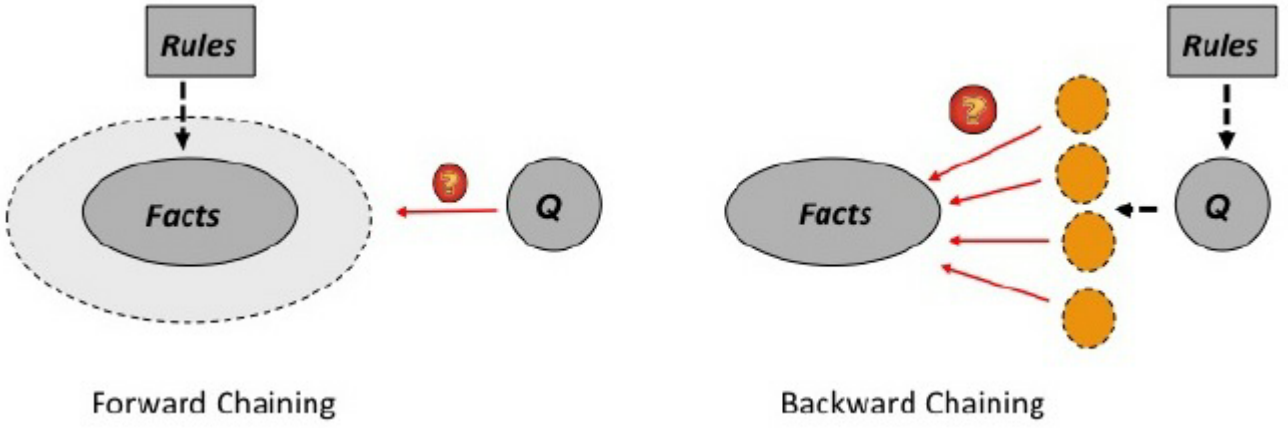
ONTOTEXT ANSWER:

Unless you are already running an “empty” ruleset repository, that’s the consequence of inference. The short explanation is that there are two types of inference – forward and backward chaining. Forward chaining starts at the data and infers all possible statements you can get. Backward chaining starts at the query and infers only the statements you can get given a particular query. Forward chaining happens at load time. Backward chaining happens at query time. This means that forward chaining has slower loads and faster queries, plus it is easier to manipulate the materialized triples. Backward chaining leads to fast loads and slower queries.



GraphDB is a forward chaining database. This means that all your statements are materialized during load time. The more statements there are, the harder it is to infer new data. This is why imports get slower over time.

If you want your import to go faster, you can always try to fine-tune the inference:

- **Use a simpler ruleset** – GraphDB comes with several flavors of rulesets. You can even disable inference altogether if you don’t need it.
- **Use a custom ruleset** – often, you will only need to use a specific thing from a given ruleset, and not all of the rules. For example, for you it may be important only that if some triple has the predicate ***example:address***, then its subject must have the class ***example:Location***. So, your ruleset may only state that rule. In general, it is good to start with a minimum viable ruleset and go from there. You can even debug any custom

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

rulesets.

- **Import now, infer later** – in GraphDB, you can import with Preload or disable inference for a particular transaction. Then you can trigger the reinfer action. This will make your partial data available for reads faster. You can also use system statements to change inference rulesets.
- **Scale vertically** – inference is a CPU-intensive task that can be parallelized. Therefore, adding more computational power would speed up your imports.

Remember that there's always a trade off in inference. Sure, the data import may be going slower than expected but, in return, queries are consistently fast and you can easily access these inferred triples programmatically.

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