

CS500222

## Validation des modèles avec PyRevit et PowerBI

KICHENIN Léo  
Cardinal Edifice

### Objectifs d'apprentissage

- Définir un processus qualité
- Naviguer dans l'API de Revit
- Configurer un onglet PyRevit
- Configurer un tableau de bord PowerBI

### Description

La présentation a permis d'exposer la réflexion qui a mené au processus qualité utilisé sur un projet en cours. Ce cours a pour but d'expliquer plus en détail les concepts et les outils associés, pour pouvoir les exploiter dans différents cas de figure.

### Intervenant(s)

KICHENIN Léo

BIM Manager chez Cardinal Edifice, j'ai eu l'occasion de travailler sur plusieurs projets à différentes phases. Cette expérience m'a permis d'appréhender le fonctionnement des projets de la conception à l'exploitation-maintenance. Mon objectif est de mettre les nouvelles technologies au service des projets et permettre de les réaliser de façon performante.

## L'API de Revit

L'API, Application Programming Interface ou Interface de Programmation d'application, est une interface qui permet de communiquer avec le logiciel. En respectant un certain nombre de principes, il est possible de donner des instructions au logiciel pour réaliser des actions.

Les actions réalisables de cette façon sont listées dans la documentation.

### La structuration de l'API de Revit

Pour rendre possible son utilisation par des développeurs, l'API de Revit est structurée autour du principe de classes.

Chaque composant du logiciel est associé à une classe qui se décompose en propriétés et en méthodes.

Les propriétés sont les caractéristiques intrinsèques de l'élément et donnent accès à des informations ou des objets associés à l'élément.

Ex : Le matériau d'un mur.

Les méthodes sont les actions réalisables à partir de l'élément. C'est par cet intermédiaire que les fonctions du logiciel sont utilisables. Elles sont utilisées pour construire des séries d'instructions qui constitueront à terme des plugins ou des surcouches logicielles à part entière.

Ex : La méthode Flip d'un mur qui permet de l'inverser.

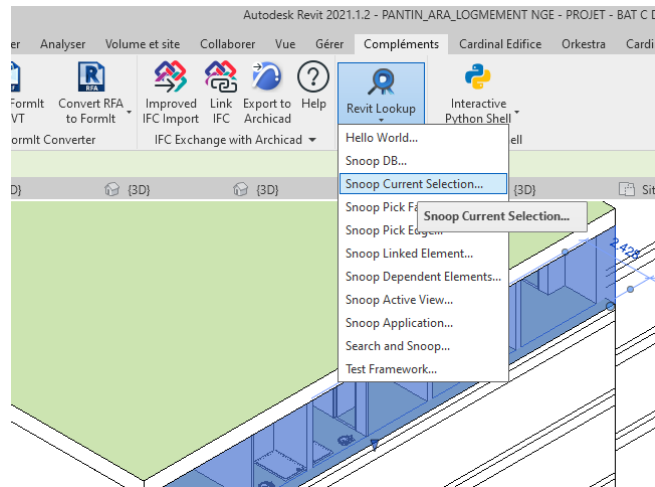
### Comprendre l'API de Revit

Une fois les principes de programmation assimilés, la seconde phase est de parcourir la documentation de l'API : <https://www.revitapidocs.com/2021.1/>

Cette première phase permet de s'imprégner de l'interface et du fonctionnement général. Pour débiter, le plus efficace sera de cibler des classes « concrètes » telles que la classe « Wall » ou « View ».

Pour approfondir cette prise en main, la deuxième phase passe par l'utilisation de Revit Lookup. Le plugin est téléchargeable ici : <https://github.com/jeremytammik/RevitLookup>

Ce plugin permet de sélectionner n'importe quel élément et de l'inspecter du point de vue de l'API. Pour cela, sélectionner l'élément à inspecter puis dans l'onglet « Compléments » → « Revit Lookup », choisir « Snoop Current selection ».



Cela donne accès à la fenêtre ci-dessous :

Snoop Objects

Wall

< Générique - Ext.350 mm 1107747 >

Field	Value
--- Element ---	
--- Properties ---	
AssemblyInstanceId	< null >
BoundaryId	< BoundaryId.N7h361010 >
Category	< Category Murs >
CreatedPhaseId	< Nouvelle construction 110001 >
DemolishedPhaseId	< null >
DesignOption	< null >
Document	< Document >
Geometry	< Geometry.Element >
GroupId	< null >
Id	1107747
IsTransient	False
IsValidObject	True
LevelId	< N7h 361010 >
Location	< LocationCurve >
Name	Générique - Ext.350 mm
OwnerViewId	< null >
Parameters	< ParameterSet >
ParametersMap	< ParameterMap >
Pinned	False
UniqueId	d07ae8e-d936-4251-a30e-90e218315e74-0010e723
VersionGuid	d07ae8e-d936-4251-a30e-90e218315e74
ViewSpecific	False
WorksetId	< WorksetId >
--- Methods ---	
ArePhasesModifiable	True
CanBeLocked	True
CanHaveAnalyticalModel	True
CanHaveTypeAssigned	True
GetAnalyticalModel	< null >
GetAnalyticalModelId	< null >
GetDependentElements	< List<1> >
GetEntitySchemaGuids	< List<1> >
GetExternalFileReference	Une exception a été levée par la cible d'un appel.
GetExternalResourceReferences	Une exception a été levée par la cible d'un appel.
GetMonitoredLinkElementIds	< List<1> >
GetMonitoredLocalElementIds	< List<1> >
GetOrderedParameters	< List<1> >
GetPhaseStatus	< phases statuses >
GetSubelements	< List<1> >

Propriétés

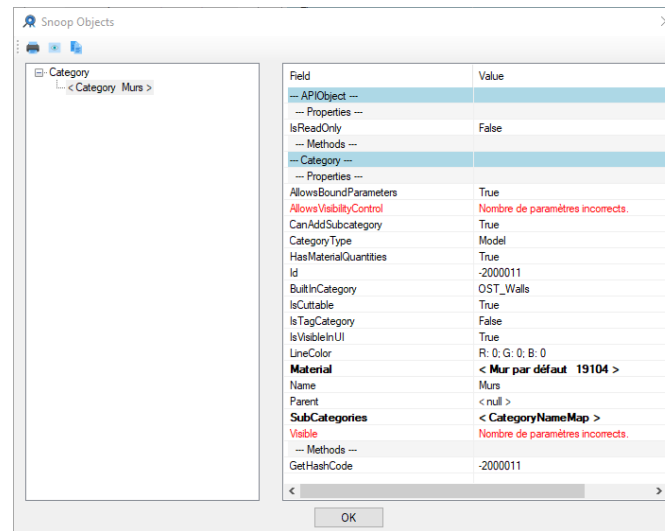
Méthodes

OK

Page 3

On y retrouve les propriétés et les méthodes de l'élément ainsi que leur valeur. En cliquant sur les valeurs on peut accéder soit aux valeurs associées soit aux objets associés.

Ainsi en cliquant sur la valeur associée à « Category », on accède au descriptif de la classe « Category » avec ses propres propriétés et méthodes :



Cette méthodologie permet d'accomplir deux choses :

- atteindre la valeur que l'on cherche à obtenir
- donner des repères sur les relations entre les différents objets et donc entre les classes qui composent l'API de Revit.

La combinaison de ces deux leviers, associée à des recherches sur les forums vous permettra d'appréhender l'architecture de l'API et les relations qui la compose.

## Interagir avec l'API

Pour pouvoir accéder aux attributs des éléments et appeler les méthodes associées à ceux-ci, il est nécessaire d'utiliser un langage de programmation. L'API de Revit est construite en langage C# et plusieurs options s'offrent à vous pour interagir avec celle-ci.

Ci-dessous un comparatif des langages selon trois critères :

- Performance → Temps d'exécution. Il varie selon la nature des langages (compilé vs interprété).
- Ressources → Evaluations des ressources documentaires disponibles. Il s'agit principalement de la documentation, des tutoriels et des forums associés au langage de programmation.
- Accessibilité → Evaluation de la facilité d'utilisation / d'appropriation pour une personne n'étant pas développeuse. Ce critère se base sur la syntaxe et la nature du langage.

Langage	Performance	Ressources	Accessibilité	Commentaire
C#	+++	+++	--	Langage le plus répandu pour développer des applications revit.
Python	+	++	+	Langage le plus répandu de manière générale.
VB.Net	++	-	-	Langage associé aux macros.
Ruby	+	---	---	Langage qui se rapproche de Python mais moins répandu avec beaucoup moins de ressources liées à Revit.
Programmation visuelle	--	+++	+++	Le plus accessible. Permet également d'utiliser du C# et du Python dans des nœuds spécifiques.

## Environnements de développements

Pour pouvoir construire et tester des fonctions à l'aide des langages ci-dessus, un environnement de développement est nécessaire. Vous trouverez ci-dessous les principaux utilisés pour chaque langage.

Langage	Environnement
C#	Visual Studio, Macros
Python	RevitPythonShell, Dynamo, PyRevit
VB.Net	Macros
Ruby	Macros, RevitRubyShell
Programmation visuelle	Dynamo, Grasshopper

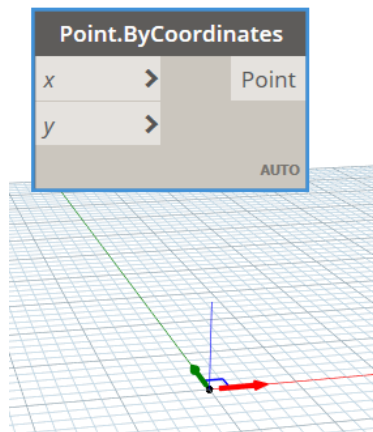
Ces environnements présentent des caractéristiques qui leurs sont propres et qui nécessitent un travail de configuration avant d'être utilisés. L'objet de ce cours est de décrire comment configurer et mettre en place un environnement de développement pour le langage Python.

## Développer en Python

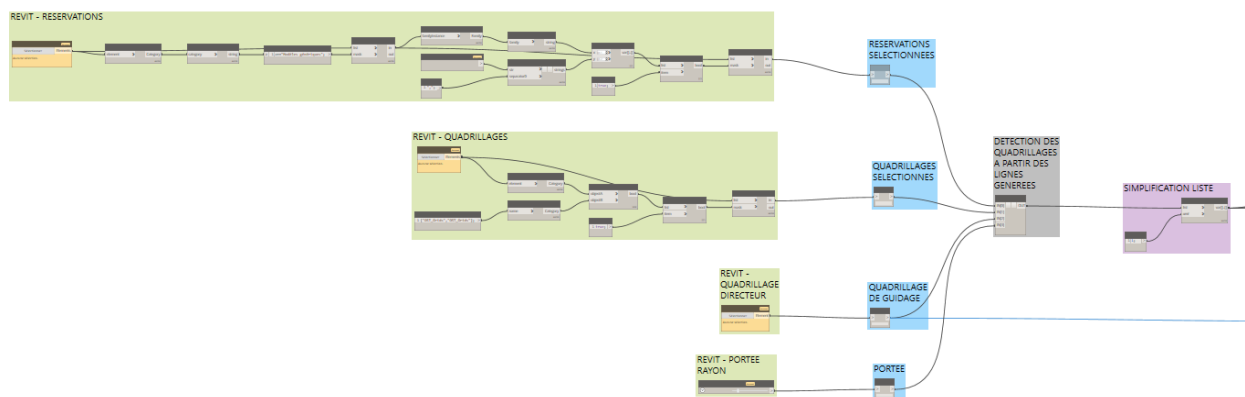
Plusieurs options s'offrent à vous pour créer des commandes en Python. Les parties suivantes détaillent celles qui ont été retenues pour ce projet.

### Dynamo

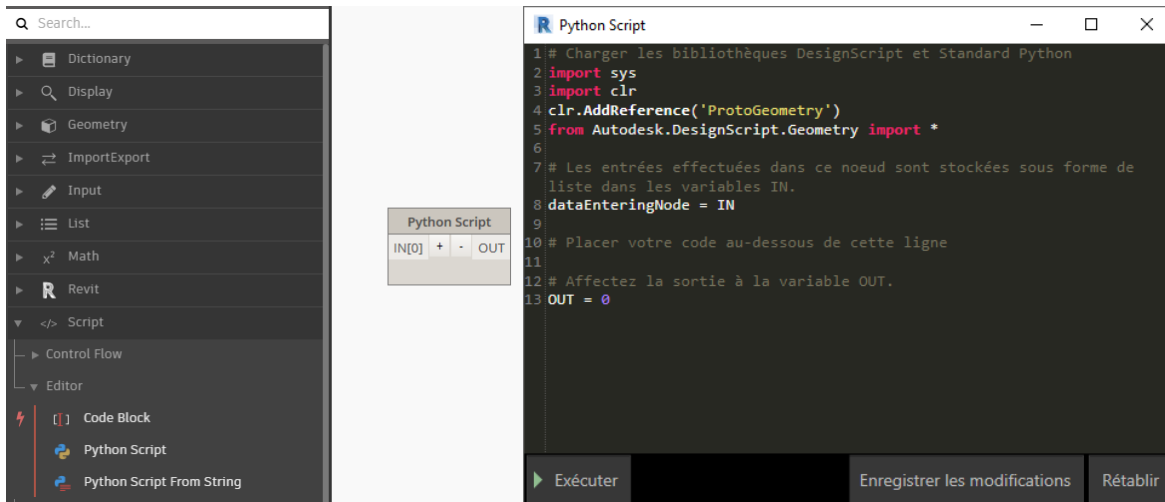
Dynamo est un module de programmation visuelle permettant d'utiliser des fonctions sous formes de blocs, plus communément appelés nœuds.



Ces fonctions sont composées d'entrées (à gauche) et donnent un résultat (à droite). L'assemblage de ces nœuds constitue ce qu'on appelle les définitions Dynamo.

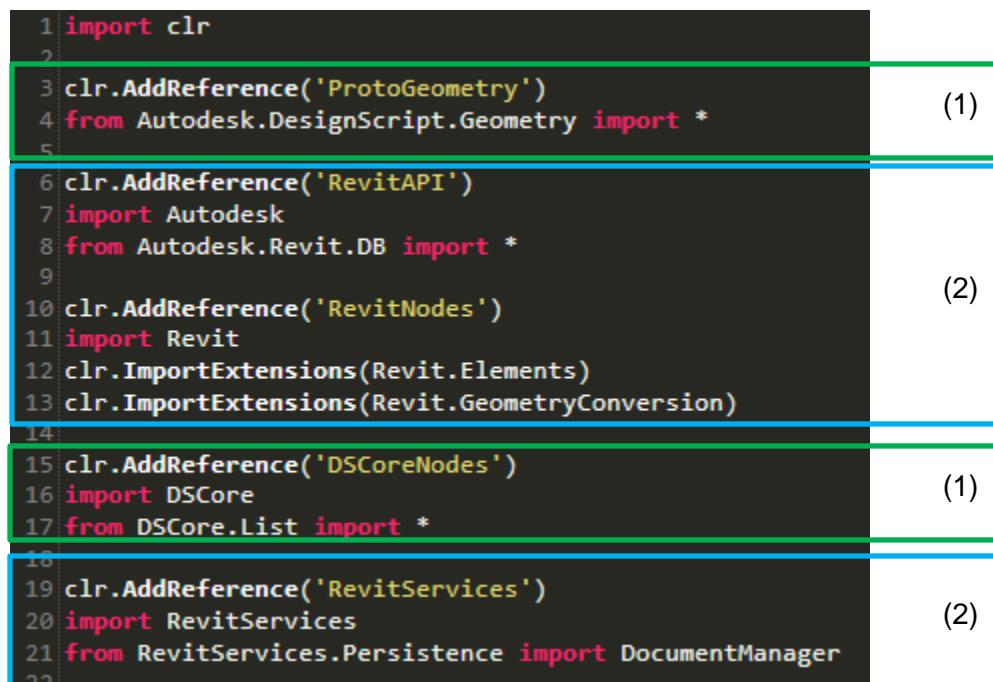


Parmi les nœuds disponibles, il existe des nœuds Python.



Ils permettent d'exécuter des portions de codes écrits en python au sein de dynamo. Le bloc permet d'appeler à la fois les fonctions de dynamo, par exemple liées aux géométries, mais aussi les fonctions liées à Revit.

Ces appels peuvent se faire en paramétrant les références aux librairies de fonctions des outils. Ce sont les premières lignes du nœud.



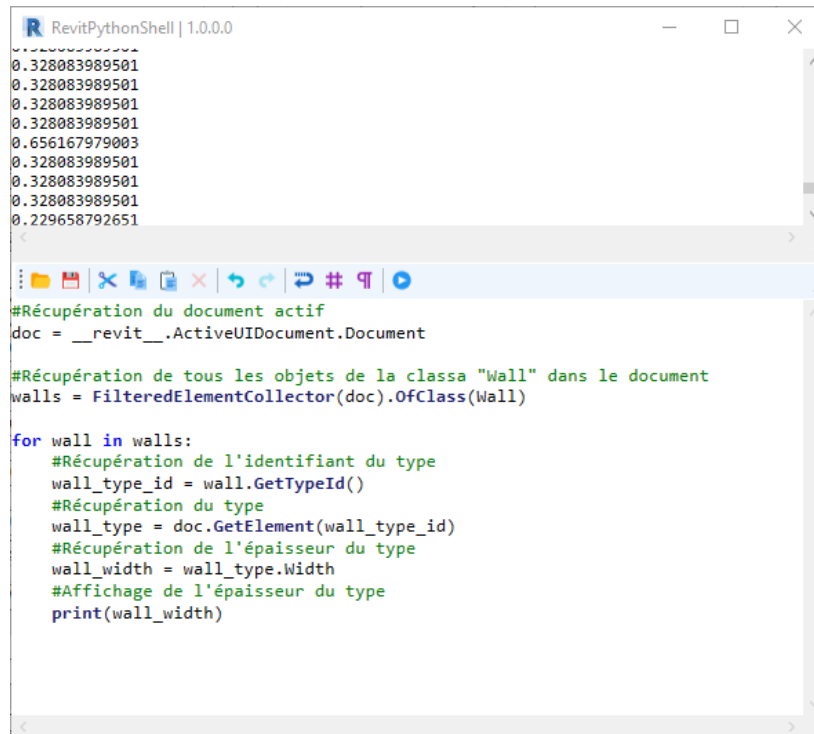
(1) : Librairies liées à Dynamo

(2) : Librairies liées à Revit

## RevitPythonShell

RevitPythonShell est un plugin Revit. Il donne accès à une console qui permettra d'exécuter des scripts Python. Ce module exploite la nature interprétée du langage Python qui exécute du code sans compilation et d'avoir un retour en direct : soit un message d'erreur, soit le résultat de la commande.

Ainsi, il est possible d'écrire directement du Python dans une console et de lancer l'exécution comme on le ferait pour une définition Dynamo. Cela joue en faveur de l'accessibilité car il n'est pas nécessaire de compiler, ce qui impliquerait de fermer/lancer Revit à chaque exécution. Ce principe est d'autant plus important pour des personnes non développeuses, pour pouvoir tatonner et savoir où s'orienter dans l'API. L'obtention de résultats intermédiaires et de messages d'erreurs à chaque étape est cruciale pour apprivoiser à la fois l'API de Revit mais aussi le langage Python en lui-même.



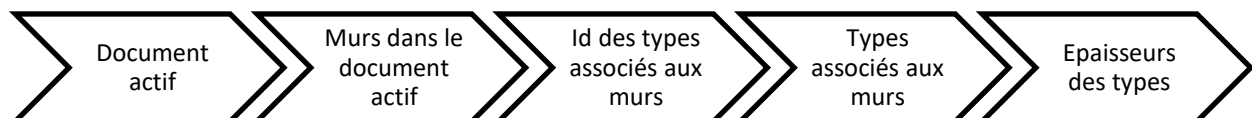
```
RevitPythonShell | 1.0.0.0
0.328083989501
0.328083989501
0.328083989501
0.328083989501
0.656167979003
0.328083989501
0.328083989501
0.328083989501
0.229658792651

#Récupération du document actif
doc = __revit__.ActiveUIDocument.Document

#Récupération de tous les objets de la classe "Wall" dans le document
walls = FilteredElementCollector(doc).OfClass(Wall)

for wall in walls:
    #Récupération de l'identifiant du type
    wall_type_id = wall.GetTypeId()
    #Récupération du type
    wall_type = doc.GetElement(wall_type_id)
    #Récupération de l'épaisseur du type
    wall_width = wall_type.Width
    #Affichage de l'épaisseur du type
    print(wall_width)
```

Dans l'exemple ci-dessus, on affiche l'épaisseur de tous les murs présents dans le document. Cette opération passe par plusieurs étapes décrites en commentaires dans le code. On aperçoit que l'accès à l'information n'est pas direct. Il faut passer par plusieurs objets intermédiaires.

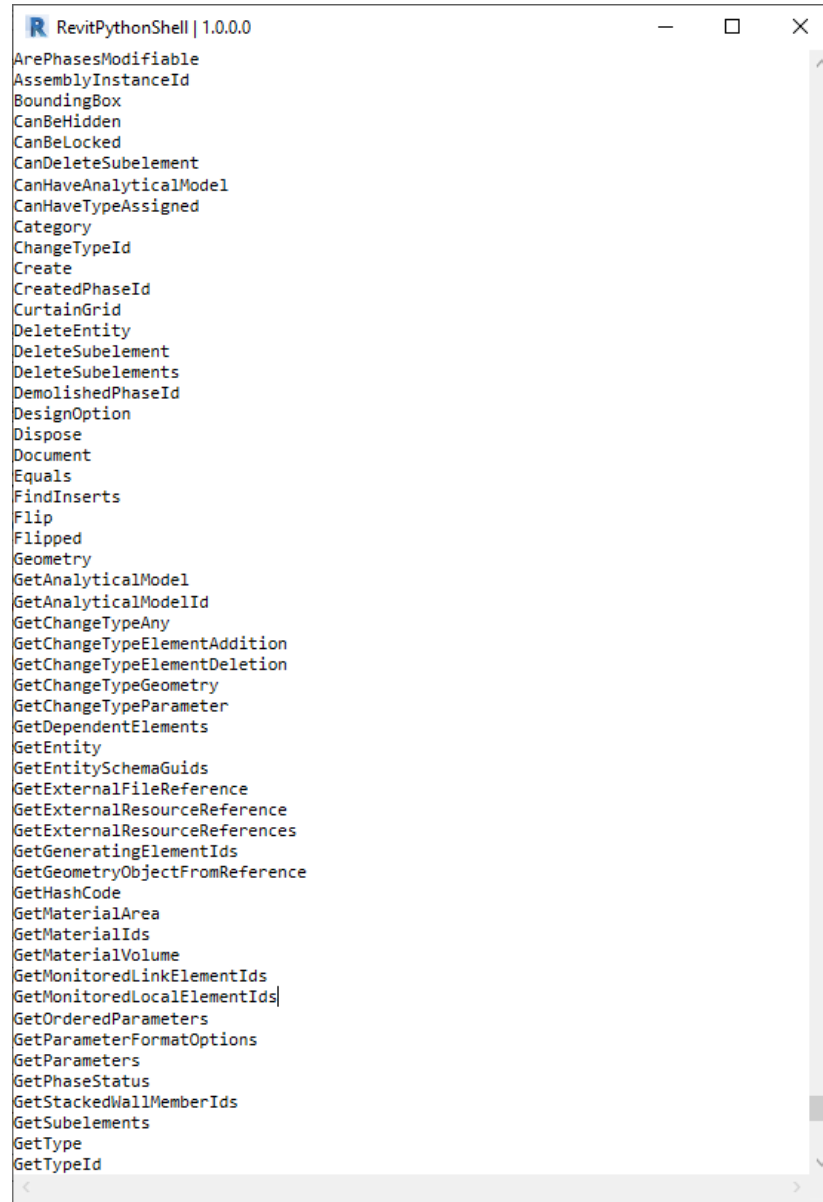


C'est dans ce cheminement que l'utilisation de la documentation et de Revit Lookup prennent tout leur sens.



En parallèle, RevitPythonShell vous permettra de vérifier vos résultats intermédiaires et de vous orienter. Pour cela, deux principes :

- principe n°1 → afficher vos valeurs via la commande print()
  - principe n°2 → utiliser la fonction dir() sur les objets que vous manipulez. Ce deuxième principe vous donnera accès aux actions qui sont réalisables à partir de l'objet que vous manipuler. C'est l'équivalent de la fonction « Snoop » de revit Lookup.
- Par exemple, dir(wall) vous donnera :



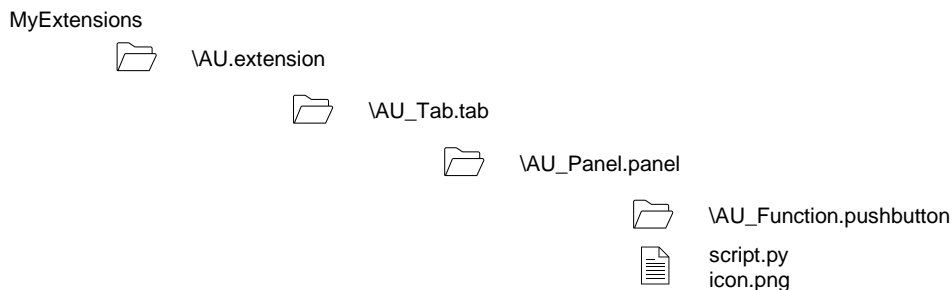
```
RevitPythonShell | 1.0.0.0
ArePhasesModifiable
AssemblyInstanceId
BoundingBox
CanBeHidden
CanBeLocked
CanDeleteSubelement
CanHaveAnalyticalModel
CanHaveTypeAssigned
Category
ChangeTypeId
Create
CreatedPhaseId
CurtainGrid
DeleteEntity
DeleteSubelement
DeleteSubelements
DemolishedPhaseId
DesignOption
Dispose
Document
Equals
FindInserts
Flip
Flipped
Geometry
GetAnalyticalModel
GetAnalyticalModelId
GetChangeTypeAny
GetChangeTypeElementAddition
GetChangeTypeElementDeletion
GetChangeTypeGeometry
GetChangeTypeParameter
GetDependentElements
GetEntity
GetEntitySchemaGuids
GetExternalFileReference
GetExternalResourceReference
GetExternalResourceReferences
GetGeneratingElementIds
GetGeometryObjectFromReference
GetHashCode
GetMaterialArea
GetMaterialIds
GetMaterialVolume
GetMonitoredLinkElementIds
GetMonitoredLocalElementIds
GetOrderedParameters
GetParameterFormatOptions
GetParameters
GetPhaseStatus
GetStackedWallMemberIds
GetSubelements
GetType
GetTypeId
```

## PyRevit

PyRevit permet également d'exécuter des scripts Python. Ses fonctionnalités vont plus loin permettent de déployer les scripts sous forme d'icône dans l'interface de Revit.

Le plugin est accessible ici : <https://github.com/eirannejad/pyRevit/releases>

Une fois le plugin téléchargé, il suffit de configurer une arborescence selon la structure suivante :

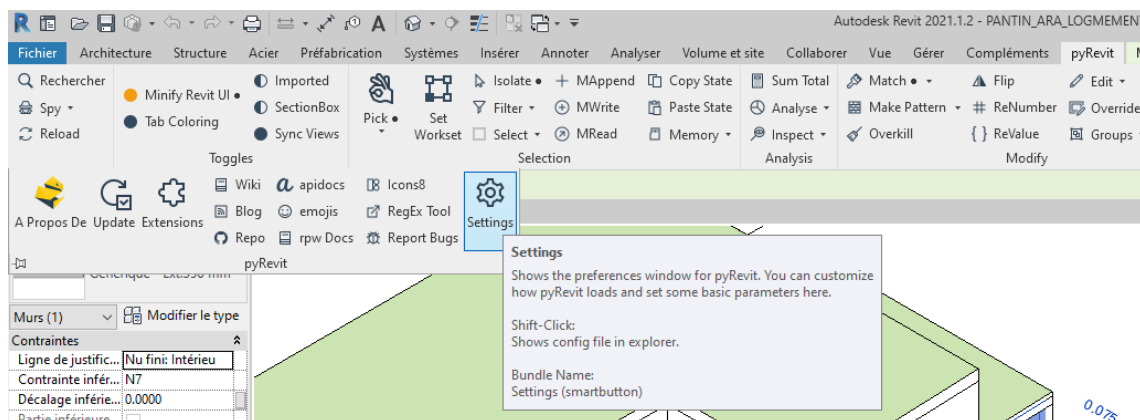


Le dossier final est le suivant :

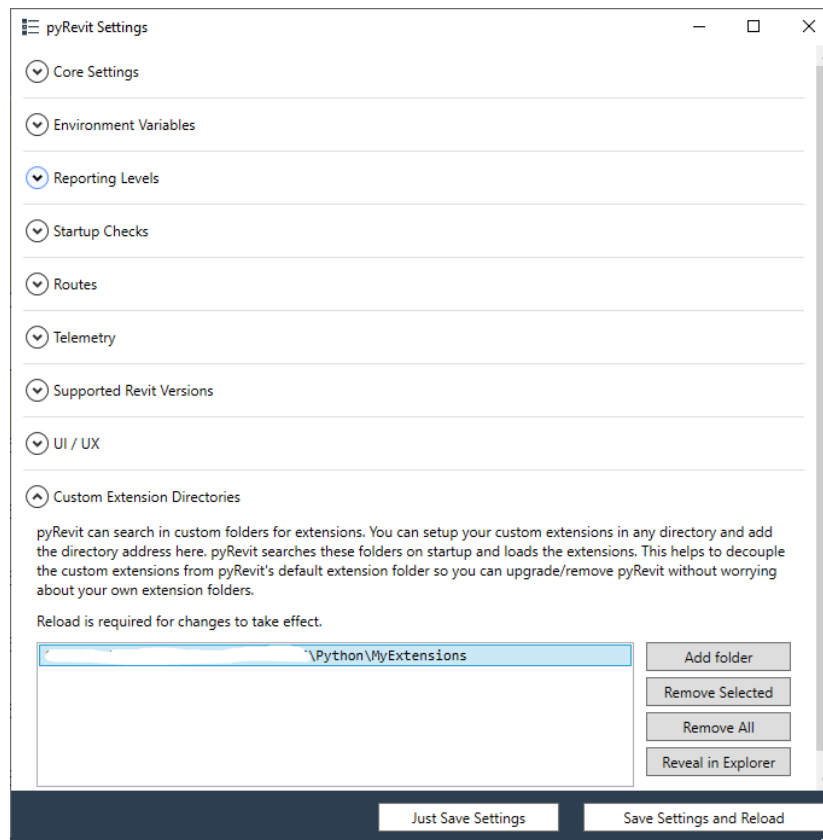
Python > MyExtensions > AU.extension > AU_Tab.tab > AU_Panel.panel > AU_Function.pushbutton				
Nom	Statut	Modifié le	Type	Taille
script.py		06/08/2020 17:44	Fichier PY	2 Ko
icon.png		09/04/2020 15:25	Fichier PNG	2 Ko

Une fois cette arborescence configurée, il ne reste plus qu'à pointer vers le dossier MyExtensions.  
Pour cela :

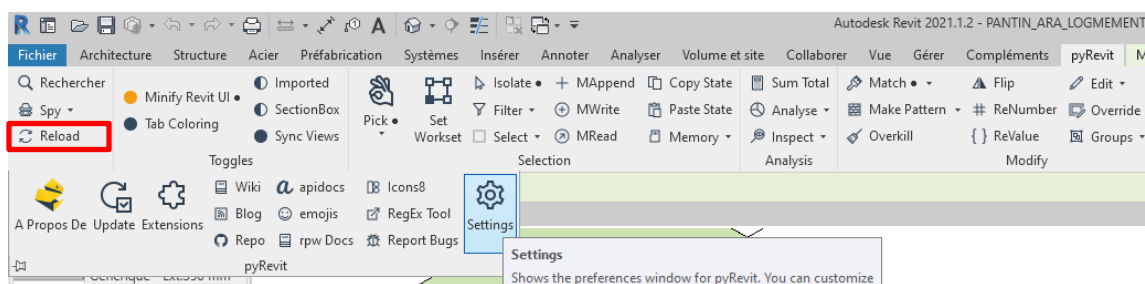
1) Aller dans pyRevit → Settings



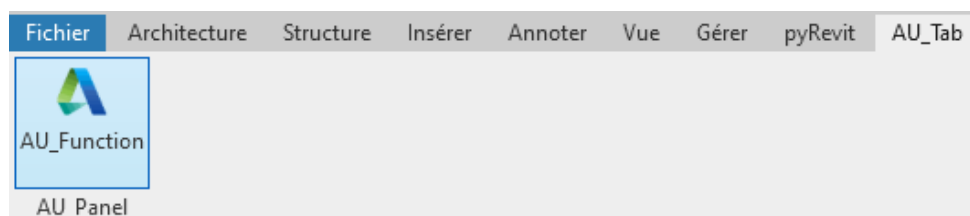
2) Dans le menu, choisir « Add folder » puis sélectionner le dossier MyExtensions.



3) Cliquer sur Reload pour charger l'extension et le contenu de ses dossiers.



4) Vérifier la présence de la fonction.



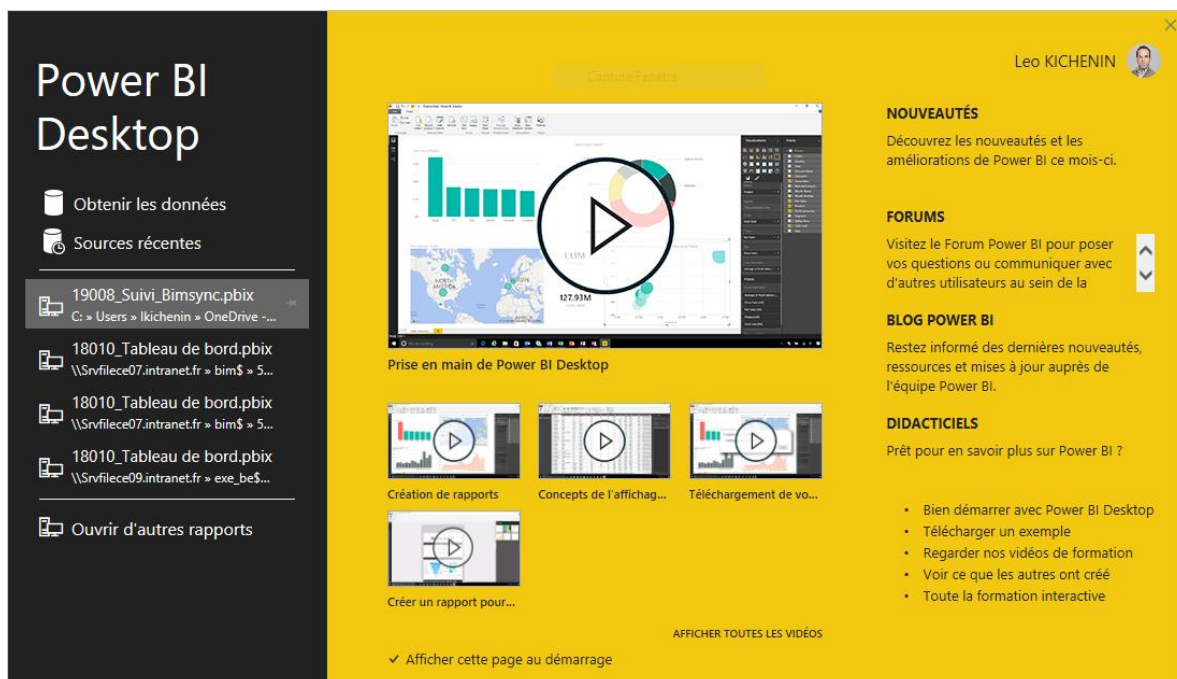
## Exploiter des données

Durant la présentation, nous avons vu comment cibler des données d'intérêt et les extraire. Cette partie permet d'approfondir la mise en forme des données.

La mise en forme des données passe par PowerBI, un logiciel développé par Microsoft. Ce logiciel est décliné en application de bureau et en application Web. Pour réaliser les étapes suivantes, nous allons utiliser l'application de bureau.

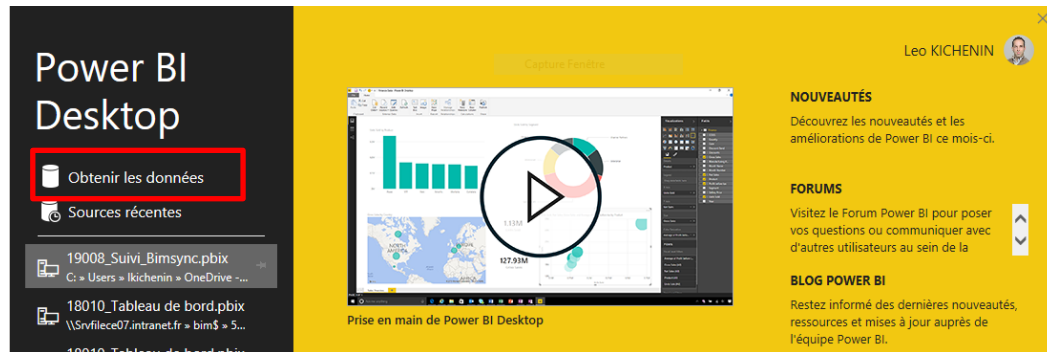
L'application en question est Power BI Desktop, téléchargeable ici : <https://powerbi.microsoft.com/fr-fr/desktop/>

Au lancement de l'application, la fenêtre ci-dessous apparaît :



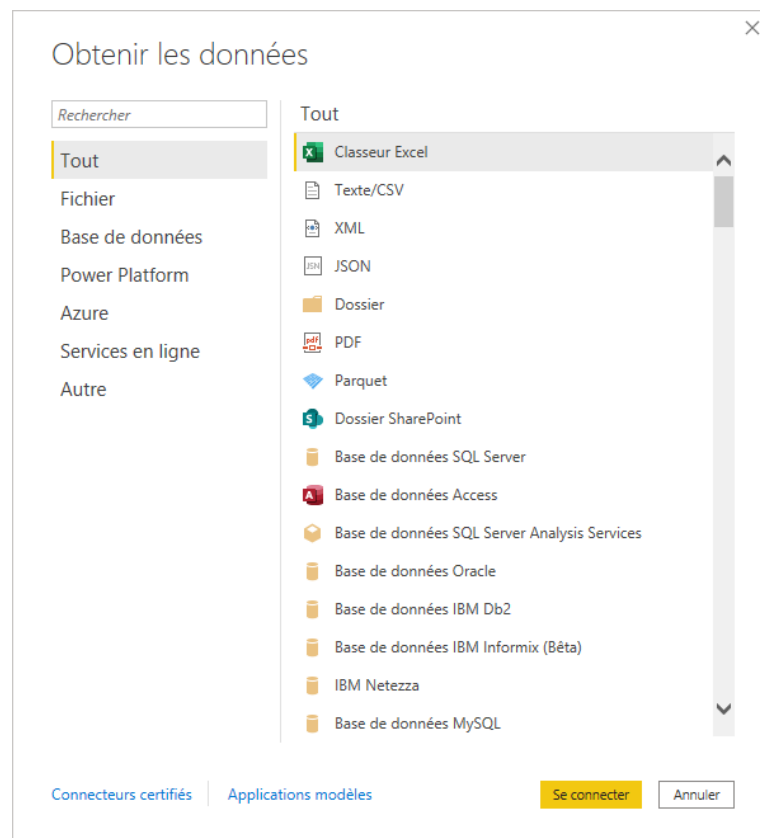
La création d'un tableau de bord passe par les étapes suivantes :

## 1) Obtenir les données



L'obtention des données est le point de départ et permettra de cibler les données sur lesquelles nous allons travailler.

Une deuxième fenêtre s'ouvre dans laquelle vous pourrez spécifier la sources de données.



Il existe une grande quantité de formats disponibles ainsi que des connecteurs avec des plateformes web de données autorisant l'échange de données en direct.

Les formats les plus classiques sont les bases de données Excel, csv, xml et JSON. Dans cet exemple, nous allons manipuler un fichier csv. Il s'agit de données séparées par des virgules.

TCE\_2021.csv - Bloc-notes

Fichier Edition Format Affichage Aide

WORKSET,CATEGORY, BIM\_LOT,CAR\_Niveau,ID REVIT,REVISION,DATE,DISCIPLINE,EXPORT DATE,ZONE,FILE NAME

"Vue ""Vue 3D: {3D},Caméras,,206180,B\_21,15/01/2021,GOE,13/01/2021 08:41,AME,VAU\_AME\_CARDINAL\_EXE\_TN\_GOE\_MAQ\_R18\_B-21"

02\_LOT2\_GROS-À'UVRE,Poteaux porteurs,EXE\_GOE,AME\_RDC\_NVP,226869,B\_21,15/01/2021,GOE,13/01/2021 08:41,AME,VAU\_AME\_CARDINAL\_EXE\_TN\_GOE\_MAQ\_R18\_B-21

02\_LOT2\_GROS-À'UVRE,Poteaux porteurs,EXE\_GOE,AME\_RDC\_NVP,226876,B\_21,15/01/2021,GOE,13/01/2021 08:41,AME,VAU\_AME\_CARDINAL\_EXE\_TN\_GOE\_MAQ\_R18\_B-21

02\_LOT2\_GROS-À'UVRE,Poteaux porteurs,EXE\_GOE,AME\_RDC\_NVP,226883,B\_21,15/01/2021,GOE,13/01/2021 08:41,AME,VAU\_AME\_CARDINAL\_EXE\_TN\_GOE\_MAQ\_R18\_B-21

02\_LOT2\_GROS-À'UVRE,Poteaux porteurs,EXE\_GOE,AME\_RDC\_NVP,226895,B\_21,15/01/2021,GOE,13/01/2021 08:41,AME,VAU\_AME\_CARDINAL\_EXE\_TN\_GOE\_MAQ\_R18\_B-21

02\_LOT2\_GROS-À'UVRE,Poteaux porteurs,EXE\_GOE,AME\_RDC\_NVP,226901,B\_21,15/01/2021,GOE,13/01/2021 08:41,AME,VAU\_AME\_CARDINAL\_EXE\_TN\_GOE\_MAQ\_R18\_B-21

02\_LOT2\_GROS-À'UVRE,Poteaux porteurs,EXE\_GOE,AME\_RDC\_NVP,226907,B\_21,15/01/2021,GOE,13/01/2021 08:41,AME,VAU\_AME\_CARDINAL\_EXE\_TN\_GOE\_MAQ\_R18\_B-21

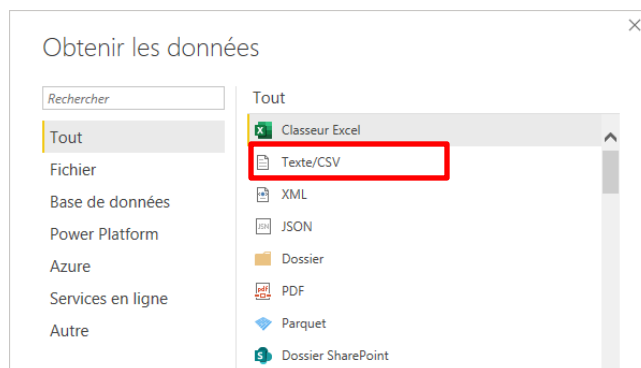
02\_LOT2\_GROS-À'UVRE,Poteaux porteurs,EXE\_GOE,AME\_RDC\_NVP,226913,B\_21,15/01/2021,GOE,13/01/2021 08:41,AME,VAU\_AME\_CARDINAL\_EXE\_TN\_GOE\_MAQ\_R18\_B-21

02\_LOT2\_GROS-À'UVRE,Poteaux porteurs,EXE\_GOE,AME\_RDC\_NVP,226919,B\_21,15/01/2021,GOE,13/01/2021 08:41,AME,VAU\_AME\_CARDINAL\_EXE\_TN\_GOE\_MAQ\_R18\_B-21

02\_LOT2\_GROS-À'UVRE,Poteaux porteurs,EXE\_GOE,AME\_RDC\_NVP,226925,B\_21,15/01/2021,GOE,13/01/2021 08:41,AME,VAU\_AME\_CARDINAL\_EXE\_TN\_GOE\_MAQ\_R18\_B-21

02\_LOT2\_GROS-À'UVRE,Poteaux porteurs,EXE\_GOE,AME\_RDC\_NVP,226931,B\_21,15/01/2021,GOE,13/01/2021 08:41,AME,VAU\_AME\_CARDINAL\_EXE\_TN\_GOE\_MAQ\_R18\_B-21

2) Choisir la source de données « Texte/CSV » puis cibler le fichier de données.



L'aperçu des données s'affiche.

TCE\_2021.csv

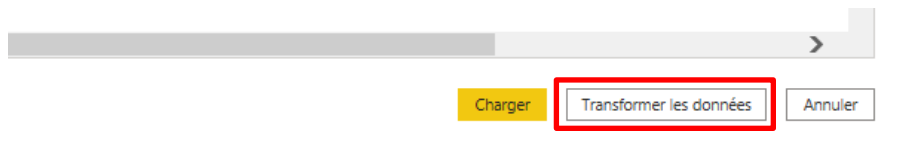
Origine du fichier: 1252: Europe de l'Ouest (Windows) | Délimiteur: Virgule | Détection du type de données: Selon les 200 premières lignes

Column1	Column2	Column3	Column4	Column5	Column6	Column7	Column8	Column9	Column10
WORKSET	CATEGORY	BIM_LOT	CAR_Niveau	ID REVIT	REVISION	DATE	DISCIPLINE	EXPORT DATE	ZONE
Vue "Vue 3D: {3D},Caméras,,206180,B_21,15/01/2021,...									
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226869	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226876	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226883	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226895	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226901	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226907	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226913	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226919	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226925	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226931	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226937	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226949	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226955	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226961	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226967	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	AME_RDC_NVP	226973	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	TRS_SSI_NVP	226979	B_21	15/01/2021	GOE	13/01/2021 08:41	AME
02_LOT2_GROS-À'UVRE	Poteaux porteurs	EXE_GOE	TRS_SSI_NVP	226985	B_21	15/01/2021	GOE	13/01/2021 08:41	AME

Les données dans l'aperçu ont été tronquées en raison de limites de taille.

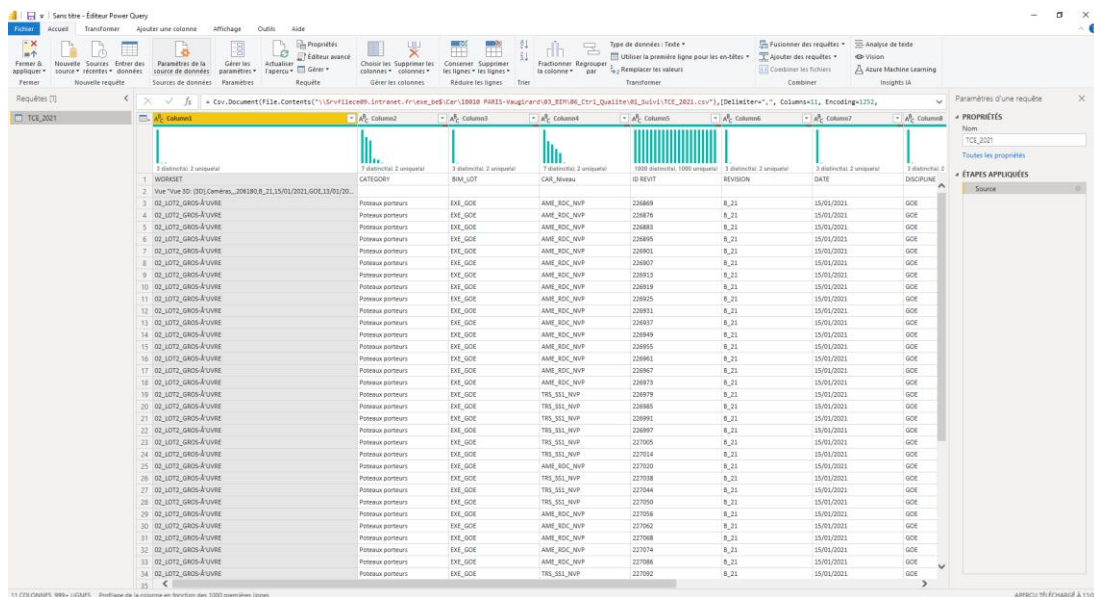
Extraire une table avec des exemples | Charger | Transformer les données | Annuler

## 3) Cliquer sur Transformer les données



Ici, nous allons pouvoir travailler la base de données brutes pour pouvoir obtenir le rendu souhaité dans le tableau de bord.

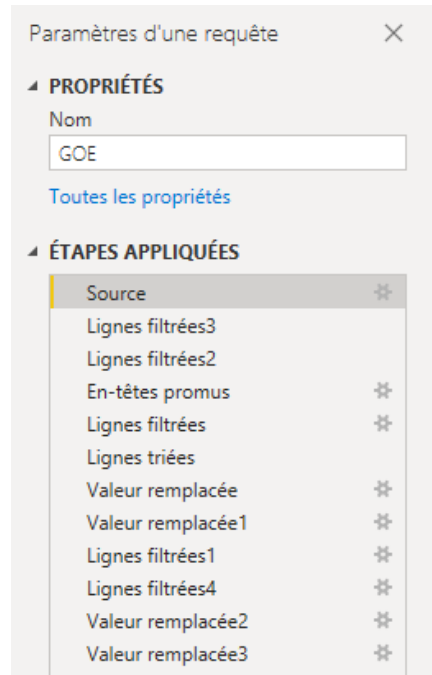
## 4) Mettre en forme la base de données via le module PowerQuery. Cette interface va permettre de réaliser des actions sur la base de données, plus communément appelées des requêtes.



Parmi les requêtes possibles vous retrouverez les suivantes :

- Filtres
- Tris
- Remplacements de valeurs
- Ajout de colonnes selon des formules
- Suppression de colonnes
- Regroupement de valeurs
- ...

Les requêtes sont similaires aux fonctions Excel classiques à la différence près que chaque requête est historisé dans le menu « Étapes ».

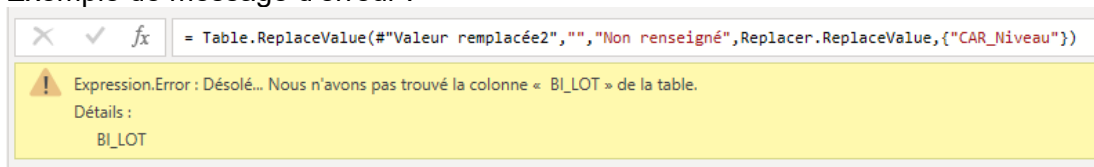


Chaque étape représentera l'état de la base de donnée à un instant t. De cette façon, il est possible de naviguer dans les différents états de la base de données jusqu'à l'étape finale. La base de donnée à l'étape finale est celle qui servira de support à la création des tableaux de bord.



La modification des requêtes intermédiaires impacte les requêtes suivantes. Lors de ces modifications, il est impératif de vérifier que l'état final de la base de données est toujours cohérent.

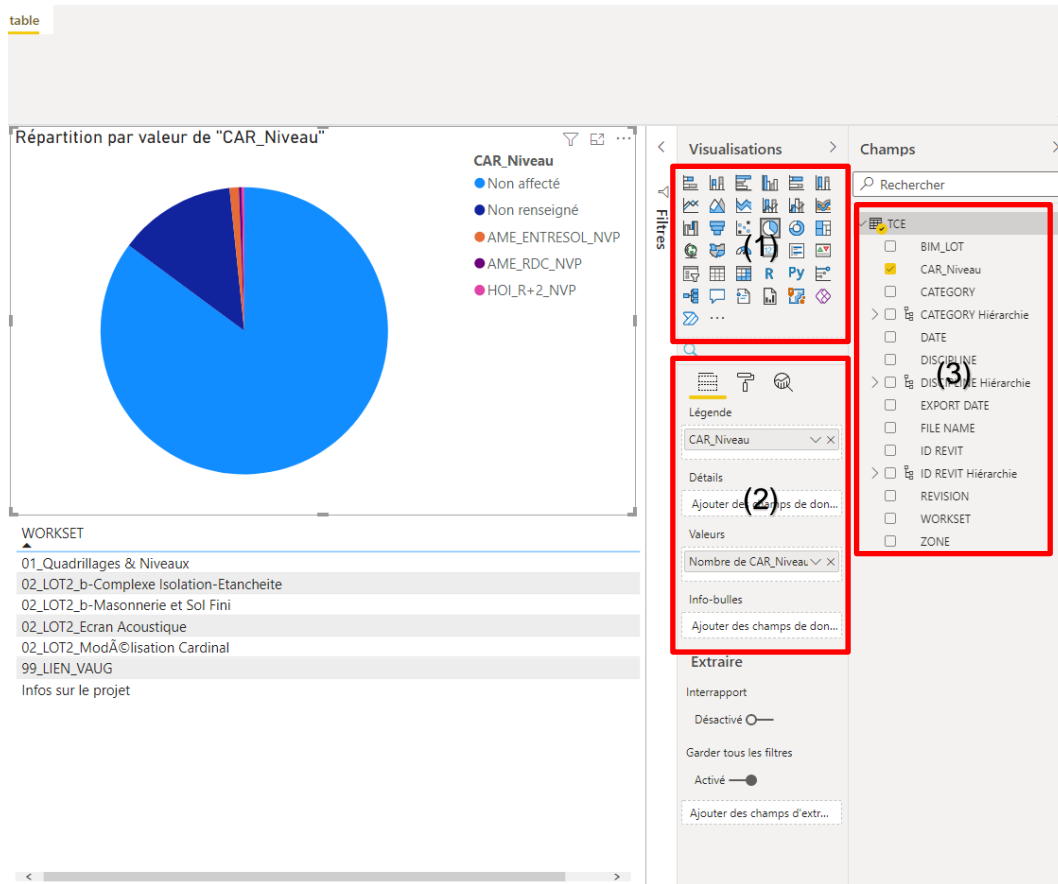
Exemple de message d'erreur :





## 5) Créer les visuels

Une fois la base de données contenant toutes les informations nécessaires et mises en forme, vous pourrez créer les visuels.



- (1) Visuels disponibles
- (2) Fenêtre de configuration des visuels
- (3) Champs disponibles depuis la base de données mise en forme

Pour configurer un visuel, vous devrez en sélectionner un parmi les visuels disponibles (1) selon ce que vous souhaitez afficher. L'étape suivante consiste à glisser-déposer les champs de la partie (3) vers la partie (2). Cela permet de définir quelles valeurs seront affichées dans le visuel et de spécifier la légende ainsi que des sous-décompositions.



La fenêtre (2) sert à régler l'aspect graphique (couleur des valeurs, titres, légendes ...) mais aussi le rendu des valeurs. Il est ainsi possible d'appliquer des fonctions statistiques de base sur les valeurs reportées dans les visuels.