Passerelle - Development #20535

opengis: prevoir un système de requetes pour le endpoint "features" pour simplifier les appels au webservice

08 décembre 2017 11:57 - Serghei Mihai

Statut:	Fermé	**Début:**	08 décembre 2017
Priorité:	Normal	**Echéance:**	
Assigné à:	Valentin Deniaud	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

~~Afin de ne pas avoir à gérer les paramètres spécifiques dans les appels au webservice: genre taper du CQL dans wcs.~~

cf #20535#note-9

Révisions associées

Révision 4c4bc697 - 14 avril 2020 11:29 - Valentin Deniaud

misc: share query management template code (#20535)

Révision 0bc0bd45 - 14 avril 2020 11:29 - Valentin Deniaud

base: add generic BaseQuery (#20535)

Révision 286a12d4 - 14 avril 2020 11:29 - Valentin Deniaud

opengis: add query system for features (#20535)

Révision 10ce34eb - 14 avril 2020 11:29 - Valentin Deniaud

opengis: cache custom queries (#20535)

Révision 68d745ad - 14 avril 2020 11:29 - Valentin Deniaud

opengis: add bbox filter for custom queries (#20535)

Historique

#1 - 11 décembre 2017 22:51 - Serghei Mihai

- Assigné à mis à Serghei Mihai

#2 - 12 décembre 2017 01:40 - Thomas Noël

Ça a un rapport avec #20546 ?

#3 - 12 décembre 2017 10:09 - Serghei Mihai

Oui, mais je préfère bosser sur ce ticket car dans la plupart des cas le filtre se fera sur plusieurs propriétés.

#4 - 12 décembre 2017 17:58 - Serghei Mihai

- Fichier 0001-opengis-add-feature-queries-20535.patch ajouté

- Statut changé de Nouveau à En cours

- Patch proposed changé de Non à Oui

Une première version ici.

Mon idée, analogue à celle des requetes dans le connecteur CSV, est de pouvoir notamment construire des requetes en CQL pour ne pas y réflechir dans wcs ou ailleurs. Les paramètres passés en GET serviront à construire la requete.
Et aussi au retour décider du formatage des données.

Je suis en train de faire les tests.

#5 - 12 décembre 2017 18:07 - Frédéric Péters

> eval

Plutôt que construire une requête interprétée côté SIG (ce que j'imaginais par "requête CQL"), ça me semble tout récupérer puis lancer du Python dessus.

#6 - 12 décembre 2017 18:14 - Serghei Mihai

La requete CQL pourra être écrite dans query_filters. Par exemple:

```
typeName:"ref_nom_voies_par_commune"
cql_filter: "strEqualsIgnoreCase(nom_commune, '" + commune + "') = true"
```

et tout ça sera envoyé en paramètres au SIG.

#7 - 12 décembre 2017 18:33 - Thomas Noël

Je pensais moi qu'on proposerait,pour une requête donnée, de spécifier TYPENAMES, PROPERTYNAME sous forme de deux charfield classiques, et un CQL_FILTER sous forme d'un textarea contenant un template Django. Template qui recevra request, et "basta". Ça sera plus simple àmha.

#8 - 13 décembre 2017 12:14 - Serghei Mihai

Je ne veux pas inventer une nouvelle façon de redefinir de paramètres. Comme la syntaxe des expressions/projections dans le connecteur CSV est déjà plus ou moins familière, je préfère me baser dessus.

#9 - 16 mars 2020 10:29 - Frédéric Péters

- Statut changé de En cours à Nouveau

- Assigné à changé de Serghei Mihai à Valentin Deniaud

- Patch proposed changé de Oui à Non

Je vais considérer ce ticket à zéro et le prendre avec le détail de ce qui est aujourd'hui concrètement imaginé pour Toodego. Et je l'assigne à Valentin qui a dit qu'il allait regarder.

On définirait ici une requête "aires de covoiturage", paramètres :

- typename : pvo_patrimoine_voirie.pvoparking, * filter : <Filter><PropertyIsEqualTo><PropertyName>typeparking...<Literal>Aire de covoiturage... (en vrai XML)

et ceux-ci seraient utilisés pour un appel (request=GetFeature) à OpenGIS.

On définirait aussi dans ces requêtes un paramètre de cache, et le code de filtrage géographique serait directement intégré dans le connecteur; ainsi on l'aurait genre toutes les heures récupérer les données de data.grandlyon puis nos requêtes pouvant s'exécuter uniquement en local.

Par filtrage géographique, j'entends juste la possibilité de passer un paramètre type ?bbox=x1,y1,x2,y2 et retourner ce qui est dans le rectangle, ça ne me semble même pas nécessaire d'activer PostGIS.

Exemple réel pour tester :

- https://download.data.grandlyon.com/wfs/grandlyon?SERVICE=WFS&VERSION=2.0.0&outputformat=GEOJSON&request=GetFeature
  - &typename=pvo_patrimoine_voirie.pvoparking
  - &filter=<Filter><PropertyIsEqualTo><PropertyName>typeparking</PropertyName><Literal>Aire de covoiturage</Literal></PropertyIsEqualTo></Filter>

Sur la forme, voir ce qui vient d'être fait côté ArcGIS. (#27782).

#11 - 25 mars 2020 12:00 - Valentin Deniaud

Petites interrogations fonctionnelles. Dans ton exemple d'appel, tu ne spécifie pas de propriété, pourtant c'est un paramètre requis de l'endpoint features. Est-ce qu'on garde ce caractère obligatoire, et si oui, est-ce qu'on le spécifie dans la requête qu'on crée ou dans l'URL ?

Le système de requêtes dans arcgis permet les deux, mais ici j'ai l'impression que tu imagines quelque chose de moins souple. Ça a des conséquences pour ce qui est de la mise en cache : si on autorise des paramètres d'url supplémentaires à destination du serveur, comme c'est fait dans arcgis, il va falloir mettre en cache suivant une clé dynamique, type un hash des ces paramètres. Sinon, pas la peine de s'embêter.

Et question technique en passant, le cache de django suffit, pas la peine de mettre les chose en db comme dans base_adresse ?

#12 - 25 mars 2020 12:40 - Frédéric Péters

- Description mis à jour

> tu ne spécifie pas de propriété, pourtant c'est un paramètre requis de l'endpoint features

Aujourd'hui on a un appel comme celui-ci :

https://download.data.grandlyon.com/wfs/grandlyon?SERVICE=WFS&VERSION=2.0.0&outputformat=GEOJSON&maxfeatures=30&request=GetFeature&typename=pvo_patrimoine_voirie.pvoparking&filter=%3CFilter%3E%3CPropertyIsEqualTo%3E%3CPropertyName%3Etypeparking%3C/PropertyName%3E%3CLiteral%3EAire%20de%20covoiturage%3C/Literal%3E%3C/PropertyIsEqualTo%3E%3C/Filter%3E

Et ça marche, sans propertyname, je dirais que qui est fait dans /features/ est à ignorer (mais à laisser intact).

> si on autorise des paramètres d'url supplémentaires à destination du serveur

Non, requête, zéro paramètres. (si ce n'est ceux qui arriveront pour filtrer la recherche, mais filtre côté passerelle)

> Et question technique en passant, le cache de django suffit, pas la peine de mettre les chose en db comme dans base_adresse ?

Ça doit être en base de données, parce qu'on parle de données qui peuvent être nombreuses et qu'il y aura des requêtes à faire dessus derrière (pour le filtrage géographique cité).

#13 - 25 mars 2020 16:42 - Valentin Deniaud

OK, et encore une question par rapport à ta description au dessus :

Frédéric Péters a écrit :

> On définirait aussi dans ces requêtes un paramètre de cache, et le code de filtrage géographique serait directement intégré dans le connecteur; ainsi
> on l'aurait genre toutes les heures récupérer les données de data.grandlyon puis nos requêtes pouvant s'exécuter uniquement en local.

Il n'y a pas une contradiction ici ? Tel que je le vois c'est soit on utilise la méthode hourly et les données sont récupérées toutes les heures, soit on introduit un paramètre de cache et la récupération se fait autrement (en supposant que « paramètre de cache » veuille dire « quelque chose qui spécifie la durée du cache »).

Perso je préférerais faire autrement qu'avec hourly, ça risque d'être bien bien frustrant voire inutilisable d'attendre une heure entre chaque modif de la requête pour vérifier qu'elle marche. On peut viser un truc hybride, synchro toutes les heures + à chaque modif de la requête le cache est invalidé et une requête quand le cache est invalide déclenche la vraie récup des données (et si c'est trop long et que la page timeout, on fait confiance à la personne derrière pour comprendre pourquoi).

#14 - 25 mars 2020 17:03 - Frédéric Péters

Oui, ça peut être cron horaire + job asynchrone lors de la création/modification pour bootstrapper ça. Quand je parlais de duréee de cache c'était de l'ordre de la journée (c'est ce qu'on a aujourd'hui pour d'autres données récupérées de data.grandlyon aujourd'hui), tout à fait compatible avec un job toutes les heures.

#15 - 26 mars 2020 17:59 - Valentin Deniaud

- Fichier 0001-misc-share-query-management-template-code-20535.patch ajouté

- Fichier 0004-opengis-add-query-system-for-features-20535.patch ajouté

- Fichier 0006-opengis-add-bbox-filter-for-custom-queries-20535.patch ajouté

- Fichier 0003-base-add-generic-BaseQuery-20535.patch ajouté

- Fichier 0002-api-use-dict-comprehension-20535.patch ajouté

- Fichier 0005-opengis-cache-custom-queries-20535.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

Voilà.

Vu ce qui a été fait dans Arcgis la première étape est clairement de mutualiser le code. Le patch 0001 s'occupe des templates, et englobe aussi csvdatasource. 0003 fait l'autre partie, plus compliquée. Ça ne permet pas encore d'ajouter facilement un système de requête à n'importe quel connecteur mais ça en prend le chemin, il manque surtout des vues génériques, mais je préfère attendre qu'il y ait un troisième cas pour se lancer là dedans.
0004 ajoute le système de requête, peu de code grâce aux patches d'avant, et views.py et urls.py sont copié collés de ce qui a été fait dans arcgis.

0005 ajoute le cache, en s'appuyant sur le système de job asynchrone.
0006 ajoute le filtrage par bbox.

J'ai un 0007 en local pour parer au cas où le cache est vide parce que la requête au serveur ne renvoie rien ; à ce moment là pour l'instant on affiche « Données en attente de récupération etc » parce qu'on voit le cache vide et qu'on suppose qu'il va être rempli, mais en fait non. Mon idée c'est d'afficher en plus dans le message l'URL qui doit être appelée, comme ça la personne peut cliquer dessus et se rendre compte que le serveur ne renvoie rien. Mais bon, ça fait un patch un peu gros pour ce que c'est, donc sauf avis contraire on dit que ce n'est pas la peine de tant s'embêter.

#16 - 29 mars 2020 20:51 - Frédéric Péters

Commentaires sur des premiers tests, sans encore avoir regardé attentivement le code (juste en passant, renommer "q" en "query", parce que les méthodes d'une lettre c'est vraiment trop court).

À la création d'une requête, trace,

```
  File "/home/fred/src/eo/venv1.11/local/lib/python2.7/site-packages/django/views/generic/edit.py", line 45, i
n get_form
    return form_class(**self.get_form_kwargs())
  File "/home/fred/src/eo/venv1.11/local/lib/python2.7/site-packages/django/forms/models.py", line 298, in __i
nit__
    self.instance = opts.model()
  File "/home/fred/src/eo/passerelle/passerelle/apps/opengis/models.py", line 434, in __init__
    self.endpoint = self.resource.features
  File "/home/fred/src/eo/venv1.11/local/lib/python2.7/site-packages/django/db/models/fields/related_descripto
rs.py", line 194, in __get__
    "%s has no %s." % (self.field.model.__name__, self.field.name)
RelatedObjectDoesNotExist: Query has no resource.
```

contournée en local,

```
     def __init__(self, *args, **kwargs):
         super(Query, self).__init__(*args, **kwargs)
-        self.endpoint = self.resource.features
-        self.called_from = self.resource.q
+        if self.resource_id:
+            self.endpoint = self.resource.features
+            self.called_from = self.resource.q
```

Ensuite fail avec #41143.

Ensuite fail parce que erreur 400, "'JSON' is not a permitted output format for layer"; c'est en fait geojson qui est attendu, est-ce là une différence entre versions ? Ou geojson marcherait toujours ? Ou il faut parser https://download.data.grandlyon.com/wfs/grandlyon?service=WFS&request=GetCapabilities pour voir ? (TODO: regarder où le connecteur est utilisé et les réponses obtenues).

En modifiant,

```
-            'OUTPUTFORMAT': 'json',
+            'OUTPUTFORMAT': 'geojson',
```

ça passe et j'obtiens bien les infos en base.

Pour tester, mon paramétrage :

- URL du service WFS : https://download.data.grandlyon.com/wfs/grandlyon
- feature type : pvo_patrimoine_voirie.pvoparking
- xml filter : <Filter><PropertyIsEqualTo><PropertyName>typeparking</PropertyName><Literal>Aire de covoiturage</Literal></PropertyIsEqualTo></Filter>

(en passant, il y aurait sans doute à rendre query_layer optionnel dans le paramétrage du connecteur).

#17 - 30 mars 2020 12:36 - Benjamin Dauvergne

On me demande mon avis sur le 0003:

- [redacted]

  ```
            q = Query.import_json(query)
  -           q.resource = instance
  ```

  ça semble gratuit comme changement, les deux fonctionnent.

- J'aime pas vraiment le self.endpoint / self.called_from c'est très spécifique à ces deux implémentations que les query correspondent à un endpoint existant dont on va pré-remplir des paramètres, je ne sais pas trop ce qu'on y gagne à part que c'est plus complexe à comprendre (trop d'indirections) ; on avait un appel direct à self.resource.mapservice_query qui était assez clair et ça devient un appel à self.endpoint qui est en fait self.resource.mapservice_query :/

- Ne pas garder query_set comme "reverse related name" définir "queries" plutôt (related_name="queries" sur la foreignkey vers resources, je reconnais que ça rajoute de la convention mais query_set c'est trop proche de queryset)
- Tu n'utilises par tes nouveaux delete_view / edit_view dans arcgis.Query
- Dans BaseQuery.as_endpoint() ne pas appeler la copie de endpoint_info, endpoint, ça reste un endpoint_info.
- Je ne comprends pas à quoi sert extra_params..

Je trouve ça bien mais je ne voudrais pas que ça complexifie les vues, le reste ça va.

#18 - 31 mars 2020 16:33 - Valentin Deniaud

Merci pour les remarques, je vais les intégrer.

Benjamin Dauvergne a écrit :

> - J'aime pas vraiment le self.endpoint / self.called_from

Oui c'est la partie pas sympa du patch.

> c'est très spécifique à ces deux implémentations que les query correspondent à un endpoint existant dont on va pré-remplir des paramètres

OK, moi j'imaginais que ce cas d'usage pourrait être amené à se répandre.

> je ne sais pas trop ce qu'on y gagne à part que c'est plus complexe à comprendre (trop d'indirections)

Factoriser du code, mais c'est peut-être vouloir trop en faire, je vais m'y repencher, probablement bouger le code dans une méthode statique qui sera appelée si besoin au cas par cas.

> - Dans BaseQuery.as_endpoint() ne pas appeler la copie de endpoint_info, endpoint, ça reste un endpoint_info.

Pourtant,

```
-> endpoint = copy.copy(self.endpoint.endpoint_info)
(Pdb) endpoint.__class__.__name__
'endpoint'
```

Mais oui c'est confusant par rapport à self.endpoint qui est autre chose, mais comme il va dégager vu ta remarque du dessus... Je vais voir pour clarifier.

> - Je ne comprends pas à quoi sert extra_params.

Le endpoint_info.parameters est utilisé pour construire l'URL d'exemple, mais dans la description de l'endpoint, la liste des paramètre vient du argspec de l'endpoint, donc le extra_params sert dans ce second cas.

#19 - 06 avril 2020 18:28 - Valentin Deniaud

- Fichier 0001-misc-share-query-management-template-code-20535.patch ajouté

- Fichier 0003-opengis-add-query-system-for-features-20535.patch ajouté

- Fichier 0004-opengis-cache-custom-queries-20535.patch ajouté

- Fichier 0002-base-add-generic-BaseQuery-20535.patch ajouté

- Fichier 0005-opengis-add-bbox-filter-for-custom-queries-20535.patch ajouté

J'ai pris en compte toutes les remarques sauf

> (en passant, il y aurait sans doute à rendre query_layer optionnel dans le paramétrage du connecteur).

J'ajouterais un patch quand j'aurais fouillé pour comprendre pourquoi ce paramètre est utilisé pour spécifier des typenames dans l'endpoint reverse, mais ça ne devrait pas impacter le reste.

J'ai aussi ouvert #41386 sur lequel je m'appuie.

#20 - 08 avril 2020 14:40 - Valentin Deniaud

- Fichier 0001-misc-share-query-management-template-code-20535.patch ajouté

- Fichier 0003-opengis-add-query-system-for-features-20535.patch ajouté

- Fichier 0004-opengis-cache-custom-queries-20535.patch ajouté

- Fichier 0002-base-add-generic-BaseQuery-20535.patch ajouté

- Fichier 0005-opengis-add-bbox-filter-for-custom-queries-20535.patch ajouté

Modifs mineures et rebasage.

#21 - 14 avril 2020 08:50 - Frédéric Péters

Ok, il me semble juste manquer :

```
+    class Meta:
+        verbose_name = _('Query')
```

#22 - 14 avril 2020 08:50 - Frédéric Péters

- Statut changé de Solution proposée à Solution validée

#23 - 14 avril 2020 11:37 - Valentin Deniaud

- Statut changé de Solution validée à Résolu (à déployer)

```
commit 68d745ad28c8991aaceec671da664c0adea85902
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Wed Mar 25 16:11:46 2020 +0100

    opengis: add bbox filter for custom queries (#20535)

commit 10ce34eb31abf1dbfc35e9c6cba9ccb2054d9bdf
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Wed Mar 25 14:51:21 2020 +0100

    opengis: cache custom queries (#20535)

commit 286a12d4d4354ab42f1df85aae50ed126f7034b6
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Tue Mar 24 16:36:19 2020 +0100

    opengis: add query system for features (#20535)

commit 0bc0bd45fee85d90d286093db306f76eb14feef0
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Tue Mar 24 16:39:37 2020 +0100

    base: add generic BaseQuery (#20535)

commit 4c4bc6970700cbdafdcc3dea8c63254a644ba52e
Author: Valentin Deniaud <vdeniaud@entrouvert.com>
Date:   Tue Mar 24 14:53:20 2020 +0100

    misc: share query management template code (#20535)
```

#24 - 15 avril 2020 14:17 - Frédéric Péters

- Statut changé de Résolu (à déployer) à Solution déployée

Fichiers

0001-opengis-add-feature-queries-20535.patch	14,6 ko	12 décembre 2017	Serghei Mihai
0001-misc-share-query-management-template-code-20535.patch	5,61 ko	26 mars 2020	Valentin Deniaud
0004-opengis-add-query-system-for-features-20535.patch	13,8 ko	26 mars 2020	Valentin Deniaud
0006-opengis-add-bbox-filter-for-custom-queries-20535.patch	3,35 ko	26 mars 2020	Valentin Deniaud
0003-base-add-generic-BaseQuery-20535.patch	14,6 ko	26 mars 2020	Valentin Deniaud
0002-api-use-dict-comprehension-20535.patch	1,69 ko	26 mars 2020	Valentin Deniaud
0005-opengis-cache-custom-queries-20535.patch	11,7 ko	26 mars 2020	Valentin Deniaud
0001-misc-share-query-management-template-code-20535.patch	10,9 ko	06 avril 2020	Valentin Deniaud
0003-opengis-add-query-system-for-features-20535.patch	15 ko	06 avril 2020	Valentin Deniaud

0004-opengis-cache-custom-queries-20535.patch	11,5 ko	06 avril 2020	Valentin Deniaud
0002-base-add-generic-BaseQuery-20535.patch	13,2 ko	06 avril 2020	Valentin Deniaud
0005-opengis-add-bbox-filter-for-custom-queries-20535.patch	3,36 ko	06 avril 2020	Valentin Deniaud
0001-misc-share-query-management-template-code-20535.patch	10,9 ko	08 avril 2020	Valentin Deniaud
0003-opengis-add-query-system-for-features-20535.patch	14,6 ko	08 avril 2020	Valentin Deniaud
0004-opengis-cache-custom-queries-20535.patch	11,5 ko	08 avril 2020	Valentin Deniaud
0002-base-add-generic-BaseQuery-20535.patch	11,9 ko	08 avril 2020	Valentin Deniaud
0005-opengis-add-bbox-filter-for-custom-queries-20535.patch	3,35 ko	08 avril 2020	Valentin Deniaud