Concevoir une base de données relationnelle à partir d'un cahier des charges

6. Ajouter une table JOURNAL dont le but est d'historiser les modifications qui seront faites dans la base (date, heure, température, précipitations, observations)

```
CREATE TABLE IF NOT EXISTS `journal` (
  `id_journal` int NOT NULL AUTO_INCREMENT,
  `date_modif` date,
  `heure_modif` time,
  `temperature` decimal(3,2),
  `precipitations` decimal(4,2),
  `observations` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`id_journal`)
) ENGINE=MyISAM DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

Lors de la SAÉ 2.04 « Exploitation d'une base de données », mes camarades et moi, nous avons défini un table « JOURNAL » qui permet de stocker la date, l'heure, la température, les précipitations et observations.

Mettre à jour et interroger une base de données relationnelle

```
DROP TRIGGER IF EXISTS journalisation;

DELIMITER $$
CREATE TRIGGER journalisation
AFTER INSERT ON releves
FOR EACH ROW
BEGIN
INSERT into journal(date_modif,temperature,precipitations)
VALUES(new.date,new.temperature,new.precipitations);
END$$

DELIMITER ;
DROP TRIGGER IF EXISTS verif_temperature;

DELIMITER $$
CREATE TRIGGER verif_temperature
BEFORE INSERT ON journal
FOR EACH ROW
BEGIN
    IF new.temperature > (SELECT avg(temperature) from releves WHERE
year(date)=year(new.date_modif)-1 and MONTH(date)=MONTH(new.date_modif) GROUP
BY month(date)) THEN
    set new.observations="ANORMAL";
END IF;
END$$

DELIMITER ;
```

Je suis capable de mettre à jour automatiquement une base de données relationnelle par le biais de « déclencheurs (Triggers) ». Dans la SAÉ 2.04 « Exploitation d'une base de données », mes camarades et moi-même avons mis en place un trigger sur la table « relevés », journalisation se chargeait, après chaque insert, d'insérer les données nécessaires dans la table journal.

Et la 2ème nommée « verfi_temperature », vérifie si la nouvelle température est supérieure à la moyenne des températures du même mois au cours de l'année précédente (et marque anormal si c'est le cas).

Visualiser des données

```sql
-- Vue pour les températures maximales par mois
CREATE OR REPLACE VIEW TemperaturesMaxParMois AS
SELECT
    YEAR(date) AS annee,
    mois,
    MAX(temperature_max) AS temperature_maximale
FROM releves
GROUP BY YEAR(date), mois
ORDER BY annee, mois;
```

```python
    def temperatures_maximales(self):
        self.conn("meteo_du_lamentin")
        self.curseur.execute('SELECT * FROM TemperaturesMaxParMois')

        # Récupération des résultats
        s = f"| Températures maximales mensuelles |"
        print("-"*len(s))
        print(s)
        print("-"*len(s))

        for ligne in self.curseur:
            annee, mois, temperature = ligne
            s=f"| {mois}/{annee}: {f'{temperature:.1f}' if temperature is not
None else 'N/A'}°C |"
            print(s)
            print("-"*len(s))
        print()
```

```
-----------------------------------
| Températures maximales mensuelles |
-----------------------------------
| 1/2009: N/A°C |
-----------------
| 1/2010: 31.0°C |
------------------
| 2/2010: 32.5°C |
------------------
| 3/2010: 34.4°C |
------------------
| 4/2010: 32.9°C |
------------------
| 5/2010: 33.6°C |
------------------
| 6/2010: 33.7°C |
------------------
```

Je suis capable de visualiser des données en structurant leur affichage de manière claire et pertinent. Dans la SAÉ 2.04 « Exploitation d'une Bases de données », mes camarades et moi nous avons créé des « VUES » SQL tels que « TemperaturesMaxParMois » permettant de synthétiser les températures extrêmes par mois et par année. Ces vues facilitent l'analyse rapide des données climatiques et leur exploitation par des utilisateurs non techniques.