Passerelle - Development #48074

Logs - recherche des logs - lenteurs

27 octobre 2020 23:17 - Lauréline Guérin

Statut:	Fermé	**Début:**	27 octobre 2020
Priorité:	Normal	**Echéance:**	
Assigné à:	Lauréline Guérin	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

cf https://passerelle.eservices.toulouse-metropole.fr/manage/toulouse-axel/axel-famille/logs/?q=51c46d37-db36-451b-97dd-a023ab2529ac

=> 67 secondes pour afficher la page, qui ne contient que 2 résultats

Révisions associées

Révision 3f57cc4e - 12 novembre 2020 11:30 - Lauréline Guérin

logs: better queryset to search for logs (#48074)

Révision 6c8bdb70 - 12 novembre 2020 11:54 - Lauréline Guérin

logs: change paginator to avoid count (#48074)

Historique

#1 - 28 octobre 2020 11:07 - Valentin Deniaud

- Sujet changé de Logs - recherche des logs provenant du même appel - lenteurs à Logs - recherche des logs - lenteurs

Même chose pour une recherche textuelle, normal c'est le même code :

```
passerelle/views.py
252                 qs = qs.annotate(
253                     text_extra=Cast('extra', TextField())
254                 ).filter(Q(text_extra__icontains=query) | Q(message__icontains=query))
```

C'est sûrement le cast sur un jsonfield qui est long. Ici il faut savoir que ça a été fait du temps de django-jsonfield, mais maintenant on est passé au JSONField standard de contrib.postgres. Il y a donc les lookup genre extra__message, extra__transaction_id etc qui permettraient de faire un truc moins bourrin et sûrement beaucoup plus rapide.

#2 - 28 octobre 2020 16:37 - Valentin Deniaud

Valentin Deniaud a écrit :

> sûrement beaucoup plus rapide

J'ai davantage regardé et de mes essais en local éviter le Cast c'est la moitié du problème, le __icontains se montre tout aussi gourmand.

Ici il faut peut-être juste nettoyer les logs et limiter la taille max des réponses dans les paramètres de journalisation.

#3 - 29 octobre 2020 02:11 - Benjamin Dauvergne

Valentin Deniaud a écrit :

> Valentin Deniaud a écrit :
>
> > sûrement beaucoup plus rapide
>
> J'ai davantage regardé et de mes essais en local éviter le Cast c'est la moitié du problème, le __icontains se montre tout aussi gourmand.
>
> Ici il faut peut-être juste nettoyer les logs et limiter la taille max des réponses dans les paramètres de journalisation.

On a déjà que 8 jours de logs :

```
passerelle=> select max(timestamp), min(timestamp) from base_resourcelog where appname = 'toulouse-axel' and s
lug = 'axel-famille';
              max               |               min
--------------------------------+--------------------------------
 2020-10-29 01:40:06.060319+01 | 2020-10-21 03:00:13.702787+02
```

mais la table fait 1Go et sur 700k messages, 650k appartiennent au connecteur Axel. Je ne pense pas qu'on ait terriblement de solution 30 secondes pour faire chercher une sous chaîne dans 1Go de log par postgresql c'est déjà pas mal (juste que ça irait plus vite avec grep), rajouter un index nous coûtera autant voir plus en volume (le seul moyen de chercher des sous chaînes c'est soit un index FTS soit un index trigramme), un index JSON coûtera autant mais n'apportera pas la recherche par sous chaîne.

J'ai regardé coté compression, pg compresse les données quand elles dépassent 2K d'après la doc, mais 2/3 des valeurs sont en dessous de la limite :

```
passerelle=> select count(1) from base_resourcelog where appname = 'toulouse-axel' and slug = 'axel-famille' a
nd length(extra::text) < 2048;
 count
--------
 401261
(1 ligne)
```

Pour donner des ordres de grandeur et voir que postgresql/kernel ne gardent pas grand chose (et c'est tant mieux finalement) de cette table en cache pour nous donner ces temps :

```
passerelle=> \copy (select * from base_resourcelog) to '/tmp/resourcelog.csv';
COPY 679362
passerelle=>

# 1,4Go au lieu de 1Go
bdauvergne@cutm-publik-prod-web1:/tmp$ ls -l resourcelog.csv
-rw-r--r-- 1 bdauvergne bdauvergne 1451146194 oct.  29 02:00 resourcelog.csv

# suppression du cache disque
bdauvergne@cutm-publik-prod-web1:/tmp$ dd of=resourcelog.csv oflag=nocache conv=notrunc,fdatasync count=0
0+0 enregistrements lus
0+0 enregistrements écrits
0 octet copié, 0,166387 s, 0,0 kB/s

# temps cache froid
bdauvergne@cutm-publik-prod-web1:/tmp$ ls -l resourcelog.csv
-rw-r--r-- 1 bdauvergne bdauvergne 1451146194 oct.  29 02:00 resourcelog.csv
bdauvergne@cutm-publik-prod-web1:/tmp$ time grep 51c46d37-db36-451b-97dd-a023ab2529ac resourcelog.csv
6984312    2020-10-27 22:35:06.28677+01    toulouse-axel    axel-famille    40    \N    GET https://passerelle
.eservices.toulouse-metropole.fr/pfs/Axel_WS/AxelWS.php?wsdl (=> 503)    {"request_url": "https://passerelle.e
services.toulouse-metropole.fr/pfs/Axel_WS/AxelWS.php?wsdl", "transaction_id": "51c46d37-db36-451b-97dd-a023ab
2529ac", "request_headers": {"Accept": "*/*", "Connection": "keep-alive", "User-Agent": "python-requests/2.21.
0", "Accept-Encoding": "gzip, deflate"}, "response_status": 503, "response_content": "b'<!DOCTYPE HTML PUBLIC
\\"-//IETF//DTD HTML 2.0//EN\\">\\\\n<html><head>\\\\n<title>503 Service Unavailable</title>\\\\n</head><body>
\\\\n<h1>Service Unavailable</h1>\\\\n<p>The server is temporarily unable to service your\\\\nrequest due to m
aintenance downtime or capacity\\\\nproblems. Please try again later.</p>\\\\n</body></html>\\\\n'", "response
_headers": {"Date": "Tue, 27 Oct 2020 21:35:06 GMT", "Server": "nginx/1.14.2", "Content-Type": "text/html; cha
rset=iso-8859-1", "Transfer-Encoding": "chunked", "Strict-Transport-Security": "max-age=15768000"}}
6984313    2020-10-27 22:35:06.464468+01    toulouse-axel    axel-famille    40    \N    connector "Axel Famil
le" (ToulouseAxel) is now down: 503 Server Error: Service Unavailable for url: https://passerelle.eservices.to
ulouse-metropole.fr/pfs/Axel_WS/AxelWS.php?wsdl    {"transaction_id": "51c46d37-db36-451b-97dd-a023ab2529ac"}

real    1m40,230s
user    0m1,162s
sys    0m1,108s

# temps cache chaud
bdauvergne@cutm-publik-prod-web1:/tmp$ time grep 51c46d37-db36-451b-97dd-a023ab2529ac resourcelog.csv
6984312    2020-10-27 22:35:06.28677+01    toulouse-axel    axel-famille    40    \N    GET https://passerelle
.eservices.toulouse-metropole.fr/pfs/Axel_WS/AxelWS.php?wsdl (=> 503)    {"request_url": "https://passerelle.e
services.toulouse-metropole.fr/pfs/Axel_WS/AxelWS.php?wsdl", "transaction_id": "51c46d37-db36-451b-97dd-a023ab
2529ac", "request_headers": {"Accept": "*/*", "Connection": "keep-alive", "User-Agent": "python-requests/2.21.
0", "Accept-Encoding": "gzip, deflate"}, "response_status": 503, "response_content": "b'<!DOCTYPE HTML PUBLIC
\\"-//IETF//DTD HTML 2.0//EN\\">\\\\n<html><head>\\\\n<title>503 Service Unavailable</title>\\\\n</head><body>
\\\\n<h1>Service Unavailable</h1>\\\\n<p>The server is temporarily unable to service your\\\\nrequest due to m
aintenance downtime or capacity\\\\nproblems. Please try again later.</p>\\\\n</body></html>\\\\n'", "response
```

```
_headers": {"Date": "Tue, 27 Oct 2020 21:35:06 GMT", "Server": "nginx/1.14.2", "Content-Type": "text/html; cha
rset=iso-8859-1", "Transfer-Encoding": "chunked", "Strict-Transport-Security": "max-age=15768000"}}
6984313    2020-10-27 22:35:06.464468+01    toulouse-axel    axel-famille    40    \N    connector "Axel Famil
le" (ToulouseAxel) is now down: 503 Server Error: Service Unavailable for url: https://passerelle.eservices.to
ulouse-metropole.fr/pfs/Axel_WS/AxelWS.php?wsdl    {"transaction_id": "51c46d37-db36-451b-97dd-a023ab2529ac"}

real    0m1,572s
user    0m1,256s
sys    0m0,311s
```

#4 - 29 octobre 2020 08:40 - Frédéric Péters

> rajouter un index nous coûtera autant voir plus en volume

Sobriété etc. je suis à fond pour mais ici je dirais qu'on ne devrait pas se soucier de 1 Go supplémentaire, ni 10, d'ailleurs.

#5 - 09 novembre 2020 18:01 - Lauréline Guérin

- Fichier 0001-logs-better-queryset-to-search-for-logs-48074.patch ajouté

- Statut changé de Nouveau à En cours

En local avec 20K entrées dans la table des logs, ça passe quand même mieux comme ça (280ms => 50ms sur un select)

Un peu d'archéologie: avant d'introduire la recherche par transaction_id on ne filtrait que sur message. Si on limite le match à extra__transaction_id (exact) + message (icontains), ça devrait être suffisant.

Autre piste: changer la pagination pour ne pas faire de count (parce que ça coûte), mais alors on perdra la notion de nombre de pages (on n'aura que des boutons previous/next)

Qu'en pensez-vous ?

#6 - 09 novembre 2020 18:06 - Benjamin Dauvergne

Lauréline Guerin a écrit :

> Qu'en pensez-vous ?

C'est dans extra qu'il y a les copies des entrées/sortie HTTP, si des gens sont habitués à chercher là dedans ça va leur manquer.

Un contournement qui garderait peut-être une bonne partie des usages: si la recherche sans text_extra__icontains= ne renvoie rien, faire une nouvelle recherche avec text_extra__icontains= ? Ça ralentirait uniquement les recherches visant extra sur un connecteur aux logs chargés, dans tous les autres cas (dont le tien) ça ira vite.

#7 - 09 novembre 2020 18:34 - Lauréline Guérin

mwai j'avais pas vu #39563, la recherche dans tout extra vient de là

et pour la pagination, un avis ?

#8 - 09 novembre 2020 20:33 - Benjamin Dauvergne

Lauréline Guerin a écrit :

> mwai j'avais pas vu #39563, la recherche dans tout extra vient de là
>
> et pour la pagination, un avis ?

La recherche étant très spécifique (2 résultats, plus court que le 'LIMIT 25' de la pagination), la requête de base ou le count doivent prendre le même temps, un parcours entier de la base.

```
In [12]: %time list(ResourceLog.objects.filter(appname='toulouse-axel', slug='axel-famille').order_by('-timest
amp').annotate(text_extra=Cast('extra', TextField())).filter(te
    ...: xt_extra__icontains='51c46d37-db36-451b-97dd-a023ab2529ac')[:25])
CPU times: user 2 ms, sys: 7.9 ms, total: 9.9 ms
Wall time: 48.7 s
Out[12]: []

In [13]: %time ResourceLog.objects.filter(appname='toulouse-axel', slug='axel-famille').order_by('-timestamp')
```

```
.annotate(text_extra=Cast('extra', TextField())).filter(text_ex
    ...: tra__icontains='51c46d37-db36-451b-97dd-a023ab2529ac').count()
CPU times: user 1.78 ms, sys: 7.57 ms, total: 9.35 ms
Wall time: 48 s
```

Donc oui ça diviserai le temps par 2 sur ton cas mais pas plus, ça restera trop long pour être supportable. Dans le cas d'une requête moins spécifique (qui retournerait très vite plus de 25 lignes), ça améliorera bien les choses. Donc je dirai qu'on peut faire les deux, la pagination ne sert pas à grand chose en fait, précédent / suivant c'est bien suffisant, pour ça tu peux simplement paginer par timestamp+id.

#9 - 12 novembre 2020 09:18 - Lauréline Guérin

- Fichier 0002-snapshots-keep-only-latest-version-gor-big-objects-4.patch ajouté

- Fichier 0001-snapshots-remove-last_modification-time-user-from-in.patch ajouté

- Fichier Capture d'écran de 2020-11-12 09-15-12.png ajouté

- Statut changé de En cours à Solution proposée

- Assigné à mis à Lauréline Guérin

- Patch proposed changé de Non à Oui

Proposition:

0001: réécriture du queryset pour ne chercher qu'en dernier recours dans extra::text
0002: autre Paginator qui ne fait pas de count

#10 - 12 novembre 2020 09:51 - Benjamin Dauvergne

Pas les bons patchs je pense.

#11 - 12 novembre 2020 09:56 - Lauréline Guérin

- Fichier 0002-snapshots-keep-only-latest-version-gor-big-objects-4.patch supprimé

#12 - 12 novembre 2020 09:56 - Lauréline Guérin

- Fichier 0001-snapshots-remove-last_modification-time-user-from-in.patch supprimé

#13 - 12 novembre 2020 09:56 - Lauréline Guérin

- Fichier 0002-logs-change-paginator-to-avoid-count-48074.patch ajouté

- Fichier 0001-logs-better-queryset-to-search-for-logs-48074.patch ajouté

grumpf

#14 - 12 novembre 2020 10:22 - Benjamin Dauvergne

J'ai regardé sur la branche :

```
+                           When(extra__transaction_id=query, then=Value(True)),
+                           When(message__icontains=query, then=Value(True)),
+                           When(extra__icontains=query, then=Value(True)),
```

alors ça va rapporter plus de résultats en théorie c'est sûr, mais ça ne débranche pas du tout la recherche dans extra, ça fait exactement la même recherche WHERE (UPPER("passerelle_resourcelog"."extra"::text) LIKE UPPER('%' || %(query)s || '%') dans la plupart des cas est un parcourt de toute la table.

En faisant une première recherche sur transaction_id et si c'est vide les autres types de recherche, on perd en théorie des résultats possible mais comme c'est un uuid, les chances de collisions sont nulles, je pense que c'est plus intéressant de simplement spécialiser sur ton cas d'usage. Ça restera lent dans le cas général (on peut aussi spécialiser sur message qui va plus vite mais là on risque de perdre plus de résultats) de toute façon.

```
qs = qs.filter(extra__transaction_id=query)
if not qs.exists():
    qs = qs.filter(Q(extra__icontains=query) | Q(message__icontains=query))
```

le gain ne sera visible qu'avec un index JSON (et en fait ce serait plus simple de rajouter une colonne UUIDField pour le transaction_id et de le déplacer là, ol 'index sera plus petit et servira juste pour ce dont on a besoin, indexer tout le json pour un champ c'est overkill):

```
In [8]: %time list(ResourceLog.objects.filter(appname='toulouse-axel', slug='axel-famille').filter(extra__tran
saction_id=trid))
CPU times: user 9 ms, sys: 0 ns, total: 9 ms
Wall time: 15.4 s
Out[8]:
```

```
[<ResourceLog: 2020-11-12 09:15:40.854498+00:00 20 toulouse-axel axel-famille>,
 <ResourceLog: 2020-11-12 09:15:41.102031+00:00 20 toulouse-axel axel-famille>]
```

parce que là ça prend encore 15 secondes juste pour chercher par transaction_id.

Pour la pagination, c'est ok, mais j'ai l'impression que tu aurais pu surcharger moins (juste def count(..): return 2**32; def page_range(...): return [0] devrait suffire à dézinguer le comptage et laisser le reste fonctionner correctement avec un template simplifié).

#15 - 12 novembre 2020 12:04 - Lauréline Guérin

- Fichier 0002-logs-change-paginator-to-avoid-count-48074.patch ajouté

- Fichier 0001-logs-better-queryset-to-search-for-logs-48074.patch ajouté

#16 - 12 novembre 2020 12:05 - Lauréline Guérin

- Fichier Capture d'écran de 2020-11-12 11-30-32.png ajouté

- Fichier Capture d'écran de 2020-11-12 12-03-10.png ajouté

pagination sur une page qui n'est no la première ni la dernière, + pagination sur la dernière page

#17 - 12 novembre 2020 16:02 - Benjamin Dauvergne

- Statut changé de Solution proposée à Solution validée

#18 - 13 novembre 2020 09:18 - Lauréline Guérin

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

```
commit 6c8bdb707d3ecfeaf3cfb512d61709f48c935e1f
Author: Lauréline Guérin <zebuline@entrouvert.com>
Date:   Thu Nov 12 09:16:38 2020 +0100

    logs: change paginator to avoid count (#48074)

commit 3f57cc4ec689515e30b9a3392905b3ab7167ebbe
Author: Lauréline Guérin <zebuline@entrouvert.com>
Date:   Mon Nov 9 17:55:32 2020 +0100

    logs: better queryset to search for logs (#48074)
```

#19 - 17 novembre 2020 19:16 - Frédéric Péters

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

Fichiers

0001-logs-better-queryset-to-search-for-logs-48074.patch	1,07 ko	09 novembre 2020	Lauréline Guérin
Capture d'écran de 2020-11-12 09-15-12.png	41 ko	12 novembre 2020	Lauréline Guérin
0002-logs-change-paginator-to-avoid-count-48074.patch	4,92 ko	12 novembre 2020	Lauréline Guérin
0001-logs-better-queryset-to-search-for-logs-48074.patch	4,42 ko	12 novembre 2020	Lauréline Guérin
0002-logs-change-paginator-to-avoid-count-48074.patch	4,43 ko	12 novembre 2020	Lauréline Guérin
0001-logs-better-queryset-to-search-for-logs-48074.patch	8,78 ko	12 novembre 2020	Lauréline Guérin
Capture d'écran de 2020-11-12 11-30-32.png	1,58 ko	12 novembre 2020	Lauréline Guérin
Capture d'écran de 2020-11-12 12-03-10.png	1,77 ko	12 novembre 2020	Lauréline Guérin