ONTOTEXT ANSWER:

The official documentation on the topic of migration to GraphDB 10 is quite correct in explaining that in the general case, the migration is very straightforward. Just move your data and work directories and you are good to go. Usually, you wouldn't need to touch the configuration at all. However, there are a few pitfalls you may discover along the way.

Local clustering

For a long time, the easiest way to experiment with the cluster was to spin up a local instance and create a "cluster" that runs only on it. This is no longer possible – a cluster requires at least two instances. Of course, they can still be hosted on the same server, but they have to be separate processes.

Instance-wide cluster

Every repository that is part of the instance will now be clustered. You can no longer have some stray repositories on one instance and not on the others that are part of the cluster.

Instance naming

Mostly, this is a problem when working with Kubernetes. This issue may arise if the external name of the instance does not correspond to its hostname. So, one of the workers in the cluster tries to reach, e.g., "worker2", but it reports back that its name is actually "worker_02_remote". In order to combat this, you can overwrite the configuration using graphdb.vhosts and graphdb.hostname and setting them to the address you try to access the worker from.

The GRPC port

You may have missed the fact that GraphDB now works with two ports. One of them is the standard HTTP port that you use for the workbench. This defaults to 7200 and handles pretty much every GraphDB API call that you should worry about as a user. However, there is also

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

the remote procedure call port. It defaults to the value of the main port + 100, so usually it would be 7300. This port is responsible for internal cluster communication. So, if you employ any sort of network-level security, you may need to whitelist communication between the workers in your cluster on this port. If done with AWS, for example, this option would be part of your EC2 management tools.

Initial cluster startup with large data

If you have billions of triples, the initial cluster startup can be a bit slow. Under the hood, when GraphDB starts a repository, it first has to reindex all the entities and prepare its in-memory entity pool “dictionary”. However, that doesn’t stop you from trying to create a cluster. However, if the reindexing takes more than 10 minutes, the cluster creation will time out. One option to avoid this is to start the repository and wait for it to reindex. You will know either from the logs or when a trivial query you run responds near-instantaneously. The other option is to increase the timeout by setting graphdb.cluster.sync.timeoutS to a value greater than 600 (10 minutes).

Those are the most common changes you’ll have to take into account.

Don’t worry, the improved performance, reliability and easier overall administration are well worth it. And if you encounter some new problems that stop you from picking up GraphDB 10, why not drop us an email on graphdb-support@ontotext.com? We are always glad to help!

Help us get better: review GraphDB on Gartner

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