

Écriture du code Projet 5 développement Python

```
# coding: utf-8
import requests
import json
import pymysql
import pymysql.cursors
import math
import copy

"""CONNECT TO THE DATABASE"""
try :
except :
    print("erreur de connexion")

"""RESET THE COUNTERS"""

with connection.cursor() as cursor:
with connection.cursor() as cursor:
with connection.cursor() as cursor:

"""CLEANING TABLES"""

with connection.cursor() as cursor:
with connection.cursor() as cursor:
with connection.cursor() as cursor:
```

Création de la base de données

```
1 CREATE DATABASE P5;
2
3 USE P5;
4
5 CREATE TABLE CATEGORIES
6 (ID INT UNSIGNED AUTO_INCREMENT NOT NULL,
7  NOM VARCHAR(80),
8  LINK_OFF VARCHAR(80),
9  PRIMARY KEY (ID)
10 )ENGINE=INNODB;
11
12 CREATE TABLE PRODUIT(
13  PRODUIT_ID INT UNSIGNED AUTO_INCREMENT NOT NULL,
14  NOM VARCHAR(200),
15  PRODUIT_URL VARCHAR(150),
16  STORE VARCHAR(300),
17  NUTRIScore INT,
18  CATEGORIE_ID INT UNSIGNED,
19  PRIMARY KEY (PRODUIT_ID),
20  CONSTRAINT fk_ok
21  FOREIGN KEY (CATEGORIE_ID) REFERENCES CATEGORIES (ID)
22 )ENGINE=INNODB;
23
24
25 CREATE TABLE SUBSTITUT(
26  SUBSTITUT_ID INT UNSIGNED AUTO_INCREMENT NOT NULL,|
27  NOM VARCHAR(80),
28  STORE VARCHAR(80),
29  SUBSTITUT_URL VARCHAR(100),
30  DESCRIPTION VARCHAR(200),
31  PRIMARY KEY (SUBSTITUT_ID)
32 )ENGINE=INNODB;
33
```

Connexion à la base de données et initialisation

```
8
9 """CONNECT TO THE DATABASE"""
10 try :
11     connection = pymysql.connect(host='127.0.0.1',
12                                 user='root',
13                                 password='azerty',
14                                 db='P5',
15                                 |
16                                 charset='utf8mb4',
17                                 port = 3306,
18                                 cursorclass=pymysql.cursors.DictCursor)
19 except :
20     print("erreur de connexion")
21
22 """RESET THE COUNTERS"""
23 with connection.cursor() as cursor:
24     sql = "ALTER TABLE `CATEGORIES` AUTO_INCREMENT=0;"
25     cursor.execute(sql, ())
26     connection.commit()
27
28 with connection.cursor() as cursor:
29     sql = "ALTER TABLE `PRODUIT` AUTO_INCREMENT=0;"
30     cursor.execute(sql, ())
31     connection.commit()
32
33 with connection.cursor() as cursor:
34     sql = "ALTER TABLE `SUBSTITUT` AUTO_INCREMENT=0;"
35     cursor.execute(sql, ())
36     connection.commit()
37
38 """CLEANING TABLES"""
39
40 with connection.cursor() as cursor:
41     sql = "DELETE FROM `PRODUIT`"
42     cursor.execute(sql, ())
43     connection.commit()
44
45 with connection.cursor() as cursor:
46     sql = "DELETE FROM `CATEGORIES`;"
47     cursor.execute(sql, ())
48     connection.commit()
49
50 with connection.cursor() as cursor:
51     sql = "DELETE FROM `SUBSTITUT`"
52     cursor.execute(sql, ())
53     connection.commit()
54
```



Création d'un menu utilisateur 1/2 : Boucle principale

```
247 """MAIN LOOP"""
248
249 continu = 0
250 while continu == 0 :
251     try :
252         mode_terminal = input("1 - Quel aliment souhaitez-vous remplacer ? \n2 - Retrouver mes aliments substitués. \n3 - Sortir du programme ?")
253         mode_terminal = int(mode_terminal)
254
255         """mode remplacement"""
256         while mode_terminal == 1 :
257             """mode BDD"""
258
259             while mode_terminal == 2 :
260
261                 """Mode sortie"""
262
263                 if mode_terminal == 3 :
264
265                     elif mode_terminal >= 3 :
266                         print("{} n'est pas dans les numéros proposés\n".format(mode_terminal))
267
268             except ValueError :
269                 if len(mode_terminal) > 1 :
270                     print("\nOops! {} est un mot, veuillez recommencer : \n".format(mode_terminal))
271                 else :
272                     print("\nOops! {} est une lettre, veuillez recommencer : \n".format(mode_terminal))
```

Création d'un menu utilisateur 2/2

choix d'une catégorie

```
"""mode remplacement"""
while terminal_mode == 1 :
    print(transition)
    """condition for mode_substitut categorie"""
    print(mode_substitut_categorie)
    temp_categorie = int
    try :
        keyboard_input = input ("choisissez votre catégorie de produit : " + proposition)
        keyboard_input = int(keyboard_input)
        print(type(keyboard_input))
        choice = str
        name_categories = str

        if keyboard_input in nb_series :
            choice = categories[keyboard_input - 1]
            name_categories = categories[keyboard_input - 1]
            link_categories = link[l = keyboard_input - 1]
            temp_categorie = keyboard_input
            print(temp_categorie)
            print("model")
        elif keyboard_input not in nb_series :
            print("{} n'est pas dans les numéros proposés\n".format(keyboard_input))
            break

        if keyboard_input in mode_substitut_categorie :
            print("catégories {} existante dans la BDD".format(name_categories))
            select_and_substitut()
        else :
            with connection.cursor() as cursor:
                sql = "INSERT INTO `CATEGORIES` (`NOM`,`LINK OFF`) VALUES (%s, %s)"
                cursor.execute(sql, (name_categories, link_categories))
            mode_substitut_categorie.append(temp_categorie)
            connection.commit()
            Add_product()
            print(transition)
            select_and_substitut()

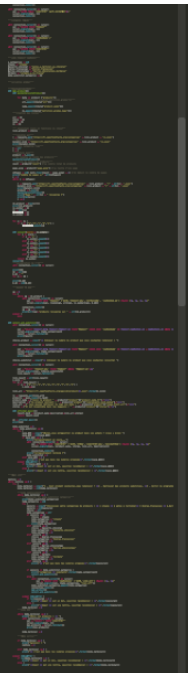
    except ValueError :
        print("\nOops! {} est une lettre, veuillez recommencer : \n".format(keyboard_input))

    terminal_mode = 0

"""Consult database"""
while terminal_mode == 2 :
    print(transition)
    print("Voici vos produits : \n ")
    with connection.cursor() as cursor:
        sql = "SELECT `NOM`,`STORE`,`SUBSTITUT_ID` FROM `SUBSTITUT`"
        cursor.execute(sql, ())
        my_product = cursor.fetchall()
    print(my_product)
```

Utilisation du module json pour récupérer les données d'une API 1/2

```
68 """PROGRAM FUNCTIONS"""
69 def Add_product():
70     def AlimenterListeProduit():
71         for data in product ["products"]:
72             """nutriscore = (data["nutrition_grades"])"""
73             url.append((data["url"]))
74             name.append((data["product_name"]))
75             ns.append((data["nutrition_grades_tags"]))
76         """creation des listes"""
77         url = []
78         store = []
79         name = []
80         ns = []
81         """ adaptation liens en fonctions du choix"""
82         link_product = choice
83         """requêtes get"""
84         r = requests.get('https://fr.openfoodfacts.org/categorie/' + link_product + '/1.json')
85         """dynamic link"""
86         dynamic_link = 'https://fr.openfoodfacts.org/categorie/' + link_product + '/1.json'
87         print(dynamic_link)
88         """liens catégorie figé"""
89         l = str
90         l = r
91         """lecture json"""
92         product = r.json()
93         """on remplit la liste des produits"""
94         AlimenterListeProduit()
95         """calcul du nombre de produits"""
96         count = product['count'] # Le nombre total de produits
97         page_size = product['page_size'] # La taille d'une page
98         nbPages = int( math.floor(count / page_size) + 1) # On déduit le nombre de pages
99         print("nombre de pages = " +str(nbPages))
100         i = 2
101         while i <= nbPages:
102             r = requests.get('https://fr.openfoodfacts.org/categorie/' + link_product + "/" + str(i)+ '.json')
103             dynamic_link = 'https://fr.openfoodfacts.org/categorie/' + link_product + "/" + str(i)+ '.json'
104             print(dynamic_link)
105             product = r.json()
106             AlimenterListeProduit()
107             print("Page " + str(i) + " récupérée !")
108             print(len(url))
109             i += 1
110
```



Utilisation du module json pour récupérer les données d'une API 2/2

```
119 1 == 1
120
121 nb_product = (len(url))
122 print(nb_product)
123 ns_number = []
124 ns = str(ns)
125 ns_simple = []
126 ns_number = []
127
128
129 for x in ns :
130     if x in ("a","b","c","d","e") :
131         ns_simple.append(x)
132
133
134 def convert(score = ns_simple):
135     for i in score :
136         if i == 'a' :
137             ns_number.append(1)
138         elif i == "b" :
139             ns_number.append(2)
140         elif i == "c" :
141             ns_number.append(3)
142         elif i == "d" :
143             ns_number.append(4)
144         elif i == "e" :
145             ns_number.append(5)
146     return ns_number
147 convert()
148 """récupération de l'ID"""
```

Utilisation de la Fonction INSERT en MYSQL

```
145         ns_number.append(5)
146     return ns_number
147     convert()
148     """récupération de l'ID"""
149     with connection.cursor() as cursor:
150
151     print(ID)
152     ID = str(ID)
153     N_ID = ""
154     for x in ID :
155
156     print(N_ID)
157     N_ID = int(N_ID)
158
159     """INSERT TO BDD"""
160
161     xy = 0
162     try :
163         while xy < nb_product :
164             with connection.cursor() as cursor:
165                 sql = "INSERT INTO `PRODUIT` (`NOM`,`PRODUIT_URL`,`NUTRISCORE`,`CATEGORIE_ID`) VALUES (%s, %s, %s, %s)"
166                 cursor.execute(sql, (name[xy], url[xy], ns_number[xy], N_ID))
167
168                 connection.commit()
169                 xy = xy + 1
170                 print(str(xy)+"produits récupérés sur " + str(nb_product))
171     except :
172         pass
173
174     def select_and_substitut():
175
176     """MAIN LOOP"""
177
178     continu = 0
179     while continu == 0 :
180
```


Utilisation des opérations de jointures en MYSQL

```
69 """PROGRAM FUNCTIONS"""
70 def Add_product():
178 def select_and_substitut():
179     with connection.cursor() as cursor:
180
181         sql = "SELECT PRODUIT.NOM, PRODUIT.PRODUIT_ID FROM `PRODUIT` INNER JOIN `CATEGORIES` ON PRODUIT.CATEGORIE_ID = CATEGORIES.ID WHERE CATEGORIES.NOM = %s AND NUTRIScore >= 3"
182         cursor.execute(sql, (name_categories))
183         result = cursor.fetchall()
184         print(result)
185
186     choice_produit = input("\n Indiquer le numéro du produit que vous souhaitez remplacer : ")
187
188     with connection.cursor() as cursor:
189
190         sql = "SELECT PRODUIT.NOM, PRODUIT.PRODUIT_ID FROM `PRODUIT` INNER JOIN `CATEGORIES` ON PRODUIT.CATEGORIE_ID = CATEGORIES.ID WHERE CATEGORIES.NOM = %s AND NUTRIScore < 3"
191         cursor.execute(sql, (name_categories))
192         result = cursor.fetchall()
193         print(result)
194
```

Utilisation de la Fonction SELECT en MYSQL

```
250 """MAIN LOOP"""
251
252 continu = 0
253 while continu == 0 :
254     try :
255         mode_terminal = input("1 - Quel aliment souhaitez-vous remplacer ? \n2 - Retrouver mes aliments substitués. \n")
256         mode_terminal = int(mode_terminal)
257
258         """mode remplacement"""
259         while mode_terminal == 1 :
260             """mode BDD"""
261
262             while mode_terminal == 2 :
263                 print("Voici vos produits : \n ")
264                 with connection.cursor() as cursor:
265                     sql = "SELECT * FROM `SUBSTITUT`"
266                     cursor.execute(sql, ())
267                     mes_produits = cursor.fetchall()
268                     print(mes_produits)
269
270             mode_terminal = 0
271
272         """Mode sortie"""
273
274         if mode_terminal == 3 :
275
276         elif mode_terminal >= 3 :
277             print("{} n'est pas dans les numéros proposés\n".format(mode_terminal))
278
279     except ValueError :
280         pass
```