ROW LEVEL SECURITY POLICIES

Filtrage par ligne

appliqué à des applications géospatiales

Régis Haubourg - Oslandia

OSLANDIA

PME Française crée en 2009

Haute technologie

Open source100% télétravail

OSLANDIA

15 collaborateurs

100% télétravail

COMMUNAUTÉS OPEN SOURCE & LIBRES

QGIS

PostGIS

Giro3D-iTowns (pointcloud / imagerie / 3D immersive)

python, SQL, C++, javascript

Stack OSGeo

LE BESOIN MÉTIER

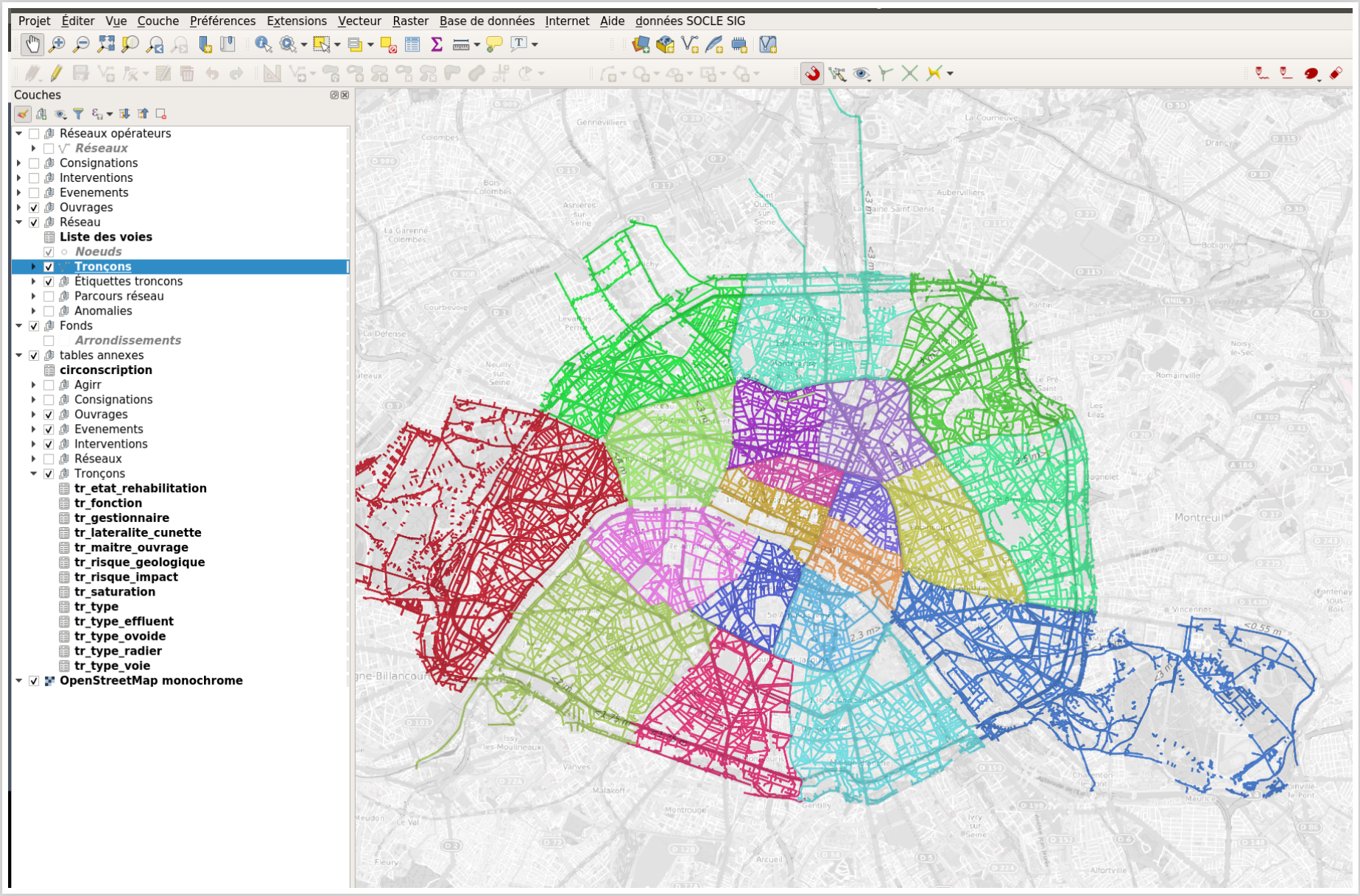
FILTRER DES LIGNES

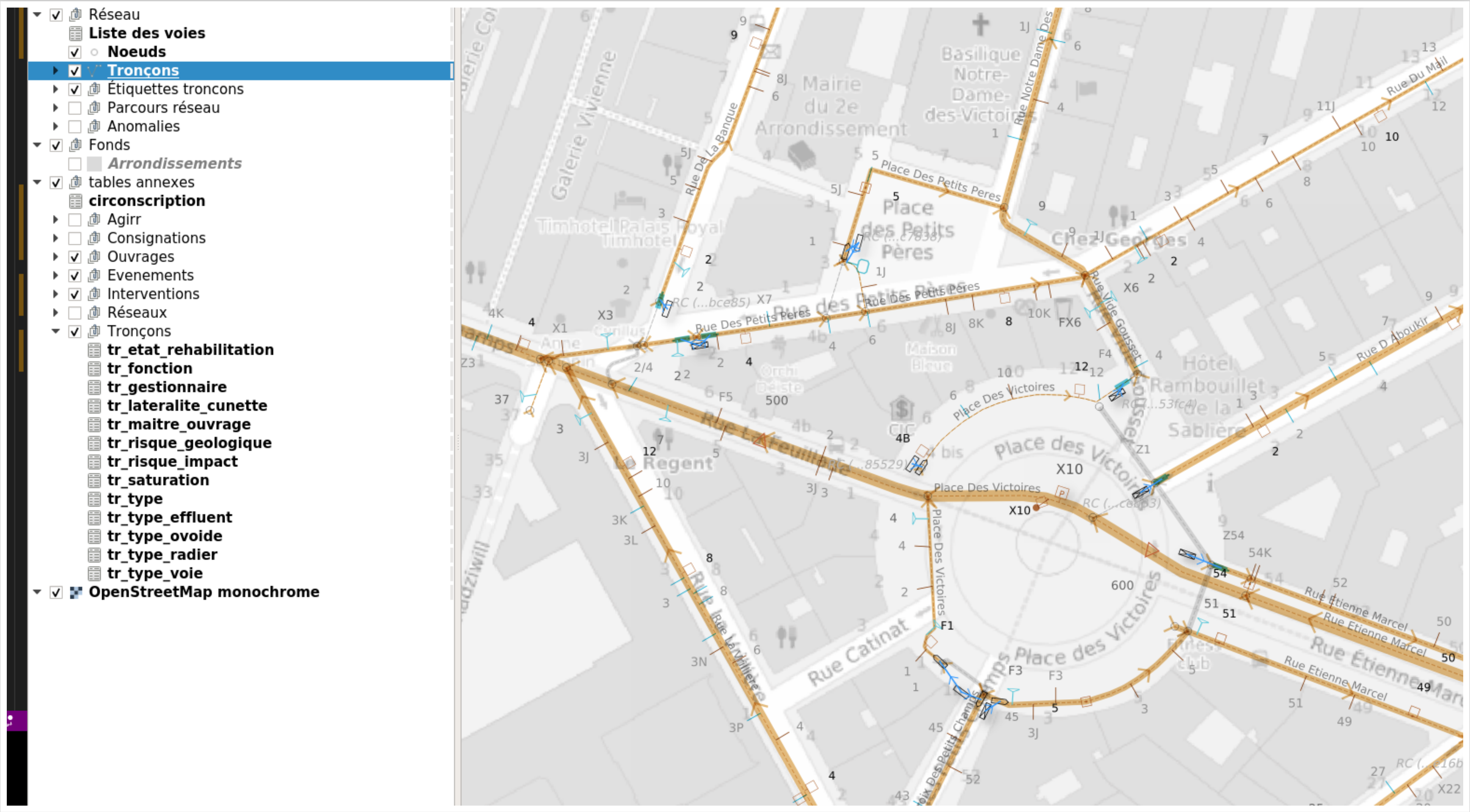
GESTION DE RÉSEAU D'EAU, CADASTRE, ROUTE

Ex:

Un utilisateur ne doit pouvoir éditer que les données de son territoire.

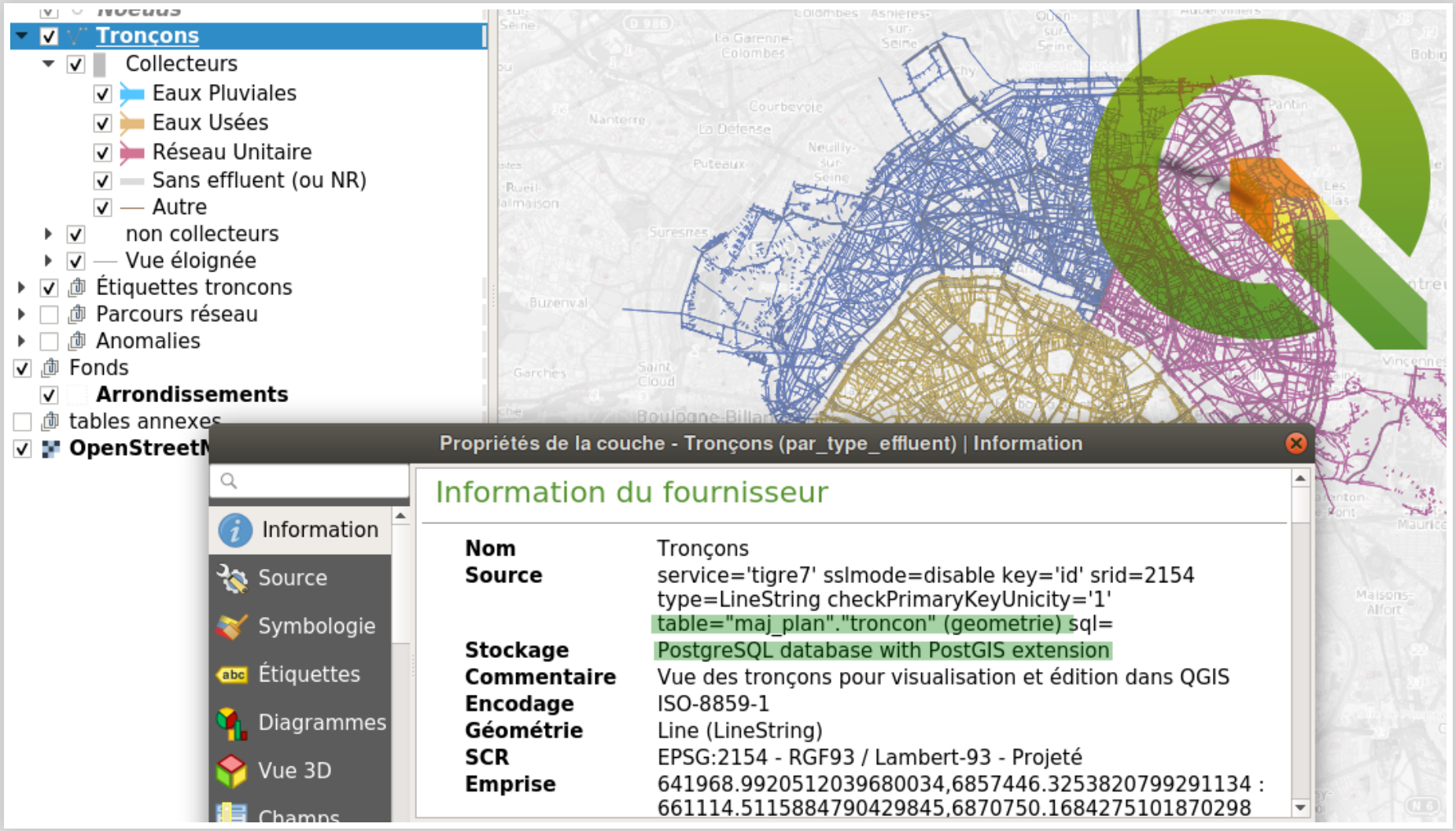
Un administrateur d'un syndicat intercommunal doit pouvoir accéder à toutes ses communes, mais pas les autres


Projet Éditer Vue Couche Préférences Extensions Vecteur Raster Base de données Internet Aide données SOCLE SIG
Couches
Réseaux opérateurs
Réseaux
Consignations
Interventions
Evenements
Ouvrages
Réseau
Liste des voies
Noeuds
Tronçons
Étiquettes troncons
Parcours réseau
Anomalies
Fonds
Arrondissements
tables annexes
circonscription
Agirr
Consignations
Ouvrages
Evenements
Interventions
Réseaux
Tronçons
tr_etat_rehabilitation
tr_fonction
tr_gestionnaire
tr_lateralite_cunette
tr_maitre_ouvrage
tr_risque_geologique
tr_risque_impact
tr_saturation
tr_type
tr_type_effluent
tr_type_ovoide
tr_type_radier
tr_type_voie
OpenStreetMap monochrome

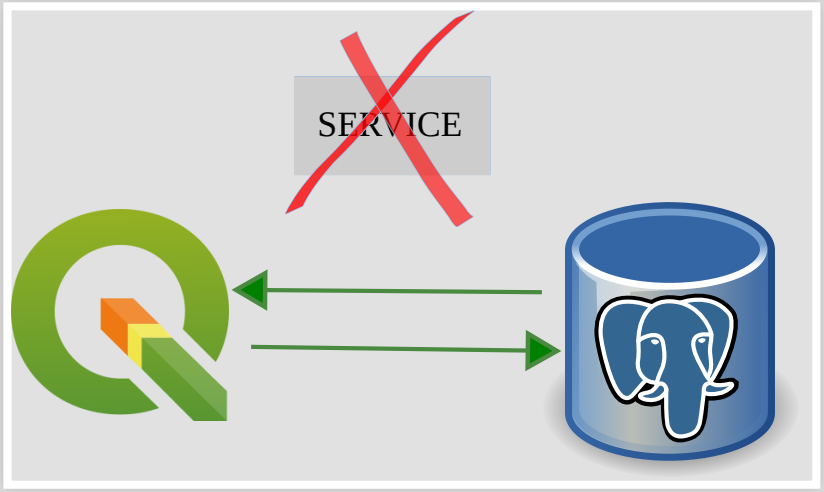

Réseau
Liste des voies
Noeuds
Tronçons
Étiquettes troncons
Parcours réseau
Anomalies
Fonds
Arrondissements
tables annexes
circonscription
Agirr
Consignations
Ouvrages
Evenements
Interventions
Réseaux
Tronçons
tr_etat_rehabilitation
tr_fonction
tr_gestionnaire
tr_lateralite_cunette
tr_maitre_ouvrage
tr_risque_geologique
tr_risque_impact
tr_saturation
tr_type
tr_type_effluent
tr_type_ovoide
tr_type_radier
tr_type_voie
OpenStreetMap monochrome
Mairie du 2e Arrondissement
Basilique Notre-Dame-des-Victoires
Place des Petits Pères
Place Des Petits Peres
Rue Des Petits Peres
Rue De La Banque
Rue Notre Dame Des
Rue Du Mail
Rue D Aboukir
Rue Vide Gousset
Place Des Victoires
Place des Victoires
Rue Catinat
Rue Etienne Marcel
Rue Etienne Marcel
Hôtel Rambouillet de la Sablière
Galerie Vivienne
Le Regent

DES OUTILS DESKTOP

- QGIS, le client de référence PostgreSQL - PostGIS
- **Accès direct aux couches PG**


Tronçons
Collecteurs
Eaux Pluviales
Eaux Usées
Réseau Unitaire
Sans effluent (ou NR)
Autre
non collecteurs
Vue éloignée
Étiquettes troncons
Parcours réseau
Anomalies
Fonds
Arrondissements
tables annexes
Propriétés de la couche - Tronçons (par_type_effluent) | Information
Information
Source
Symbologie
Étiquettes
Diagrammes
Vue 3D
Information du fournisseur
Nom Tronçons
Source service='tigre7' sslmode=disable key='id' srid=2154 type=LineString checkPrimaryKeyUnicity='1' table="maj_plan"."troncon" (geometrie) sql=
Stockage PostgreSQL database with PostGIS extension
Commentaire Vue des tronçons pour visualisation et édition dans QGIS
Encodage ISO-8859-1
Géométrie Line (LineString)
SCR EPSG:2154 - RGF93 / Lambert-93 - Projeté
Emprise 641968.9920512039680034,6857446.3253820799291134 : 661114.5115884790429845,6870750.1684275101870298

Gestion de droit pilotée par la base de données



-

LES DROITS POSTGRESQL CLASSIQUES

- Table
- Vue
- Colonne

Nécessaire pour filtrer les droits de base

Admin / Édition / consultation

INSUFFISANT

FILTRER LES LIGNES EN FONCTION DE L'UTILISATEUR COURANT

Variable dynamique nécessaire

```
[local] regis@qwat= > SELECT current_user, session_user;
 current_user | session_user
--------------+--------------
 yann         | regis
(1 ligne)
```

`session_user` : utilisateur de la session qui a servi à la connexion

`current_user` : l'utilisateur courant ou effectif

```
[local] postgres@qwat= > SET session authorization postgres;
SET
Temps : 0,730 ms
[local] postgres@qwat= # SET ROLE yann;
SET
Temps : 0,862 ms
[local] postgres@qwat= > SELECT current_user, session_user
;
 current_user | session_user
--------------+--------------
 yann         | postgres
(1 ligne)
```

LA MÉTHODE TRADITIONNELLE

GESTION DES DROITS APPLICATIVE

```
CREATE TABLE user (...)

CREATE TABLE user_group (...)

CREATE TABLE group  (...)

CREATE TABLE group_commune (...)
```

L'applicatif filtre les données

Mais toujours avec un seul role PG

SOLUTION HACKY AVEC DES VUES

`current_user` et `session_user` dans les filtres `WHERE` des vues

VERSION SIMPLIFIÉE

```
CREATE VIEW my_user AS

SELECT *

FROM "user"

WHERE

   "user"."user" = current_user; -- or current_session

--
```

EXEMPLE PÉDAGOGIQUE

on ne travaille jamais avec des utilisateurs avec droit de connexion

Et on utilise de groupes

MODÈLE DE DONNÉES AVEC TERRITOIRE / GROUPES

```
SELECT *
FROM communes
WHERE
  id_commune IN
  ( SELECT id_commune
    FROM group_zone gz JOIN user_group ug
    ON ug.group_id = gz.fk_group_id

    -- here we filter:
    WHERE gz.user_id = current_session )
```

note : on peut éviter l'utilisation d'une table explicite d'utilisateurs avec des requêtes système

LIMITES

- Nécessite de modifier les vues en place (MIGRATION)
- Pas sécure (security_barrier)
- Pas désactivable

RLS - ROW LEVEL SECURITY POLICIES

La belle manière de faire.

Politiques de sécurité niveau ligne

LA SYNTAXE

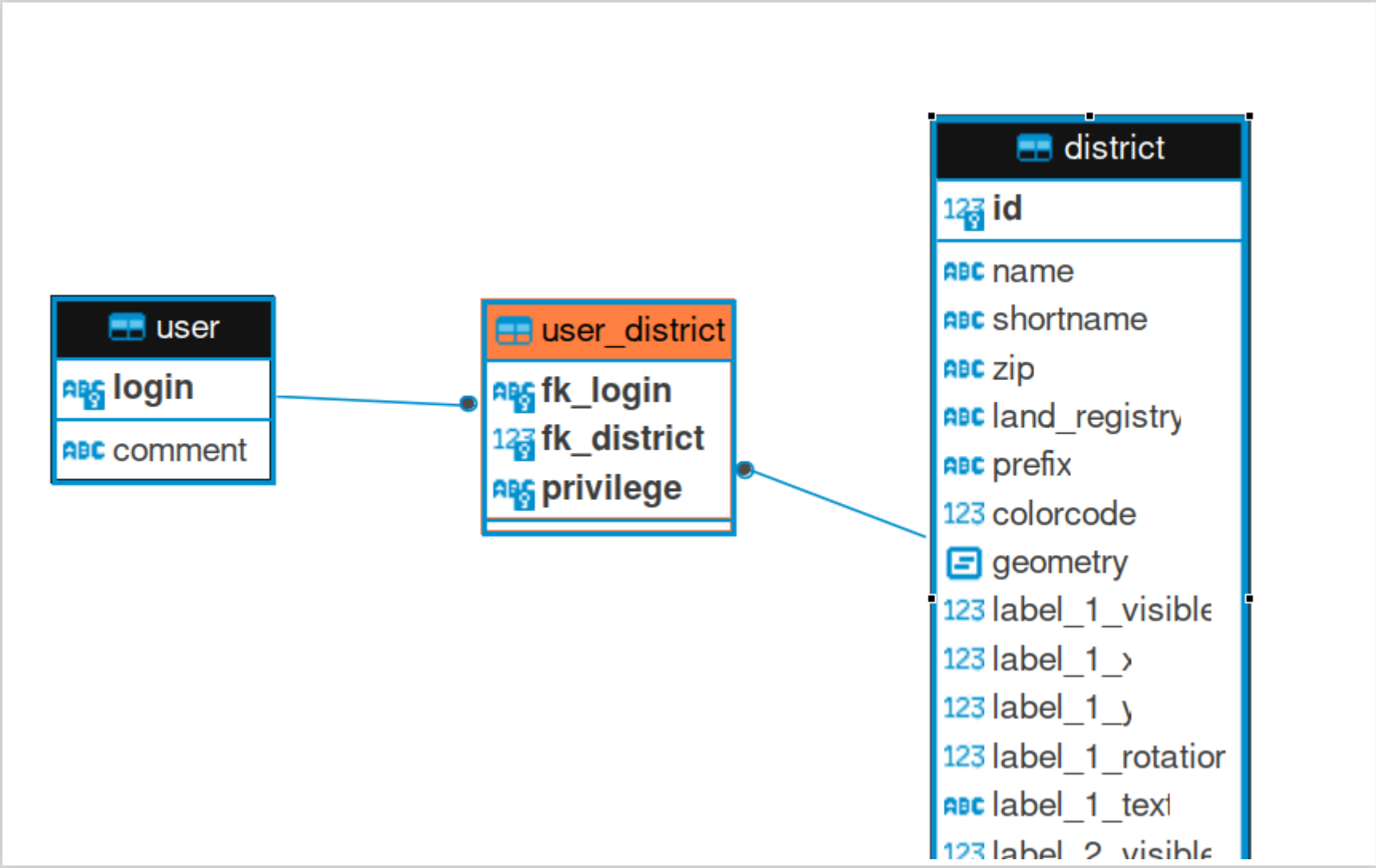
```
CREATE TABLE comptes (admin text, societe text, contact_email text);

-- activation des politiques de sécurité sur la table
ALTER TABLE comptes ENABLE ROW LEVEL SECURITY;

-- à ce stade, les utilisateurs ne voient plus les données

-- activation d'une politique

CREATE POLICY compte_admins ON comptes TO admins
    USING (admin = current_user);
```

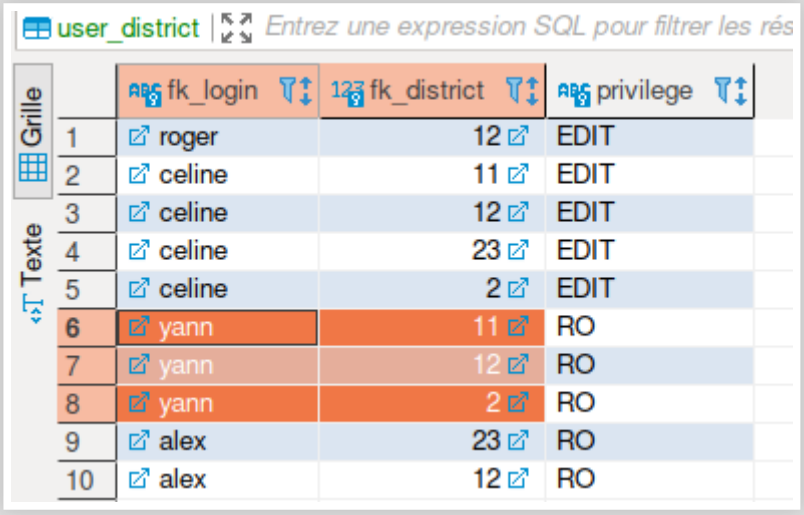
VARIATIONS

```
-- activation de deux politiques combinées

-- lecture pour tous
CREATE POLICY compte_admins ON comptes TO admins
 FOR SELECT
    USING (true);

-- modification uniquement pour ses propres données
-- Combinaison OR par défaut

CREATE POLICY user_mod_policy ON users
    -- critère de lecture
    USING (user_name = current_user)
    -- vérification des données à enregistrer
    WITH CHECK (user_name = current_user)
```

DÉMO

FILTRER DES TRONÇONS DE CANALISATION - APPLICATION QWAT


user
login
comment
user_district
fk_login
fk_district
privilege
district
id
name
shortname
zip
land_registry
prefix
colorcode
geometry
label_1_visible
label_1_x
label_1_y
label_1_rotatior
label_1_text


user_district | Entrez une expression SQL pour filtrer les rés

Grille | Texte

	fk_login	fk_district	privilege
1	roger	12	EDIT
2	celine	11	EDIT
3	celine	12	EDIT
4	celine	23	EDIT
5	celine	2	EDIT
6	yann	11	RO
7	yann	12	RO
8	yann	2	RO
9	alex	23	RO
10	alex	12	RO

psql /home/regis/OSLANDIA/oslandia_perso/presentations/pgsession_2020-11-17

```
[local] postgres@qwat= # SELECT current_user, session_user
;
 current_user | session_user
--------------+--------------
 postgres     | postgres
(1 ligne)

Temps : 0,877 ms
[local] postgres@qwat= # SELECT count(*) from pipe;
 count
-------
  4223
(1 ligne)

Temps : 8,694 ms
[local] postgres@qwat= #
```

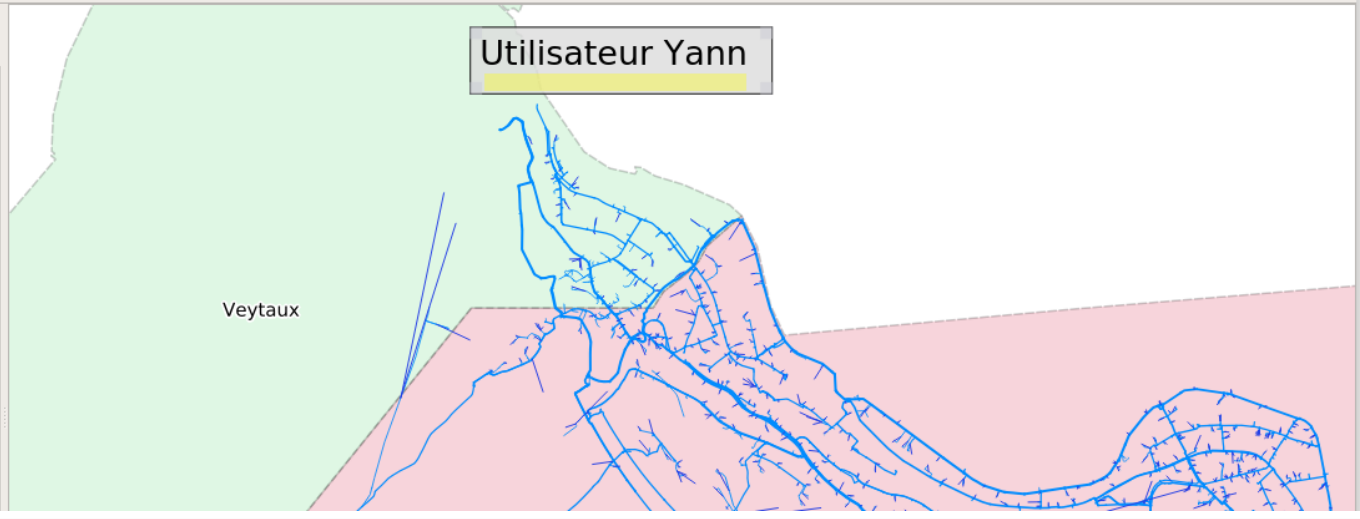
fish /home/regis/OSLANDIA/oslandia_perso/presentations/pgsession_2020-11-17 ×

psql /home/regis/OSLANDIA/oslandia_perso/presentations/pgsession_2020-11-17 ×

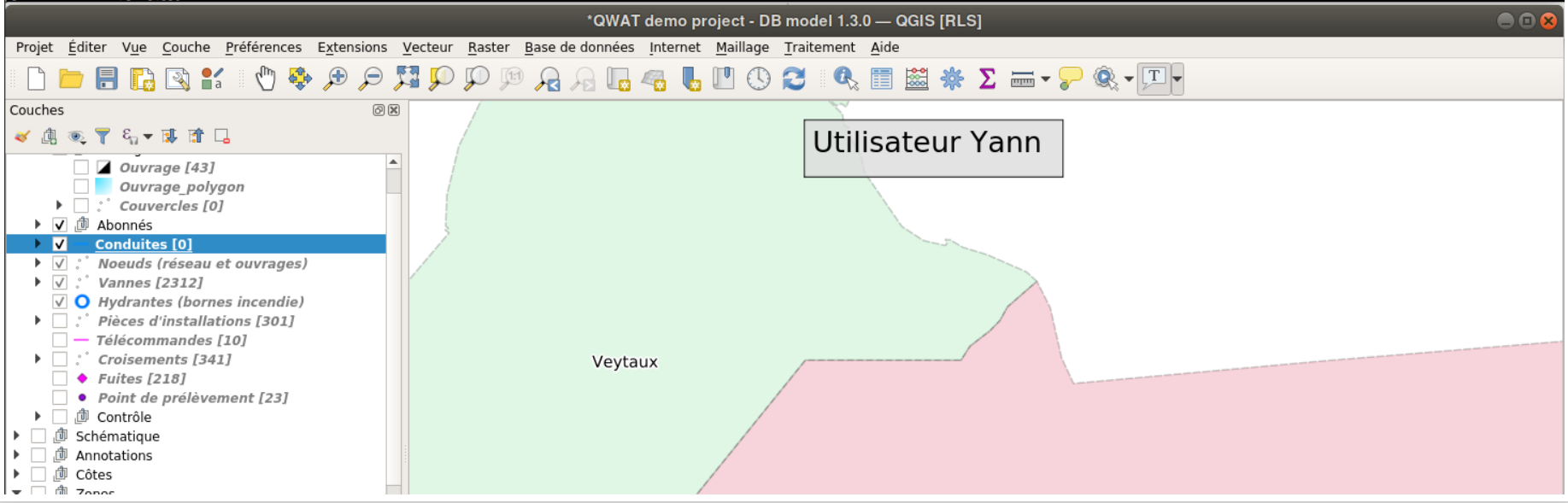
Projet Éditer Vue Couche Préférences Extensions Vecteur Raster Base de données Internet Maillage Traitement Aide

Couches

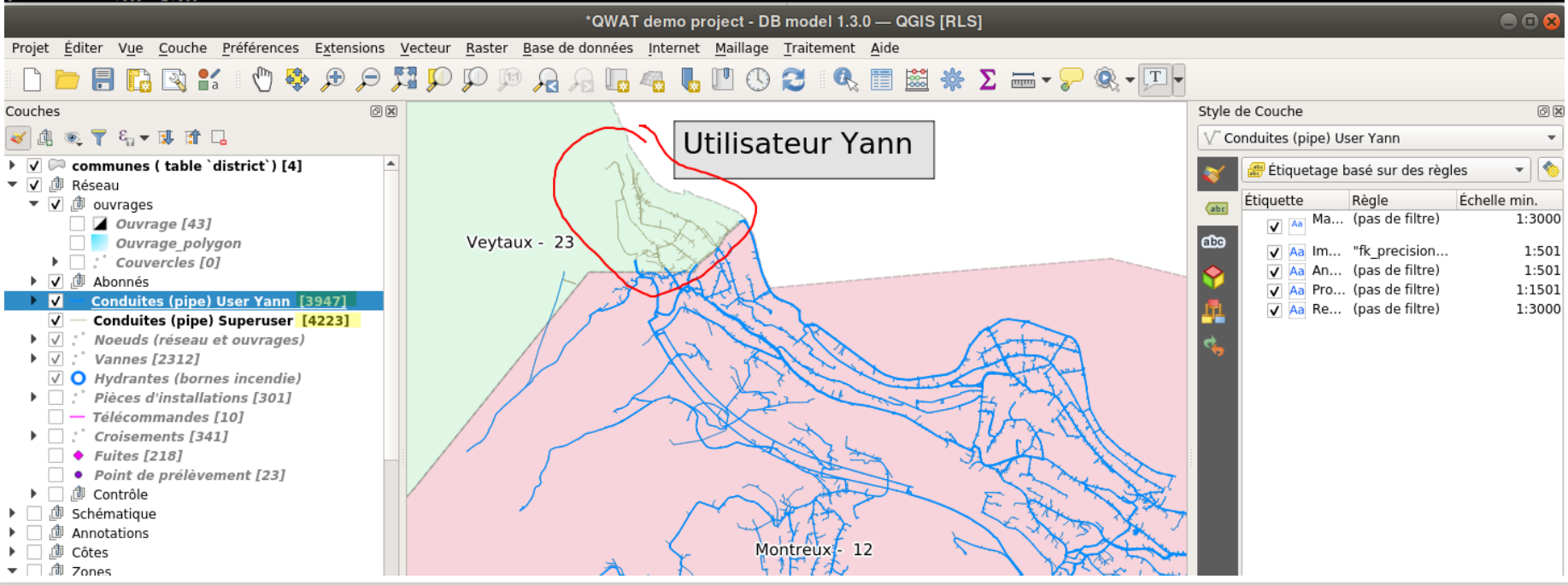
- communes (table `district`) [4]
- Réseau
  - ouvrages
    - Ouvrage [43]
    - Ouvrage_polygon
    - Couvercles [0]
  - Abonnés
  - Conduites [4223]
  - Noeuds (réseau et ouvrages)
  - Vannes [2312]
  - Hydrantes (bornes incendie)
  - Pièces d'installations [301]
  - Télécommandes [10]
  - Croisements [341]
  - Fuites [218]
  - Point de prélèvement [23]
  - Contrôle
- Schématique
- Annotations
- Côtes



```
Temps : 0,742 ms
[local] postgres@qwat= # SELECT count(*) from pipe;
 count
-------
  4223
(1 ligne)

Temps : 5,046 ms
[local] postgres@qwat= # ALTER TABLE qwat_od.pipe ENABLE ROW LEVEL SECURITY;
ALTER TABLE
Temps : 4,767 ms
[local] postgres@qwat= # SET ROLE yann; SELECT count(*) from pipe; RESET ROLE;
SET
Temps : 0,626 ms
 count
-------
     0
(1 ligne)
```

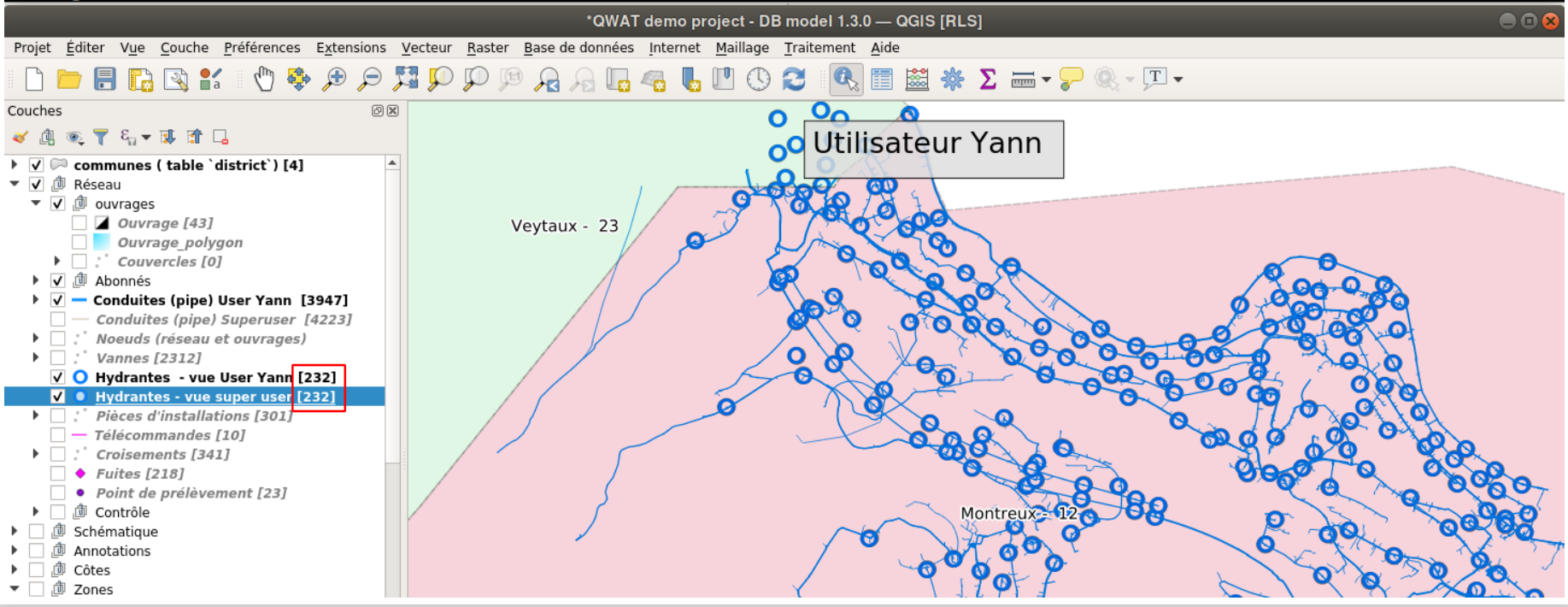


```
[qwat] # CREATE POLICY district_filter_pipe ON qwat_od.pipe
AS PERMISSIVE
FOR ALL  USING ( pipe.fk_district =
( SELECT distinct fk_district  FROM qwat_od.user_district ud
 WHERE fk_login = current_user and ud.fk_district = pipe.fk_district ));
CREATE POLICY
Temps : 6,349 ms
[qwat] #
[qwat] # SET ROLE yann; SELECT count(*) from pipe; RESET ROLE;
SET
Temps : 0,740 ms
 count
-------
  3947
(1 ligne)
```



FILTRAGE SUR UNE AUTRE TABLE - BORNES INCENDIES

```
[qwat] # SET ROLE yann; SELECT count(*) from vw_element_hydrant; RESET ROLE;
SET
 count
-------
   232
(1 ligne)

RESET
[qwat] # CREATE POLICY district_filter_node ON qwat_od.node  AS PERMISSIVE FOR ALL
USING ( node.fk_district = ( SELECT distinct fk_district
FROM qwat_od.user_district ud
WHERE ud.fk_login = current_user and ud.fk_district = node.fk_district ))
;
CREATE POLICY
[qwat] # SET ROLE yann; SELECT count(*) from vw_element_hydrant; RESET ROLE;
SET
 count
-------
   232
(1 ligne)
```



WTF ????

Compte identique avec ou sans RLS

C'EST UNE VUE !

```
CREATE OR REPLACE VIEW qwat_od.vw_element_installation
```

24

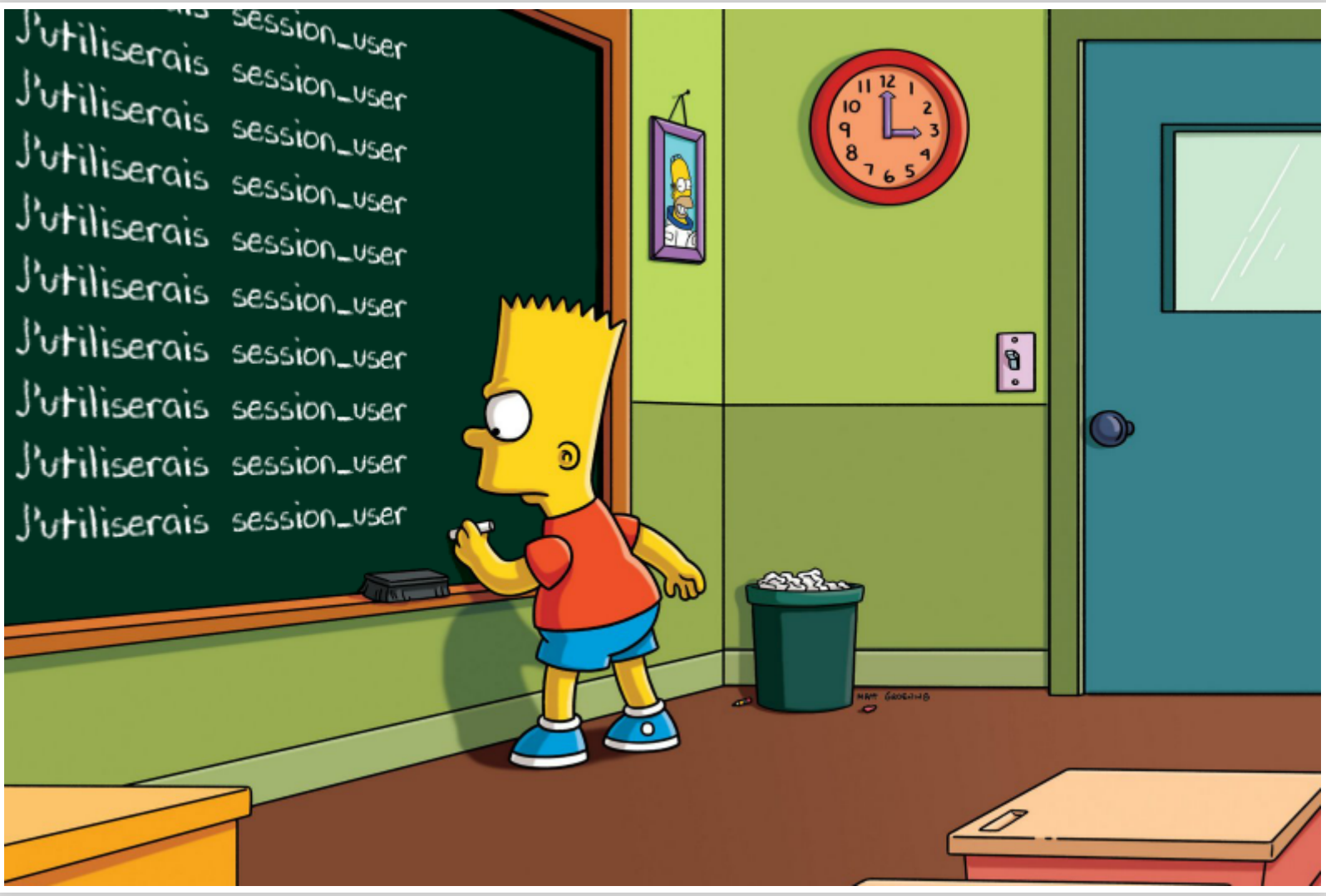
Basically because it wasn't possible to retroactively change how views work. I'd like to be able to support `SECURITY INVOKER` (or equivalent) for views but as far as I know no such feature presently exists.

You can filter access to the view its self with row security normally.

The tables accessed by the view will also have their row security rules applied. However, they'll see the `current_user` as the *view creator* because views access tables (and other views) with the rights of the user who created/owns the view.

Maybe it'd be worth raising this on pgsql-hackers if you're willing to step in and help with development of the feature you need, or pgsql-general otherwise?

That said, while views access tables as the creating user and change `current_user` accordingly, they don't prevent you from using custom GUCs, the `session_user`, or other contextual information in row security policies. You can use row security with views, just not (usefully) to filter based on `current_user`.


J'utiliserais session_user
J'utiliserais session_user
J'utiliserais session_user
J'utiliserais session_user
J'utiliserais session_user
J'utiliserais session_user
J'utiliserais session_user
J'utiliserais session_user
J'utiliserais session_user
J'utiliserais session_user

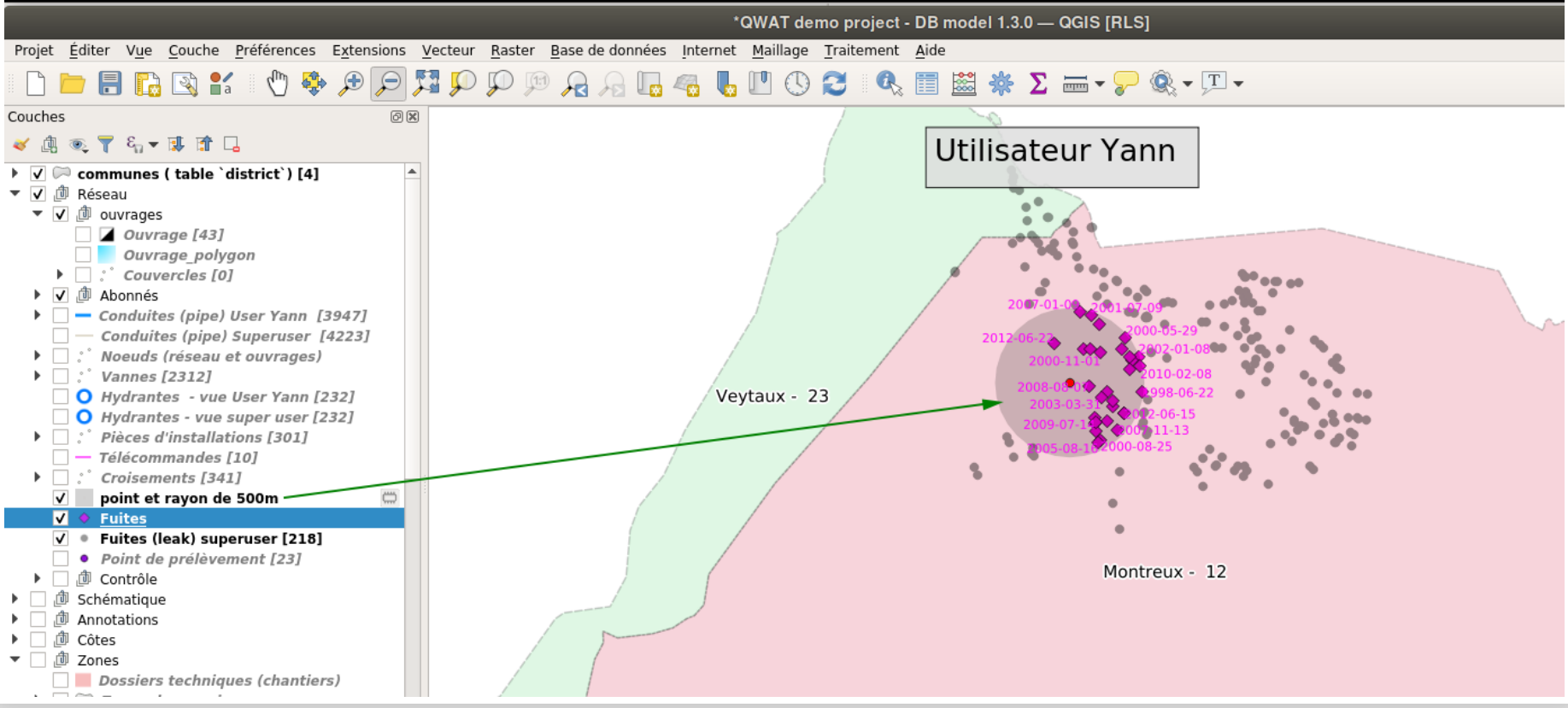
DROITS MIXTES

	fk_login	fk_district	privilege
1	roger	12	EDIT
2	celine	11	EDIT
3	celine	12	EDIT
4	celine	23	EDIT
5	celine	2	EDIT
6	yann	11	RO
7	yann	12	RO
8	yann	2	RO
9	alex	23	RO
10	alex	12	RO

```
CREATE POLICY district_filter_edit ON qwat_od.district  for UPDATE  to public
USING (id=
        ( SELECT distinct fk_district  FROM qwat_od.user_district ud
          WHERE fk_login = session_user and ud.fk_district = district.id
          and privilege = 'EDIT' ))
WITH CHECK (id=
        ( SELECT distinct fk_district  FROM qwat_od.user_district ud
          WHERE fk_login = session_user and ud.fk_district = district.id
          and privilege = 'EDIT' ));
    ;

CREATE POLICY district_filter_insert ON qwat_od.district  for INSERT  to public
WITH CHECK (id=
        ( SELECT distinct fk_district  FROM qwat_od.user_district ud
          WHERE fk_login = session_user and ud.fk_district = district.id
          and privilege = 'EDIT' ));
    ;
```

FILTRAGE GÉOGRAPHIQUE POSTGIS

```
[qwat] # CREATE POLICY district_filter_select ON qwat_od.leak  FOR SELECT
USING ( st_dwithin(st_setsrid(st_makepoint(515156,157771), 21781), leak.geometry, 500))  ;
CREATE POLICY
[qwat] # SET ROLE yann; SELECT count(*) from leak; RESET ROLE;
SET
 count
-------
    31
(1 ligne)

RESET
[qwat] #
```



EN SYNTHÈSE

- Assez technique à rédiger
- Choix de règles AND (restrictif) / OR (permissif)
- Combinaison avec droits par colonne

KEEP. IT. SIMPLE. STUPID.

TESTEZ. TOUJOURS.

Ce sont des évaluations WHERE.

Avec impact sur la performance.

OPTIMISEZ VOS REQUÊTES ET INDEXES

Performances si PG < 10

(Scan séquentiel avant filtrage RLS)

ET AVEC DES FILTRES SPATIAUX ?

- performance : point dans polygone OK. KNN OK.
- Grandes géométries - requêtes coûteuses de recouvrement > **DANGER!**
- Erreur de topologie? : **Réponse VIDE et erreurs SILENCIEUSES** !

CRÉER UN ROLE DE BASE POUR CHAQUE UTILISATEUR ?

Pas toujours possible ni souhaitable (appli web)

Filtrage avec **Variables de session**

```
SET rls.ename = 'smith';

---

CREATE POLICY emp_rls_policy ON employee FOR all TO public

USING(ename=current_setting('filtre.ename'));
```

AVEC LES VUES

- il suffit de filtrer les tables sous jacentes! rien à faire sur le vues.
- La lecture des tables d'une vue exploite l'utilisateur propriétaire de la vue:
  - ~~current_user~~ > `session_user`
- Si le propriétaire est superuser, les RLS sont bypassées
  - Explicitement avoir des propriétaires NON superuser

RÉFÉRENCES

Bennie Swart - Octobre 2018 - PostgresConf South Africa 2018 :

https://postgresconf.org/conferences/SouthAfrica2018/prog level-security

https://dba.stackexchange.com/questions/83216/how-to-restrict-access-to-current-user-record-and-its-rights-in-postgres

http://rhaas.blogspot.com/2012/03/security-barrier-views.html

https://stackoverflow.com/questions/33858030/why-isnt-row-level-security-enabled-for-postgres-views

MERCI DE VOTRE ATTENTION

ET N'OUBLIEZ PAS LES JOURNÉES QGIS-FR LES 15 ET 16 DÉCEMBRE !

https://twitter.com/JourneesQgis


Rencontres utilisateurs
QGIS-fr
Les 15 et 16 décembre 2020
A distance
15/12 : Ateliers
16/12 : Conférences