

```
INSERT INTO T_PRESTATION_PST VALUES (1, 'Tour en voiture',          '[3]');
INSERT INTO T_PRESTATION_PST VALUES (2, 'Compétition à plusieurs', '[3].[2]');
INSERT INTO T_PRESTATION_PST VALUES (3, 'Course d''endurance ',     '[1].[2]');
INSERT INTO T_PRESTATION_PST VALUES (4, 'Bapteme ',                 '[1]');
INSERT INTO T_PRESTATION_PST VALUES (5, 'Course en tandem ',        '[2]');
```

Il modélise des prestations sportives en visant différents publics.
Les données sont donc les suivantes :

```
SELECT * FROM T_PUBLIC_PBL
PBL_ID      PBL_NOM
----------- ----------------
1           particulier
3           entreprise
2           groupe
```

```
SELECT * FROM T_PRESTATION_PST
PST_ID      PST_LIBELLE
       PST_PUBLIC

----------- ------------------------- ----------------------------
1           Tour en voiture
       [3]
2           Compétition à
 plusieurs   [3].[2]
3           Course
 d'endurance        [1].[2]
4           Bapteme                   [1]
5           Course en tandem          [2]
```

Et notre internaute voudrait réaliser la jointure entre la colonne PBL_ID de la table T_PUBLIC_PBL et chacune des valeurs situées entre crochets de la colonne PST_PUBLIC de la table T_PRESTATION_PST., Bref, en une requête sauriez vous afficher les données comme ceci :

```
PBL_ID      PBL_NOM       PST_ID      PST_LIBELLE               PST_PUBLIC
----------- ------------- ----------- ------------------------- ----------------
3           entreprise    1           Tour en voiture           [3]
2           groupe        2           Compétition à plusieurs   [3].[2]
3           entreprise    2           Compétition à plusieurs   [3].[2]
1           particulier   3           Course d'endurance        [1].[2]
2           groupe        3           Course d'endurance        [1].[2]
1           particulier   4           Bapteme                   [1]
2           groupe        5           Course en tandem          [2]
```

cliquez pour répondre et titrez votre mail "**Problème n°22 - Jointure hétérogène**"**mailto:sqlpro_NO@SPAM_club-internet.fr?subject=Jointure%20heterogene%20multiple**

IV-C - Problème n° 23 - Insertion conditionnelle

En voilà une question intéressante : comment insérer une ligne dans une table, uniquement si elle n'y est pas déjà ? Autrement dit il me faut une requête capable d'insérer la ligne si elle n'existe pas, sinon, de ne pas procéder à l'insertion...
Pour mieux comprendre la chose voici le modèle de données et les différents jeu d'essais.

```
CREATE TABLE MATABLE
( COL1 INTEGER,
  COL2 VARCHAR(16))  ;
```

```
INSERT INTO MATABLE VALUES (1, 'toto') ;
INSERT INTO MATABLE VALUES (1, 'titi') ;
```

Pourriez-vous :
1) créer une requête INSERT dans cette table avec les valeurs 1, toto et faire en sorte que rien ne soit inséré sans générer d'erreur ?
2) insérer 3, tutu avec cette même requête ??

cliquez pour répondre et titrez votre mail "**Problème n°23 - Insertion conditionnelle**"

IV-D - Problème n° 24 - Un arbre à deux niveaux

Posté par the tigrou...,dans le forum SQL de developpez :
Je dois écrire des requêtes qui font appel à des lignes appartenant à des tables differentes. Par exemple une table "liste entreprise" et une table "liste salarié". Comment selectionner les salariés de chaque entreprise de façon a faire une arboresence à deux niveaux ?
Pour vous aider, voici le jeu d'essais concocté par mes soins :

```
CREATE TABLE T_ENTREPRISE_ETP
(ETP_ID    INTEGER NOT NULL PRIMARY KEY,
 ETP_NOM   VARCHAR(16)) ;

CREATE TABLE T_EMPLOYEE_EMP
(EMP_ID    INTEGER NOT NULL PRIMARY KEY,
 EMP_NOM   VARCHAR(16),
 ETP_ID    INTEGER NOT NULL
               FOREIGN KEY REFERENCES T_ENTREPRISE_ETP (ETP_ID)) ;
```

```
INSERT INTO T_ENTREPRISE_ETP
VALUES (1, 'IBM') ;
INSERT INTO T_ENTREPRISE_ETP
VALUES (2, 'EDF') ;

INSERT INTO T_EMPLOYEE_EMP
VALUES (1, 'Durand', 1) ;
INSERT INTO T_EMPLOYEE_EMP
VALUES (2, 'Dupont', 1) ;
INSERT INTO T_EMPLOYEE_EMP
VALUES (3, 'Dubois', 1) ;
INSERT INTO T_EMPLOYEE_EMP
VALUES (4, 'Duval', 1) ;
INSERT INTO T_EMPLOYEE_EMP
VALUES (5, 'Dupond', 2) ;
INSERT INTO T_EMPLOYEE_EMP
VALUES (6, 'Duhamel', 2) ;
INSERT INTO T_EMPLOYEE_EMP
VALUES (7, 'Dufour', 2) ;
```

Ce qui Tigrou voudrait, doit se présenter comme ceci :

```
NOM
-----------------
IBM
 Durand
 Dupont
 Dubois
 Duval
EDF
 Dupond
 Duhamel
 Dufour
```

A vous de jouer !

cliquez pour répondre et titrez votre mail "**Problème n°24 - Arbre à deux niveaux**"

IV-E - Problème n° 25 - Éclater des lignes

C'est un de mes clients qui m'a mis au défi de trouver comment réaliser cette transformation...
Partant d'une table de "LIGNE" ainsi modélisée et des données suivantes :

```
CREATE TABLE LIGNE
(ID_LIGNE INTEGER,
```

```
 ID_REF   INTEGER,
 QUANTITE INTEGER);
```

```
INSERT INTO LIGNE (ID_LIGNE, ID_REF, QUANTITE) VALUES (1, 1, 4);
INSERT INTO LIGNE (ID_LIGNE, ID_REF, QUANTITE) VALUES (2, 2, 1);
INSERT INTO LIGNE (ID_LIGNE, ID_REF, QUANTITE) VALUES (3, 3, 2);
INSERT INTO LIGNE (ID_LIGNE, ID_REF, QUANTITE) VALUES (4, 1, 1);
INSERT INTO LIGNE (ID_LIGNE, ID_REF, QUANTITE) VALUES (5, 2, 2);
INSERT INTO LIGNE (ID_LIGNE, ID_REF, QUANTITE) VALUES (6, 4, 2);
```

Que l'on présentera mieux sous sa forme extraite :

```
SELECT *
FROM   LIGNE
ORDER  BY ID_REF ;
```

```
ID_LIGNE    ID_REF      QUANTITE
----------- ----------- -----------
1           1           4
4           1           1
5           2           2
2           2           1
3           3           2
6           4           2
```

Auriez-vous l'amabilité de produire la table suivante :

```
ID_ARTICLE  ID_REF      SERIE       ID_LIGNE
----------- ----------- ----------- -----------
1           1           1           1
2           1           2           1
3           1           3           1
4           1           4           1
5           1           5           4
6           2           1           2
7           2           2           5
8           2           3           5
9           3           1           3
10          3           2           3
11          4           1           6
12          4           2           6
```

Observez bien la manière dont est construite cette table réponse : SERIE est un incrément dépendant uniquement de ID_REF, avec comme particularité qu'il doit éclater et numéroter à l'intérieur d'un même ID_REF et dans l'ordre ID_LIGNE, des lignes dont le nombre figure dans QUANTITE.
Autrement dit si pour ID_REF = 1, QUANTITE = 4, il y aura 4 lignes avec ce même ID_REF. Si pour ce même ID_REF de 1 il y a une ID_LIGNE différente, alors la numérotation doit de poursuivre en séquence...
Voici une présentation qui met en correspondance les lignes de la table sources avec la table réponse :

```
ID_REF      ID_LIGNE
 QUANTITE
----------- ----------- -----------
1           1           4
1           1           4
1           1           4
1           1           4
1           4           1
2           2           1
2           5           2
2           5           2
3           3           2
3           3           2
4           6           2
4           6           2
```

```
ID_REF      ID_LIGNE    SERIE
     ID_ARTICLE
----------- ----------- ----------- -----------
1           1           1
          1
1           1           2
          2
1           1           3
          3
1           1           4
          4
1           4           5
          5
2           2           1
          6
2           5           2
          7
```

```
2               5               3
        8
3               3               1
        9
3               3               2
        10
4               6               1
        11
4               6               2
        12
```

Les lignes grise représentent des doublons de la ligne précédente. D'ou lidée d'éclater chaque ligne en autant de fois que l'indique la valeur de QUANTITE...

cliquez pour répondre et titrez votre mail "**Problème n°25 - Éclater des lignes**"

IV-F - Problème n° 26 - Noms incrémentés

Dans un site Web un problème récurent vient des noms qui peuvent être homonymes. Pour lever toute ambiguité, notre quidam veut qu'en cas d'insertion le nom dont la colonne est muni d'une clause d'unicité ne soit par rejeté, mais qu'il lui soit ajouté un numéro calculé automatiquement en séquence...
Voici le modèle de table et les données de notre quidam :

```
CREATE TABLE T_UTILISATEUR_USR
(USR_ID    INTEGER NOT NULL PRIMARY KEY,
 USR_NOM   CHAR(16) NOT NULL UNIQUE);
```

```
INSERT INTO T_UTILISATEUR_USR VALUES (1, 'DUPONT');
INSERT INTO T_UTILISATEUR_USR VALUES (2, 'DURAND');
INSERT INTO T_UTILISATEUR_USR VALUES (3, 'DURAND1');
```

```
SELECT *
FROM   T_UTILISATEUR_USR
```

```
USR_ID      USR_NOM
----------- ----------------
1           DUPONT
2           DURAND
3           DURAND1
```

Par exemple; si je tente de rentrer DUPOND, il doit insérer DUPOND. Si je tente de rentrer DUPONT, il doit insérer DUPONT1. Si je tente de rentrer DURAND, il doit insérer DURAND2...
Comment faire cela en une seule requête ?

cliquez pour répondre et titrez votre mail "**Problème n°26 - Noms incrémentés**"

IV-G - Problème n° 27 - Une lettre un nom

Notre internaute recherche à construire une requete sql qui me permette d'extraire
d'une table la première entrée de chaque lettre alphabétique, donc de a à z, par rapport à une colonne contenant le nom d'une personne.
Par exemple, avec la table T_CONTACT_CTC contenant une colonne CTC_NOM avec les valeur suivantes :(Aaron, Abel, Babe, Boudet, Cabi...), il voudrait obtenir la
première personne ayant l'initiale A, la première personne ayant l'initiale B, etc... jusqu'à Z.
Voici la table pour ce faire :

```
CREATE TABLE T_CONTACT_CTC
(CTC_NOM   VARCHAR(32));
```

```
INSERT INTO T_CONTACT_CTC VALUES ('Aaron');
INSERT INTO T_CONTACT_CTC VALUES ('Abel');
INSERT INTO T_CONTACT_CTC VALUES ('Babet');
INSERT INTO T_CONTACT_CTC VALUES ('Boudet');
INSERT INTO T_CONTACT_CTC VALUES ('Brouard');
INSERT INTO T_CONTACT_CTC VALUES ('Cabu');
INSERT INTO T_CONTACT_CTC VALUES ('Cortès');
INSERT INTO T_CONTACT_CTC VALUES ('Cardeau');

INSERT INTO T_CONTACT_CTC VALUES ('Zoltan');
```

```
LETTRE     CTC_NOM
---------- -------------------------
A          Aaron
B          Babet
C          Cabu
...
Z          Zoltan
```

Mais n'ayons pas peur d'aller plus loin en posant quelques questions supplémentaires...
variante 1 : obtenir les noms des "secondes" personnes
variante 2 : obtenir les noms des niemes personnes
variante 3 : obtenir les noms des niemes personnes, mais si elle n'existe pas, alors la dernière !

variante 1 : obtenir les noms des "secondes" personnes	LETTRE CTC_NOM ------ ------------------------------ A Aaron B Babet C Cabu Z Zoltan
variante 2 : (n = 3)	LETTRE CTC_NOM ------ ------------------------------ B Brouard C Cortès
variante 3 : (n = 2) obtenir les noms des niemes personnes, mais si elle n'existe pas, alors la dernière !	LETTRE CTC_NOM ------ ------------------------------ A Abel B Boudet C Cardeau Z Zoltan

Comment faire cela et toujours en une seule requête ?

cliquez pour répondre et titrez votre mail "**Problème n°27 - Une lettre un nom**"

IV-H - Problème n° 28 - Filtrer les adresses IP

Pêché sur le forum SQL de www.developpez.com :

Bonjour à tous,
je travaille sur des adresses IP et je n'arrive pas a faire une selection comme ceci dans une requête sql :

```
Select adresseSource
from Y
where 10.120.12.1 <= adresseSource <= 10.130.23.1
```

- *10.120.12.1 correspond à l'attribut DebutIPValide dans la table X*
- *10.130.23.1 correspond à l'attribut FinIPValide dans la table X adresseSource fait partie de la table Y*
  Merci pour vos suggestions

Sauriez vous l'aider ?
Voici la table des adresses IP et quelques lignes bien suffisantes pour tester votre travail :

```
CREATE TABLE TIP
(TIP_ADR VARCHAR(15));
```

```
INSERT INTO TIP VALUES ('10.120.12.1');
INSERT INTO TIP VALUES ('10.130.23.1');
INSERT INTO TIP VALUES ('10.130.201.1');
INSERT INTO TIP VALUES ('10.13.11.1');
```

Le résultat de votre requête doit donner :

```
TIP_ADR
-------------------
10.120.12.1
10.130.23.1
```

Simple ? Pas sûr ! A vos codes...

cliquez pour répondre et titrez votre mail "**Problème n°28 - Filtrer les adresses IP**"

IV-I - Problème n° 29 - Calculer l'adresses IP suivante

L'exercice précédent m'a donné l'idée d'un exercice complémentaire...
Partant d'une table de machines ayant des adresses IP, cette fois ci bien modélisées, comment trouver l'adresse IP suivante pour insérer une nouvelle machine ?

Voici la nouvelle table des machines avec leur adresse IP et quelques lignes de test :

```
CREATE TABLE T_MACHINE_MAC
(MAC_ID     INT NOT NULL PRIMARY KEY,
 MAC_NOM    VARCHAR(16),
 MAC_ADRIP1 SMALLINT CHECK(MAC_ADRIP1 BETWEEN 0 AND 255),
 MAC_ADRIP2 SMALLINT CHECK(MAC_ADRIP2 BETWEEN 0 AND 255),
 MAC_ADRIP3 SMALLINT CHECK(MAC_ADRIP3 BETWEEN 0 AND 255),
 MAC_ADRIP4 SMALLINT CHECK(MAC_ADRIP4 BETWEEN 0 AND 255)
 CONSTRAINT CU_ADRIP UNIQUE (MAC_ADRIP1, MAC_ADRIP2, MAC_ADRIP3, MAC_ADRIP4));
```

```
INSERT INTO T_MACHINE_MAC VALUES (1, 'PC', 123,  12,   1, 200);
INSERT INTO T_MACHINE_MAC VALUES (2, 'PC', 123,  12,   1, 255);
INSERT INTO T_MACHINE_MAC VALUES (3, 'PC', 123,  12, 255, 255);
INSERT INTO T_MACHINE_MAC VALUES (4, 'PC', 123,  13,   0,   0);
INSERT INTO T_MACHINE_MAC VALUES (5, 'PC', 123, 255, 255, 255);
```

Le résultat de votre requête doit être :

```
MAC_ID      MAC_NOM          NEW_ADRIP1  NEW_ADRIP2  NEW_ADRIP3  NEW_ADRIP4
----------- ---------------- ----------- ----------- ----------- -----------
1           PC               123         12          1           201
2           PC               123         12          2           0
3           PC               123         13          0           1
4           PC               123         13          0           1
5           PC               124         0           0           0
```

Raisonnez à petits pas. Emboitez vos requêtes...

cliquez pour répondre et titrez votre mail "**Problème n°29 - Calculer l'adresse IP suivante**

IV-J - Problème n° 30 - Nombre de jours contigus

Pas facile, la question de Stéphane T...
Y a t il une formule magique qui permet de compter des dates consécutives en sql ?

Comme vous le savez certainement déjà, cette formule magique est une requête !

Voici les données qui nous servirons de test et la structure de la table associée :

```
CREATE TABLE T_PLANNING_PNG
(PNG_DATE DATE);
```

```
INSERT INTO T_PLANNING_PNG VALUES ('2004-01-01');
INSERT INTO T_PLANNING_PNG VALUES ('2004-01-02');
INSERT INTO T_PLANNING_PNG VALUES ('2004-01-03');
INSERT INTO T_PLANNING_PNG VALUES ('2004-01-15');
INSERT INTO T_PLANNING_PNG VALUES ('2004-01-16');
INSERT INTO T_PLANNING_PNG VALUES ('2004-01-17');
INSERT INTO T_PLANNING_PNG VALUES ('2004-01-18');
INSERT INTO T_PLANNING_PNG VALUES ('2004-01-30');
```

Le résultat de votre requête doit être :

```
DateDebut       NbJours
--------------- -------------
2004-01-01      3
2004-01-15      4
2004-01-30      1
```

Aidez-vous d'une table des dates...

cliquez pour répondre et titrez votre mail "**Problème n°30 - Nombre de jours contigus**

V - Exercices - 4° Partie

V-A - Problème n° 31 - plus proche valeur

Il s'agit de rechercher la plus proche valeur dans une table.
Par exemple une table propose des jours de rendez-vous. Le client souhaiterais un rendez-vous aux alentours du 12 janvier 2008. Comment obtenir la date libre la plus proche de cette date ?
Voici la table des rendez-vous :

```
CREATE TABLE T_RENDEZ_VOUS_RDV
(RDV_DATE      TIMESTAMP);
```

```
INSERT INTO T_RENDEZ_VOUS_RDV VALUES ('2008-01-07');
INSERT INTO T_RENDEZ_VOUS_RDV VALUES ('2008-01-11');
INSERT INTO T_RENDEZ_VOUS_RDV VALUES ('2008-01-14');
INSERT INTO T_RENDEZ_VOUS_RDV VALUES ('2008-01-16');
INSERT INTO T_RENDEZ_VOUS_RDV VALUES ('2008-01-18');
INSERT INTO T_RENDEZ_VOUS_RDV VALUES ('2008-01-21');
INSERT INTO T_RENDEZ_VOUS_RDV VALUES ('2008-01-22');
INSERT INTO T_RENDEZ_VOUS_RDV VALUES ('2008-01-24');
INSERT INTO T_RENDEZ_VOUS_RDV VALUES ('2008-01-25');
```

C'est plus facile que cela en à l'air, si l'on se souvient qu'un nombre premier est un nombre qui n'est divisible que par lui même et 1. Ou plutôt qu'un nombre qui n'est pas premier est divisible par un nombre compris entre 2 et lui-même moins un...

En l'occurrence, le résultat doit être le suivant :

```
RDV_DATE
-------------
2008-01-11
```

Testez donc votre solution avec une demande de rendez-vous le 23 janvier 2008...

cliquez pour répondre et titrez votre mail " **Problème n°31 - plus proche valeur** "

V-B - Problème n° 32 - primes de salaires

Les employés de l'entreprise doivent recvoir une prime de salaire d'un montant donné et cette prime doit être répartie sur un nombre de mois donné. Cependant il faut que la totalité de la prime soit donnée, chaque prime étant arrondie au centime, la dernière devant compenser les arrondis.. Par exemple si une prime de 1000 euros est versé et que cela se fait sur 12 mois, alors il faudra verser par exemple 83.33 € les 11 premiers mois et 83,37 € le dernier.
Comment faire cela en une seule requête ?
Voici la table pour le calcul des primes :

```
CREATE TABLE T_PRIME_PRM
(PRM_EMPLOYE       VARCHAR(16),
 PRM_MONTANT       DECIMAL(16,2),
 PRN_NB_ECHEANCES  INT);
```

```
INSERT INTO T_PRIME_PRM VALUES ('JAK', 1000, 1);
INSERT INTO T_PRIME_PRM VALUES ('POL', 1000, 3);
INSERT INTO T_PRIME_PRM VALUES ('LUC', 1000, 12);
```

Et le résultat attendu :

```
PRM_EMPLOYE  ECHEANCE  MONTANT
------------ --------- ------------
JAN          1         1000.00
JOE          1         333.33
JOE          2         333.33
JOE          3         333.34
LUC          1         83.33
LUC          2         83.33
LUC          3         83.33
LUC          4         83.33
LUC          5         83.33
LUC          6         83.33
LUC          7         83.33
LUC          8         83.33
LUC          9         83.33
LUC          10        83.33
LUC          11        83.33
LUC          12        83.37
```

A vos claviers !

cliquez pour répondre et titrez votre mail " **Problème n°32 - primes de salairesr** "

V-C - Problème n° 33 - affectations comptables

Notre utilisateur travaille dans une entreprise comptable et doit affecter au plus juste certains éléments à des comptes plus ou moins "flous"
Un petit exemple vallant mieux qu'un long discours, voici les données de son problème :

Exemple des données comptables :

```
compte   ana      montant
-------- -------- ------------
612000   26045    260.50
612000   12345    130.40
612101   33556    330.50
654000   26346     33.55
654102   28333    180.10
```

Règles d'affectation :

```
compte     ana      nouvelle affectation
---------- -------- --------------------
6?????     ?????    T124
6?????     ??5??    T028
6?????     26???    T033
612???     ?????    T125
612???     ??5??    T126
```

Résultat que l'on doit obtenir :

```
compte    montant    affectation
--------- ---------- -----------
612000    260.50     T125        -->(Regle 612??? ?????)
612000    130.40     T125        -->(Règle 612??? ?????)
612101    330.50     T126        -->(Règle 612??? ??5??)
654000     33.55     T033        -->(Règle 6????? 26???)
654102    180.10     T124        -->(Règle 6????? ?????)
```

Bref, à partir de règles d'affectation comptable un peu "floues" du fait de caractères génériques, il convient de trouver la meilleure affectation comptable des données...
Voici les tables avec lesquelles nous allons devoir établir notre requête, et les données du jeu d'essai :

```
CREATE TABLE T_COMPTE_CPT
(CPT_COMPTE          CHAR(6),
 CPT_ANA             CHAR(5),
 CPT_MONTANT         DECIMAL(16,2));

CREATE TABLE T_AFFECTATION_AFC
(AFC_COMPTE          CHAR(6),
 AFC_ANA             CHAR(5),
 AFC_AFFECT          CHAR(4));
```

```
INSERT INTO T_COMPTE_CPT VALUES ('612000', '26045', 260.50);
INSERT INTO T_COMPTE_CPT VALUES ('612000', '12345', 130.40);
INSERT INTO T_COMPTE_CPT VALUES ('612101', '33556', 330.50);
INSERT INTO T_COMPTE_CPT VALUES ('654000', '26346', 33.55);
INSERT INTO T_COMPTE_CPT VALUES ('654102', '28333', 180.10);

INSERT INTO T_AFFECTATION_AFC VALUES ('6_____', '_____', 'T124');
INSERT INTO T_AFFECTATION_AFC VALUES ('6_____', '__5__', 'T028');
INSERT INTO T_AFFECTATION_AFC VALUES ('6_____', '26___', 'T033');
INSERT INTO T_AFFECTATION_AFC VALUES ('612___', '_____', 'T125');
INSERT INTO T_AFFECTATION_AFC VALUES ('612___', '__5__', 'T126');
```

Je vous ais d'ailleurs déjà facilité la vie en remplacçant les ? par des _ !
Vous connaissez déjà le résultat attendu, alors en piste ! Sachez cepandant qu'une élégante solution passe par l'ajout de données...

cliquez pour répondre et titrez votre mail " **Problème n°33 - affectations comptables** "

V-D - Problème n° 34 - précédents (ou suivants)

Question simple et directe : comment numéroter les lignes d'une requête et prévoir d'une ligne sur l'autre la référence à la ligne précédente ?

Encore faut-il savoir quel ordre prendre en compte !
Partons d'une table on ne peut plus simple, puisque dotée d'une seule colonne :

```
CREATE TABLE T_LIGNE_PRECEDENTE
(COL VARCHAR(3));
```

```
INSERT INTO T_LIGNE_PRECEDENTE VALUES ('AAA');
INSERT INTO T_LIGNE_PRECEDENTE VALUES ('BBB');
INSERT INTO T_LIGNE_PRECEDENTE VALUES ('CCC');
INSERT INTO T_LIGNE_PRECEDENTE VALUES ('DDD');
```

Comment obtenir le résultat suivant :

```
COL  LIGNE_PRECEDENTE
---- ----------------
AAA  NULL
BBB  AAA
CCC  BBB
DDD  CCC
```

D'ailleurs est-il possible d'informer de la ligne suivante et de la précédente en même temps ?

cliquez pour répondre et titrez votre mail " **Problème n°34 - précédente et suivants** "

V-E - Problème n° 35 - matchs et victoires

Voici un développeur confronté à des classements sportifs... Nous lui laissons la parole :
Je voudrais générer un classement entre des joueurs à partir des résultats de matchs de tennis de table.
J'ai pour l'instant créé les tables suivantes :

TABLE T_JOUEUR_JOR

```
JOR_ID       JOR_NOM
------------ -----------
1            Dupont
2            Camus
3            Mercier
```

TABLE T_MATCH_MCH

```
MCH_ID       JOR_ID1      JOR_ID2      MCH_SCORE_JOUEUR1  MCH_SCORE_JOUEUR2
------------ ------------ ------------ ------------------ ------------------
1            1            3            21                 15
2            3            2            17                 21
3            2            1            21                 13
4            3            1            21                 10
```

Résultat attendu :

```
JOR_ID       JOR_NOM        NB_MATCHS    NB_VICTOIRES
------------ -------------- ------------ ---------------
1            Dupont         2            1
2            Camus          3            2
3            Mercier        3            1
```

A vous de jouer !
Voici la défintion des tables et le jeu d'essais :

```
CREATE TABLE T_JOUEUR_JOR
(JOR_ID      INT,
 JOR_NOM     VARCHAR(32));
```

```
CREATE TABLE T_MATCH_MCH
(MCH_ID              INT,
 JOR_ID1             INT,
 JOR_ID2             INT,
 MCH_SCORE_JOUEUR1   INT,
 MCH_SCORE_JOUEUR2   INT);
```

```
INSERT INTO T_JOUEUR_JOR VALUES (1, 'Dupont');
INSERT INTO T_JOUEUR_JOR VALUES (2, 'Camus');
INSERT INTO T_JOUEUR_JOR VALUES (3, 'Mercier');

INSERT INTO T_MATCH_MCH VALUES (1, 1, 3, 21, 15);
INSERT INTO T_MATCH_MCH VALUES (2, 3, 2, 17, 21);
INSERT INTO T_MATCH_MCH VALUES (3, 2, 1, 21, 13);
INSERT INTO T_MATCH_MCH VALUES (4, 3, 1, 21, 10);
```

cliquez pour répondre et titrez votre mail " **Problème n°35 - matchs et victoires** "

V-F - Problème n° 36 - tri bâtard

Une mauvaise modélisation de données à fait que notre internaute désire un tri corresponsant à l'ordre numérique pour une colonne ne contenant que des chiffres mais modélisée en varchar...
Venez lui en aide afin de trier ces données comme s'il s'agissiait de nombres...
Attention, prenez en compte le fait qu'il peut ne pas y avoir que des chiffres dans cette colonne de type caractères !

```
CREATE TABLE T_PARCELLE_PCL (PCL_NUM VARCHAR(16));
```

```
INSERT INTO T_PARCELLE_PCL VALUES ('1');
INSERT INTO T_PARCELLE_PCL VALUES ('11');
INSERT INTO T_PARCELLE_PCL VALUES ('111');
INSERT INTO T_PARCELLE_PCL VALUES ('2');
INSERT INTO T_PARCELLE_PCL VALUES ('21');
INSERT INTO T_PARCELLE_PCL VALUES ('22');
INSERT INTO T_PARCELLE_PCL VALUES ('221');
INSERT INTO T_PARCELLE_PCL VALUES ('28A');
INSERT INTO T_PARCELLE_PCL VALUES ('28B');
INSERT INTO T_PARCELLE_PCL VALUES ('3');
```

Voici la réponse attendue :

```
PCL_NUM
----------------
1
2
3
11
21
22
111
221
28A
28B
```

cliquez pour répondre et titrez votre mail " **Problème n°36 - tri bâtard** "

V-G - Problème n° 37 - vote contraint

Un exercice inspiré des fameux puzzle de Joe Celko...
Une compétition de schnorkelzig ne comporte jamais que 4 compétiteurs. Dans la compétition exemple, les athlètes sont Paul, Marc, Jean et Léon.
Chaque membre du jury, et ils sont très nombreux, doit indiquer quel serait l'ordre dans lequel il voudrait placer ses champions.

Par exemple, le jury DUPONT aimerait que le podium soit : 1er Jean, 2nd Marc, 3e Léon, 4e Paul.
Mais les membres du jury peuvent laisser leur podium incomplet. Ainsi le jury MARTIN, aimerait le podium suivant : 1er Léon, 2nd Paul, 3e Marc. Dans ce cas Jean aurait un marqueur NULL (absence de valeur).
Si le podium est incomplet, il doit aun moins être en séquence. Par exemple 1, 2, 3 ou 1, 2, ou 1, mais pas 1, 3, 4.
Aucun podium ne peut se faire avec des ex aecquo. Par exemple 1, 2, 2, 4.
Il n'est pas possible que le podium soit vide...
La table destinée à recevoir les votes est ainsi constituée :

```
CREATE TABLE T_SCHNORKELZIG_SKZ
(SKZ_JURY VARCHAR(16) NOT NULL PRIMARY KEY,
 SKZ_MARC  INT,
 SKZ_PAUL  INT,
 SKZ_JEAN  INT,
 SKZ_LEON  INT)
```

Votre mission, si vous l'acceptez, sera d'écrire les contraintes nécessaires à valider la saisie et empêcher notamment des erreurs telles que :

```
SKZ_JURY    SKZ_MARC    SKZ_PAUL    SKZ_JEAN    SKZ_LEON
----------- ----------- ----------- ----------- -----------
MOULIN      1           3           5           11          --> restreindre le vote à 1,2,3,4
DUVAL       1           NULL        3           4
  --> séquence incorrecte, aurait dû être 1, NULL, 2, 3
SCHMIDT     1           1           2           3           --> doublon (présence du 1 deux fois)
```

Voici deux jeux d'essais. Le premier doit voir toutes ses lignes acceptées, le second aucune...

podiums corrects

```
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY001', 1, 2, 3, 4)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY002', 1, 2, 4, 3)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY003', 1, 3, 2, 4)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY004', 1, 3, 4, 2)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY005', 2, 1, 3, 4)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY006', 2, 1, 4, 3)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY007', 3, 1, 2, 4)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY008', 4, 3, 1, 2)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY009', 1, NULL, 3, 2)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY010', NULL, 2, 1, 3)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY011', 1, 3, NULL, 2)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY012', 1, 3, 2, NULL)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY013', NULL, 2, 1, NULL)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY014', 1, NULL, NULL, 2)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY015', 1, NULL, NULL, NULL)
```

podiums incorrects

```
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY999', 1, 2, 3, 3)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY998', 1, 2, 3, 5)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY997', 1, 1, 2, 2)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY996', 1, 3, 3, 3)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY995', 1, 2, 2, 6)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY994', NULL, 2, 3, 4)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY993', NULL, 3, 3, 4)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY992', 2, 2, 3, 4)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY991', 1, 1, 1, 1)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY990', NULL, NULL, 6, 4)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY989', NULL, 6, 4, NULL)
INSERT INTO T_SCHNORKELZIG_SKZ VALUES ('JRY988', NULL, NULL, 2, NULL)
```

Pour cela vous devez créer une contrainte SQL, et une seule, la plus concise possible. Pour ma part j'ai réussi cette contrainte avec 114 caractères en faisant appel une seule fois à chacune des 4 colonnes...

cliquez pour répondre et titrez votre mail " **Problème n°37 - vote contraint** "

V-H - Problème n° 38 - propriété bien gardée

Monsieur Fulmington est un milliardaire qui possède une charmante propriété à Saint Jean Cap Ferrat. 168 pièces, 382 hectares de jardins, 2 874 oeuvres d'art et 6 gardiens.
Le problème est que Monsieur Fulmington ne sait pas si sa propriété est gardée en permanence. Pourriez vous l'aider et regarder sur une semaine, celle allant du 1er au 7 septembre 2008, quelles sont la ou les périodes pendant lesquelles la propriété n'est pas surveillée ? Le planning de gardiennage figure dans la table suivante :

```
CREATE TABLE T_GARDIENNAGE_GDN
(GDN_NOM              VARCHAR(16),
 GDN_DATEHEURE_DEBUT  TIMESTAMP,
 GDN_DATEHEURE_FIN    TIMESTAMP) ;
```

```
INSERT INTO T_GARDIENNAGE_GDN VALUES ('LEON', '2008-09-01 22:30:00', '2008-09-02 08:00:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('LEON', '2008-09-03 18:00:00', '2008-09-04 04:30:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('LEON', '2008-09-07 06:00:00', '2008-09-07 12:30:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('MARC', '2008-09-01 07:30:00', '2008-09-01 14:30:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('MARC', '2008-09-03 08:30:00', '2008-09-03 16:00:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('MARC', '2008-09-05 09:30:00', '2008-09-05 12:00:00')
INSERT INTO T_GARDIENNAGE_GDN VALUES ('MARC', '2008-09-06 08:30:00', '2008-09-06 13:30:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('PAUL', '2008-09-02 06:00:00', '2008-09-02 17:30:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('PAUL', '2008-09-03 06:00:00', '2008-09-03 17:30:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('PAUL', '2008-09-04 06:00:00', '2008-09-04 17:30:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('PAUL', '2008-09-06 12:00:00', '2008-09-06 22:30:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('JACK', '2008-09-02 08:30:00', '2008-09-02 16:00:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('JACK', '2008-09-05 08:30:00', '2008-09-05 21:30:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('JACK', '2008-09-06 21:00:00', '2008-09-07 10:30:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('DICK', '2008-09-01 12:30:00', '2008-09-02 00:00:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('DICK', '2008-09-03 22:30:00', '2008-09-04 10:00:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('DICK', '2008-09-06 16:30:00', '2008-09-07 10:30:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('DICK', '2008-09-07 18:30:00', '2008-09-08 06:30:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('CHAD', '2008-09-01 23:30:00', '2008-09-02 11:00:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('CHAD', '2008-09-04 14:30:00', '2008-09-05 08:30:00') ;
INSERT INTO T_GARDIENNAGE_GDN VALUES ('CHAD', '2008-09-06 10:30:00', '2008-09-06 21:00:00') ;
```

Voici le résultat attendu...

```
GDN_DATEHEURE_DEBUT      GDN_DATEHEURE_FIN
------------------------ ------------------------
2008-09-01 00:00         2008-09-01 07:30
2008-09-02 17:30         2008-09-03 06:00
2008-09-03 17:30         2008-09-03 18:00
2008-09-05 21:30         2008-09-06 08:30
2008-09-07 12:30         2008-09-07 18:30
```

A l'évidence, Monsieur Fulmington à du mourron à se faire !

cliquez pour répondre et titrez votre mail " **Problème n°38 - propriété bien gardée** "

V-I - Problème n° 39 - abstract et mots clefs

Encore une base mal modélisée...
Dans une base documentaire figure dans une table le nom d'un article dans une colonne et dans l'autre colonne un conglomérat des mots clef (abstract) de l'article. Pour séparer les mots clef, le développeur à trouvé judicieux d'utiliser le caractère # (dièse) entre les expressions. Voici le jeu de données sur lequel nous allons travailler :

```
CREATE TABLE T_PAPIER_PPR
(PPR_TITRE     VARCHAR(36),
 PPR_ABSTRACT  VARCHAR(120)) ;
```

```
INSERT INTO T_PAPIER_PPR VALUES ('Les derniers roi de France', 'Histoire#Politique') ;
```

```
INSERT INTO
 T_PAPIER_PPR VALUES ('De Gaulle, un héro de l''histoire', 'Histoire#Politique#Guerre') ;
INSERT INTO T_PAPIER_PPR VALUES ('Les deux guerres mondiales', 'Histoire#Guerre') ;
INSERT INTO T_PAPIER_PPR VALUES ('Les années 50 en Europe', 'Histoire#Union européenne') ;
INSERT INTO T_PAPIER_PPR VALUES ('Les présidents de la république', 'Histoire#Politique') ;
INSERT INTO T_PAPIER_PPR VALUES ('Histoire de France', 'Histoire') ;
```

Comme vous le voyez la première forme normale n'est pas respectée puisque plusieurs informations figurent dans une seule et même colonne !
Cependant, notre quidam doit renvoyer comme réponse le nombre d'apparition des mots clefs dans une sélection d'article. Par exemple, ici, le résultat devrait être :

```
MOT_CLEF           NOMBRE
------------------ ----------
Histoire           5
Politique          3
Guerre             2
Union européenne   1
```

Sauriez-vous l'aider à accomplir sa tâche ?

cliquez pour répondre et titrez votre mail " **Problème n°39 - abstract et mots clefs** "

V-J - Problème n° 40 - gestion des stocks

Un problème bien classique : à partir d'une table des réception de produits et des commandes clients, notre magazinier aimerait avoir un état des stocke au jour le jour... Voyons cela en détail...
Les tables de notre jeu d'essai :

```
CREATE TABLE T_RECEPTION_RCP
(PRD_ID       INT,
 RCP_DATE     DATE,
 RCP_QUANTITE FLOAT) ;

CREATE TABLE T_COMMANDE_CMD
(PRD_ID        INT,
 CMD_DATE      DATE,
 CMD_QUANTITE  INT) ;
```

```
INSERT INTO T_RECEPTION_RCP VALUES (1, '2008-01-01', 10) ;
INSERT INTO T_RECEPTION_RCP VALUES (1, '2008-02-01', 20) ;
INSERT INTO T_RECEPTION_RCP VALUES (1, '2008-03-15',  5) ;
INSERT INTO T_RECEPTION_RCP VALUES (2, '2008-02-10', 20) ;
INSERT INTO T_RECEPTION_RCP VALUES (2, '2008-03-20', 25) ;
INSERT INTO T_RECEPTION_RCP VALUES (3, '2008-01-12', 20) ;
INSERT INTO T_RECEPTION_RCP VALUES (4, '2008-01-01', 10) ;
INSERT INTO T_RECEPTION_RCP VALUES (4, '2008-01-15', 20) ;
INSERT INTO T_RECEPTION_RCP VALUES (4, '2008-02-01', 30) ;

INSERT INTO T_COMMANDE_CMD VALUES (1, '2008-02-11', 45) ;
INSERT INTO T_COMMANDE_CMD VALUES (2, '2008-02-15', 35) ;
INSERT INTO T_COMMANDE_CMD VALUES (3, '2008-02-01'  20) ;
INSERT INTO T_COMMANDE_CMD VALUES (4, '2008-01-05'  75) ;
```

Les lignes de réception sont celles des produits alimentant le stock. Les lignes des commandes sont celles sortant du stock...
Notre magazinier aimerait obtenir la présentation suivante :

```
PRD_ID     DATE_STOCK     QTE_PRISE     QTE_RECUE     QTE_DISPO     RESTE_A_COMPLETER
---------- -------------- ------------- ------------- ------------- -----------------
1          2008-01-01     NULL          10            10            0
1          2008-02-01     NULL          20            20            0
1          2008-02-11     45            NULL          0             15
```

1	2008-03-15	NULL	5	0	10
2	2008-02-10	NULL	20	20	0
2	2008-02-15	35	NULL	0	15
2	2008-03-20	NULL	25	10	0
3	2008-01-12	NULL	20	20	0
3	2008-02-01	20	NULL	0	0
4	2008-01-01	NULL	10	10	0
4	2008-01-05	75	NULL	0	65
4	2008-01-15	NULL	20	0	45
4	2008-02-01	NULL	30	0	15

Qui lui permet de connaître au jour le jour ou il est est avec toutes les variations de stock.
Sauriez vous élaborer une telle requête ?

cliquez pour répondre et titrez votre mail " **Problème n°40 - gestion des stocks** "