w.c.s. - Development #57623

sql: ne pas itérer ligne par ligne sur un SqlKlass.select(iterator=False)

05 octobre 2021 18:51 - Benjamin Dauvergne

Statut:	Rejeté	**Début:**	05 octobre 2021
Priorité:	Normal	**Echéance:**	
Assigné à:	Benjamin Dauvergne	**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Oui	**Planning:**	Non

Description

Cf. #56562 (c'est un cas super limite, mais de faire les itérateurs coté serveurs coûtent cher, on faire des "FETCH FORWARD 1" avec la latence qui va avec pour chaque ligne de table, le problème est ici aussi que sur un live on initialise quand même

Comme on va de toute façon on va tout charger et retourner tout ça dans une liste, itérer coté serveur ligne par ligne ne sert pas à grand chose (et est plus lent à cause des allers-retours autour de FETCH FORWARD).

J'attache un log des requêtes pour le ticket #56562[1], même si je me restreins aux requêtes sur les fiches, histoire de revenir dans un cas plus normal, (pour alimenter les sources de donnée je suppose) on y passe 500ms en 10 requêtes :

```
In [23]: for k in endpoints:
    ...:     print('pid', k, 't0', endpoints[k][0], 't1', endpoints[k][1])
    ...:
pid 31900 t0 18:09:05.103 t1 18:09:05.189
pid 31901 t0 18:09:05.229 t1 18:09:05.232
pid 31905 t0 18:09:07.213 t1 18:09:07.289
pid 31906 t0 18:09:07.350 t1 18:09:07.353
pid 31907 t0 18:09:07.365 t1 18:09:07.368
pid 31909 t0 18:09:07.401 t1 18:09:07.510
pid 31913 t0 18:09:09.463 t1 18:09:09.538
pid 31914 t0 18:09:09.577 t1 18:09:09.583
pid 31917 t0 18:09:11.532 t1 18:09:11.607
pid 31921 t0 18:09:13.602 t1 18:09:13.673

In [24]: sum
Out[24]: 0.50700000001234 (ms)
```

Il y a certainement un problème aussi avec le nombre de fois où on récupère la liste des vues (12 fois) alors que ça n'est pas du tout utilisé ici (ou alors pour certaines sources de données, mais il n'y a que 2 champs listes dans le formulaire de workflow, on devrait avoir seulement 2 requêtes).

[1] log des requêtes SQL pour une seule requête à https://demarches-departement13.test.entrouvert.org/backoffice/management/contacter-le-departement/144/live, ça donne 55000 lignes de SQL.

Demandes liées:

Lié à w.c.s. - Development #57637: Trop de requêtes SQL sur la tables roles p...	**Fermé**	**06 octobre 2021**
Lié à w.c.s. - Development #57635: Sur un appel à /backoffice/management/cont...	**Nouveau**	**06 octobre 2021**
Lié à w.c.s. - Development #58013: Itérer sur les ids au lieu d'utiliser les ...	**Fermé**	**20 octobre 2021**

Historique

#2 - 05 octobre 2021 18:52 - Benjamin Dauvergne

- Description mis à jour

#3 - 05 octobre 2021 18:52 - Benjamin Dauvergne

- Assigné à mis à Benjamin Dauvergne

#4 - 05 octobre 2021 19:10 - Frédéric Péters

Je préfère les tickets publics; si le problème est le fichier attaché il n'a qu'à l'être côté ticket client.

#5 - 05 octobre 2021 20:02 - Benjamin Dauvergne

- Fichier log supprimé

#6 - 05 octobre 2021 20:33 - Benjamin Dauvergne

- Fichier log-with-cursor-itersize-10000 ajouté

- Fichier log-without-cursor ajouté

En interdisant complètement les curseurs coté serveur, on tombe à 3,79s au lieu de 14s (fichier de log log-without-cursor).

Avec des curseurs serveurs à 10000, on est à 3,78s soit dans l'épaisseur du benchmark (fichier de log log-with-cursor-itersize-10000).

Avec une répartition des requêtes sur les tables suivantes (pas bien important pour ce ticket mais c'est intéressant en général) :

```
1 carddata_11_13ave_direction_traitante
1 formdata_50_contacter_le_departement
1 formdata_50_contacter_le_departement_evolutions
1 sessions
2 carddata_2_13ave_kb_copie
2 users
9 carddata_5_13ave_dispositifs
12 custom_views
85 roles
```

#7 - 05 octobre 2021 20:57 - Benjamin Dauvergne

- Statut changé de Nouveau à Information nécessaire

- Assigné à changé de Benjamin Dauvergne à Frédéric Péters

Dans #54242 qui introduit les curseurs nommés et les connections "isolées", je ne vois pas la justifications des connections isolées, quelle était la raison ?

#8 - 05 octobre 2021 21:25 - Benjamin Dauvergne

- Statut changé de Information nécessaire à Nouveau

- Assigné à changé de Frédéric Péters à Benjamin Dauvergne

Benjamin Dauvergne a écrit :

> Dans #54242 qui introduit les curseurs nommés et les connections "isolées", je ne vois pas la justifications des connections isolées, quelle était la raison ?

Et je me réponds à moi même, c'est à cause des erreurs "named cursor isn't valid anymore" parce qu'un curseur est attaché à une transaction et qu'on fait des commits un peu n'importe où parce qu'on a pas l'équivalent de atomic() de Django, donc le isolate est important, sauf pour un select(iterator=False), là on est sûr qu'il n'y aura pas de commit pendant le parcours du curseur.

#9 - 05 octobre 2021 21:30 - Frédéric Péters

Tu pourrais fermer ce ticket et redémarrer un ticket public qui expose brièvement le problème concret ?

#10 - 05 octobre 2021 22:35 - Benjamin Dauvergne

- Assigné à Benjamin Dauvergne supprimé

- Priorité changé de Normal à Immediat

Frédéric Péters a écrit :

> Tu pourrais fermer ce ticket et redémarrer un ticket public qui expose brièvement le problème concret ?

Qu'est-ce qui n'est pas concret ? On ne doit pas itérer ligne par ligne ce n'est pas performant, sauf quand c'est nécessaire (faire un select massif sur data_class()).

#11 - 05 octobre 2021 22:41 - Frédéric Péters

- Priorité changé de Immediat à Normal

La situation précédente était tout charger en mémoire et éclater les serveurs du Grand Lyon. Ce n'était pas "performant".

Ça ne peut donc pas juste être un retour en arrière.

Ici il y a une référence à un ticket privé dont l'intitulé n'a pas de rapport et ça voudrait dire en lire les commentaires pour tenter de peut-être voir de quels objets en quelles circonstances on parle, par exemple, concrètement.

#12 - 05 octobre 2021 22:52 - Frédéric Péters

Sérieusement, recule dix secondes et relis la description de ce ticket, sa première ligne :

> Cf. #56562 (c'est un cas super limite, mais de faire les itérateurs coté serveurs coûtent cher, on faire des "FETCH FORWARD 1" avec la latence qui va avec pour chaque ligne de table, le problème est ici aussi que sur un live on initialise quand même

~~

Bref, à chercher à comprendre malgré tout, le propos réécrit me semble être : on utilise des curseurs côté serveur (et c'est très bien c'était une nécessité ceci n'est pas remis en question) mais ça récupère les données ligne par ligne, ce qui multiplie de manière inutile le nombre de requêtes. Il pourrait être possible de conserver les avantages actuels sur la consommation mémoire tout en accélérant sensiblement les choses en récupérant les données par lots.

#13 - 05 octobre 2021 23:43 - Benjamin Dauvergne

- Priorité changé de Normal à Immediat

- Privée changé de Oui à Non

#14 - 05 octobre 2021 23:43 - Benjamin Dauvergne

- Priorité changé de Immediat à Normal

#15 - 06 octobre 2021 09:51 - Benjamin Dauvergne

- Assigné à mis à Benjamin Dauvergne

#16 - 06 octobre 2021 09:51 - Benjamin Dauvergne

- Fichier 0001-sql-use-a-for-loop-to-iterate-on-cursors.patch ajouté

- Fichier 0003-sql-define-an-adapted-itersize.patch ajouté

- Fichier 0002-sql-use-iterate-on-server-only-for-iterators.patch ajouté

- Statut changé de Nouveau à Solution proposée

- Patch proposed changé de Non à Oui

- le premier patch remplace quelques boucles while/fetchone() par des boucles for, parce que ça désactive le chunking qui aurait été précisé au moment de la création du curseur et de plus les boucles ne sont pas très pythonesque. Pour ne pas changer le comportement réel à ce stade je force un .itersize à 1 au moment de la création d'un curseur nommé.
- le deuxième patch limite _iterate_on_server aux cas où c'est nécessaire, i.e. on utilise vraiment un générateur dans le code de w.c.s et pas simplement pour remplir une liste qui sera renvoyée;
- le troisième pose un .itersize plus raisonnable de 2000 en général (c'est de toute façon la valeur par défaut dans psycopg2, si on utilise pas fetchone() ou fetchmany() explicitement) et de 100 pour les formulaires et les fiches, ça peut s'affiner ou bien revenir à une valeur plus faible pour toute le monde, ça n'a pas d'effet sans le premier patch, je pourrai merger les deux.

#17 - 06 octobre 2021 10:11 - Frédéric Péters

Reste quelque chose par rapport à des vues personnalisées dans le ticket lié, ce que tu notes dans

> Il y a certainement un problème aussi avec le nombre de fois où on récupère la liste des vues (12 fois) alors que ça n'est pas du tout utilisé ici (ou alors pour certaines sources de données, mais il n'y a que 2 champs listes dans le formulaire de workflow, on devrait avoir seulement 2 requêtes).

et qui devrait du coup avoir son ticket propre.

#18 - 06 octobre 2021 11:28 - Benjamin Dauvergne

Pour info je viens de supprimer les 7704 vues personnalisées créer par l'audit de sécurité, et on tombe à 1,5s sans ces patchs et autour de 1,16s avec. Je vais produire une liste de requêtes et/ou de stracktraces toujours pour cet appel à live/ voir ce que je peux ouvrir comme ticket.

#19 - 06 octobre 2021 11:41 - Benjamin Dauvergne

- Lié à Development #57637: Trop de requêtes SQL sur la tables roles pendant une seule requête HTTP ajouté

#20 - 06 octobre 2021 11:51 - Benjamin Dauvergne

- Lié à Development #57635: Sur un appel à /backoffice/management/contacter-le-departement//live des appels à FormPage.set_default_view() sont faits qui ne semblent pas tous nécessaires voir redondant ajouté*

#21 - 15 octobre 2021 14:11 - Frédéric Péters

- Statut changé de Solution proposée à Solution validée

#22 - 18 octobre 2021 10:34 - Frédéric Péters

- Statut changé de Solution validée à Solution proposée

Apparemment c'est cette version qui crashait donc non.

#23 - 18 octobre 2021 10:53 - Frédéric Péters

Une trace reçue d'un serveur qui avait une version de ce patch dans #57639

#24 - 19 octobre 2021 11:27 - Benjamin Dauvergne

- Statut changé de Solution proposée à Rejeté

Ce n'est pas la bonne approche, sans pouvoir déterminer le début et la fin d'une transaction on ne peut pas utiliser des curseurs nommés correctement et donc revenir à une connexion par requête comme en Django. Il faudrait un patch qui remplace tous les conn.commit()/.rollback() explicites par l'utilisation d'un context manager comme atomic() en Django (et par défaut que la connexion soit en mode autocommit, c'est beaucoup plus simple à gérer). J'ouvrirai un autre ticket avec cette idée, je garde le lien vers la trace sous le coude.

PS: Le problème existe aussi en Django, il est juste plus difficile de tomber dedans, mais avec un code de cette forme on peut y arriver :

```
with atomic():
     iterator = Model.objects.iterator()

for instance in interator: <-- normalement une DatabaseError "named cursor isn't valid anymore"
   ...
```

#25 - 20 octobre 2021 16:03 - Benjamin Dauvergne

- Lié à Development #58013: Itérer sur les ids au lieu d'utiliser les curseurs nommés ajouté

Fichiers

log-without-cursor	39,8 ko	05 octobre 2021	Benjamin Dauvergne
log-with-cursor-itersize-10000	56 ko	05 octobre 2021	Benjamin Dauvergne
0001-sql-use-a-for-loop-to-iterate-on-cursors.patch	3,38 ko	06 octobre 2021	Benjamin Dauvergne
0003-sql-define-an-adapted-itersize.patch	1,46 ko	06 octobre 2021	Benjamin Dauvergne
0002-sql-use-iterate-on-server-only-for-iterators.patch	3,89 ko	06 octobre 2021	Benjamin Dauvergne