Theme: Shanghai library FOLIO project

Time: May 24, 2022 07:00pm (EST) / May 25, 2022 07:00am (GMT+8)

Attendees:
Vincent Bareau (Enterprise Architect, EBSCO)
Gang Zhou (Project manager, Shanghai library)
Sha Jiang (Technical Director, Jiatu)
Lucy Liu (Product Owner, Folio China)

Notes:

We are testing circulation modules in the SHL's environment and are trying to add stress to the system. When there are 10 queries per second, the response time is about one second per query, which is acceptable. However, when there are 20 queries per second, the average response time is about 3-4 seconds per query. The longest time can be over 10 seconds. We noticed that the response time in the DB was not very high. What might be the problem? How can we reduce the response time? (Sha Jiang)

Vince: What modules are the queries for?

Sha Jiang: Checkout.

Vince: Are you doing anything else on the system at the same time? Are there other things going on, like data import, for example.

Gang Zhou: It's in the performance testing scenario. We are only testing the checkout.

Vince:

- We've seen that pattern before. But it's usually in performance testing with combination activities. We can test checkin and checkout in isolation. We can also test it when we are doing something else. And the biggest part in the system is data import. That's when something has been observed.
- The one second response sounds normal. I am not sure why increasing the number of queries per second will bog down the system. It seems to be non-linear in what you're describing. If you double the query per second, you shouldn't be getting 3-4 times longer on a query or more.
- CQL has to go through some code parsing of the logic of the module, which is the RMB logic for the module for CQL parsing. And then there are database operations. But before you go there, there's traffic to go through Okapi. And then there's the database itself.
- So do you have any sense of where the bottleneck might be? Are you able to measure DB performance and the performance of Okapi at the same time to see if those are also increasing in a non-linear session?

Sha Jiang: We found it might be waiting for Okapi to respond. When we check out items, there are a lot of Okapi apis in this action. And every api is waiting.

Vince:

- So it might be waiting. But it's important to see which part is waiting and which part is not.
- What version of FOLIO are you using right now for the test?

Sha Jiang: Goldenrod.

Vince:

- That can be part of the problem. The current release is Lotus. There have definitely been some performance improvements in the versions between Goldenrod and Lotus. That's definitely one thing to consider.
- It included changes to Okapi to fix some potential issues, which Okapi might be a problem. And there are probably some fixes down at the database level, too, either with indexes or with the queries themselves that are generated by RMB to do the transaction on the database.
- If you want to verify where the problem might lie, I know that in the checkout operation one of the last steps in the workflow is a write operation that happens in inventory and the very last thing that happens is that an item status is updated in the database inventory. So a write operation may lead to locks on the database. If you increase the volume, it may cause a backup because it's still busy writing. That might be one place that you can look at to see if you have that particular issue.
- You need to identify at which level the operation is being blocked. If it's in the database, you need to implement some scaling on the database. If it's Okapi, that is where you need some scaling. So you need to know how far down your bottleneck is going to be.

Sha Jiang: The FOLIO version is Goldenrod. But we have updated Okapi to Kiwi version.

Vince:

- Kiwi is better, but also a little bit behind. It reduces the possibility that Okapi is the problem. Okapi may be or may not be the issue. Again, the first thing you need to do is investigate your installation to find out where the bottleneck is.
- You run your test and observe IO, CPU usage and memory usage on the different servers of database, Okapi and also the modules. You should see a nonlinear increase in the response times. And you'll see if you can see it at the same time in Okapi. And if you do, you can look and see if you have it on the database. And if you see it on the database, then you'll know it's the database problem. if you see it's in Okapi but not the database, then it's an Okapi problem. That's my advice.
- Nevertheless, even if you have a more recent version of Okapi, the modules themselves are still Goldenrod, which implies that you have an old version of RMB. So RMB has a parser RMB for CQL. RMB generates queries that will look at the database. Those may be an area causing the problems on the application level because it's an old version and may be not sufficient to just increase the version of Okapi.

Sha Jiang: We did some analysis on the servers and found that there was quite high usage on the CPU. Any suggestions for this?

Vince: That's an indication that you need more capacity on the database, larger CPUs, qurums, or whatever.

Sha Jiang: This server is only for FOLIO, not DB.

Vince: Is the database on the same server of Okapi and modules?

Sha Jiang: No.

Vince: Good. I don't know the specs of your server for the database, how many quorums you have and how much memory you have. Is it sufficient?

Gang Zhou: FOLIO server - 16-Core in the CPU. 32 bits memory. Database server - 16-Core; 64 bits memory.

Vince:

- That seems you should do 20 queries a second. Just my guess.
- It sounds like your problems are on the database server because of what you're saying about high CPU.
- I don't know exactly where the problem is at this point. I am just giving you some advice on where to look. One particular problem is on this kind of checkout test, you are going to do updates on the items. When you do the check, change item status. When you do a write operation on the database, you will trigger the recalculation of the indexes. If you have too many indexes and you do too many write operations, you will slow down because you get high CPU usage and because the database is reindexing frequently. It becomes nonlinear at some point because you are putting in more requests faster than the database can process. That's one possibility you might want to look into.
- If that turns out to be the case, there are solutions for that sort of thing by changing the configuration of reindexing or using replicas on the database. You have to speak to somebody who is closer to the database configuration than I am for that.
- Have you modified the database with additional indexes?

Sha Jiang: That might be a problem. We have many indexes in item databases.

Vince: Are those normal indexes provided by FOLIO, or you added your own indexes?

Sha Jiang: We added our own indexes.

Vince: That might be an area to look at to see if you can optimize your indexing because that will cause performance loss if you have reached the threshold of the query rate.

Gang Zhou: FOLIO used JSONB columns in the postgreSQL database. We have to create some additional indexes for some huge tables such as item and instance. The additional indexes are necessary in some scenarios. What suggestions do you have if the bottleneck is in the DB or is an index problem?

Vince:

- This is more deep into database optimization. I am not an expert on the optimization of databases.
- There are multiple types of indexes you can create. You can control the conditions on which the reindexing occurs. Just do a test. You might turn off an index or most of them.

And run your performance test. If the problem goes away, it just confirms the indexing is causing the problem. And you have to reexam your indexes and see if they can be changed or combined. Sometimes combining indexes can eliminate a number of different ones. Just to see if you can make the system less demanding.

- In the Lotus version, there has been an effort to remove some of the indexes existing in inventory. The reason is that now we have adopted Elasticsearch to do the search. The indexes were there before because the database itself was providing search functionality. But now, since the database is not needed to do an immediate search. Instead, we use ElasticSearch, we don't need the indexes anymore. So there was an effort to remove some of the indexes, which is intended to help with the performance. You are also using ElasticSearch, so maybe you don't need some of the indexes here.

Sha Jiang:

- We can have a try and have someone look into the database configuration.
- Do you think adding instances for modules will help improve performance?

Vince: It depends on where the problem is. If the bottleneck is really the database, it can't make a difference. If the issue is that the module can't handle 20 queries a second and you add one more instance for modules, it may help since each will see 10 queries a second. But both modules will go through Okapi. Okapi still sees 20 queries per second. And both modules will go to one database. And the database also sees 20 requests per second. So It will only help if the problem is only in the modules.

Sha Jiang: The logs indicate the problems might be the modules.

Vince: You can try that and see what difference it makes.

Sha Jiang: What are the practices at other institutions? How many instances are enough for 20 queries per second?

Vince: The PTF team has done a number of tests to qualify the capacity and the performance. You may check their wiki page to get more info. Martin Tran from EBSCO is the lead for PTF.

Lucy: We had a meeting with Martin two years ago. We can invite him to another meeting if you need. A Slack chat will be created to start the discussions.