TP2 : manipulation relationnelle (SQL)

UCBL - Base de données pour la bioinformatique - 2024 / 2025

Objectif du TP : écrire des requêtes SQL sur une BD relationnelle

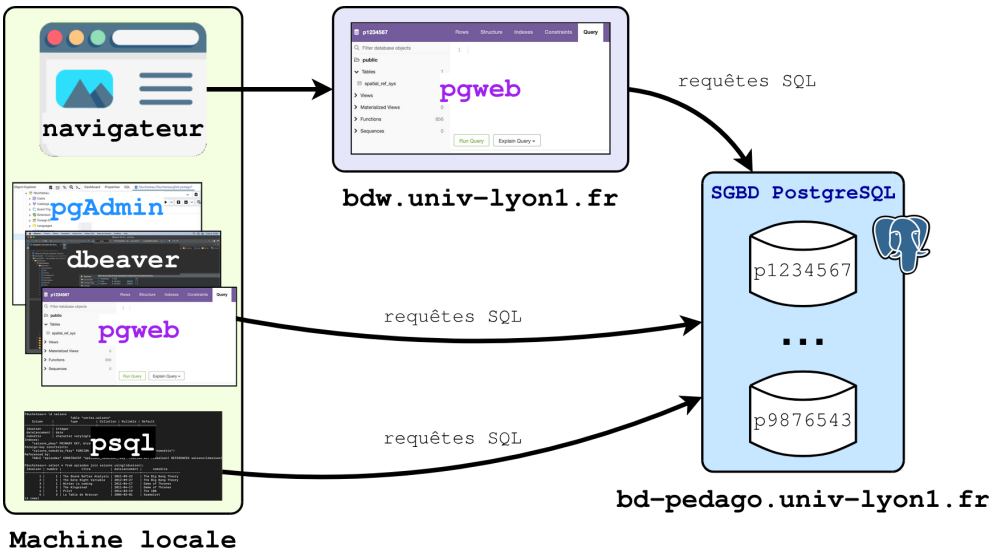
1 Informations de connexion au SGBD

Chaque étudiant·e dispose d'un compte sur le SGBD PostgreSQL `bd-pedago.univ-lyon1.fr`[1]. Quelque soit l'outil utilisé pour interagir avec ce SGBD, vous avez besoin des informations suivantes pour vous connecter au SGBD avec votre compte :

- **Serveur :** `bd-pedago.univ-lyon1.fr`
- **Utilisateur :** `p1234567` (à remplacer par votre numéro étudiant)
- **Mot de passe :** case `mdp_bdbio` sur Tomuss (ce n'est **pas** votre mot de passe UCBL !)
- **Base de données :** `p1234567` (idem que le nom d'utilisateur)

2 Interagir avec le SGBD (requêtes SQL)

Vous avez plusieurs options pour vous connecter au serveur `bd-pedago` et l'utiliser.

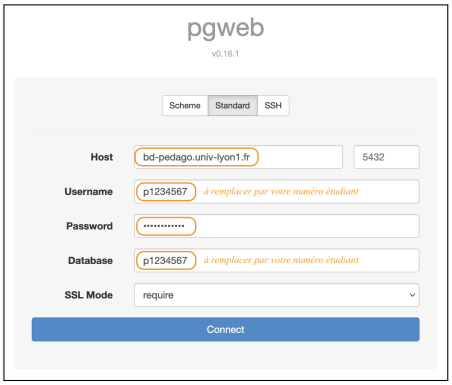


- Option facile : utiliser l'outil **pgweb** installé sur https://bdw.univ-lyon1.fr/.

  Vous devez simplement remplir les informations de connexion dans le formulaire.

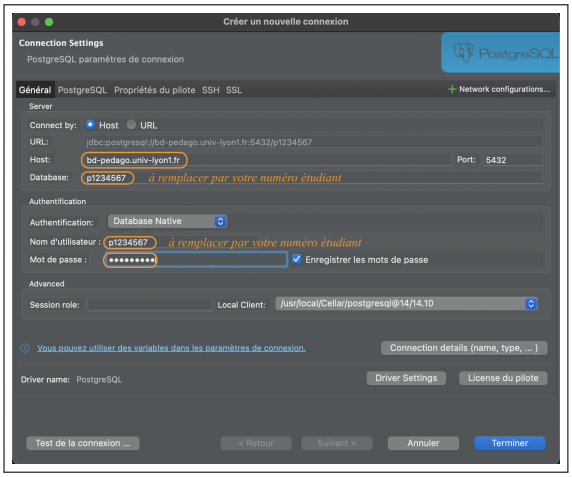
  Un rappel sur la commande SQL pour changer de répertoire schéma (ici pour utiliser le schéma nommé *tp1*, à adapter selon le nom de votre schéma) :

```sql
SET SEARCH_PATH TO tp1;
```

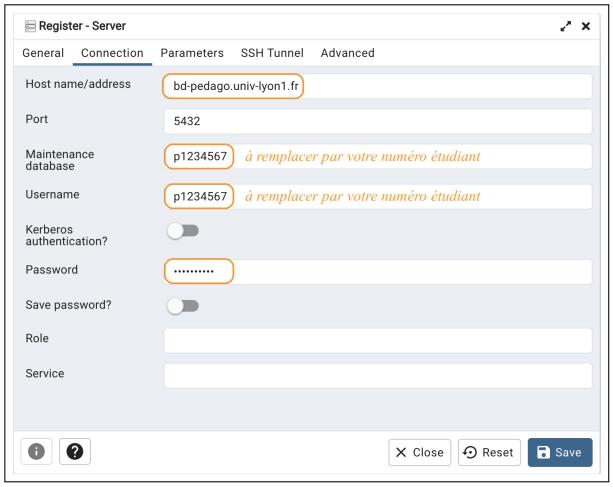


[1] Plus de détails sur la documentation du serveur bd-pedago.

- Option ”exécution locale” : lancer manuellement un outil graphique comme DBeaver, pgAdmin ou pgweb. Il est probable que vous deviez **installer l'outil** choisi.



Interface DBeaver : créer une nouvelle connexion PostgreSQL



Interface pgAdmin : créer une nouvelle connexion avec `Object>Register>Server`

- Option ”j'aime la ligne de commande” : installer (si besoin) et lancer psql, l'outil officiel de PostgreSQL en ligne de commande.

```
psql -h bd-pedago.univ-lyon1.fr -U p1234567 -d
p1234567 --password
```

```
$ psql -h bd-pedago.univ-lyon1.fr -U fduchateau -d fduchateau --password
Password:
psql (16.3, server 14.5 (Ubuntu 14.5-1.pgdg20.04+1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, compression: off)
Type "help" for help.

fduchateau=>
```

3 Création du jeu de données

Commencez par télécharger le script PostgreSQL.
Le jeu de données représente les concepts de gènes, exons et transcripts utilisés en bio-informatique. Il provient à l'origine d'un fichier au format GenBank[2], qui a été converti en SQL.

Copiez-coller le script SQL et exécutez-le dans l'éditeur SQL de votre outil, ou directement au niveau du *prompt* de psql[3]. Vous devez obtenir 152 gènes, 39 exons et 95 transcripts.

Schéma relationnel de la base de données :

Gènes	(gene_id, *start*, *stop*, *strand*, *gene*, *GeneID*, *locus_tag*, *gene_synonym*)
Exons	(exon_id, *start*, *stop*, *strand*, *number*, *#gene*, *product*, *locus_tag*)
Transcripts	(CDS_id, *start*, *stop*, *strand*, *codon_start*, *protein_id*, *product*, *#gene*, *translation*, *UniProtKBSwissProt*, *InterPro*, *GOA*, *GI*, *PDB*, *GeneID*, *locus_tag*, *note*, *UniProtKBTrEMBL*, *function*)

Questions sur la modélisation :

1. Quel problème y a t-il dans cette base de données, en particulier au niveau des clés étrangères ?
2. Cette base de données est-elle en troisième forme normale ?

4 Exécution de requêtes SQL

Pour écrire des requêtes SQL, vous devez lancer l'éditeur SQL de l'outil, ou directement dans la console sur psql. Il est conseillé d'écrire les requêtes dans un fichier texte (sauvegarde) et de les copier-coller dans l'outil.
Documentations : SQL sur PostgreSQL, DBeaver, pgAdmin, SQL.sh.

[2] Génome EBV de GenBank, http://www.ncbi.nlm.nih.gov/nuccore/NC_007605.1
[3] Sur psql, ne pas oublier le point-virgule en fin de requête. Tapez \? pour la liste des commandes.

Traduire les requêtes suivantes en SQL :

1. Nom des protéines de la table transcripts, que l'on obtient avec la requête `SELECT product FROM Transcripts`. Pourquoi obtient-on des doublons ? Modifiez la requête pour obtenir 84 tuples.
2. Informations sur les gènes. Les attributs *start, stop et strand* seront concaténés et séparés par des virgules au sein d'un seul attribut nommé *coords* (152 tuples résultat).
3. Informations sur les gènes qui possèdent un locus_tag. Le résultat sera ordonné par nom de gène décroissant (45 tuples résultat)
4. Identifiant des transcripts qui codent pour une protéine dont l'existence est supposée, i.e. contenant *hypothetical*. L'identifiant (le gene_id de GENES) et le nom du gène associés à ce transcript seront également donnés (65 tuples résultat)
5. Identifiant des transcripts dont le locus_tag est celui d'un gène (45 tuples résultat)
6. Nom des gènes dont le locus_tag contient un P ou qui sont associés à un exon codant pour une protéine (5 tuples résultat)
7. Nombre de transcripts appartenant au même gène. Les résultats seront ordonnés par nombre décroissant (85 tuples résultat)
8. Noms des gènes qui ont au moins deux *geneid* différents. On affichera aussi ce nombre de *geneid* (34 tuples résultat)
9. Longueurs minimale et maximale des translations, mais uniquement pour les transcripts dont le gène est également associé à un exon (1 tuple résultat : 378 et 944)
10. Identifiant et locus_tag des paires de transcripts qui possèdent le même complément / *strand* et dont les locus_tags sont celui d'un gène commençant par *BALF* (1 tuple résultat)