w.c.s. - Development #66816

Race condition possible dans le stockage sql des "transient data"

30 June 2022 01:54 PM - Pierre Ducroquet

Status:	Fermé	**Start date:**	30 June 2022
Priority:	Normal	**Due date:**	
Assignee:	Pierre Ducroquet	**% Done:**	0%
Category:		**Estimated time:**	0:00 hour
Target version:			
Patch proposed:	Yes	**Planning:**	No

Description

En regardant les commits récemment apparus dans main, j'observe le code suivant (#66620):

```
    @guard_postgres
    def store(self):
        sql_dict = {
            'id': self.id,
            'session_id': self.session_id,
            'data': bytearray(pickle.dumps(self.data, protocol=2)),
            'last_update_time': now(),
        }

        conn, cur = get_connection_and_cursor()
        column_names = sql_dict.keys()
        sql_statement = '''UPDATE %s SET %s WHERE id = %%(id)s RETURNING id''' % (
            self._table_name,
            ', '.join(['%s = %%(%s)s' % (x, x) for x in column_names]),
        )
        cur.execute(sql_statement, sql_dict)
        if cur.fetchone() is None:
            sql_statement = '''INSERT INTO %s (%s) VALUES (%s)''' % (
                self._table_name,
                ', '.join(column_names),
                ', '.join(['%%(%s)s' % x for x in column_names]),
            )
            cur.execute(sql_statement, sql_dict)

        conn.commit()
        cur.close()
```

Ce code souffre d'au moins une race condition, qu'on peut heureusement régler facilement tout en le simplifiant grâce à l'UPSERT introduit dans PostgreSQL 9.5. Si deux instances veulent stocker la donnée en même temps, les deux instances verront leur update échouer, et l'une d'elles aura un insert qui échouera.

Il suffit de remplacer le code SQL par un insert on conflict update pour éviter tout de suite ce bug, et en prime simplifier le code et réduire le nombre de requêtes.

Associated revisions

Revision fa129af6 - 05 July 2022 11:41 AM - Pierre Ducroquet

sql: use insert on conflict to store transient data (#66816)

History

#1 - 30 June 2022 02:06 PM - Frédéric Péters (de retour le 10/10)

À noter que cette construction se trouve répandue, si une correction pouvait simplifier globalement plutôt qu'introduire une manière de faire particulière pour ces objets particuliers, ça serait top.

#2 - 01 July 2022 07:20 AM - Frédéric Péters (de retour le 10/10)

Je vois maintenant le patch de la branche et je me dis qu'on peut en fait y aller avec juste ça, cette classe a quand même la particularité d'un id

random attribué côté python, plutôt que les séquences postgresql qu'on trouve ailleurs.

#3 - 04 July 2022 10:56 AM - Pierre Ducroquet

- File 0001-sql-use-insert-on-conflict-to-store-transient-data-6.patch added

- Status changed from Nouveau to Solution proposée

- Patch proposed changed from No to Yes

Du coup je propose le patch. J'ai pas vu d'autres endroits évidents à changer, mais je garde ça en tête si je vois un schéma se répéter...

#4 - 05 July 2022 08:45 AM - Frédéric Péters (de retour le 10/10)

- Status changed from Solution proposée to Solution validée

#5 - 05 July 2022 11:42 AM - Pierre Ducroquet

- Status changed from Solution validée to Résolu (à déployer)

Mergé

```
commit fa129af6c994801b4057a3bde94e211c2580cd08 (HEAD -> main, origin/main, origin/HEAD, wip/66816-possible-ra
cecondition)
Author: Pierre Ducroquet <pducroquet@entrouvert.com>
Date:   Thu Jun 30 13:59:31 2022 +0200

    sql: use insert on conflict to store transient data (#66816)
```

#6 - 05 July 2022 10:14 PM - Transition automatique

- Status changed from Résolu (à déployer) to Solution déployée

#7 - 04 September 2022 04:42 AM - Transition automatique

Automatic expiration

Files

0001-sql-use-insert-on-conflict-to-store-transient-data-6.patch	1.37 KB	04 July 2022	Pierre Ducroquet