w.c.s. - Development #68413

SQL: exploiter la connaissance des options possibles dans FormPage.get_item_filter_options

25 août 2022 12:36 - Pierre Ducroquet

Statut:	Nouveau	**Début:**	25 août 2022
Priorité:	Normal	**Echéance:**	
Assigné à:		**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Non	**Planning:**	Non

Description

Aujourd'hui, la fonction FormPage.get_item_filter_options() exécute la requête SQL suivante pour lister les valeurs utilisées dans la base, y compris pour les champs où la liste des valeurs possibles est connue :

```
SELECT DISTINCT ON (fbofd87dd65) fbofd87dd65, fbofd87dd65_display FROM formdata_101 WHERE anonymis
ed IS NULL AND status != 'draft' AND fbofd87dd65 IS NOT NULL ORDER BY fbofd87dd65;
```

Or, quand la liste des valeurs possibles, une autre écriture est envisageable.

```
SELECT x, x FROM (VALUES('X1'), ('X2')) AS v(x) WHERE EXISTS(SELECT 1 FROM formdata_101 WHERE fbof
d87dd65 = v.x AND anonymised IS NULL AND status != 'draft');
```

Sur au moins un client, la première requête prend plus de 400ms alors que la deuxième ne va prendre qu'une milliseconde.
Le seul cas où cette écriture va dégénérer est le cas où l'optimiseur déciderait de scanner une fois la table par valeur listée, et si ces valeurs sont toutes absentes. Mais même dans le cas d'une longue liste d'options, l'optimiseur choisirait alors de n'effectuer qu'un seul scan de la table et de faire un hash, retombant alors sur les performances du select distinct actuel.

Autre solution, plus simple éventuellement mais impactant l'interface : aujourd'hui, les filtres ne listent pour ces champs que les options réellement utilisées, et donc extraites du SQL. Pourrait-on dans certains cas lister toutes les options et donc n'effectuer aucune requête SQL ?

Historique

#1 - 25 août 2022 12:41 - Frédéric Péters

> lister toutes les options et donc n'effectuer aucune requête SQL

Non. (par exemple les options peuvent venir de réservations d'événements et ces événements ne sont plus listés une fois la date limite de réservation atteinte).

#2 - 25 août 2022 12:44 - Pierre Ducroquet

Autre possibilité, exploiter les statistiques de PostgreSQL.
Par exemple :

```
# select most_common_vals from pg_stats where attname = 'fbofd87dd65' and tablename = 'formdata_101';
 most_common_vals
------------------
 {X1,X2}
(1 ligne)
```

Si toutes les valeurs possibles sont listées ici, alors il n'est pas nécessaire d'aller consulter la table pour vérifier leur présence.

#3 - 24 avril 2023 17:01 - Pierre Ducroquet

Nouvelle alternative : PostgreSQL ne gère pas nativement les "index skip scan" ou "loose index scan". Du coup quand on demande l'ensemble des valeurs distinctes d'une colonne, il ne va pas exploiter l'index, même si ça lui permettrait de scanner une quantité de données bien moindre.
On a donc la requête suivante :

```
SELECT DISTINCT on (fbo00f68ab4_9152_4baa_ba38_e3ca5482acae) fbo00f68ab4_9152_4baa_ba38_e3ca5482acae, fbo00f68
ab4_9152_4baa_ba38_e3ca5482acae_display
FROM formdata_101_allo_toulouse_signalements
WHERE anonymised IS NULL AND status != 'draft' AND fbo00f68ab4_9152_4baa_ba38_e3ca5482acae is not null
order by fbo00f68ab4_9152_4baa_ba38_e3ca5482acae;
```

Qui prend au moins 500ms sur l'instance toulouse.
Alors qu'avec l'écriture (compliquée) suivante, on passe sous les 2ms:

```
WITH RECURSIVE t AS (
   SELECT min(fbo00f68ab4_9152_4baa_ba38_e3ca5482acae) AS fbo00f68ab4_9152_4baa_ba38_e3ca5482acae FROM formdat
a_101_allo_toulouse_signalements WHERE anonymised IS NULL AND status != 'draft' AND fbo00f68ab4_9152_4baa_ba38
_e3ca5482acae is not null
   UNION ALL
   SELECT (SELECT min(fbo00f68ab4_9152_4baa_ba38_e3ca5482acae) FROM formdata_101_allo_toulouse_signalements WH
ERE anonymised IS NULL AND status != 'draft' AND fbo00f68ab4_9152_4baa_ba38_e3ca5482acae > t.fbo00f68ab4_9152_
4baa_ba38_e3ca5482acae)
   FROM t WHERE t.fbo00f68ab4_9152_4baa_ba38_e3ca5482acae IS NOT NULL
   )
SELECT fbo00f68ab4_9152_4baa_ba38_e3ca5482acae, (SELECT fbo00f68ab4_9152_4baa_ba38_e3ca5482acae_display FROM f
ormdata_101_allo_toulouse_signalements WHERE fbo00f68ab4_9152_4baa_ba38_e3ca5482acae = t.fbo00f68ab4_9152_4baa
_ba38_e3ca5482acae LIMIT 1) FROM t WHERE fbo00f68ab4_9152_4baa_ba38_e3ca5482acae IS NOT NULL;
```

Problème, cette écriture est plus rapide si la colonne est indexée (et j'ai indexé à l'époque des problèmes de perf de toulouse certaines colonnes).
Il faudrait donc deux choses :
1) pouvoir définir proprement quels champs en fbo*** doivent être indexés,
2) si un champ est indexé et qu'on demande ses valeurs distinctes, passer par la requête en SELECT DISTINCT.