w.c.s. - Development #58219

Éviter d'écraser la session quand seulement une clé d'un dico a changé

27 octobre 2021 11:23 - Benjamin Dauvergne

Statut:	Rejeté	**Début:**	27 octobre 2021
Priorité:	Normal	**Echéance:**	
Assigné à:		**% réalisé:**	0%
Catégorie:		**Temps estimé:**	0:00 heure
Version cible:			
Patch proposed:	Non	**Planning:**	Non

Description

L'objet session de w.c.s. contient pas mal de stockage clé/valeur assez basique (magictokens, anonymous_formdata_keys, visiting_objects_keys, data_source_query_url_tokens, et des choses hérités de quixote comme _form_tokens) mais qui interagissent mal avec le fait que différents workers peuvent se retrouver à écrire la session comme un tout alors qu'il n'y eu qu'un ajout/modification/suppression d'une clé dans un de ces dictionnaires / une de ces listes.

Il y aurait pour certaines API à stocker en plus de la donnée dans le dictionnaire, un diff dans un historique (genre self._changelog.append(('magictokens-set', key, value ou self._changelog.append(('mark-visited-object', formdata)), qui pourrait être rejoué lors du Session.store() au lieu d'écraser la ligne de la table session (si d'autres champs ont été modifié il faudra l'écraser quand même).

- Pour magictokens on pourrait imaginer une table toute bête :

```
CREATE TABLE sesssion_magictoken(
     session_id VARCHAR NOT NULL,
     key VARCHAR NOT NULL,
     value BYTEA,
     PRIMARY KEY (session_id, key))
```

Et utiliser l'historique pour y faire des modifications en cas de .store().

- Pour visiting_objects on pourrait rester une colonne de la table SQL session mais il faudrait s'arranger pour faire des UPDATE de seulement cette colonne en cas de changement sur celle-ci uniquement et stocker les timestamps dans une colonne JSONB sur le coté, ça donnerait le SQL suivant :

```
-- suppression des objets visités expirés
UPDATE session SET
   visiting_objects_timestamps = (SELECT json_object_agg(key, value) FROM json_each(
visiting_objects_timestamps) WHERE key NOT IN %(deleted_formdata_object_keys)s),
   visiting_objects_keys = (SELECT ARRAY_AGG(key) FOR UNNEST(visiting_objects_keys) AS k(key)
WHERE key NOT IN %(deleted_formdata_objects_key)s)
WHERE id = %(session_id)s;
-- ajout des nouvelles entrées
UPDATE session SET
   visiting_objects_timestamp = visiting_objects_timestamps || %(new_visiting_objects_timestamps)s
::jsonb
   visiting_objects_keys = (SELECT array_agg(key) FROM (SELECT * FROM UNNEST(visiting_objects_keys
) AS a UNION SELECT * FROM UNNEST(%(new_visiting_objects_keys)s) AS b) AS c(key))
WHERE id = %(session_id)s;
```

Tout ça pour donner une idée du chemin, la priorité pour moi étant surtout magictokens.

Demandes liées:

Lié à w.c.s. - Development #24635: Générer un identifiant pour un formdata dè...	**Nouveau**	**19 juin 2018**

Historique

#2 - 27 octobre 2021 11:36 - Benjamin Dauvergne

Si en plus on peut s'assurer que magictokens ne contient que des données sérialisables en JSON ce serait cool, ça nous donnerait plus de visibilité sur le contenu.

#4 - 29 octobre 2021 11:41 - Benjamin Dauvergne

- Lié à Development #24635: Générer un identifiant pour un formdata dès la saisie ajouté

#5 - 29 octobre 2021 11:42 - Benjamin Dauvergne

Je lie #24635 parce que le chemin proposé dans ce ticket fait disparaître le besoin du dico magictoken normalement.

#7 - 05 janvier 2022 18:13 - Benjamin Dauvergne

- Statut changé de Nouveau à Rejeté

Pas important, c'est corrigé autrement sur le ticket client.