TP SQL — la base `education.db`

Écoles, collèges, lycées, spécialités et communes de France : la base `education.db` est décrite en détail dans `README_bases.md`. Ce qu'il faut savoir pour ce TP :

— **aucune valeur `NULL`** ; dans les tables d'effectifs, l'absence de ligne signifie un effectif nul ;
— les **codes** (`code_insee`, `code_commune`, `dep_code`, `uai`…) sont des **chaînes de caractères** : on écrit `dep_code = '78'`, avec des apostrophes ;
— les établissements sont **datés** : la clé primaire de `ecoles`, `colleges` et `lycees` est le couple `(uai, rentree)`.

Rappels hors programme : `NULL`, `LEFT JOIN`, `WITH`, `IN`/`NOT IN`, `EXISTS`, `GROUP_CONCAT`.

Tout ce TP se traite avec `SELECT … FROM … JOIN … ON … WHERE … GROUP BY … HAVING … ORDER BY … LIMIT/OFFSET`, les sous-requêtes et `UNION`/`INTERSECT`/`EXCEPT`.

Chargement de la base

```
from sql_fonctions import GestionnaireSQL

db = GestionnaireSQL("education.db")
db.liste_tables()
```

```
name
-----------------
communes
ecoles
colleges
effectifs_colleges
lycees
effectifs_lycees
specialites
combinaisons
cpge
```

```
# Les schémas des tables utilisées dans ce TP.
for table in ["communes", "ecoles", "colleges", "effectifs_colleges",
              "lycees", "effectifs_lycees", "specialites", "cpge"]:
    print(f"{table} : {db.schema('SELECT * FROM ' + table)}")
```

```
communes : ['code_insee', 'nom_standard', 'nom_sans_pronom', 'nom_a',
    'nom_de', 'nom_sans_accent', 'nom_majuscule', 'reg_code', 'reg_nom',
    'dep_code', 'dep_nom', 'epci_code', 'epci_nom', 'code_postal',
    'academie_code', 'academie_nom', 'population', 'superficie_km2',
    'altitude_moyenne', 'altitude_min', 'altitude_max',
    'latitude_mairie', 'longitude_mairie', 'latitude_centre',
    'longitude_centre', 'grille_densite', 'grille_densite_texte',
    'gentile', 'url_wikipedia']
ecoles : ['uai', 'rentree', 'denomination', 'patronyme', 'secteur',
    'code_commune', 'rep', 'rep_plus', 'nb_classes', 'pre_elementaire',
    'elementaire', 'ulis']
colleges : ['uai', 'rentree', 'denomination', 'patronyme', 'secteur',
    'code_commune', 'rep', 'rep_plus']
```

```
effectifs_colleges : ['uai', 'rentree', 'niveau', 'filles', 'garcons']
lycees : ['uai', 'rentree', 'denomination', 'patronyme', 'secteur',
    'code_commune']
effectifs_lycees : ['uai', 'rentree', 'niveau', 'serie', 'filles',
    'garcons']
specialites : ['uai', 'rentree', 'niveau', 'code_specialite',
    'specialite', 'filles', 'garcons']
cpge : ['rentree', 'ministere', 'filiere', 'annee_etude',
    'garcons_public', 'filles_public', 'garcons_prive', 'filles_prive']
```

Exécuter une requête dans ce TP

Placer ce notebook, `education.db` et `sql_fonctions.py` dans le même dossier, puis exécuter les deux cellules de chargement ci-dessus avec **Maj + Entrée**. Dans une cellule de code, écrire une requête SQL entre les triples guillemets de `db.requete("""…""")`, puis exécuter la cellule : les noms des colonnes et les lignes du résultat s'affichent en dessous. Les triples guillemets permettent d'écrire la requête sur plusieurs lignes ; les valeurs textuelles SQL restent entre apostrophes. Par exemple :

```
db.requete("""
SELECT nom_standard
FROM communes
LIMIT 5
""")
```

Modifier la requête et réexécuter la cellule pour essayer une autre solution. Pour la partie graphique, `db.liste("""…""")` renvoie une liste de tuples à conserver dans une variable, tandis que `db.schema("""…""")` renvoie les noms des colonnes. Après un redémarrage du noyau, réexécuter les cellules de chargement.

1. Lire le schéma

Question 1. Sans écrire de requête :

1. Quelle est la clé primaire de `communes` ? Celle de `lycees` ? Pourquoi `uai` seul ne suffit-il pas ?
2. Quelle colonne de `lycees` est une clé étrangère, et vers quelle colonne de quelle table ?
3. Donner le type d'association ($1-1$, $1-*$ ou $*-*$) entre : `communes` et `lycees` ; `lycees` et `effectifs_lycees` ; les lycées et les spécialités.

2. Une seule table

Question 2 *(projection, sélection, tri)*. Le nom et la population des communes des Yvelines (`dep_code = '78'`) de plus de 20 000 habitants, par population décroissante.

Question 3 *(`DISTINCT`)*.

1. Comparer le nombre de lignes de `colleges` et son nombre d'établissements distincts. Expliquer l'écart.
2. Combien de communes distinctes comptent au moins un collège REP+ à la rentrée 2025 ?

Question 4 *(`ORDER BY`, `LIMIT`, `OFFSET`)*. Les trois communes les plus peuplées de France, puis les communes des rangs 4 à 10 (nom et population).

Question 5 *(renommage `AS`)*. Les dix communes les plus denses parmi celles de plus de 10 000 habitants (nom et densité en habitants/km²).

Agrégats

Question 6 *(agrégats simples)*. En une seule requête : le nombre de communes des Yvelines, leur population totale, leur superficie totale et l'altitude du plus haut sommet du département.

Question 7 *(`GROUP BY`)*. Le nombre de communes et la population totale de chaque région, par population décroissante (nom de région, nombre de communes, population de la région).

Question 8 *(`WHERE` et `HAVING`)*. Les académies comptant au moins cinq communes de plus de 50 000 habitants, avec ce nombre, par ordre décroissant.

Question 9 *(division entière !)*. La proportion de filles en terminale générale (série `G`) en France à la rentrée 2025.

Sous-requêtes

Question 10 *(sous-requête scalaire)*. La commune la plus peuplée des Yvelines, de deux façons : avec `ORDER BY … LIMIT 1`, puis avec une sous-requête scalaire. Quelle version garde les ex æquo ? Pourquoi `SELECT nom_standard, MAX(population)` serait-il incorrect ?

On affichera (nom, population).

Question 11 *(rang)*. Le rang de Versailles au classement des communes françaises par population décroissante.

Question 12 *(agrégat d'agrégat)*. Le nombre moyen d'écoles par commune à la rentrée 2025 (parmi les communes qui en ont au moins une).

Question 13 *(compter des groupes)*. Dans un département, appelons *doublon* un ensemble de communes ayant exactement la même population. Les départements comptant au moins cent doublons, avec ce nombre, par nombre décroissant.

3. Jointures

Toute cette partie se joue dans l'univers des lycées, avec des tables introduites une à une : d'abord le couple `lycees`–`communes`, puis `effectifs_lycees`, et enfin la table d'association `specialites`.

3.1 Un premier couple : `lycees` et `communes`

Question 14 *(jointure simple)*. La dénomination, le patronyme et le secteur des lycées de Versailles à la rentrée 2025.

Question 15 *(désigner par la clé, pas par le nom)*. Combien la commune nommée Saint-Denis compte-t-elle de lycées à la rentrée 2025 ? La réponse obtenue est-elle correcte ?

3.2 Un deuxième couple : `lycees` et `effectifs_lycees`

Question 16 *(condition de jointure composée)*. Le patronyme et l'effectif de terminale G des lycées de Versailles à la rentrée 2025, par effectif décroissant. Le code INSEE de Versailles, lu dans `communes` à la question 14, est `'78646'`.

Question 17 *(joindre avant d'agréger)*. Le plus grand effectif de terminale G d'un lycée privé à la rentrée 2025.

Question 18 *(portée d'une sous-requête scalaire)*. Le ou les lycées privés atteignant ce record en 2025 (patronyme, effectif). Que renvoie la requête si la sous-requête calcule le maximum sur tous les lycées ?

3.3 La même table des deux côtés : autojointures

Question 19 *(autojointure)*. Les paires de lycées distincts de Versailles à la rentrée 2025. Combien de lignes obtiendrait-on avec `<>` au lieu de `<` ? Et sans aucune condition sur les `uai` ?

Question 20 *(autojointure temporelle)*. Pour chaque lycée de Versailles, l'effectif de terminale G à la rentrée 2019 et à la rentrée 2025, et la variation entre les deux, par variation décroissante.

3.4 Trois tables et plus

Question 21 *(jointure triple, agrégat)*. Les dix académies dont l'effectif total de terminale G est le plus élevé à la rentrée 2025 (nom, effectif).

Question 22 *(maximum par groupe)*. Pour chaque académie, le lycée dont l'effectif de terminale G est le plus élevé à la rentrée 2025 (académie, patronyme, effectif), par effectif décroissant. Pourquoi la sous-requête scalaire de la question 18 ne suffit-elle plus ici ?

Question 23 *(deux jointures triples)*. Les dix académies où la proportion d'élèves de terminale inscrits en voie générale (série `G`) est la plus élevée à la rentrée 2025 (nom, proportion).

Question 24 *(table d'association : traduire un « et »)*. Les lycées de l'académie de Versailles proposant, en terminale à la rentrée 2025, à la fois « Numérique et sciences informatiques » et « Arts plastiques » (patronyme du lycée, nom de la commune). Pourquoi la condition `specialite = 'Numérique et sciences informatiques' AND specialite = 'Arts plastiques'` renvoie-t-elle une table vide ?

4. Ensembles et antijointure

Question 25 *(`INTERSECT`)*. Les codes INSEE des communes des Yvelines ayant au moins un lycée à la rentrée 2025.

Question 26 *(`EXCEPT` et antijointure)*.

1. Combien de communes de plus de 10 000 habitants n'ont aucun lycée à la rentrée 2025 ?
2. Donner le nom, le département et la population des dix plus peuplées d'entre elles.

Question 27 *(distance)*. La commune des Yvelines, autre que Versailles, dont le centre est le plus proche de celui de Versailles (au sens de la distance euclidienne sur les coordonnées `latitude_centre`, `longitude_centre`).

5. Synthèse et compléments

Question 28 *(la requête complète)*. Pour chaque académie, le nombre de lycées publics de la rentrée 2025 situés dans une commune de moins de 20 000 habitants ; ne garder que les académies en comptant au moins vingt, par nombre décroissant de lycées.

Question 29 *(bonus, une table pour finir)*. Pour chaque rentrée, le nombre total d'étudiants en CPGE et la proportion de filles. Attention à la division…

Question 30 *(bonus, retour sur la question 15)*. Plusieurs communes peuvent porter le même nom. On s'intéresse ici aux couples de communes homonymes.

1. Combien y a-t-il de couples de communes distinctes portant le même nom ?
2. En donner les vingt premiers par ordre alphabétique (nom, et les deux départements).

6. Utilisation des données *(bonus)*

Cette partie ne sera pas traitée en TP. Elle montre ce que l'on fait des résultats d'une requête une fois revenu dans Python : le SQL fait le calcul, matplotlib le donne à voir.

Importation de matplotlib et de la fonction $\log_{10}$

```
import matplotlib.pyplot as plt
from math import log10
```

On écrit les requêtes en langage SQL comme précédemment, mais on utilise la fonction `db.liste` pour récupérer la liste des données demandées (sans le schéma de la table).

Création d'une carte

Question 31 *(du SQL au graphique)*

1. Récupérer dans une liste nommée `V` la longitude et la latitude du centre de chaque commune de France métropolitaine, ainsi que sa densité (hab/km²).
2. Stocker dans `x` la liste des longitudes, dans `y` celle des latitudes et dans `L_d` les images des densités par $t \mapsto \log_{10}(t+1)$.
3. À l'aide d'une requête SQL, récupérer dans `D` la densité maximale des communes de France métropolitaine.

Une fois `x`, `y`, `L_d` et `D` calculés, exécuter la cellule suivante pour tracer la carte.

```
fig, ax = plt.subplots()

# Tracé
points = ax.scatter(x, y, c=L_d, cmap="YlOrRd", vmin=0, vmax=log10(D+1),␣
 ↪marker="s", s=0.75)

# Création de la légende
cbar = fig.colorbar(points, label="Densité (hab/km²)")

# Graduations entières de 0 à la valeur maximale possible
ticks = list(range(0, int(log10(D+1)) + 1))

# Étiquettes en puissances de 10 correspondantes
labels = [f"$10^{k}$" for k in ticks]

cbar.set_ticks(ticks)
cbar.set_ticklabels(labels)
```

```
ax.set_axis_off()
```

4. Récupérer dans une liste nommée `G` les centres de gravité démographiques (barycentres des centres des communes, pondérés par leur population) des départements de France métropolitaine (`dep_code` strictement inférieur à `'96'` : c'est une chaîne de caractères).

Une fois `G` calculée, extraire les longitudes dans `xG` et les latitudes dans `yG`, puis exécuter la cellule suivante pour ajouter ces points à la carte.

```
ax.scatter(xG, yG, marker="s", s=0.75,c='k')
fig
```