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

Incident documentation/20200407-Wikidata's wb_items_per_site table dropped

< Incident documentation

document status: in-review

Summary

Impact: For about 20 minutes, almost all content page parses were broken. They received, along with any other user interaction with the `wb_items_per_site` table in some way, a database error message instead of page content. This affected many (but not all or even most) page views for logged-in users, and many (possibly most, but not all) edits made to wikis. For 19 hours, all users looking at pages rendered since the incident start received some incorrect metadata on pageviews (infobox content, cross-language links, wikdata item relation). Also, Wikidata's checks for duplicate site links failed, leading to many hundreds of duplicate items being created.

Cause: Wikidata's `wb_items_per_site` secondary table, used for the rendering of every page connected to Wikidata, was dropped by a mis-configured weekly cron script which executed the `update.php` code path, which itself had been misconfigured for eight years to drop this table. This immediately led to a `DBQueryError` on content page loads (reads). The table was re-created as blank, at which point pages began to paint again (though wrongly).

Contents [hide]

Timeline

All times in UTC.

2020-04-06

- 23:00 weekly cron from puppet `cron/wikibase/dumpwikibaserdf.sh` executes
- 23:02 The cron calls cron/wikibase/wikibasedumps-shared.sh, which calls `sql.php` to run a minor *ad hoc* query.
  - Due to phab:T157651, noted in February 2017, this ran `LoadExtensionSchemaUpdates`, bypassing the checks in `update.php` to prevent this ever happening in production
  - Due to a bug in Wikibase's LoadExtensionSchemaUpdates code dating from eight years ago, this wrongly dropped the `wb_items_per_site` table. **OUTAGE BEGINS**
- 23:02:13: first log message indicating the table is missing
- 23:03:57: first user report of an issue: `<NotASpy> just got an error - [Xou1IQpAIDEAAC74T5wAAAOQ] 2020-04-06 23:03:09: Fatal exception of type "Wikimedia\Rdbms\DBQueryError"` (log)
- 23:05:33: first automated report of the issue: `<icinga-wm> PROBLEM - MediaWiki exceptions and fatals per minute on icinga1001 is CRITICAL`
- …
- 23:26 Amir re-creates the table as blank. Users no longer receive error messages instead of content (but metadata might be inaccurate). **OUTAGE ENDS, USER IMPACT CONTINUES**
- 23:31 Amir triggers rebuildItemsPerSite to begin
- 23:59 Adam removes broken code from Wikibase's LoadExtensionSchemaUpdates in production

2020-04-07

- 01:06 - 01:08 James F removes sql.php in production
- 10:27 jynus: starting recovery on db1099:3318
- 10:41 addshore deploy update to rebuildItemsPerSite.php script that allows lists of ids (needed for repopulating the gap)
- 11:07 jynus: starting recovery on all s8 hosts
- 11:36 Amir1: stopped the rebuilt script
- 11:42 marostegui: Depool db1092, db1111, db1099:3318 for table rename
- 11:45 jynus: stopping s8 replication on db1116:3318, db1095:3318, db2079
- 11:50 jynus: renaming wb_items_per_site_recovered to wb_items_per_site on s8
  - marostegui Repool db1092, db1111, db1099:3318 after table rename **USER IMPACT REDUCED**
  - User impact reduced here as now the bulk of the data is back but some will be outdated (by 14.5 hours)
  - Bad entries of rendered pages also still exist in the parser cache
- 11:51 marostegui: depool db1126
- 11:52 marostegui: repool db1126
- 12:06 addshore: start rebuild script take 1
- 12:42 addshore: restart run 1 of the rebuild script
- 12:50 addshore: start run 2 of rebuild script (with no redirects in the list of items)
- 13:23 rebuild of tables from maint script is done (addshore) **USER IMPACT REDUCED**
  - At this stage the table contains mostly complete and up to date data (with a few thousand items with incorrectly stored links, see phab:T249613)
- 13:55 addshore: bad sync of change to CommonSettings aborted and reverted
- 14:08 addshore, RejectParserCacheValue entries during wb_items_per_site drop incident (1/14.5h)
- 14:15 addshore, RejectParserCacheValue entries during wb_items_per_site drop incident (2/14.5h)
- 14:25 addshore, RejectParserCacheValue entries during wb_items_per_site drop incident (4/14.5h)
- 14:35 addshore, RejectParserCacheValue entries during wb_items_per_site drop incident (8/14.5h)
- 14:56 addshore, RejectParserCacheValue entries during wb_items_per_site drop incident (10/14.5h)
- 15:17 addshore, RejectParserCacheValue entries during wb_items_per_site drop incident (12/14.5h)
- 17:55 addshore, RejectParserCacheValue entries during wb_items_per_site drop incident (14.5/14.5h) **USER IMPACT REDUCED**
  - At this stage all parser cache entries from the time that the table was missing or partial will be rejected
  - For the next 24 hours, anonymous users may still be served badly-rendered pages that have persisted in the Varnish cache
- …

TODO: Add epilogue/cleanup.

Mediawiki exceptions/fatals

Edge traffic error rate

Detection

Did the appropriate alert(s) fire? Was the alert volume manageable? Did they point to the problem with as much accuracy as possible?

TODO: If human only, an actionable should probably be to "add alerting".

- First report to #wikimedia-operations at 23:03:57 UTC `<NotASpy> just got an error - [Xou1IQpAIDEAAC74T5wAAAAQ] 2020-04-06 23:03:09: Fatal exception of type "Wikimedia\Rdbms\DBQueryError"` (log)
- icinga-wm reported problems a minute later at 23:05:33 UTC `<icinga-wm> PROBLEM - MediaWiki exceptions and fatals per minute on icinga1001 is CRITICAL: cluster=logstash job=statsd_exporter level=ERROR site=eqiad https://wikitech.wikimedia.org/wiki/Application_servers https://grafana.wikimedia.org/d/000000438/mediawiki-alerts?panelId=2&fullscreen&orgId=1&var-datasource=eqiad+prometheus/ops`
- There was **not** an alert that indicated this was happening for most parser cache renders, or that there was a high rate (>700/second) of Mediawiki serving 50X errors to the cache servers [1]. There should have been automated detection of such a condition, including paging SRE -- if this had happened at off-hours, it could have gone unnoticed by engineers for far too long.

Underlying problems

`wb_terms`, when introduced 7.5 years ago, was actually made by merging several tables into one. The code to update the database when you migrate from an old version checks if `wb_terms` exists and if not, it creates the `wb_terms` table and drops those old tables (like `wb_aliases`). It mistakenly had an important table in the list that we still use today: `wb_items_per_site` (here's the original patch). This bug went unnoticed for 7.5 years.

There's another aspect of the problem. The code that runs the database update (`update.php`) should never ever be run in production. After too many outages, `update.php` checks if you're in WMF production and it errors out if that's the case. But here comes another bug, this time in core: `sql.php` runs `LoadExtensionSchemaUpdates` which basically means one of the production scripts called `sql.php` (not `update.php`) that is being used to debug production database (an interactive shell so you can run db queries against production) mistakenly runs `update.php` for extensions behind the scene and as a bonus without the check that actually `update.php` does (to error if it's in WMF production). This is a quote from Tgr on the ticket three years ago: "This is a very, very ugly accident waiting to happen".

2020-04-07 around midnight CET, a cronjob for building the dumps called `sql.php` to run a query (instead of doing it directly), this has been being run weekly for years. Meaning update.php in production has been running every week on Monday 23:00 UTC for years without us noticing. At the same day, `wb_terms` table got renamed (as a step to drop the table phab:T208425). The `update.php` code path, thinking that our production database was an old version that didn't even have `wb_terms`, tried to create it and drop old tables and mistakenly dropped an important table with 70M rows.

Conclusions

What went well?

- Outage cause was root-caused quickly
- WMF Backup/restore system worked as intended, in terms of data recovery
- Developers familiar with the relevant codebase were readily available and actively working on a fix/mitigation
- DBAs were responsive despite late night/early morning time for them

What went poorly?

- Options for restoring such a large table from backup late at night with limited coverage were not straightforward or appealing. More availability/coverage would have allowed us to proceed with this earlier.

  Comment (jynus): I disagree- I think there was a missunderstanding thinking the last backup was 1 week old. However, that is not true, we could have done a full recovery of the full wikidata database in under 2 hours to all hosts (requiring read only), last backup was less than 24 hours old, and we had also binlogs. A decision was taken to do a partial, point in time recovery, as only non-user data was affected while keeping the site read-write (which I agree with, but which of course, takes more time). Of course, someone available has to know how and what to restore.

  More comments on the Discussion page
- No pages were sent
- Unnecessary GRANTS (DROP) on wikiadmin user

Where did we get lucky?

- This was "only" secondary data, and could be re-created by a maintenance script (albeit one that will take 1-2 weeks to run for 80 million items (rough estimate)).
- We've been running this cron for months/years, effectively running update.php on wikidatawiki every week, and this was the first time it shot us in the foot.

How many people were involved in the remediation?

- The initial response: 1 IC (SRE), 1 DBA, 1 SRE, 2 WMDE Developers, 3 WMF Developers, 2 volunteers
- Aftermath fixing a few hours later: 2 WMDE Developers, 2 DBAs

Links to relevant documentation

Add links to information that someone responding to this alert should have (runbook, plus supporting docs). If that documentation does not exist, add an action item to create it.

- Table documentation https://doc.wikimedia.org/Wikibase/master/php/md_docs_sql_wb_items_per_site.html

- Rebuild script https://github.com/wikimedia/mediawiki-extensions-Wikibase/blob/master/repo/maintenance/rebuildItemsPerSite.php

Actionables

Create a list of action items that will help prevent this from happening again as much as possible. Link to or create a Phabricator task for every step.

- Remove `ALTER` and `DROP` permissions from the `wikiadmin` user (which is used by automated crons). Create a separate user for humans performing schema changes. https://phabricator.wikimedia.org/T249683
  - Dumps processes probably shouldn't run with a SQL user that can alter any data, let alone drop tables.
  - Potentially, permissions should be much more fine-grained: separate users for maintenance scripts/crons; for testing ro queries; one for dumps generation; core vs extensions maintenance, etc.
  - Maybe `DROP TABLE` should not granted to anyone except DBAs?
  - Maybe "Database::dropTable" should check for production environment (similar to update.php) and fail if something like that happens.
- `sql.php` must re-implement `update.php`'s not-in-production checks if we're going to restore it.
- Having both `mwscript sql.php` and `sql` (aka `mwscript mysql.php`) which run *very* different code paths is confusing. Fix this.
- Change Wikidata cronjob to use `mwscript mysql.php` instead of `mwscript sql.php`: gerrit:587218
- `sql.php` must not run `LoadExtensionSchemaUpdates` hooks: phab:T157651
- Pursue restructuring Wikibase repository's SQL directory: phab:T205094
- Consider adding paging based on ats-be-perceived 50X error rate, which considering recent history looks to be a good signal. TODO make a task
- Wikibase schema updaters must not modify database directly phab:T249598

TODO: Add the #Wikimedia-Incident Phabricator tag to these tasks and move them to the "Follow-up" column.

Categories: Incident documentation | Incident documentation drafts