

Projet 6 – Open Data

Table des matières

Activité 1 : Choix du jeu de données.....	1
Activité 1 bis: nettoyage du jeu.....	1
Activité 2 : Import du jeu dans DBBrowser.....	2
Activité 3 : Requêtes SQL.....	2
Activité 4 : Python et base de données SQLite3.....	7
Activité 5 : Exploitation visuelle.....	7
Activité 6: Jointure de tables.....	8

Activité 1 : Choix du jeu de données

Fichier concerné : [arrets.csv](#)

J'ai choisi la table « Points d'arrêt du réseau Transports en Commun Lyonnais »,

<https://data.grandlyon.com/jeux-de-donnees/points-arret-reseau-transports-commun-lyonnais/telechargements> ,

(l'originale est trouvable dans le dossier `fichiers_pre.modif`).

Après conversion via Internet du format JSON au format CSV, voici les informations que l'on peut en tirer :

- Sans compter la première ligne, cette table possède 4635 enregistrements
- Il y a 12 attributs. Les plus intéressants sont bien sûr `id`, `nom`, `desserte`, et les coordonnées. Ceux traitant l'accessibilité peuvent aussi être utiles.
- Beaucoup de caractères en trop et d'attributs inutiles.

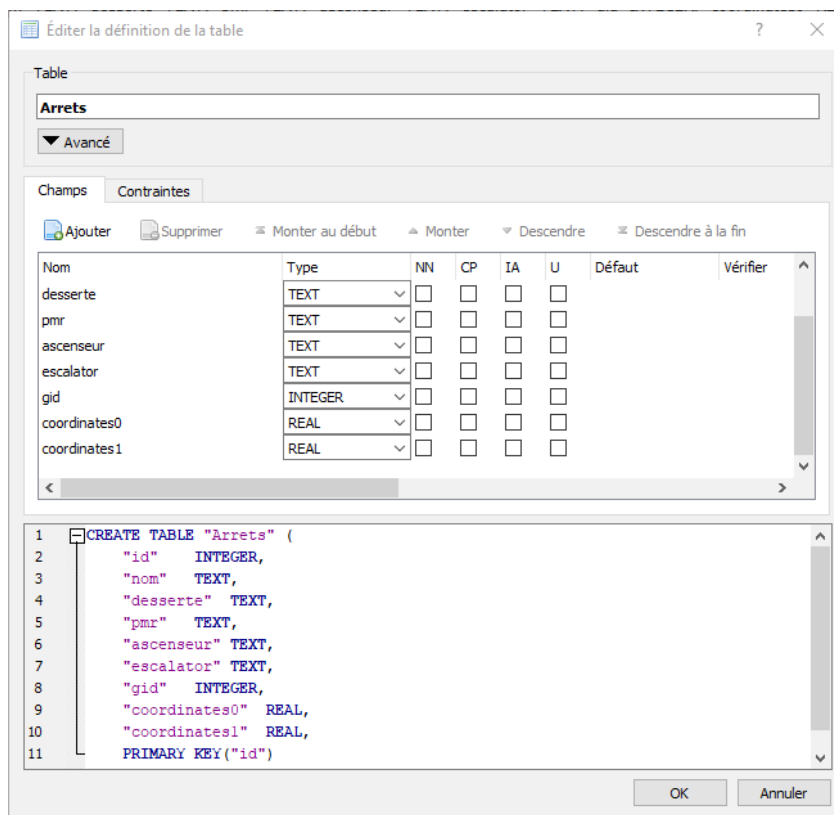
Activité 1 bis: nettoyage du jeu

Le nettoyage étant quasiment impossible à effectuer en Python à mon niveau, il fut réalisé à la main via le tableur :

- Les attributs de lignes du type A:A, B:R, A:R, B:A, etc. . ont été retirés. Le site TCL a été consulté pour éviter de supprimer ce genre d'attributs chez certaines lignes telles que 73E, 89D, C24E...
- Certaines lignes étaient notées en double, problème réglé désormais
- Les séparateurs des lignes de transport dans l'attribut `desserte` étaient des virgules, qui transformaient ces lignes en décimaux ou en puissances selon les cas, ou qui retiraient des 0 importants (lignes de bus 40, 70, etc). Après avoir essayé les tirets (opéraient comme des soustractions), les slashes (opéraient comme des dates) et même les points médians (problème d'encodage), c'est « & » l'esperluette qui fut retenue.
- Les attributs `type`, `last_update` et `last_update_fme` ont été supprimés.

Activité 2 : Import du jeu dans DBBrowser

Fichier concerné : [requetes.sqlite3](#)



La table *Arrets* définie dans DBBrowser

Activité 3 : Requêtes SQL

Fichier concerné : [requetes.sqlite3](#)

Exemples de questions auxquelles on peut répondre avec cette table :

1) Données de base, nombre d'enregistrements

1	SELECT COUNT(*)
2	FROM Arrets;
	COUNT(*)
1	4635

2) Combien y a-t-il d'arrêts avec un nom différent ?

```
1 SELECT COUNT(DISTINCT nom)
2 FROM Arrêts;
```

	COUNT(DISTINCT nom)
1	2102

Il y en a 2102.

3) Quels sont les différents arrêts de la ligne C19 contenant « Sainte-Foy » dans leur nom ?

On notera que Sainte-Foy s'écrit à la fois « Sainte Foy » et « Ste Foy » dans le tableur. Ainsi :

```
2 FROM Arrêts
3 WHERE nom LIKE "%Sainte Foy%"
4 OR nom LIKE "%Ste Foy%"
5 AND desserte LIKE "%C19%";
```

	nom
1	Cimetiere de Ste Foy
2	Petit Sainte Foy
3	Ste Foy Centre
4	Ste Foy Châtelain
5	Ste Foy Eglise
6	Ste Foy Hopital
7	Ste Foy Platanes
8	Mairie de Sainte Foy

4) Quelles lignes desservent l'arrêt Cuire ?

```
1 SELECT desserte
2 FROM Arrêts
3 WHERE nom = "Cuire";
```

	desserte
1	S5
2	C
3	33 & 38
4	33 & S5
5	C1
6	C13
7	38 & C13



Il s'agit donc des lignes C1, C13, 33, 38 et S5, ainsi que du métro C.

5) Combien de points d'arrêts sont accessibles aux PMR ? Et combien ne le sont pas ?

1	SELECT COUNT(*)
2	FROM Arrêts
3	WHERE pmr = "true";

	COUNT(*)
1	3391

1	SELECT COUNT(*)
2	FROM Arrêts
3	WHERE pmr = "false";

	COUNT(*)
1	1244

6) Quels sont les arrêts de la ligne C3 ?

1	SELECT DISTINCT nom
2	FROM Arrêts
3	WHERE desserte LIKE "%C3%";

	nom
1	Vaulx HDV Campus
2	Grand Vire
3	Cuzin - Stalingrad
4	Lesire
5	Vaulx - La Grappiniere
6	Mas du Taureau
7	Lefevre
8	Vaulx Les Grolieres
9	Laurent Bonnevey
10	Pont des Planches
11	Bon Coin - Medipole

12	Cyprian - Leon Blum
13	Bernaix
14	Grandclement
15	Blanqui - Le Rize
16	Verlaine
17	Instit. Art Contemporain
18	Charmettes
19	Thiers - Lafayette
20	Part-Dieu Jules Favre
21	Halles Paul Bocuse
22	Saxe - Lafayette
23	Hotel de Ville L. Pradel
24	Cordeliers
25	Terreaux La Feuillie
26	Gare Saint-Paul

7) Combien d'arrêts sont desservis par la ligne 17 ?

Les lignes C17 et 171 viennent perturber la requête, il faut donc exclure ces lignes avec la méthode NOT LIKE :

```

1 SELECT COUNT(DISTINCT nom)
2 FROM Arrets
3 WHERE desserte LIKE "%17%"
4 AND desserte NOT LIKE "%C17%"
5 AND desserte NOT LIKE "%171%";

```

	COUNT(DISTINCT nom)
1	53

Ainsi, la ligne 17 comporte 53 arrêts.

8) Quelles sont les coordonnées de l'arrêt *Les Marches du Rhône* ?

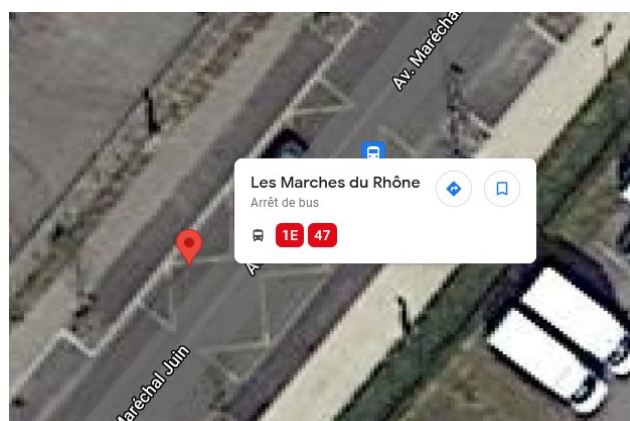
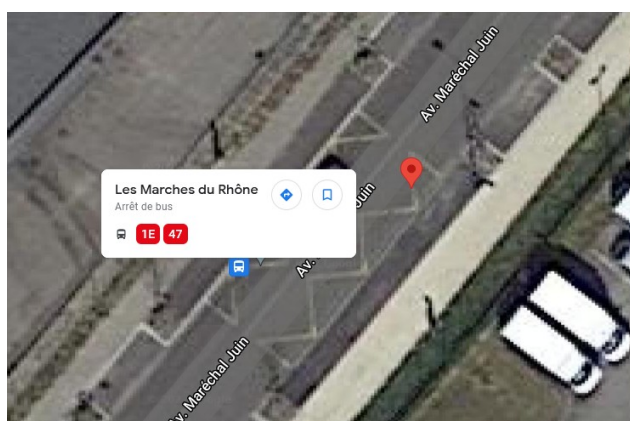
```

1 SELECT coordinates0, coordinates1
2 FROM Arrets
3 WHERE nom = "Les Marches du Rhone";

```

	coordinates0	coordinates1
1	45.6883533824318	5.0668844570195
2	45.6883073805192	5.06675142237945

Deux points d'arrêts → deux latitudes, deux longitudes :



9) Quels arrêts sont desservis par le tram T3 ?

Ici encore, la ligne de car T36 gêne la requête.
Même réponse au problème, donc :

```

1 SELECT DISTINCT nom
2 FROM Arrets
3 WHERE desserte LIKE "%T3%"
4 AND desserte NOT LIKE "%T36%";

```

	nom
1	Meyzieu les Panettes
2	Meyzieu Z.i.
3	Meyzieu Gare
4	Decines Grand Large
5	Decines Centre
6	Vaulx-en-Velin La Soie

7	Bel Air - Les Brosses
8	Gare de Villeurbanne
9	Reconnaissance - Balzac
10	Dauphine -Lacassagne
11	Gare Part-Dieu Vilette

10) Combien d'arrêts possèdent des ascenseurs ?

```

1 SELECT COUNT(DISTINCT nom)
2 FROM Arrets
3 WHERE ascenseur = "true";

```

	COUNT(DISTINCT nom)
1	41

11) J'ai aperçu un doublon rebelle

```

1 SELECT nom, desserte
2 FROM Arrets
3 WHERE nom = "Bir Hakeim"

```

	nom	desserte
1	Bir Hakeim	C11 & C11
2	Bir Hakeim	C11 & C11

```

1 UPDATE Arrets
2 SET desserte = "C11"
3 WHERE nom = "Bir Hakeim"

```

```

1 SELECT nom, desserte
2 FROM Arrets
3 WHERE nom = "Bir Hakeim"

```

	nom	desserte
1	Bir Hakeim	C11
2	Bir Hakeim	C11

Activité 4 : Python et base de données SQLite3

Fichiers concernés : [act4_bdd.py/requetes.sqlite3](#)

Le fichier comporte plusieurs variantes d'exécution pour obtenir le même résultat.

Activité 5 : Exploitation visuelle

Fichiers concernés :

[act5_folium.py/act5_p2.py/act5.html/act5_tcl.html](#)

Le fichier `act5_folium.py` reprend l'exemple de l'énoncé et donne la carte `act5_exemple.html`.

Le fichier `act5_p2.py` utilise ma table CSV et donne la carte `act5_tcl.html`. C'est ici que j'ai remarqué que les séparateurs sous forme de points-virgules posent des problèmes lors de lecture de listes de listes, ils ont donc été remplacés par des virgules.

Légende du fichier (j'ai essayé d'en rajouter une sur la carte mais c'est complexe) :

	Métro A		Tramway
	Métro B		Funiculaire
	Métro C		Bus
	Métro D		

Activité 6: Jointure de tables

Fichiers concernés :

[communes.csv/act6.py/act6_tcl2.html/jointure.sqlite3](#)

Une table intéressante à joindre était celle représentant les communes du département du Rhône (69) puisque les arrêts TCL ne dépassent pas (ou presque) les limites de l'agglomération lyonnaise.

J'ai donc pris cette table :

<https://www.data.gouv.fr/fr/datasets/communes-de-la-metropole-de-lyon-et-ses-alentours/>

(l'originale est trouvable dans le dossier `fichiers_pre.modif`).

- Le problème : rien ne permettait de joindre les deux tables.

Après de sacrées recherches, j'ai finalement saisi à la main les 4635 valeurs nécessaires pour réaliser une jointure...

- Deux attributs ont été rajoutés à cette table : `cp` pour les codes postaux des communes en question, et `is_metropole` qui montre si telle commune fait partie des 59 communes de la Métropole de Lyon.

- La jointure se fait entre la clé étrangère `commune_gid` de la table `Arrets`, qui se rapporte à la clé primaire `gid` de la table `Communes`.

- Pour une drôle de raison, il me manquait une seule commune, celle de Thurins, qui était nécessaire pour compléter la jointure. Je l'ai donc rajoutée grâce à Internet.

Quelques requêtes SQL désormais possibles ([jointure.sqlite3](#)):

- On peut afficher les arrêts se situant dans une commune spécifique :

```
1 SELECT a.nom, a.desserte
2 FROM Arrets AS a
3 JOIN Communes as c
4 ON a.commune_gid = c.gid
5 WHERE c.nom = "VAULX-EN-VELIN";
```

	nom	desserte
1	Chardonnet	ZI3
2	Chardonnet	ZI3
3	Chemin des Pivolles	52
4	Chemin des Pivolles	52
5	Cuzin - Stalingrad	C3 & C8
6	Cuzin - Stalingrad	C3
7	Desgrand	37

- On peut afficher les arrêts ne se situant pas dans les communes de la Métropole de Lyon :

```
1 SELECT DISTINCT a.nom, a.desserte
2 FROM Arrets AS a
3 JOIN Communes as c
4 ON a.commune_gid = c.gid
5 WHERE c.is_metropole is False;
```

	nom	desserte
1	24 Aout	87 & ZI1
2	4 Chemins - La Salette	C20 & C20E
3	4 Chemins - La Salette	90
4	8 Mai - Parc du Roy	33
5	Ambroise Pare-Laennec	24
6	Ambroise Pare-Seignemartin	24 & 34
7	Corbas Gabriel Peri	76
8	Acacias	19
9	Acacias	19 & 28 & 58 & 61 & 73

- On peut afficher le code postal de la commune dans laquelle se trouve un arrêt spécifique (ici exemple : l'arrêt *Pennachy*) :




1	SELECT DISTINCT c.nom, c.cp
2	FROM Communes AS c
3	JOIN Arrêts AS a
4	ON c.gid = a.commune_gid
5	WHERE a.nom = "Pennachy"

	nom	cp
1	SAINT-GENIS-LAVAL	69230

Une carte HTML possible ([act6_tcl2.html](#)) :

Répertoire des arrêts TCL de manière « épiciéntrique », c'est-à-dire de manière à hiérarchiser les arrêts selon la présence ou non des villes dans la Métropole de Lyon, et si ils se situent à Lyon ou Villeurbanne le cas échéant.

Légende de la carte :

-  Dans la Métropole (à Lyon ou Villeurbanne)
-  Dans la Métropole (ni Lyon ni Villeurbanne)
-  Pas dans la Métropole