


Research database options to support table display functions

 [DM-9888](#) - Jira project doesn't exist or you don't have permission to view it.

Loi Ly finished this ticket and made the report:

With the need to support more advanced table manipulations, we need to replace our internal fixed-length IPAC table format with something more suitable. This ticket is to research and report on the pros and cons of selected embedded/in-memory databases as well as preliminary performance expectations.

The 3 popular java-based embedded databases I looked at are Derby, H2 and HyperSQL. I list the pros and cons of each below.

Derby:

Pros:

- full-functioned relation database with lots of features
- lots of documentation

Cons:

- slowest

H2:

Pros:

- simple and fast
- full-text search
- embedded sql client via servlet

Cons:

- one author

HyperSQL

Pros:

- great performance with lots of features
- good documentation

Cons:

- slow down proportionally to the size of data
- I've also recorded some performance numbers during the evaluation. The operation I am recording is the time it takes to persist an in-memory table. Table size is the file size in ipac table format. Times are in seconds.

The test is done on my laptop with SSD drive.
The databases are tuned to use disk-base tables with a large cache size.
Code is written using jdbc driver with pre-compiled prepare statement and bulk insert.

table size	Derby	H2	HyperSQL	IPAC Table
50k rows, 25 cols, 21MB	4.8	1.0/0.5	0.6/0.3	1.8
300k rows, 25 cols, 128MB	28.4	3.6/2.1	4.7/6.1	10.9
1million rows, 25 cols, 432MB	108.4	10.5	31.7/59.0	35.7

~* if there are 2 numbers, then it's first try vs subsequence tries~

In conclusion, I feel comfortable recommending H2 based on our needs. Although, I am sure HyperSQL would be just as good after additional tuning.