PostgrSQL

Plan de séance : Premier contact avec PostgreSQL.

Prérequis :

Connaitre nodeJS et Express dans un usage simple.

Connaitre l'éxistence de la ligne de commande et ne pas avoir peur de l'utiliser.

Objectif général :

Le but est de comprendre comment créer une base de données très simple sur PostgreSQL pour ensuite l'utiliser dans un serveur nodeJS.

L'apprenant doit être capable de connecter une base de données qu'il aura créer lui-même à un serveur NodeJS.

Objectifs intermédiaires :

L'apprenant doit être capable de réaliser des actions en ligne de commande avec psql.

L'apprenant doit être capable de naviguer dans PgAdmin4.

L'apprenant doit être capable d'installer le module pg dans nodeJS.

Objectifs bonus :

L'apprenant doit être capable de reconnaitre une requête/ordre SQL et d'aller chercher l'information à son sujet.

L'apprenant doit être capable de récupérer des informations en base de données dans un serveur NodeJS.

L'apprennant doit être capable d'ancitiper les erreurs liées à la connection pour maintenir le serveur en état de fonctionnement.

Situation d'apprentissage :

L'apprenant montre comment réaliser l'action. Puis il propose aux apprenants de réaliser une action similaire, avec soutient de l'apprenant.

Pour la ligne de commandes / l'interface graphique :

Créer une base de données. Créer une table avec quelques colonnes.

Pour le serveur nodeJS.

Connecter cette base de données au serveur. Récupérer de l'information avec un SELECT * FROM table_name;

Support pédagogique :

Ce document Markdown.

Temps :

Total : 2h

Présentation de PostgreSQL + ligne de commande : 30 min.

Présentation de PgAdmin : 10min;

Exercice pratique : 30min.

Présentation de la connection avec le module PG : 20 min

Exercice pratique : 30min.

J1

Installer postgreSQL

Sur le site de postgreSQL

PostgreSQL: Downloads

Sur le site EnterpriseDB

C'est quoi PostgreSQL ?

Un système de gestion de bases de données relationnelles et objet.

Créer en 1996

Géré par le PostgreSQL Global Development Group (PGDG)

Depuis 2017, les versions majeures sont numérotées : la version 10.

Nous en sommes à la version 15.

Chaque version majeure est maintenu pendant 5 ans avec des versions mineures.

Donc en ce moment, 5 versions majeures sont utilisables sans risque de sécurité : 15,14,13,12 et 11.

Une app utilisant une version plus ancienne devrait passer à une nouvelle :)

PostgreSQL est un logiciel libre et ouvert, dans son usage mais aussi dans son développement : le code est en ligne et chacun peut y participer.

Voir : https://commitfest.postgresql.org/

Se connecter a une base de données PostgreSQL.

Avec pgadmin 4

C'est un outil avec interface graphique. Cela pourra vous aider si vous avez du mal avec la ligne de commande :) !

Avec psql, l'outil en ligne de commande.

Pour vérifier que l'outil est dispo (installé de base avec PostgreSQL)

```
psql --version
```

Pour se connecter avec le super utilisateur postgres

```
psql -U postgres
```

-U est le drapeau pour se connecter avec un utilisateur spécifique (pour User)

Un mot de passe pour cet utilisateur sera demandé.

Ici, comme une DB et un utilisateur ont le même nom, il suffit de donner le nom de l'utilisateur pour que psql sache à quelle DB se connecter.

Quelques commandes de base :

Attention : il faut distinguer les commandes psql des "ordres" SQL, qui respecte la structure du langage SQL. (Structured Query Language).

La structure d'un ordre/requête SQL :

Sur les objets :

- `CREATE` t n pour créer un objet de type t et de nom n
- `ALTER` t n pour modifier le schéma d'un objet de type t et de nom n
- `DROP` t n pour supprimer un objet de type t et de nom n

les objets peuvent être une DATABASE, un SCHEMA, une TABLE.

Sur les tables :

- `INSERT` pour ajouter des lignes à une table
- `REPLACE` pour modifier des lignes d'une table
- `DELETE` pour supprimer des lignes d'une table
- `SELECT` pour extraire des données à partir de tables existantes

Apprendre les bases de données et SQL

Vous pouvez créer une base de données ainsi :

```
CREATE DATABASE nom_de_la_bdd;
```

Par exemple :

```
CREATE DATABASE books_db;
```

Pour avoir la liste de toutes les bases de données de l'utilisateur.

```
\l
```

Avec plus d'infos :

```
\l+
```

Pour supprimer une base de données (attention, irréversible !)

Le IF EXISTS n'est pas obligatoire mais ça évite les bugs si vous faites une erreur dans le nom.

```
DROP DATABASE IF EXISTS nom_de_la_bdd
```

Comment créer une table dans la base de données où l'on se situe.

```
CREATE TABLE table_name();
```

Voici des exemples :

```
CREATE TABLE authors(
    author_id SERIAL PRIMARY KEY,
    first_name VARCHAR(100) NOT NULL,
    last_name VARCHAR(100) NOT NULL
);

CREATE TABLE books(
    book_id           SERIAL PRIMARY KEY,
    title             VARCHAR(100) NOT NULL,
    published_year    INT
);
```

On peut ajouter une relation a ces deux tables : c'est l'un des caractéristiques des bases de données relationnelles.

```
ALTER TABLE books
    ADD COLUMN author_id INT REFERENCES authors(author_id);
```

Une fois que l'on a créer les tables, on peut injecter des données dedans !

```
INSERT INTO authors (first_name, last_name)
VALUES (‘Tamsyn’, ‘Muir’), (‘Ann’, ‘Leckie’), (‘Zen’, ‘Cho’);
```

```
INSERT INTO books(title, published_year, author_id)
VALUES ('Gideon the Ninth', 2019, 1), ('Ancillary Justice', 2013, 2), ('Black Water
Sister', 2021, 3);
```

dans psql, pour voir toutes les tables de la base de données : il faut utiliser la commande display tables

```
\dt
```

Pour voir précisement une table, il faut utiliser la commande display avec le nom de la table

```
\d authors
```

Une fois qu'on a vu comment c'est galère de taper les commandes SQL dans l'invite de commande, on peut utiliser la commande editor

```
\e
```

Qui permet d'ouvrir un bloc note et d'écrire son code dedans.

Sites intéréssants :

https://sql.sh/ Apprendre les bases de données et SQL