TP Manipulations SQL

Pour ce faire nous allons étudier la table PERSONNE suivante :

PERSONNE
id
nom
prenom
mail
adresse
ville

Partie 1 : Installation

Créez une nouvelle base de donnés sur MySQL et exécutez-y le script joint. Ce script vous permet de construire et alimenter la table ci-dessous avec un jeu de données que vous utiliserez dans les questions suivantes.

Partie 2 : Généralités et traitements

1) Combien de lignes sont présentes dans la table ?
2) Combien de villes différentes sont présentes dans la table ?
3) Ecrire une requête permettant de récupérer les noms des villes apparaissant en doublon dans la table.
4) Quel nom de famille compte le plus de personnes dans la table ? Donnez le nom et le nombre de personnes de la famille.

Partie 3 : Manipulation de chaines de caractères

Pour l'ensemble de ces requêtes, aidez-vous des fonctions SQL sous MySQL pour la manipulation de chaines de caractères (voir annexe à la fin du sujet)

1) Afficher les noms et prénoms des personnes ayant un nom composé de minimum 5 caractères.
2) Afficher les noms et prénoms des personnes ayant un nombre total de caractères (nom et prénom) multiple de 7.
3) Quel est le nombre de personnes habitant ayant un prénom avec deux L qui se suivent ?
4) Quel est le nom et prénom de la personne ayant le nom le plus long ?
5) Quel est le nombre de personnes habitant dans une ville ayant un nom composé ? On dit qu'une ville possède un nom composé à partir du moment où ce nom contient au moins un – ou un espace.

6) Ecrire une requête permettant de récupérer le nom de domaine de chacune des adresses email présentes dans la table. Pour information :

7) Afficher la liste des utilisateurs (nom, prénom, adresse) pour lesquels il y a « Rd. » dans leur adresse.
8) Remplacer « Rd. » par « Road. » pour tous les utilisateurs issus de la question précédente. Vérifiez le résultat de votre requête en exécutant de nouveau la requête écrite lors de la question précédente.
9) Ajoutez une colonne « sexe » permettant de stocker le sexe des personnes. Le contenu de cette colonne est représenté par un caractère (H ou F). Une fois la colonne ajoutée on choisira arbitrairement que toutes les personnes ayant un prénom se terminant par une voyelle seront de sexe F et les autres de sexe M. Remplir la colonne créée à l'aide de la condition décrite (il est possible de le faire en deux requêtes, une pour chaque valeur de la colonne)
10) On souhaite stocker les initiales de chacune des personnes dans une nouvelle colonne.
    Les initiales sont composées de la première lettre du nom suivi de la première lettre du prénom, toujours dans cet ordre. Les deux lettres sont séparées par un « . » (Exemple : Robert Dupuis donnera en initiales => D.R)
    - Ajoutez la colonne à votre table (veillez à bien choisir le type de données adapté)
    - Exécutez une requête permettant d'écrire les initiales de chaque personne dans la table
11) On souhaite créer une boite mail sur le domaine esgi.fr pour tous les utilisateurs. Ajoutez une nouvelle colonne que vous appellerez « mail_esgi » dans laquelle vous devrez stockez les adresses de chacune de ces personnes qu'il faudra construire en suivant le modèle suivant : **nom.prénom@esgi.fr** où nom et prénom correspondent aux valeurs de chacune des personnes. Vous devrez stocker l'intégralité de la chaine de caractères en minuscules.

MySQL CHEAT SHEET: STRING FUNCTIONS

by **mysqlbackupftp.com** with ♥

MEASUREMENT

Return a string containing binary representation of a number

```
BIN (12) = '1100'
```

Return length of argument in bits

```
BIT_LENGTH ('MySql') = 40
```

Return number of characters in argument

```
CHAR_LENGTH ('MySql') = 5
CHARACTER_LENGTH ('MySql') = 5
```

Return the length of a string in bytes

```
LENGTH ('Ö') = 2
LENGTH ('A') = 1
OCTET_LENGTH ('Ö') = 2
OCTET_LENGTH ('X') = 1
```

Return a soundex string

```
SOUNDEX ('MySql') = 'M240'
SOUNDEX ('MySqlDatabase') = 'M24312'
```

Compare two strings

```
STRCMP ('A', 'A') = 0
STRCMP ('A', 'B') = -1
STRCMP ('B', 'A') = 1
```

SEARCH

Return the index of the first occurrence of substring

```
INSTR ('MySql', 'Sql') = 3
INSTR ('Sql', 'MySql') = 0
```

Return the position of the first occurrence of substring

```
LOCATE ('Sql', 'MySqlSql') = 3
LOCATE ('xSql', 'MySql') = 0
LOCATE ('Sql', 'MySqlSql', 5) = 6
POSITION('Sql' IN 'MySqlSql') = 3
```

Pattern matching using regular expressions

```
'abc' RLIKE '[a-z]+' = 1
'123' RLIKE '[a-z]+' = 0
```

Return a substring from a string before the specified number of occurrences of the delimiter

```
SUBSTRING_INDEX ('A:B:C', ':', 1) = 'A'
SUBSTRING_INDEX ('A:B:C', ':', 2) = 'A:B'
SUBSTRING_INDEX ('A:B:C', ':', -2) = 'B:C'
```

CONVERSION

Return numeric value of left-most character

```
ASCII ('2') = 50
ASCII (2) = 50
ASCII ('dx') = 100
```

Return the character for each number passed

```
CHAR (77.3,121,83,81, '76, 81.6') = 'MySQL'
CHAR (45*256+45) = CHAR (45,45) = '--'
CHARSET(CHAR (X'65' USING utf8)) = 'utf8'
```

Decode to / from a base-64 string

```
TO_BASE64 ('abc') = 'YWJj'
FROM_BASE64 ('YWJj') = 'abc'
```

Convert string or number to its hexadecimal representation

```
X'616263' = 'abc'
HEX ('abc') = 616263
HEX(255) = 'FF'
CONV(HEX(255), 16, 10) = 255
```

Convert each pair of hexadecimal digits to a character

```
UNHEX ('4D7953514C') = 'MySQL'
UNHEX ('GG') = NULL
UNHEX (HEX ('abc')) = 'abc'
```

Return the argument in lowercase

```
LOWER ('MYSQL') = 'mysql'
LCASE ('MYSQL') = 'mysql'
```

Load the named file

```
SET blob_col=LOAD_FILE ('/tmp/picture')
```

Return a string containing octal representation of a number

```
OCT (12) = '14'
```

Return character code for leftmost character of the argument

```
ORD ('2') = 50
```

Escape the argument for use in an SQL statement

```
QUOTE ('Don\'t!') = 'Don\'t!'
QUOTE (NULL) = 'NULL'
```

Convert to uppercase

```
UPPER ('mysql') = 'MYSQL'
UCASE ('mysql') = 'MYSQL'
```

MODIFICATION

Return concatenated string

```
CONCAT ('My', 'S', 'QL') = 'MySQL'
CONCAT ('My', NULL, 'QL') = NULL
CONCAT (14.3) = '14.3'
```

Return concatenate with separator

```
CONCAT_WS (',', 'My', 'Sql') = 'My,Sql'
CONCAT_WS (',','My',NULL,'Sql') = 'My,Sql'
```

Return a number formatted to specified number of decimal places

```
FORMAT (12332.123456, 4) = 12,332.1235
FORMAT (12332.1, 4) = 12,332.1000
FORMAT (12332.2, 0) = 12332.2
FORMAT (12332.2, 2, 'de_DE') = 12.332,20
```

Insert a substring at the specified position up to the specified number of characters

```
INSERT ('12345', 3, 2, 'ABC') = '12ABC5'
INSERT ('12345', 10, 2, 'ABC') = '12345'
INSERT ('12345', 3, 10, 'ABC') = '12ABC'
```

Return the leftmost number of characters as specified

```
LEFT ('MySql', 2) = 'My'
```

Return the string argument, left-padded with the specified string

```
LPAD ('Sql', 2, ':)') = 'Sq'
LPAD ('Sql', 4, ':)') = ':Sql'
LPAD ('Sql', 7, ':)') = ':):)Sql'
```

Remove leading spaces

```
LTRIM ('      MySql') = 'MySql'
```

Repeat a string the specified number of times

```
REPEAT ('MySQL', 3) = 'MySQLMySQLMySQL'
```

Replace occurrences of a specified string

```
REPLACE ('NoSql', 'No', 'My') = 'MySql'
```

Reverse the characters in a string

```
REVERSE ('MySql') = 'lqSyM'
```

Return the specified rightmost number of characters

```
RIGHT ('MySql', 3) = 'Sql'
```

Returns the string argument, right-padded with the specified strin.

```
RPAD ('Sql', 2, ':)') = 'Sq'
RPAD ('Sql', 4, ':)') = 'Sql:'
RPAD ('Sql', 7, ':)') = 'Sql:):)'
```

Remove trailing spaces

```
RTRIM ('MySql      ') = 'MySql'
```

Return a string of the specified number of spaces

```
SPACE ('6') = '      '
```

Return the substring as specified

```
SUBSTRING=SUBSTR=MID('MySql',3) = 'Sql'
SUBSTRING=SUBSTR=MID('MySql' FROM 4) = 'ql'
SUBSTRING=SUBSTR=MID('MySql',3,1) = 'S'
SUBSTRING=SUBSTR=MID('MySql',-3) = 'Sql'
SUBSTRING=SUBSTR=MID('MySql' FROM -4 FOR 2)
= 'yS'
```

Remove leading and trailing spaces

```
TRIM(' MySql ') = 'MySql'
TRIM(LEADING 'x' FROM 'xxxSqlMy') = 'MySql'
TRIM(BOTH 'My' FROM 'MySqlMy') = 'Sql'
TRIM(TRAILING 'Sql' FROM 'MySql') = 'My'
```

SETS

Return string at index number

```
ELT (1, 'ej', 'Heja', 'hej', 'foo') = 'ej'
ELT (4, 'ej', 'Heja', 'hej', 'foo') = 'foo'
```

Return a string such that for every bit set in the value bits, you get an on string and for every unset bit, you get an off string

```
EXPORT_SET (5,'Y','N',',',4) = 'Y,N,Y,N'
EXPORT_SET (6,'1','0',',',6) = '0,1,1,0,0,0'
```

Return the index (position) of the first argument in the subsequent arguments

```
FIELD ('ej','Hj','ej','Heja','hej','oo') = 2
FIELD ('fo','Hj','ej','Heja','hej','oo') = 0
```

Return the index position of the first argument within the second argument

```
FIND_IN_SET ('b', 'a,b,c,d') = 2
FIND_IN_SET ('z', 'a,b,c,d') = 0
FIND_IN_SET ('a,', 'a,b,c,d') = 0
```

Return a set of comma-separated strings that have the corresponding bit in bits set

```
MAKE_SET (1,'a','b','c') = 'a'
MAKE_SET (1|4,'ab','cd','ef') = 'ab,ef'
MAKE_SET (1|4,'ab','cd',NULL,'ef') = 'ab'
MAKE_SET (0,'a','b','c') = ''
```