**TM1py**

TM1py est une bibliothèque Python développée pour interagir avec IBM Planning Analytics.

Son objectif : automatiser les tâches liées à la modélisation, à l'administration et à l'intégration de données dans IBM Planning Analytics.

TM1py permet de lier IBM Planning Analytics(tm1) et Python.

TM1py permet de :

* gérer les dimensions, les cubes et les sous-ensembles ;
* exécuter des requêtes MDX pour récupérer des données et des analyses ;
* créer et exécuter des processus d’écriture et de calcul ;
* automatiser les tâches ;
* intégrer des données de différentes sources dans IBM Plannings Analytics.

*used\_dimensions=set()* : initialisée comme un ensemble vide.

*used\_dimensions.update(cube.dimensions)* : mettre à jour l’ensemble *used\_dimensions* avec les dimensions utilisées par ces cubes

with TM1Service(address=*'decilim19paca01'*, port=30049, user=*'admin'*, password=*'apple'*, ssl=False) as tm1:

Pour chaque programme on a besoin de cette ligne pour se connecter au serveur.

**Test TM1py**

output\_file = *'cubes.csv'*: création du fichier csv

print(*f"Le fichier {output\_file} a été créé avec succès."*) : le f permet de dire que la chaine est une chaine formatée

 with open(output\_file, *'w'*, newline=*''*) as file:

writer = csv.writer(file, delimiter=*';'*)

writer.writerow([*'Cube'*, *'Règle'*]) : Création des colonneswrite

le délimiteur permet d’avoir des colonnes distinctes.

all\_dimensions = tm1.dimensions.get\_all\_names()

Récupération des données soit les dimensiosn ici

for dimension in all\_dimensions:

writer.writerow([*'Dimension'*, dimension])

Ecriture des données récupérer dans le fichier csv

Pour écrire dans une boucle (for ou while) il suffit de faire une indentation et pour en sortir il suffit juste de sortir de cette indentation.

Python est sensible aux indentations

**Pour les processus :**

process\_name = *'IMP\_COUTS2’* mettre le nom du processus

p\_ascii.add\_variable(*'PRODUIT'*, *'String'*) pour créer une variable

p\_ascii.add\_parameter(name=*'pVersion'*, prompt=*'test'*, value=*''*) pour créer un paramètre

p\_ascii.prolog\_procedure = *""" """*

p\_ascii.metadata\_procedure

p\_ascii.data\_procedure

p\_ascii.epilog\_procedure

pour créer le prologue, métadonnées , données et épilogue

cube\_name = str(input(*"Entrez le nom du cube : "*)) demande à l’utilisateur d’entrer une information de type string (chaine de caractère)

b= recup(cube\_name) appel de la fonction récup et mettre son résultat dans une variable nommé b

d= a+b+c concaténation de plusieurs variables

tm1.processes.create(p\_ascii) Création du processus

**Excel**

from openpyxl import Workbook importer la bibliothèque afin de créer des fichier excel

sheet1 = workbook.active Création de l’onglet principal

sheet1.title = *"MDX Info"* nommage de l’onglet

sheet3.append([*'Cube'*, *'Règle'*])ajout des titres des colonnes

sheet3.append([cube\_name, str(rules) if rules is not None else *""*])

permet d’ajouter toute les info dans l’onglet 3

workbook.save(fichier\_xlsx)

enregistrement du fichier