Table of Contents

Configure CG for application servers

We didn't find memory leaks in JVx, but sometimes memory consumption is higher than expected. We tried a lot of settings and profiling tools to find problems in JVX, without success. In the end we configured the gc to free memory reliable. With one of our JVM settings, application server memory consumption is as expected:

- G1GC *(available in Java 8 and later) memory efficient, good for small memory configurations*

  ```
  -XX:+UseG1GC -XX:+UnlockExperimentalVMOptions -
  XX:+UseStringDeduplication -XX:InitiatingHeapOccupancyPercent=1
  ```

  This will release all the memory sooner or later even when idle (8 minutes)

- ZGC *(JVM >= 11, for very large memory configurations, needs more memory)*

  ```
  -XX:+UseZGC -XX:+ExplicitGCInvokesConcurrent -XX:ZCollectionInterval=30
  ```

- ZGC *(JVM >= 21, for very large memory configurations, needs more memory)*

  ```
  -XX:+UseZGC -XX:+ZGenerational -XX:+UnlockExperimentalVMOptions -
  XX:+ExplicitGCInvokesConcurrent -XX:+ZProactive -
  XX:ZCollectionInterval=30
  ```

Postgres: set sequence to max ID

Use following statement in your postgres database to get a list of **setval** statements which will set the sequences to the max value.

```
SELECT t.table_name,
       'SELECT setval(''seq_'|| t.table_name || '_id'', max(id)::integer)
FROM ' ||t.table_name
  FROM information_schema.tables t
 WHERE table_schema = 'public'
   AND table_type = 'BASE TABLE';
```