TRAVAUX PRATIQUES SUR LA BASE DE DONNÉES DU SITE IMDB
Le site imdb.com met des bases de données à disposition :
https://www.imdb.com/interfaces/
https://datasets.imdbws.com

Il faudra utiliser le fichier imdb.db .

```
import sqlite3
conn = sqlite3.connect('imdb.db')
c = conn.cursor()
c.execute("select * from name_basics limit 10")
for row in c:
    print(row)
conn.close()
```

Dans ce projet vous devrez rédiger des fonctions Python afin d'interroger une base de données.
Compléter le schéma de la base de données imdb.db en traçant les liens entre les tables. On pourra parcourir les données sur DB Browser pour constater la correspondance de certains champs.

name_basics	
nconst	int
primaryName	varchar(105)
birthYear	smallint
deathYear	smallint
primaryProfession	varchar(66)

title_akas	
titleId	int
ordering	smallint
title	varchar(831)
region	varchar(4)
language	varchar(3)
types	varchar(20)
Attributes	varchar(62)
isOriginalTitle	bit

name_titles	
nconst	int
knownForTitles	int

title_directors	
tconst	int
directors	int

title_principals	
tconst	int
ordering	smallint
nconst	int
category	varchar(20)
job	varchar(286)
characters	varchar(463)

title_basics	
tconst	int PRIMARY KEY
titleType	varchar(12)
primaryTitle	varchar(419)
originalTitle	varchar(419)
isAdult	bit
startYear	smallint
endYear	smallint
runtimeMinutes	smallint
genres	varchar(32)

title_writers	
tconst	int
writers	int

title_episode	
tconst	int PRIMARY KEY
parentTconst	int
seasonNumber	smallint
episodeNumber	smallint

title_ratings	
tconst	int PRIMARY KEY
averageRating	float(24)
numVotes	int

Dans une première version de cette base de données, les tables name_basics et name_titles étaient fusionnées en une table name_basics ayant le schéma suivant.
Expliquer les avantages du système actuel à deux tables

name_basics	
nconst	int
primaryName	varchar(105)
birthYear	smallint
deathYear	smallint
primaryProfession	varchar(66)
knownForTitles	int

Pour votre culture personnelle, vous pouvez étudier le potentiel de la bibliothèque pprint pour pouvoir afficher ces longues listes de manière lisible. Ne rendez pas compte de cette étude dans votre rapport .
Vos requêtes sont dans un répertoire requêtes sous la forme :

req1.sql
req2.sql
etc ….
Ce même répertoire contient un fichier markdown alire.md contenant les informations et la liste des requêtes.
Exemple :

```
requ2.sql
#02.Donnez tous les pays d'Amérique du Sud ← sur la première ligne, il y a la question
SELECT Name                                ← Ensuite, la requête sur plusieurs lignes pour une meilleure
FROM Country                                 lisibilité.
WHERE Country.Continent="South America" ;
```

Vous chargez les requêtes dans un dictionnaire sous la forme :
req[1]=[‘la question’,’ sql 1’]
req[2]=[‘la question’,’ sql 2’]
……

Il faut tester par la suite, si la requête est valide , s’il y a une erreur dans la requête, il faut en tenir compte et le signaler par un message du type erreur au moment de l’exécution dans la reqx, mais je continue le travail.

Remarque sur l’affichage : Créez une fonction afficher_table qui affiche le contenu d’une table sous un format ASCII dont voici un modèle (vous pouvez regarder le module pprint pour information):

```
+---+--------------+----+--------------+
|1  |Kabul         |AFG |Kabol         |
+---+--------------+----+--------------+
|2  |Qandahar      |AFG |Qandahar      |
+---+--------------+----+--------------+
|3  |Herat         |AFG |Herat         |
+---+--------------+----+--------------+
|4  |Mazar-e-Sharif|AFG |Balkh         |
+---+--------------+----+--------------+
|5  |Amsterdam     |NLD |Noord-Holland |
+---+--------------+----+--------------+
|6  |Rotterdam     |NLD |Zuid-Holland  |
+---+--------------+----+--------------+
|7  |Haag          |NLD |Zuid-Holland  |
+---+--------------+----+--------------+
|8  |Utrecht       |NLD |Utrecht       |
+---+--------------+----+--------------+
|9  |Eindhoven     |NLD |Noord-Brabant |
+---+--------------+----+--------------+
|10 |Tilburg       |NLD |Noord-Brabant |
+---+--------------+----+--------------+
|11 |Groningen     |NLD |Groningen     |
+---+--------------+----+--------------+
```

La fonction (en fait, une procédure, à strictement parler), prend en argument une table et deux arguments optionnels : debut, qui vaut 0 par défaut et fin, qui vaut None par défaut. La table est affichée de la ligne debut à la ligne fin. Si fin vaut None, la table est affichée jusqu’à la fin. Donc par défaut, la table est entièrement affichée.
Créez une fonction projection_table qui prend en argument une table et une suite d’arguments correspondant à des numéros de colonnes. Cette fonction renvoie une table correspondant à la table en entrée, mais dont on ne garde que les colonnes spécifiées par la liste des numéros. Ainsi afficher_table(projection_table(villes,1,3),0,10) affiche :

```
+--------------+--------------+
|Kabul         |Kabol         |
+--------------+--------------+
|Qandahar      |Qandahar      |
+--------------+--------------+
|Herat         |Herat         |
+--------------+--------------+
|Mazar-e-Sharif|Balkh         |
+--------------+--------------+
|Amsterdam     |Noord-Holland |
+--------------+--------------+
|Rotterdam     |Zuid-Holland  |
+--------------+--------------+
|Haag          |Zuid-Holland  |
+--------------+--------------+
|Utrecht       |Utrecht       |
+--------------+--------------+
|Eindhoven     |Noord-Brabant |
+--------------+--------------+
|Tilburg       |Noord-Brabant |
+--------------+--------------+
|Groningen     |Groningen     |
+--------------+--------------+
```

Créez une fonction produit_cartesien qui prend en argument deux tables et renvoie le produit cartésien de ces deux tables. Cette fonction sera utile par la suite. Ainsi, si vous opérez le produit cartésien d'une table de 3 lignes de 4 champs et d'une table de 5 lignes de 2 champs, vous obtenez une table de 15 lignes de 6 champs (donc une liste de tuples de taille 6). Vous aurez donc noté qu'en termes mathématiques, ce n'est pas à strictement parler un produit cartésien !

Les requêtes :
Créez la fonction qui renvoie la table contenant les informations demandées :
Il y a affichage de la requête (la question) et du résultat via afficher_table.

1) Ecrire les requêtes SQL permettant de répondre aux questions suivantes.
2) Quels sont les différents types de titres dans cette base de données ?
3) Combien y a-t-il de titres dans cette base de données ?
4) En quelle année est sortie le film The Godfather ?
5) En quelle année est sortie le premier film Superman ?
6) Quel est le titre original du film 'Les dents de la mer' ?
7) Quel est le métier d'Olivier Nakache ?
8) Quels sont les films d'Olivier Nakache ?
9) Quel est le film ayant recueilli le plus de votes ?
10) Qui a écrit le scénario du film Taxi sorti en 1998 ?
11) Quelles sont les noms et rôles (category et job) des personnes intervenant dans la production du film Return of the Jedi ?
12) Quels sont les titres des films notés plus de 9 sur 10 avec plus de 10 000 votes ?
13) Quelle sont les 5 comédies romantiques les mieux notées ?
14) Quels sont les 10 films d'animation ayant reçu plus de 1000 votes les mieux notés ?
15) Combien de films durent plus de 3 heures ?
16) Quelle est la durée moyenne d'un film ?
17) Quel est le film le plus long ?
18) Quels sont les 5 films les plus longs ?
19) Quels sont les titres des films les plus connus de Sean Connery ?
20) Quels sont les acteurs ayant joué le rôle de James Bond, et dans quels films ?
21) Quel sont les réalisateurs ayant fait les cinq film les mieux notés ? Indiquer les noms des films correspondants.
22) Quels sont les noms des épisodes de Game of Thrones ?

Compte-rendu du travail
Le rendu du travail sera constitué du code Python, qui devra être commenté un affichage des tables produites (utilisation de afficher_table). Il faudra donner à chaque fois le nombre de lignes de la table, ainsi que l'affichage de seulement les 10 premières lignes.
Il faut déposer votre projet sur https://gitlab.com/ pour que je puisse suivre votre travail.
Vous m'indiquerez l'adresse de votre gitlab.
https://education.github.com/students
https://docs.gitlab.com/ee/user/profile/account/create_accounts.html

Sinon, vous pouvez utiliser monbureaunumerique.fr et créer un répertoire d'échanges

La partie programmation sera évaluée sur les critères suivants :
Correction des fonctions : le résultat donné est-il correct ?
Lisibilité des fonctions : le code est-il commenté ? Les variables, le nom des fonctions ont-elles un nom lisible ?
Généricité du code : le prototype des fonctions vous permettent-ils de les réutiliser d'une question à l'autre ?
Modularité : avez-vous pensé à créer des fonctions supplémentaires pour les parties de code répétitives ?

```
import sqlite3
conn = sqlite3.connect('imdb.db')
c = conn.cursor()
c.execute("select * from name_basics limit 10")
```

```
for row in c:
    print(row)
conn.close()
```

```
import sqlite3

class database:
  #https://docs.python.org/3/library/sqlite3.html
  def __init__(self, base):
    self.base = ""

  def connexion(self):
    self.con = .sqlite3.connect(self.base)

    self.cur = self.con.cursor()

  def deconnexion(self):
    self.con.close()

  def fetch(self, sql):
    self.connexion()
    self.cur.execute(sql)
    result = self.cur.fetchall()
    self.deconnexion()
    return result

  def execute(self, sql):
    self.connexion()
    self.cur.execute(sql)
    self.deconnexion()

  def chargersql():
    pass

  def afficher_table():
    pass

  def listedesrequetes():
    pass

  def infotable():
    pass


  def informations_base():
    pass


  def ....
    pass
```

Vous pouvez faire une interface avec tk ou qt5.