ONTOTEXT ANSWER:

Unfortunately, millions of customers on a single GraphDB deployment would usually be impossible. This is because each repository, when active, even at 0 triples, has to introduce some data structures and processing. We have estimated that this contributes to about 200 MB of heap usage. So, even though your database could be impossibly large, there would be a physical limit to how much you can grow. On the software side, though, GraphDB would not block you from making an arbitrary number of repositories.

Now, there's a caveat to this. This memory usage is for active repositories. Those are repositories which are in use. However, there's no way to deactivate a repository via the UI. The only surefire way to do it is a GraphDB restart, but calling repo.shtudown() with the RDF4J client would work as well for local repositories.

There's something else to consider – activating a repository is very easy to do "by accident". And accidents will happen with end-users. Any interaction with a repository, besides viewing its configuration, would activate it. So, a size check, or a health check would both activate the repository. And the workbench is not aware of a repository shutting down, so it would initiate a ping towards the currently "active" repository – where "active" is what is kept on the browser cache and not what is actually on the cluster – every 30 seconds. So, some user with a forgotten browser tab will keep their repository active in perpetuity.

To sum it up, while it is technically possible to have an astronomical number of repositories on GraphDB and function well, practically, there is a limit. Assume the worst case scenario, where all repositories are active all the time and budget your memory for that. You also need to take into consideration the total number of triples in your installation. If you want to run a setup for hundreds of users, we'd recommend a Kubernetes deployment which can scale dynamically. It is close to how we used to run the Ontotext Cognitive Cloud.

The repository per-customer is a fairly common situation, though we'll put it, rather as a "repository per customer group". This is because you rarely have a single person behind a given "customer". And there is no trade-off here. You can have a super-user, the administrator, who has read/write access to all repositories and can perform a federated query across them all. Or a user who only has an access to a subset of repositories. For example, the "ReadOnly For Group1", who can read repositories "Group1-1", "Group1-2" and

Copyright: Ontotext New York | Sofia | Basel
Reach out to us: info@ontotext.com

“Group1-3”.

Did this help you solve your issue? Your opinion is important not only to us but also to your peers.

Help us get better: review GraphDB on Gartner

Copyright: Ontotext New York | Sofia | Basel
Reach out to us: info@ontotext.com