**Titre**: "Analyse des Liens entre Température, Hospitalisations et COVID-19"

**Auteurs**: " Michael NYANYUIE,  
Divin IZERE,   
Lucky Bernice IKAZE,   
Mehdi HIMMICHE,   
Rayane BENCHOKROUN   
"

**1. Création des Tables**

Le processus a débuté par importer les tables PROJET\_DEP, PROJET\_REG et PROJET\_POPU fournies au commencement du projet et la création de deux tables principales conçues pour stocker les données pertinentes :

* **Table TEMPERATURES** : Cette table a été établie pour capturer les données de température quotidiennes par département, comprenant les températures minimales, maximales et moyennes.
* *Commande SQL* :

CREATE TABLE TEMPERATURES(

        DATE\_TEMP DATE,

        DEP\_ID VARCHAR(3),

        DEPARTEMENT VARCHAR(50),

        MIN\_TEMP DECIMAL(10, 2),

        MAX\_TEMP DECIMAL(10, 2),

        MEAN\_TEMP DECIMAL(10, 2),

        PRIMARY KEY (DATE\_TEMP, DEP\_ID)

    );

* **Table HOSPITALISATIONS** : Créée pour suivre les données sur les visites hospitalières par département et par date, comprenant le nombre d'hospitalisations et des résultats spécifiques tels que les soins intensifs et les décès.
* *Commande SQL* :

 CREATE TABLE HOSPITALISATIONS(

        DEP\_ID VARCHAR(3),

        SEX INT,

        DATE\_HOSP DATE,

        NUM\_HOSP INT,

        NUM\_INTENSE\_CARE INT,

        NUM\_CONV\_HOSP INT,

        NUM\_SSR\_USLD INT,

        NUM\_OTHERS INT,

        NUM\_RETURN\_HOME INT,

        NUM\_DEATHS INT,

        PRIMARY KEY (DEP\_ID, DATE\_HOSP, SEX)

    );

**2. Importation des Données dans les Tables**

Une fois les tables créées, les données ont été importées à partir de fichiers CSV segmentés :

* *Segmentation et Importation* : Le jeu de données original a été divisé en morceaux gérables pour faciliter le traitement efficace. Chaque type de données (TEMPERATURES et HOSPITALISATIONS) a été divisé en au moins cinq fichiers CSV, chacun contenant environ 20 000 enregistrements.
* *Processus d'Importation* : Les données de ces fichiers ont été importées dans les tables respectives, en veillant à ce que tous les enregistrements soient correctement positionnés pour un traitement et une analyse ultérieurs.

**3. Nettoyage des Données**

"Avant d'appliquer les contraintes relationnelles, il était essentiel de standardiser et nettoyer les données pour garantir la cohérence entre les tables :

* *Standardisation des Identifiants de Département* :
* Les champs DEP\_ID des tables TEMPERATURES et HOSPITALISATIONS ont été ajustés pour correspondre au format de PROJET\_DEP. Cela impliquait la suppression des zéros non significatifs qui pourraient causer des incohérences lors des jointures de données.
* *Commandes SQL pour le Nettoyage* :

UPDATE TEMPERATURES SET DEP\_ID = SUBSTRING(DEP\_ID, 2) WHERE DEP\_ID LIKE '0%';

UPDATE HOSPITALISATIONS SET DEP\_ID = SUBSTRING(DEP\_ID, 2) WHERE DEP\_ID LIKE '0%';

**4. Ajout des Clés Étrangères**

Après avoir rempli les tables avec des données, des contraintes de clé étrangère ont été ajoutées pour assurer l'intégrité des données et la cohérence relationnelle :

* *Contraintes de Clé Étrangère* :
* *TEMPERATURES* :

ALTER TABLE TEMPERATURES ADD CONSTRAINT temp\_depID\_fk FOREIGN KEY (DEP\_ID) REFERENCES PROJET\_DEP(codeD);

* *HOSPITALISATIONS* :

ALTER TABLE HOSPITALISATIONS ADD CONSTRAINT hosp\_depID\_fk FOREIGN KEY (DEP\_ID) REFERENCES PROJET\_DEP(codeD);

**5. Filtrage des Données Inutiles**

La dernière étape consistait à supprimer les données qui n'étaient pas nécessaires pour l'analyse :

* *Suppression des Années Spécifiques* :
* Les données au-delà de 2021 ont été jugées non pertinentes pour l'analyse et ont été supprimées des deux tables.
* *Commandes SQL pour la Suppression des Données* :

DELETE

    FROM TEMPERATURES

    WHERE DATE\_TEMP LIKE '2022-%' OR DATE\_TEMP LIKE '2023-%' OR DATE\_TEMP LIKE '2024-%';

   DELETE

FROM HOSPITALISATIONS

WHERE DATE\_HOSP LIKE '2022-%' OR DATE\_HOSP LIKE '2023-%';

* *Suppression de Catégories de Sexe Spécifiques* :
* Les entrées pour les sexes masculin et féminin ont été supprimées de la table HOSPITALISATIONS pour se concentrer éventuellement sur d'autres catégories ou pour corriger des erreurs de saisie de données.
* *Commande SQL* (hypothétique, car des valeurs spécifiques pour SEXE n'ont pas été fournies) :

DELETE FROM HOSPITALISATIONS WHERE SEX = 1 OR SEX = 2;

**6. Utilisation de CTE et Logique Conditionnelle dans les Requêtes**

Au lieu de créer des tables permanentes avec des variables spécifiques ou des ensembles de données pour notre analyse, nous avons souvent utilisé des Expressions de Table Communes (CTEs) ou une logique conditionnelle dans nos requêtes SQL. Cette approche nous permet de créer des tables temporaires, sur le moment, qui peuvent être adaptées à des analyses spécifiques et produire les résultats attendus. Voici quelques exemples de comment nous avons utilisé des CTE et des instructions conditionnelles :

**Exemple 1 : Analyse Régionale des Hospitalisations**

Cette requête calcule le nombre moyen d'hospitalisations et catégorise les départements par régions :

SELECT

    AVG(H.NUM\_HOSP) AS AVG\_NUM\_HOSP,

    CASE

        WHEN H.DEP\_ID IN ('59', '62', '80', '60', '02', '76', '14', '50', '27', '77', '95') THEN 'Nord'

        WHEN H.DEP\_ID IN ('13', '83', '84', '06', '31', '33', '66', '34', '11', '30', '24') THEN 'Sud'

        ELSE 'D.O.M'

    END AS 'REGIONS'

FROM HOSPITALISATIONS H

INNER JOIN PROJET\_DEP D ON H.DEP\_ID = D.codeD

GROUP BY REGIONS

ORDER BY AVG\_NUM\_HOSP DESC;

**Exemple 2 : Analyse Saisonnière Pendant et Hors Périodes COVID**

Cet ensemble de CTE analyse les taux d'hospitalisation à travers différentes saisons pendant et en dehors des périodes COVID-19 :

*-- SAISONS PENDANT ET HORS PÉRIODES COVID*

AVEC PERIODE\_COVID AS(

    SELECT

        AVG((NUM\_HOSP/ total) \* 10000) AS TAU\_HOSP\_PENDANT\_COVID,

        CASE

        WHEN MONTH(DATE\_HOSP) IN (12, 1, 2) THEN 'Hiver'

        WHEN MONTH(DATE\_HOSP) IN (3, 4, 5) THEN 'Printemps'

        WHEN MONTH(DATE\_HOSP) IN (6, 7, 8) THEN 'Été'

        WHEN MONTH(DATE\_HOSP) IN (9, 10, 11) THEN 'Automne'

        END AS 'SAISONS'

    FROM HOSPITALISATIONS H

    INNER JOIN PROJET\_POPU P

    ON H.DEP\_ID = P.codeD

    WHERE DATE\_HOSP LIKE '2020-%' OR DATE\_HOSP LIKE '2019-%'

    GROUP BY SEASONS

), HORS\_COVID AS(

    SELECT

        AVG((NUM\_HOSP/ total) \* 10000) AS TAU\_HOSP\_HORS\_COVID,

        CASE

        WHEN MONTH(DATE\_HOSP) IN (12, 1, 2) THEN 'Hiver'

        WHEN MONTH(DATE\_HOSP) IN (3, 4, 5) THEN 'Printemps'

        WHEN MONTH(DATE\_HOSP) IN (6, 7, 8) THEN 'Été'

        WHEN MONTH(DATE\_HOSP) IN (9, 10, 11) THEN 'Automne'

        END AS 'SAISONS'

    FROM HOSPITALISATIONS H

    INNER JOIN PROJET\_POPU P

    ON H.DEP\_ID = P.codeD

     WHERE DATE\_HOSP LIKE '2018-%' OR DATE\_HOSP LIKE '2021-%'

    GROUP BY SEASONS

)

SELECT TAU\_HOSP\_PENDANT\_COVID, TAU\_HOSP\_HORS\_COVID, CP.SEASONS

FROM PERIODE\_COVID CP

INNER JOIN HORS\_COVID HC

ON CP.SEASONS = HC.SEASONS

ORDER BY TAU\_HOSP\_PENDANT\_COVID DESC, TAU\_HOSP\_HORS\_COVID DESC;