Jour 3

- Objectif
  - Se familiariser avec le traitement programmatique via Python de données tabulaires (au format CSV ou Excel) en vue de leur publication sur le portail Open Data.
- Sommaire
  - 11. Publication de données tabulaires (Python)
  - … et exercices pratiques

11. Publication de données tabulaires: Utilisation d'outils en Python

Objectif

- A partir d'un jeu de données « brut » au format Excel
  - Non directement exploitable par une machine
  - Au format large
  - Avec éventuellement des données à caractère personnel
- Publier le jeu de données sur le portail Open Data (de pré-production)
  - Données directement exploitables par une machine
  - Au format long
  - Valeurs standardisées
  - Données validées
  - Sans données à caractère personnel
  - Accompagnées de métadonnées

=> En utilisant des outils en Python

Pourquoi?

SBC4D

- Automatisation
  - Certaines étapes de conversion sont plus difficiles à mettre en œuvre depuis Excel.
  - Une même série de transformations peut être appliquée sur un grand nombre de fichiers.
  - Facilité d'intégration dans un processus de gestion des données.
- Garanties
  - Validation des valeurs et de la transformation.
  - Création du schéma intégrée.
  - Pas de risques d'introduction manuelle d'erreurs lors de la conversion.
- Flexibilité
  - Gestion de formats additionnels (ex: CSV, Excel, JSON, etc.)
  - … en entrée comme en sortie!

Outils Python

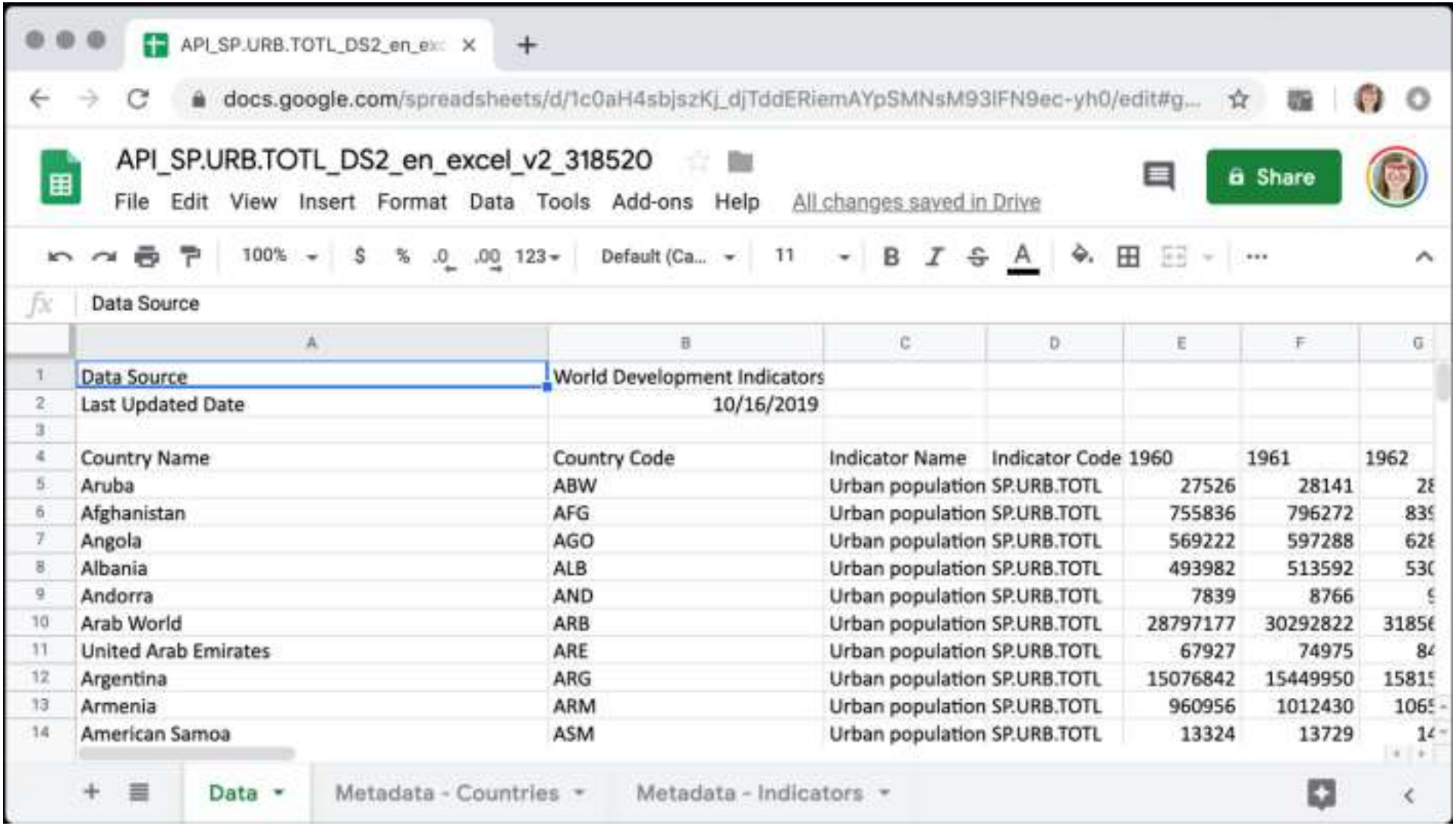
- Pour le nettoyage des données et la conversion d'un schéma initial vers un schéma final:
  Whyqd - https://whyqd.readthedocs.io/en/latest/

- Pour des opérations de manipulation de données (calculs, anonymisation, etc.):
  Pandas - https://pandas.pydata.org/

Méthodologie

- Mise en place
  - Examiner les données
- Identifier une stratégie de transformation
  - Schéma initial => schéma final; ou
  - Schéma initial => schéma intermédiaire => schéma final.
  - Nécessité de pivoter certaines données au format long.
- Définir le ou les schémas nécessaires
  - https://whyqd.readthedocs.io/en/latest/strategies/schema/
- Créer le(s) script(s) de transformation (*crosswalk*)
  - https://whyqd.readthedocs.io/en/latest/strategies/crosswalk/
- Transformer et valider les données
  - https://whyqd.readthedocs.io/en/latest/api/transform/

Examiner les données



	A	B	C	D	E	F	G
1	Data Source	World Development Indicators					
2	Last Updated Date	10/16/2019					
3							
4	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962
5	Aruba	ABW	Urban population	SP.URB.TOTL	27526	28141	28
6	Afghanistan	AFG	Urban population	SP.URB.TOTL	755836	796272	839
7	Angola	AGO	Urban population	SP.URB.TOTL	569222	597288	628
8	Albania	ALB	Urban population	SP.URB.TOTL	493982	513592	530
9	Andorra	AND	Urban population	SP.URB.TOTL	7839	8766	9
10	Arab World	ARB	Urban population	SP.URB.TOTL	28797177	30292822	31856
11	United Arab Emirates	ARE	Urban population	SP.URB.TOTL	67927	74975	84
12	Argentina	ARG	Urban population	SP.URB.TOTL	15076842	15449950	15815
13	Armenia	ARM	Urban population	SP.URB.TOTL	960956	1012430	1065
14	American Samoa	ASM	Urban population	SP.URB.TOTL	13324	13729	14

Dériver le schéma initial

```
import whyqd as qd

DATASOURCE_PATH = "…"
MIMETYPE = "xlsx"

datasource = qd.DataSourceDefinition()
datasource.derive_model(
   source=DATASOURCE_PATH,
   mimetype=MIMETYPE)
schema_source = qd.SchemaDefinition()
schema_source.derive_model(data=datasource.get)
schema_source.save()
```

Définir le schéma final

```
import whyqd as qd

schema: qd.models.SchemaModel = {
  "name": "rates_data",
  "title": "Commercial rates data",
  "description": "Standardised schema for archival and analysis of commercial rates
data."
}
fields: list[qd.models.FieldModel] = [
  { "name": "ba_ref",
    "title": "Billing reference",
    "type": "string",
    "description": "Unique code for a specific hereditament." },
  { "name": "prop_ba_rates",
    "title": "Property billing rates",
    "type": "number",
    "description": "Actual rates paid by a specific ratepayer." },
  …
]
schema_destination = qd.SchemaDefinition()
schema_destination.set(schema=schema)
schema_destination.fields.add_multi(terms=fields)
```

Script de transformation

```
"ACTION
    > 'destination_field'::'destination_term'
    < 'source_term'::['source_field', 'source_field']"
```

« Exécute cette action (**ACTION**) pour créer ce champ de destination (**destination_field**) avec cette valeur (**destination_term**) à partir de ces champs initiaux (**source_field**) »

Script de transformation

```
import whyqd as qd

ACTIONS = [
  "DEBLANK",
  "DEDUPE",
  "DELETE_ROWS < [0, 1, 2, 3]",
  f"PIVOT_LONGER > ['year', 'values'] < ["2000", "2001", "2002"],
  "RENAME > 'indicator_code' < ['Indicator Code']",
  "RENAME > 'indicator_name' < ['Indicator Name']",
  "RENAME > 'country_code' < ['Country Code']",
  "RENAME > 'country_name' < ['Country Name']",
]

crosswalk = qd.CrosswalkDefinition()
crosswalk.set(
    schema_source=schema_source,
    schema_destination=schema_destination)
crosswalk.actions.add_multi(terms=ACTIONS)
crosswalk.save()
```

Pivoter au format long

	A	B	C	D	E	F	G	H	I	J	K	
1	Data Source	World Development Indicators										
2	Last Updated Date	09/04/2020										
3												
4	Country Name	Country Code	Indicator Name	Indicator Code	1960	1961	1962	1963	1964	1965	1966	1967
5	Aruba	ABW	Urban populatio	SP.URB.TOTL	27526	28141	28532	28761	28924	29082	29253	
6	Afghanistan	AFG	Urban populatio	SP.URB.TOTL	755836	796272	839385	885228	934135	986074	1041191	1
7	Angola	AGO	Urban populatio	SP.URB.TOTL	569222	597288	628381	660180	691532	721552	749534	
8	Albania	ALB	Urban populatio	SP.URB.TOTL	493982	513592	530766	547928	565248	582374	599300	
9	Andorra	AND	Urban populatio	SP.URB.TOTL	7839	8766	9754	10811	11915	13067	14262	
10	Arab World	ARB	Urban populatio	SP.URB.TOTL	28797177	30292822	31856717	33513046	35275337	37163923	39098493	41

Action pour le script de transformation:

```
f"PIVOT_LONGER > ['year', 'values'] < {datasource.get[0].names[4:]}"
```

Transformation / Validation

SBC4D

```
import whyqd as qd

DATA_SOURCE = "…"
DESTINATION_DATA = "…"
DESTINATION_MIMETYPE = "xlsx"

transform = qd.TransformDefinition(
    crosswalk=crosswalk,
    data_source=DATA_SOURCE)
transform.process()
transform.save()

valiform = qd.TransformDefinition()
valiform.validate(
    transform=transform,
    data_destination=DESTINATION_DATA,
    mimetype_destination=DESTINATION_MIMETYPE)
```

Exemple: convertir un jeu de données publié par le HCP

	A	B	C	D	E
1	INDICATEUR				
2	CODE	LIBELLE	Désagrégation	Unite	Source
3	3482	Exportations à prix courants	[Produits,]	En millions de DH	Direction de la comptabilité nationale (HCP)
4					
5	SERIE				Données
6	CODE	Produits	2014	2015	2016
7	35697	Agriculture et sylviculture	23876	13048	16258
8	35715	Pêche et aquaculture	1483	1734	1703
9	35722	Extraction	12823	11944	14035
10	35702	Industries manufacturières	208064	152354	162175
11	35719	Fabrication de produits alimentaires et de boissons et tabacs	32181	23878	27015
12	35718	Fabrication de textiles, d'articles d'habillement, de cuir et d'articles de cuir	25094	21797	21830
13	35692	Fabrication d'articles en bois et en papier; imprimerie et reproduction de supports	1488	1286	1276
14	35696	Cokéfaction et raffinage	3129	6544	3065
15	35708	Fabrication de produits chimiques	45812	35219	37000
16	35721	Fabrication de produits pharmaceutiques de base et de préparations pharmaceutiques	1199	953	1006
17	35711	Fabrication d'articles en caoutchouc et en matières plastiques, et autres produits minéraux non métalliques	4180	3165	3368
18	35703	Fabrication de produits métallurgiques de base et d'ouvrages en métaux, sauf machines et matériel	6005	4233	5048
19	35707	Fabrication d'ordinateurs, d'articles électroniques et optiques	2200	1814	1574
20	35701	Fabrication d'équipements électriques	16916	14742	14522
21	35698	Fabrication de machines et de matériel, n.c.a	2983	2051	2051
22	35713	Fabrication de matériel de transport	62860	33929	40828
23	35706	Autres activités de fabrication (y.c fabrication de meubles), réparation et installation	4017	2743	3592
24	35717	Distribution d'électricité et de gaz-Distribution d'eau, réseau d'assainissement, traitement des déchets	2077	2212	2088
25	35693	Construction	4350	4614	5358
26	35720	Commerce de gros et de détail; réparation de véhicules automobiles et de motocycles	1252	364	436
27	35714	Transports et entreposage	31728	23761	25068
28	35699	Activités d'hébergement et de restauration	3336	2822	2798
29	35716	Information et communication	16396	13615	14263
30	35712	Activités financières et d'assurance	1805	1700	1764
31	35704	Activités immobilières	443	356	436

Exemple: convertir un jeu de données publié par le HCP

SBC4D

```
import whyqd as qd

# Lire le fichier Excel
# et en dériver le schéma initial
SOURCE_DATA = "i_3482.xlsx"
MIMETYPE = "xlsx"

datasource = qd.DataSourceDefinition()
datasource.derive_model(
    source=SOURCE_DATA,
    mimetype=MIMETYPE,
    # Les headers apparaissent ligne 6
    header=[5])
schema_source = qd.SchemaDefinition()
schema_source.derive_model(data=datasource.get)
```

Exemple: convertir un jeu de données publié par le HCP

SBC4D

```
# Définir le schéma final (crosswalk)
schema: qd.models.SchemaModel = {
  "name": "schema-exportations-prix-courants",
  "title": "Blah"
}
fields: list[qd.models.FieldModel] = [
  {
    "name": "code", "title": "Code",
    "type": "string",
    "constraints": { "required": True }
  },
  { "name": "produits",  "title": "Produits", "type": "string" },
  { "name": "annee", "title": "Année", "type": "integer" },
  { "name": "valeur", "title": "Valeur", "type": "number" }
]
schema_final = qd.SchemaDefinition()
schema_final.set(schema=schema)
schema_final.fields.add_multi(terms=fields)
```

Exemple: convertir un jeu de données publié par le HCP

SBC4D

```
# Définir le script de transformation
script = [
  "DELETE_ROWS < [0, 1, 2, 3, 4, 5, 36]",
  "RENAME > 'code' < ['CODE']",
  "RENAME > 'produits' < ['Produits']",
  f"PIVOT_LONGER > ['annee', 'valeur'] < {datasource.get.names[2:]}"
]
crosswalk = qd.CrosswalkDefinition()
crosswalk.set(
    schema_source=schema_source,
    schema_destination=schema_final)
crosswalk.actions.add_multi(terms=script)

# Effectuer la transformation
transform = qd.TransformDefinition(
    crosswalk=crosswalk,
    data_source=datasource.get)
transform.process()
transform.save(mimetype="csv")
```

Exemple: convertir un jeu de données publié par le HCP

SBC4D

```
code,produits,year,value
35697,Agriculture et sylviculture,2014,23876.0
35715,Pêche et aquaculture,2014,1483.0
35722,Extraction,2014,12823.0
35702,Industries manufacturières,2014,208064.0
35719,Fabrication de produits alimentaires et de boissons et
tabacs,2014,32181.0
35718,"Fabrication de textiles, d’articles d’habillement, de cuir
et d’articles de cuir",2014,25094.0
35692,Fabrication d’articles en bois et en papier; imprimerie et
reproduction de supports,2014,1488.0
35696,Cokéfaction et raffinage,2014,3129.0
35708,Fabrication de produits chimiques,2014,45812.0
35721,Fabrication de produits pharmaceutiques de base et de
préparations pharmaceutiques,2014,1199.0
35711,"Fabrication d’articles en caoutchouc et en matières
plastiques, et autres produits minéraux non
métalliques",2014,4180.0
…
```

Portail CKAN

- Le portail national est un portail **CKAN**: https://ckan.org/
- CKAN est un projet open source permettant de gérer un portail de données.
- CKAN est utilisé pour de nombreux portails nationaux de données ouvertes, par exemple aux Etats-Unis, au Canada, en Australie, ou au Mexique
- La documentation CKAN est disponible en ligne: https://docs.ckan.org/en/2.10/

Publication via l'API

- CKAN expose une API REST
- Cette API permet d'envoyer des requêtes au portail et de récupérer des réponses au format JSON. Par exemple, pour récupérer la liste des jeux de données disponible: https://data.gov.ma/data/api/3/action/package_list
- L'API permet d'accéder à toutes les données publiques du portail.
- Sous réserve de disposer de s'authentifier avec une clé d'API, l'API permet également de:
  - Modifier les métadonnées d'un jeu de données existant
  - Ajouter/Modifier les données (ressources) d'un jeu de données
  - Publier de nouveaux jeux de données
- La documentation de l'API CKAN est disponible en ligne: https://docs.ckan.org/en/2.10/api/index.html

Intérêts de l'API

- Utiliser l'API permet d'intégrer complètement l'étape « Publication sur le portail national de données ouvertes » dans le processus de gestion de données.
- Si nécessaire, l'API permet également de créer des ponts de synchronisation automatique entre le portail national et un portail géré au niveau d'une structure gouvernementale.

Conclusion

« Toute la littérature que les chercheurs souhaitent rechercher, récupérer et lire ne doit pas nécessairement relever de la notion de **connaissance**.

Nous voulons avoir accès à des **propositions** de connaissances sérieuses, même si elles s'avèrent fausses ou incomplètes.

Nous voulons avoir accès à des **hypothèses** sérieuses même si nous les testons toujours et discutons de leurs mérites.

Nous voulons avoir accès aux **données** et aux analyses proposées à l'appui des revendications que nous évaluons.

Nous voulons avoir accès à tous les arguments, preuves et discussions.

Nous voulons avoir accès à tout ce qui pourrait nous aider à décider de ce qui relève de la connaissance, pas seulement aux résultats que nous convenons d'appeler la connaissance.

Si l'accès dépendait du résultat du débat et de l'enquête, l'accès ne pourrait pas contribuer au débat et à l'enquête. »

Peter Suber, Open Access

Ressources

- Documentation sur le portail national : https://data.gov.ma/fr/documentations
  - Les manuels Open Data (bientôt celui sur les standards de données)
  - Les présentations
  - Le plan d'actions national
- La note de l'ADD sur le développement de référentiels
- Le guide méthodologique pour l'inventaire
- Plan d'activités
  - Les Feuilles de route Open Data des Ministères français: https://www.data.gouv.fr/en/datasets/feuilles-de-route-ministerielles-sur-la-politique-de-la-donnee-des-algorithmes-et-des-codes-sources/
  - Département du Commerce – Etats-Unis: https://www.commerce.gov/sites/default/files/2021-08/US-Dept-of-Commerce-Data-Strategy.pdf

Jour 3 - FIN