



Toolforge webservices are in the final stages of [migrating to the toolforge.org domain](#) .
Please help us clean up older documentation referring to tools.wmflabs.org!

Incident documentation/20190110-WDQS

[< Incident documentation](#)

Contents [\[hide\]](#)

- 1 [Summary](#)
- 2 [Timeline](#)
- 3 [Conclusions](#)
- 4 [Links to relevant documentation](#)
- 5 [Actionables](#)

Summary

WDQS update process started failing on wdqs1007 and shortly after on wdqs1008. No further updates were possible though read queries proceeded normally.

Further details on Phabricator: <https://phabricator.wikimedia.org/T213210>

Timeline

- Jan 7 22:57:10 wdqs1007 started producing errors on writes
- Jan 8 around 03:00:00 wdqs1008 started producing errors on writes
- Jan 8 the cause identified as Blazegraph allocator limit (see below)
- Jan 9 12:02:42 wdqs1007 and wdqs1008 database was reloaded from wdqs1004 and full functionality restored

Conclusions

Blazegraph has a limit of 256K allocators (see below), which has been hit on these database. It is unclear why these databases hit it while others did not, but due to the way that Blazegraph allocates things (e.g. literal/term dictionary entries are never removed even if triples containing them are deleted) it may depend on the workload and previous history. If the database at the limit for allocators, it will not be able to create new memory allocations, and thus writes will fail.

We need to readjust how we use the database to preclude exhausting the number of allocators.

Links to relevant documentation

- Discussion on Blazegraph issue tracker: <https://github.com/blazegraph/database/issues/114>
- <https://wiki.blazegraph.com/wiki/index.php/FixedAllocators>
- Phabricator umbrella task: <https://phabricator.wikimedia.org/T213210>

Actionables

- We need to take some measures to ensure we're not hitting allocator limits now or in the future:
 - **Done** categories namespace has non-trivial size, splitting it to separate database (Blazegraph instance) would probably free up some allocations and thus alleviate allocator problem for a while
 - We are using so called "raw record" mode, which means memory for literals is allocated directly. This consumes a lot of allocators. Turning it off would allocate literals together with index pages, reducing allocator numbers.
 - We may consider inlining more URIs - such as values and references - which switch them from allocated space to in-index space. While it may not be significant savings in terms of storage space, it would make them not consume allocators either.
- **Done** We may want to add some monitoring to allocator counts (available under `status?` `dumpJournal`) to ensure that when we're nearing a dangerous place we at least get warned about it.

[Main page](#)

[Recent changes](#)

[Server admin log \(Prod\)](#)

[Server admin log \(RelEng\)](#)

[Deployments](#)

[SRE/Operations Help](#)

[Incident status](#)

[Cloud VPS & Toolforge](#)

[Cloud VPS documentation](#)

[Toolforge documentation](#)

[Request Cloud VPS project](#)

[Server admin log \(Cloud VPS\)](#)

[Tools](#)

[What links here](#)

[Related changes](#)

[Special pages](#)

[Permanent link](#)

[Page information](#)


[Cite this page](#)

[Print/export](#)

[Create a book](#)

[Download as PDF](#)

[Printable version](#)

-  **Done** wdqs1006 also has low number of available allocators, we may want to reload its database as well.

Category: [Incident documentation](#)

This page was last edited on 7 February 2019, at 01:03.

Text is available under the [Creative Commons Attribution-ShareAlike License](#); additional terms may apply. See [Terms of Use](#) for details.

[Privacy policy](#) [About](#)

[Disclaimers](#) [Code of Conduct](#) [Developers](#) [Statistics](#) [Cookie statement](#) [Mobile view](#)

[Wikitech](#)

