Autovacuum Wraparound

Last updated by | Lisa Liu | Nov 6, 2020 at 10:35 AM PST

Auto-Vacuum:

Before discussing about wraparound in auto-vacuum, please check below links for more info about auto-vacuuming and vacuum:

- Vacuum
- Auto-vacuum

What is Auto-vacuum wraparound and how its raised:

PostgreSQL's MVCC transaction semantics depend on being able to compare transaction ID (XID) numbers: a row version with an insertion XID greater than the current transaction's XID is “in the future” and should not be visible to the current transaction. But since transaction IDs have limited size (32 bits) a cluster that runs for a long time (more than 4 billion transactions) would suffer transaction ID wraparound: the XID counter wraps around to zero, and all of a sudden transactions that were in the past appear to be in the future — which means their output become invisible. In short, catastrophic data loss. (Actually the data is still there, but that's cold comfort if you cannot get at it.)

What is "FrozenTransactionId":

Normal XIDs are compared using modulo-232 arithmetic. This means that for every normal XID, there are two billion XIDs that are “older” and two billion that are “newer”; another way to say it is that the normal XID space is circular with no endpoint. Therefore, once a row version has been created with a particular normal XID, the row version will appear to be “in the past” for the next two billion transactions, no matter which normal XID we are talking about. If the row version still exists after more than two billion transactions, it will suddenly appear to be in the future. To prevent this, PostgreSQL reserves a special XID, FrozenTransactionId, which does not follow the normal XID comparison rules and is always considered older than every normal XID. Frozen row versions are treated as if the inserting XID were FrozenTransactionId, so that they will appear to be “in the past” to all normal transactions regardless of wraparound issues, and so such row versions will be valid until deleted, no matter how long that is.

Parameters controls XID age:

vacuum_freeze_min_age

vacuum_freeze_table_age

autovacuum_freeze_max_age

How to Avoid:

To avoid this, it is necessary to vacuum every table in every database at least once every two billion transactions.

The reason that periodic vacuuming solves the problem is that VACUUM will mark rows as frozen, indicating that they were inserted by a transaction that committed sufficiently far in the past that the effects of the inserting transaction are certain to be visible to all current and future transactions.

IF auto-vaccum is running with wraparound, what to do?

1. You can work on tuning parameters above, using guidelines from PostgreSQL documention: Vacuum wraparound but please coordinate with PG.

2. If table size was big (more than 200G) and customer is doing heavy DMLs, suggest table partitioning:

   - Declarative partitioning for version 10 and above

   - Basic Partitioning for version 9.5 and 9.6

Please refer to public documentation for more info about Auto-vacuum Wraparound:
https://www.postgresql.org/docs/11/routine-vacuuming.html#VACUUM-FOR-WRAPAROUND

How good have you found this content?