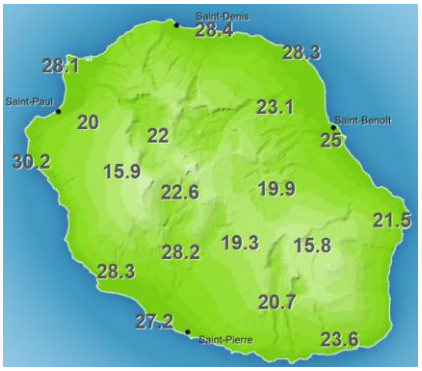
SYHTHESE DES REQUETES SQL

MISE EN SITUATION

Plusieurs stations météorologiques réalisées autour d'une carte **RASPBERRY PI 3 ou 4** et réparties sur toute l'île de la Réunion permettent d'obtenir une base de données climatologique. Les chercheurs pourront alors mettre à disposition des informations sur le climat dans le département.

Chaque station va acquérir la température, la vitesse du vent sur 24h et la pluviométrie :

- La température est mesurée en °C
- La vitesse du vent est mesurée en km/h
- La pluviométrie (ou hauteur des précipitations) est mesurée en mm (sur une plage de 24h)



La base de données « **BDDmeteo** » comporte une table appelée « **grandeur** » donnée ci-dessous.

Elle est constituée des enregistrements issus des mesures. Elles ont été effectuées le mercredi 03/02/2018 à 14h00

id	lieu	temperature	vent	pluviometrie
1	Saint Benoit	25.1	7	84
2	Saint Denis	28.4	3	19.5
3	Le Port	28.1	13	0
4	Trois Bassins	30.2	2	0
5	Sainte Rose	21.5	28	204.4
6	Saint Pierre	27.2	41	39.2
7	Saint Philippe	23.6	12	211
8	Volcan	15.8	5	383
9	Maido	15.9	12	23.2

Pointe des Trois Bassins

- Température en degrés Celsius
- Vent : vitesse en km/h
- Pluviométrie : mm/24h

1. TRAVAIL DEMANDE SUR LA BASE <BDDmeteo>

Afin d'aider les agriculteurs, un chercheur souhaite effectuer une analyse des données stockées. Pour cela, il doit réaliser une application qui interroge la base de données en utilisant le langage de requête **SQL** (se référer à la documentation fournie lors des précédents TP).

Pour le travail demandé, le serveur de base de données est installé en localhost sur votre machine virtuelle Linux.

- On demande d'utiliser le ***client mysql*** en CLI (Command Line Interface) sous Linux afin d'effectuer les requêtes suivantes.

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 217
Server version: 8.0.22-0ubuntu0.20.04.3 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

1.1. Créer la base de données **BDDmeteo** en ligne de commande avant toute opération d'importation.

1.2. Utiliser l'interface web phpMyAdmin afin d'importer la base de données **BDDmeteo** fournie par le professeur (fichier « BDDmeteo.sql »).

1.3. Donner la requête **SQL** permettant d'afficher les enregistrements de la table **grandeur**.

1.4. Donner la requête **SQL** permettant d'afficher les données climatologiques (enregistrement) de Trois Bassins

1.5. Donner la requête **SQL** permettant d'afficher les données météorologiques des lieux où la température est supérieure à 25°C.

1.6. Donner la requête **SQL** permettant d'afficher les données météorologiques des lieux où la pluviométrie est nulle.

1.7. Donner la requête **SQL** permettant d'afficher les données météorologiques des lieux où le vent est faible (< 5 km/h) avec une présentation dans l'ordre alphabétique.

1.8. Donner la requête **SQL** permettant de compter le nombre de lieux où la température est supérieure à 28 °C et le vent inférieur à 5 km/h.

- Le chercheur a prévu la possibilité pour son application d'ajouter un enregistrement à cette table (il y aura aussi la possibilité de supprimer un enregistrement).

1.9. Donner la requête **SQL** permettant d'insérer un enregistrement pour la **Petite France** (située au-dessus du Guillaume) → température = 20°C, vent = 7 km/h, pluviométrie (précipitations) = 11,4 mm

1.10. Donner la requête **SQL** permettant de supprimer l'enregistrement du **Maido**.

2. MODIFICATION DE LA BASE DE DONNEES

Une nouvelle structure de base de données est mise en place pour améliorer l'exploitation des mesures. Dans un premier temps, cette nouvelle base appelée « **Meteo974** » sera constituée de 2 tables.

Les requêtes pour créer les deux tables sont données ci-dessous :

```
mysql> CREATE TABLE station (id INT NOT NULL AUTO_INCREMENT, commune VARCHAR(40),
emplacement VARCHAR(50), PRIMARY KEY(id));
```

```
mysql> CREATE TABLE mesure(mesure_id INT NOT NULL AUTO_INCREMENT, temperature FLOAT
, vent FLOAT, pluviometrie FLOAT, station_id INT NOT NULL, PRIMARY KEY(mesure_id),
FOREIGN KEY(station_id) REFERENCES station(id));
```

2.1. Donner le nom des deux tables créées.

2.2. Identifier les clés primaires et étrangères de chaque table (si elles existent).

2.3. Donner un schéma relationnel entre les deux tables en précisant les éléments suivants :

- Le nom de la table
- Les champs
- La clé primaire
- La clé étrangère (si elle existe)
- Le lien entre les tables
- La cardinalité

Partie théorique

Pour la suite de l'activité, on demande d'utiliser la base de données mise à disposition par le professeur.

2.4. Créer la base de données **Meteo974** en ligne de commande avant toute opération d'importation.

2.5. Utiliser l'interface web phpMyAdmin afin d'importer la base de données **Meteo974** fournie par le professeur (fichier « Meteo974.sql »).

2.6. Donner la requête **SQL** permettant d'afficher les enregistrements de la table **station**.

2.7. Donner la requête **SQL** permettant d'afficher les enregistrements de la table **mesure**.

2.8. Donner la requête (**sur les deux tables**) permettant d'obtenir l'ensemble des données météorologiques (temperature, vent, pluviométrie) avec leur emplacement.

2.9. Donner la requête permettant d'obtenir les données de température et de vent sur la station de la Pointe de Trois Bassins (emplacement, commune, temperature, vent, pluviométrie).

ANNEXE 1 : EXTRAIT DE LA DOCUMENTATION DU LANGAGE SQL

Le langage SQL (Structured Query Language) est un langage normalisé permettant d'accéder aux bases de données relationnelles.

La requête SELECT

L'utilisation la plus courante de SQL consiste à lire des données issues de la base de données. Cela s'effectue grâce à la commande SELECT, qui retourne des enregistrements dans un tableau de résultat. Cette commande peut sélectionner une ou plusieurs colonnes d'une table.

Syntaxe de base : L'utilisation de cette requête s'effectue de la manière suivante :

SELECT nom_du_champ FROM nom_table

- Obtention de plusieurs colonnes :

Avec la même table il est possible de lire plusieurs colonnes à la fois. Il suffit tout simplement de séparer les noms des champs souhaités par une virgule. Pour obtenir les prénoms et les noms des clients il faut alors faire la requête suivante :

SELECT nom_du_champ1, nom_du_champ2 nom FROM nom_table

- Obtention de toutes les colonnes :

C'est un joker qui permet de sélectionner toutes les colonnes. Il s'utilise de la manière suivante :

SELECT * FROM nom_table

La clause WHERE

La clause WHERE dans une requête SQL permet d'extraire les lignes d'une base de données qui respectent une condition. Cela permet d'obtenir uniquement les informations désirées.

La commande WHERE s'utilise en complément à une requête utilisant SELECT.

Syntaxe :

SELECT nom_champ FROM nom_table WHERE condition

La condition peut être exprimée par une expression logique utilisant les opérateurs >, <, =, AND, OR …

La clause ORDER BY

La commande ORDER BY permet de trier les lignes dans un résultat d'une requête SQL. Il est possible de trier les données sur une ou plusieurs colonnes, par ordre ascendant ou descendant.

Une requête où l'on souhaite filtrer l'ordre des résultats utilise la commande ORDER BY de la sorte :

```
SELECT nom_du_champ1, nom_du_champ2
FROM table
ORDER BY colonne1
```

Par défaut les résultats sont classés par ordre ascendant, toutefois il est possible d'inverser l'ordre en utilisant le suffixe DESC après le nom de la colonne. Par ailleurs, il est possible de trier sur plusieurs colonnes en les séparant par une virgule. Une requête plus élaborée ressemblerait à cela :

```
SELECT nom_du_champ1, nom_du_champ2, nom_du_champ3
FROM table
ORDER BY nom_du_champ1 DESC, nom_du_champ2 ASC
```

Remarque : il n'est pas obligé d'utiliser le suffixe ASC sachant que les résultats sont toujours classés par ordre ascendant par défaut.

La requête INSERT INTO

L'insertion de données dans une table s'effectue à l'aide de la commande INSERT INTO. Cette commande permet au choix d'inclure une seule ligne à la base existante ou plusieurs lignes à la fois.

Pour insérer des données dans une base, il y a 2 syntaxes principales :

- Insérer une ligne en indiquant les informations pour chaque colonne existante (en respectant l'ordre)
- Insérer une ligne en spécifiant les colonnes que vous souhaitez compléter. Il est possible d'insérer une ligne renseignant seulement une partie des colonnes

Syntaxe simplifiée :

- **Insérer une ligne en spécifiant toutes les colonnes**

  INSERT INTO table VALUES ('valeur 1', 'valeur 2', ...)

- **Insérer une ligne en spécifiant seulement les colonnes souhaitées**

  INSERT INTO table (nom_colonne_1, nom_colonne_2, ...
  VALUES ('valeur 1', 'valeur 2', ...)

La requête DELETE

La commande DELETE en SQL permet de supprimer des lignes dans une table. En utilisant cette commande associé e à WHERE il est possible de sélectionner les lignes concernées qui seront supprimées.

La syntaxe pour supprimer des lignes est la suivante :

```
DELETE FROM `table`
WHERE condition
```

Attention : s'il n'y a pas de condition WHERE alors toutes les lignes seront supprimées et la table sera alors vide.

La fonction COUNT()

En SQL, la fonction d'agrégation COUNT() permet de compter le nombre d'enregistrement dans une table.

Syntaxe :

- Pour connaître le nombre de lignes totales dans une table, il suffit d'effectuer la requête SQL suivante :

  SELECT COUNT(*) FROM table

- Il est aussi possible de connaitre le nombre d'enregistrement sur une colonne en particulier. Les enregistrements qui possèdent la valeur NULL ne seront pas comptabilisés.

  SELECT COUNT(nom_colonne) FROM table

La jointure interne INNER JOIN

Dans le langage SQL la commande INNER JOIN, est un type de jointures très communes pour lier plusieurs tables entre-elles. Cette commande retourne les enregistrements lorsqu'il y a au moins une ligne dans chaque colonne qui correspond à la condition.

Pour utiliser ce type de jointure il convient d'utiliser une requête SQL avec cette syntaxe :

```
SELECT *
FROM table1
INNER JOIN table2 ON table1.id = table2.fk_id
```

La syntaxe ci-dessus stipule qu'il faut sélectionner les enregistrements des tables table1 et table2 lorsque les données de la colonne "id" de table1 est égal aux données de la colonne fk_id de table2.

La jointure SQL peut aussi être écrite de la façon suivante :

```
SELECT *
FROM table1
INNER JOIN table2
WHERE table1.id = table2.fk_id
```

La syntaxe avec la condition WHERE est une manière alternative de faire la jointure mais qui possède l'inconvénient d'être moins facile à lire s'il y a déjà plusieurs conditions dans le WHERE.

On peut aussi associer une ou plusieurs condition(s) supplémentaire(s) avec l'opérateur AND :

```
SELECT *
FROM table1
INNER JOIN table2 ON table1.id = table2.fk_id
AND condition2
```