```
import pymysql
import pandas as pd
import os

# Connexion à la base de données MySQL
def connect_to_db():
  try:
    db = pymysql.connect(
      host="localhost",
      user="root",
      password="",
      database="selmarin_tdmk_aubin",
      local_infile=True
    )
    print("✅ Connexion réussie à la base de données.")
    return db
  except pymysql.MySQLError as e:
    print(f"❌ Erreur de connexion à la base de données : {e}")
    return None

cursor = None

# Définition des chemins relatifs
excel_file = "sel_marin_2023.xlsx"  # Changez le nom de votre fichier Excel si nécessaire
output_dir = "csv_output"

# Fonction pour convertir le fichier Excel en CSV
def excel_to_csv(excel_file, output_dir):
```

```
    try:
        if not os.path.exists(excel_file):
            print(f"❌ Erreur : Le fichier Excel '{excel_file}' est introuvable.")
            return False

        xls = pd.ExcelFile(excel_file)
        os.makedirs(output_dir, exist_ok=True)

        for sheet_name in xls.sheet_names:
            df = xls.parse(sheet_name)
            csv_filename = os.path.join(output_dir, f"{sheet_name}.csv")
            df.to_csv(csv_filename, index=False, sep=';', encoding='utf-8')
            print(f"✅ Fichier CSV créé pour {sheet_name}: {csv_filename}")

        return True

    except Exception as e:
        print(f"❌ Erreur lors de la conversion : {e}")
        return False

# Fonction pour insérer les données CSV dans MySQL
def insert_data_from_csv(csv_filename, table_name, columns):
    try:
        if not os.path.exists(csv_filename):
            print(f"❌ Erreur : Le fichier '{csv_filename}' est introuvable.")
            return

        df = pd.read_csv(csv_filename, delimiter=';', encoding='utf-8')
```

```
        placeholders = ', '.join(['%s'] * len(columns))

        if table_name == "sortie":
            update_clause = "qteSort = qteSort + VALUES(qteSort)"
            unique_columns = ["NumSort", "numPdt"]
        else:
            update_clause = ', '.join([f"{col}=VALUES({col})" for col in columns if col != columns[0]])
            unique_columns = [columns[0]]

        sql = f"""
            INSERT INTO {table_name} ({', '.join(columns)})
            VALUES ({placeholders})
            ON DUPLICATE KEY UPDATE {update_clause};
        """

        for row in df.itertuples(index=False, name=None):
            try:
                cursor.execute(sql, row)
            except Exception as e:
                print(f"❌ Erreur lors de l'insertion dans {table_name} pour la ligne {row}: {e}")
                continue  # Continue même en cas d'erreur

        print(f"✅ Données insérées dans la table {table_name} avec succès.")

    except Exception as e:
        print(f"❌ Erreur lors de l'insertion des données dans {table_name}: {e}")
```

```
# Fonction pour récupérer les données de la table SORTIE
def get_sortie_data():
    try:
        cursor.execute("SELECT numSort, dateSort, numCli FROM sortie")
        sortie_data = cursor.fetchall()  # Récupère toutes les lignes
        return sortie_data
    except pymysql.MySQLError as e:
        print(f"❌ Erreur lors de la récupération des données de la table SORTIE : {e}")
        return []

# Fonction pour vérifier si un numSort existe dans la table CONCERNER
def check_concerner_exists(numSort):
    try:
        cursor.execute("SELECT COUNT(*) FROM concerner WHERE numSort = %s", (numSort,))
        result = cursor.fetchone()[0]  # Récupère le premier élément (le nombre d'occurrences)
        return result > 0  # Si result > 0, cela signifie que le numSort existe dans CONCERNER
    except pymysql.MySQLError as e:
        print(f"❌ Erreur lors de la vérification de l'existence de numSort dans CONCERNER : {e}")
        return False

# Fonction pour insérer des données dans la table CONCERNER
def insert_concerner_data(sortie_data):
    numSort_list = []  # Liste pour stocker les numSort récupérés
    inserted_sort = []  # Liste pour suivre les numSort insérés
```

```
    for entry in sortie_data:
        numSort = entry[0]  # numSort de la table SORTIE
        dateSort = entry[1]  # dateSort de la table SORTIE
        numCli = entry[2]  # numCli de la table SORTIE

        try:
            # Ajouter numSort à la liste
            numSort_list.append(numSort)

            # Vérifier si le numSort existe déjà dans CONCERNER
            if not check_concerner_exists(numSort):
                # Si le numSort n'existe pas dans CONCERNER, insérer dans CONCERNER
                produits = [(1, 500), (2, 200)]  # Exemple de produits (numPdt, qteSort)
                for numPdt, qteSort in produits:
                    sql = "INSERT INTO concerner (numSort, numPdt, qteSort) VALUES (%s, %s, %s)"
                    cursor.execute(sql, (numSort, numPdt, qteSort))
                    print(f"✅ Données insérées dans CONCERNER pour numSort {numSort}, numPdt {numPdt}")
                    inserted_sort.append(numSort)  # Ajouter à la liste des numSort insérés
            else:
                print(f"❌ Le numSort {numSort} existe déjà dans CONCERNER. Aucune donnée insérée.")
        except pymysql.MySQLError as e:
            print(f"❌ Erreur lors de l'insertion dans CONCERNER pour numSort {numSort}: {e}")
            continue  # Continue même en cas d'erreur
```

```
    return numSort_list, inserted_sort


# Fonction principale
def main():
    global cursor
    db = connect_to_db()  # Connexion à la base de données
    if not db:
        return  # Si la connexion échoue, on arrête l'exécution

    cursor = db.cursor()

    if not excel_to_csv(excel_file, output_dir):
        return

    # Charger et insérer les données dans les tables
    tables = {
        "ENTREE": ("entree", ["NumEnt", "dateEnt", "qteEnt", "numPdt", "numSau"]),
        "SAUNIER": ("saunier", ["numSau", "nomSau", "prenomSau", "villeSau"]),
        "CLIENT": ("client", ["numCli", "nomCli", "precisionCli", "villeCli"]),
        "SORTIE": ("sortie", ["NumSort", "dateSort", "numCli", "numPdt", "qteSort"])
    }

    for sheet_name, (table_name, columns) in tables.items():
        csv_filename = os.path.join(output_dir, f"{sheet_name}.csv")
        insert_data_from_csv(csv_filename, table_name, columns)

    # Récupérer les données de la table SORTIE
    sortie_data = get_sortie_data()
```

```
    if sortie_data:
        # Insérer les données dans la table CONCERNER et obtenir les numSort insérés
        numSort_list, inserted_sort = insert_concerner_data(sortie_data)

        # Affichage des numSort traités
        print(f"Liste de numSort récupérés : {numSort_list}")
        print(f"Liste des numSort insérés dans CONCERNER : {inserted_sort}")

    db.commit()
    db.close()
    print("✅ Programme terminé avec succès.")

if __name__ == "__main__":
    main()
```