

TPs Données

la pollution atmosphérique

Liste des TP

XML.....	2
CSV.....	3
Python.....	4
Éléments de correction.....	5

Bulletin Officiel

Une **donnée** est une valeur décrivant un objet, une personne, un événement (expl : le numéro de téléphone). Plusieurs **descripteurs** peuvent être utiles pour décrire un même objet (par exemple des descripteurs permettant de caractériser un contact : nom, prénom, adresse et numéro de téléphone).

Une **collection** regroupe des objets partageant les mêmes descripteurs (par exemple, la collection des contacts d'un carnet d'adresses). La structure de table permet de présenter une collection : les objets en ligne, les descripteurs en colonne et les données à l'intersection. Les données sont alors dites **structurées**.

Pour assurer la persistance des données, ces dernières sont stockées dans des fichiers. Le format CSV (Comma Separated Values, les données avec des séparateurs) est un format de fichier simple permettant d'enregistrer une table. À tout fichier sont associées des **métadonnées** qui permettent d'en décrire le contenu. Ces métadonnées varient selon le type de fichier (date et coordonnées de géolocalisation d'une photographie, auteur et titre d'un fichier texte, etc.).

Une **base de données** regroupe plusieurs collections de données reliées entre elles. Par exemple, la base de données d'une bibliothèque conserve les données sur les livres, les abonnés et les emprunts effectués.

Contenus	Capacités attendues
Données	Identifier les principaux formats et représentations de données.
Données structurées	Identifier les différents descripteurs d'un objet. Distinguer la valeur d'une donnée de son descripteur. Utiliser un site de données ouvertes, pour sélectionner et récupérer des données.
Traitement de données structurées	Réaliser des opérations de recherche, filtre, tri ou calcul sur une ou plusieurs tables.
Métadonnées	Retrouver les métadonnées d'un fichier personnel.
Données dans le nuage (<i>cloud</i>)	Utiliser un support de stockage dans le nuage. Partager des fichiers, paramétrer des modes de synchronisation. Identifier les principales causes de la consommation énergétique des centres de données ainsi que leur ordre de grandeur.
Exemples d'activités	
<ul style="list-style-type: none">— Consulter les métadonnées de fichiers correspondant à des informations différentes et repérer celles collectées par un dispositif et celles renseignées par l'utilisateur.— Télécharger des données ouvertes (sous forme d'un fichier au format CSV avec les métadonnées associées), observer les différences de traitements possibles selon le logiciel choisi pour lire le fichier : programme Python, tableur, éditeur de textes ou encore outils spécialisés en ligne.— Explorer les données d'un fichier CSV à l'aide d'opérations de tri et de filtre, effectuer des calculs sur ces données, réaliser une visualisation graphique des données.— À partir de deux tables de données ayant en commun un descripteur, montrer l'intérêt des deux tables pour éviter les redondances et les anomalies d'insertion et de suppression, réaliser un croisement des données permettant d'obtenir une nouvelle information.— Illustrer, par des exemples simples, la consommation énergétique induite par le traitement et le stockage des données.	

1. XML

Selon les sources, la pollution atmosphérique est responsable d'environ 10 % des morts humains mais entraîne également des complications sur la santé des humains, sans parler des conséquences que la pollution atmosphérique a sur la faune. Il est donc important de limiter cette pollution et pour cela la comprendre en l'étudiant.

L'État français a mis en place des outils pour visualiser cette pollution atmosphérique [<http://www2.prevoir.org/>] mais laisse également un accès libre aux mesures effectuées sur le site [www.data.gouv.fr].


À la page [<https://www.data.gouv.fr/fr/datasets/donnees-temps-reel-de-mesure-des-concentrations-de-polluants-atmospheriques-reglementes-1/>] télécharger la "Note explicative sur les données mises à disposition", le "Guide IPR - partie 1", le fichier **xls** "Dataset D : métadonnées de stations de mesures" et ouvrez le fichier de données brutes.

Vous avez accédé à un fichier xml.

Similairement aux fichiers html, le contenu d'un fichier xml est structuré par des balises (eg **<om:OM_Observation ... >**)

Sous Firefox, appuyer sur la première balise **<base:Identifier>**. Que se passe-t-il ?

```
- <gml:featureMember>
- <aqd:AQD_ReportingHeader gml:id="REP_FR_2019-05-11-10_E2">
  <aqd:change>false</aqd:change>
  <aqd:changeDescription/>
- <aqd:inspireId>
- <base:Identifier>
  <base:localId>FR_2019-05-11-10_E2</base:localId>
  <base:namespace>FR.LCSQA-INERIS.AQ</base:namespace>
  <base:versionId>1</base:versionId>
</base:Identifier>
</aqd:inspireId>
```



Par la suite, on fermera les deux premières balises **<gml:featureMember>** en appuyant dessus.

Nous allons étudier la section **<gml:featureMember>** suivante.

Parcourir cette section afin de trouver la balise fermante **</gml:featureMember>**. La section que l'on va étudier contient tout ce qui est entre la balise ouvrante **<gml:featureMember>** à sa balise fermante.

Une section correspond à un ensemble de mesures prises pour un polluant donné, à une heure donnée et un lieu donné.

À l'aide du document "Dataset D : métadonnées de stations de mesures",
à quelle station le code station **FR15043** correspond-il ?
dans quelle ville est-elle située ?

À l'aide du document "Guide IPR - partie 1" p17,
à quelle polluant le code **5** correspond-il ?
à quelle polluant le code **7** correspond-il ?

À l'aide du document "Note explicative sur les données mises à disposition" p5-6,
préciser la date et l'heure à laquelle les mesures ont été prises.
préciser le code de la station de mesure, son nom et sa localité
préciser le type du polluant mesuré
préciser le nombre de mesures faites
au niveau de la ligne **<swe:values>**, préciser la valeur d'une mesure. La mesure est-elle valide ?

Le contenu d'un fichier xml vous semble-t-il ordonné ? Lisible ?

Quels avantages et inconvénients voyez-vous à ce type de format ?

Exposés possibles : coût énergétique des serveurs

Sources possibles :

Greenpeace (2017) [<https://www.greenpeace.fr/il-est-temps-de-renouveler-internet/>]

Europe 1 (2017) [<https://www.europe1.fr/sante/pollution-atmospherique-une-carte-interactive-sur-la-qualite-de-lair-en-europe-3494342>]

Futura Sciences (2016) [<https://www.futura-sciences.com/sante/actualites/vie-pollution-air-tue-2-fois-plus-quon-ne-pensait-63256/>]

Planetoscope [<https://www.planetoscope.com/mortalite/1913-deces-dus-a-la-pollution-dans-le-monde.html>]

Qualité de l'air (État français) [<http://www2.prevoir.org/>] ou (instances européennes) [<http://airindex.eea.europa.eu/>]

2. CSV

Afin de rendre plus intelligible le contenu du fichier xml étudié, un programme python a créé un fichier CSV à partir de ce premier fichier.

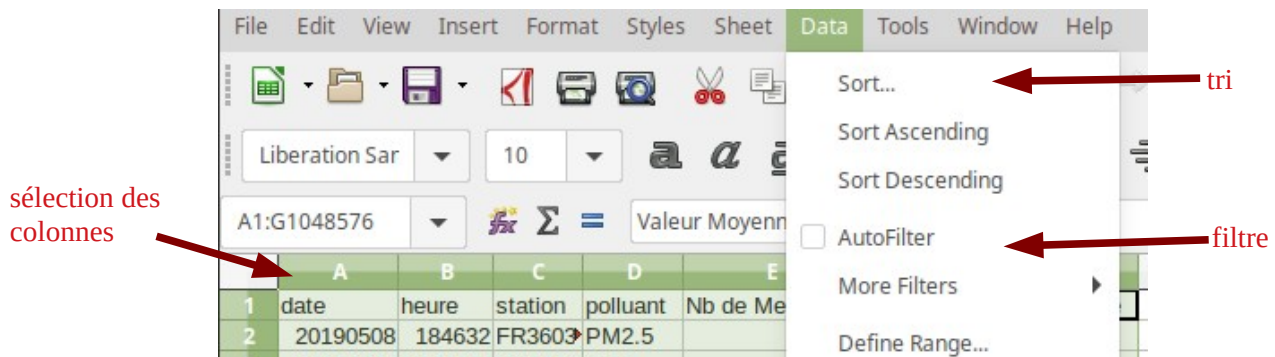
Ouvrir le fichier "**2019-Donnees-PollutionA.csv**" à l'aide d'un tableur LibreOffice Calc. Choisir pour séparateurs les points-virgules et les espaces.

Le titre de la dernière colonne vous semble-t-elle bien définie ?

Ouvrez le fichier à l'aide d'un éditeur de texte, modifier la 1^{ère} ligne afin que le titre de la dernière colonne soit bien défini.

Recharger le fichier dans LibreOffice Calc et fermer l'éditeur de texte.

Sélectionner les colonnes et faites un filtre afin de ne garder que les mesures liées à la station **FR15043** à Grenoble.



Quels sont les polluants mesurés à cette station ?

Quelles ont été les valeurs mesurées et à quelle heure ?

Calculer la moyenne des 3 valeurs moyennes a-t-il un sens ? Pourquoi ?

Annuler le filtre et sélectionner toutes les lignes (sauf la 1^{ère}). Faites alors un tri par polluant.

sélection des lignes

1	date	heure	code station	polluant	Un
2	20190508	184215	FR23140	C6H6	ug.
3	20190508	184215	FR23140	CO	mg
4	20190508	184632	FR36021	NO2	ug.
5	20190508	184632	FR36002	NO2	ug.
6	20190508	184514	FR34064	NO2	ug.
7	20190508	184514	FR34061	NO2	ug.

Dans la cellule **G1**, entrer "Nombre de stations". On notera le nombre de station mesurant le polluant au niveau de sa dernière ligne de valeurs.

Combien de station mesure le Benzène (C₆H₆) ? Compléter les cellules **G2** et **G3**.

Pour les stations mesurant le dioxyde d'azote (NO₂), entrer dans la cellule **G86** `=count(F4:F86)`.

Compléter les cellules **G90**, **G149** etc.

Dans la cellule **H1**, entrer "Moyenne par polluant". On notera la moyenne d'un polluant au niveau de sa dernière ligne de valeurs.

Quelle est la moyenne du polluant Benzène (C₆H₆) ? Compléter les cellules **H2** et **H3**.

Pour la moyenne de dioxyde d'azote (NO₂), dans la cellule **H86**, entrer `=average(F4:F86)`.

Compléter les cellules **H90**, **H149** etc.

La commande `=countif(A2:A3;"<=2")` permet de savoir combien de cases parmi celles entre **A2** et **A3** on une valeur inférieur à 2.

Comment faire pour compter le nombre de station en-dessus de la moyenne ?

Sauvegarder le fichier en tant que csv. Ouvrir le fichier sous un éditeur de texte. Les formules entrées ont-elles été sauvegardées ?

Sauvegarder le fichier sous le format odt. Quel est l'intérêt de sauvegarder sous ce format ?

Travail possible :

Faire une recherche sur l'un des polluants du tableau et préciser des risques liés à ce polluant

Déterminer le seuil légal maximum des polluants mesurés, comparer avec la moyenne calculée, compter le nombre de station au dessus du seuil, faire une étude temporelle

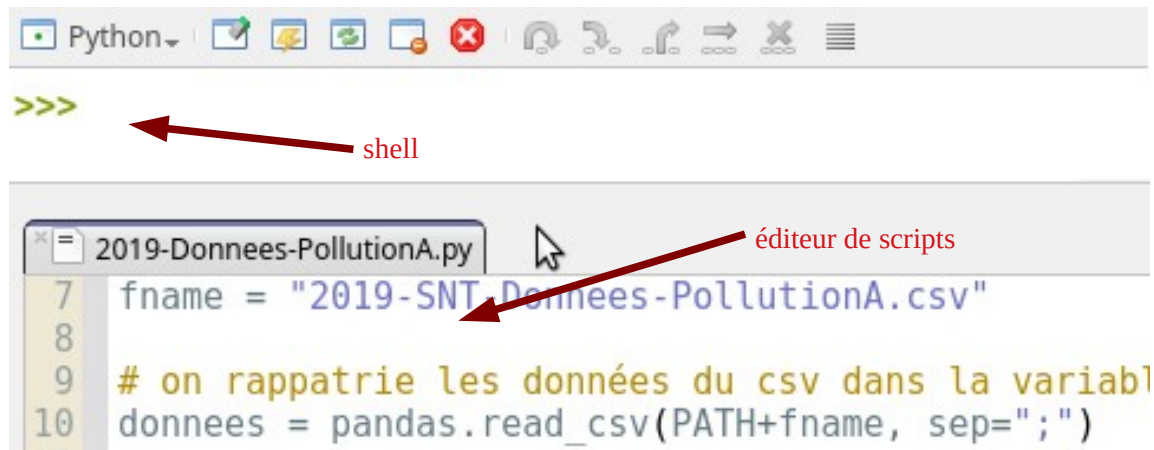
Calcul de quartiles, indicateurs de dispersion, graphes

3. Python

Il a été dit qu'un programme python avait créé le fichier csv à partir du fichier xml.
Serait-il possible d'étudier les mesures à l'aide de python ?

Lancer un éditeur python (eg pyzo) et ouvrir le fichier "**2019-Donnees-PollutionA.py**" (x0x, xlx ou xxx sont à modifier).
définir le chemin correspondant à votre répertoire dans la variable **PATH** (eg K:\MesDocuments\).

La ligne **pandas.read_csv()** permet, à l'aide de pandas, de lire le fichier csv et charger le fichier dans une variable pandaframe nommée **donnees**.



dans le shell, entrer **donnees.columns.values**. Que cela renvoie-t-il ?

à l'aide de la commande **donnees.info()**, déterminer la place mémoire occupée par la dataframe donnees.

comparer les retours des commandes **donnees.loc[3,:]** et **donnees.loc[1:3,:]**.

comment afficher les 5 premières lignes avec toutes les colonnes ?

comparer les retours des commandes **donnees.loc[3,:]** et **donnees.loc[3,2]**.

comparer les retours des commandes **donnees.loc[3,:]** et **donnees.loc[3,['date', 'heure']]**.

comment afficher les 5 premières lignes avec les deux colonnes '**polluant**' et '**Val Mesuree**' ?

Définir une variable **dTronquees** (ligne 14) avec toutes les lignes mais qu'avec les colonnes '**polluant**' et '**Val Mesuree**'.

Afin d'afficher toutes ses valeurs, on effectue une boucle et on affiche chacun de ses éléments.

```
for e in dTronquees.values :  
    print(e)
```

écrire ce code dans le script (ligne 17) et exécuter-le (touche **F5**).

Afin de ne garder que les mesures du NO₂, on effectue une boucle avec un test conditionnel.

```
dNO2 = []  
for e in dTronquees.values :  
    if e[0]=="NO2" :  
        dNO2.append( e[1] )
```

Ainsi **dNO2** contient toutes les mesures faites pour le polluant dioxyde d'azote. Compléter le code lignes 22-25.

On peut alors calculer la moyenne des mesures pour le dioxyde d'azote.

Afficher les 3 premières valeurs en entrant **dNO2[:3]** dans le shell et calculer la moyenne manuellement

Que faut-il faire pour calculer la moyenne des mesures de **dNO2** ? (rq : le nombre total de mesures est **len(dNO2)**).

Compléter le code fourni (lignes 27 à 31)

Comparer le résultat obtenu avec celui du tableur.

On souhaite compter le nombre de mesures supérieures à la moyenne.

Compléter le code fourni (lignes 34 à 38)

Comparer au résultat du tableur.

Proposer un code afin de faire la même chose avec les autres polluants (O₃, PM₁₀, PM_{2.5} et SO₂).

Approfondissement : répondre aux traitements des autres polluants à l'aide de fonctions

Sources : python pandas [<https://python-simple.com/python-pandas/panda-intro.php>]

Éléments de correction

XML :

FR15043 correspond à la station Grenoble les Frenes

le code 5 correspond aux particules inférieures à 10 micromètres (PM10).

le code 7 correspond à l'ozone (O₃)

les réponses dépendent du fichier étudié, pour le fichier "**2019-SNT-Donnees-PollutionA.xml**" cela donne les mesures ont été faites 8 mai 2019 à 18h46 (et 32s).

la station de mesure a pour code FR36021 qui correspond à la Drôme Rurale Sud-SND à St Nazaire le Désert. il a été mesure la concentration du polluant n°8 soit le dioxyde d'azote (NO₂).

5 mesures ont été effectuées

la première mesure listée vaut 1,8 µg/m⁻³ et est valide (code 1).

Avantages : très structuré, automatisable

Inconvénients : trop verbeux, difficilement lisible par un humain

CSV :

FR15043 mesure la concentration de dioxyde d'azote (NO₂), d'ozone (O₃) et des PM2,5.

Les mesures ont été prises à 18h45 le 8 mai 2019.

Calculer la moyenne de trois polluants différents ne fait pas sens. On fait la moyenne de choses similaires.

On obtient alors

polluant	C6H6	CO	NO2	NOx	O3	PM10	PM2,5	SO2
nb mesures	1	1	83	4	59	60	32	24
moyenne	0,39	0,23	9,74	4,98	70,24	11,34	7,37	4,34

On écrit en I86 =**countif(F4:F86;">9,74")** ; etc. jusqu'en I265 =**countif(F242:F265;">4,34")**.

polluant	C6H6	CO	NO2	NOx	O3	PM10	PM2,5	SO2
nb > moyen	0	0	31	2	30	24	15	5

Lorsque l'on sauvegarde au format csv seules les valeurs sont sauvegardées mais pas les formules. Par conséquent, sauvegarder en odt permet de se souvenir des opérations que l'on fait pour l'étude des données. on a donc un fichier dynamique.

Python :

donnees.columns.values renvoie

donnees.info() renvoie

`['date' 'heure' 'code station' 'polluant' 'Unite' 'Val Mesuree']`

`<class 'pandas.core.frame.DataFrame'>`

RangeIndex: 264 entries, 0 to 263

Data columns (total 6 columns):

date 264 non-null int64

heure 264 non-null int64

code station 264 non-null object

polluant 264 non-null object

Unite 264 non-null object

Val Mesuree 264 non-null float64

dtypes: float64(1), int64(2), object(3)

memory usage: 12.5+ KB

donnees.loc[:3, :] renvoie les 4 premières lignes tandis que **donnees.loc[1:3, :]** omet la 1^{ère} ligne.

pour afficher les 5 premières lignes, on écrit dans le shell **donnees.loc[:4, :]**.

la commande **donnees.loc[:3,:2]** renvoie l'erreur **Cannot do slice indexing on <class pandas...>**.

pour afficher les 5 premières lignes avec les 2 colonnes, on écrit **donnees.loc[:4, ['polluant', 'Val Mesuree']]**.

pour la suite, cf fichier "**2019-Donnees-PollutionA-corrige.py**"