MySQL, PostgreSQL, and LevelDB Performance

To test the performance of MySQL and PostgreSQL persistence compared to LevelDB, I ran a series of tests against Fedora 4 with each backend configured. The tests performed 200 batches, with each batch consisting of:

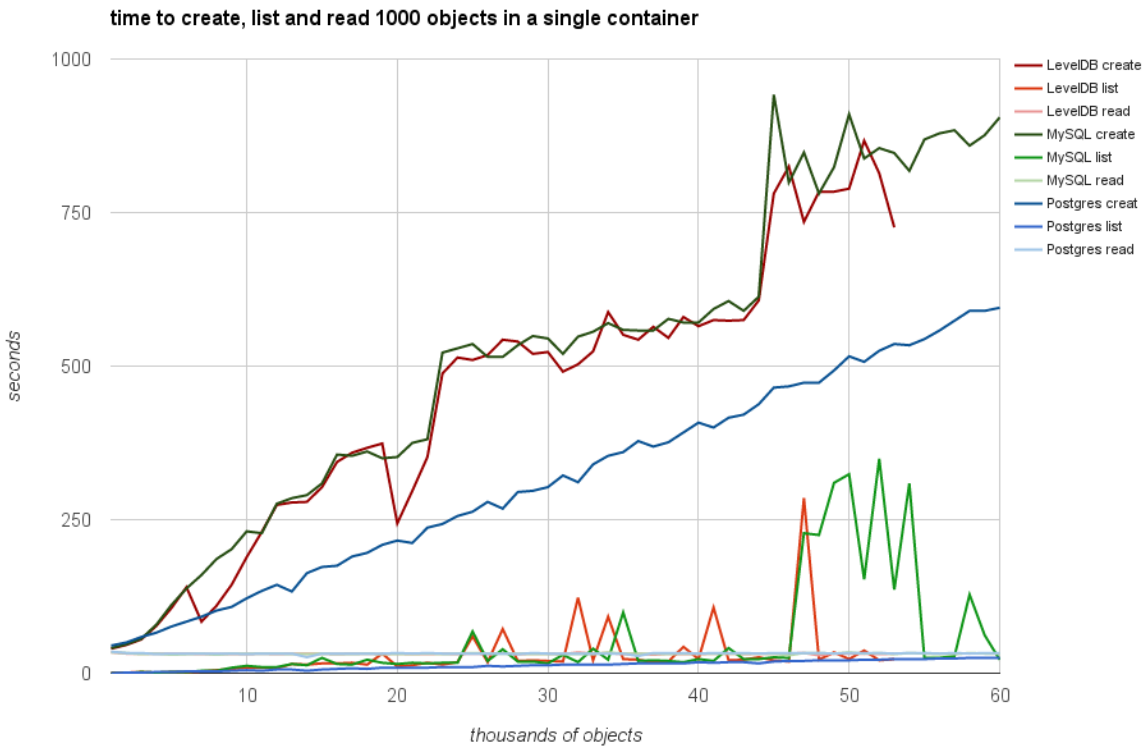
1. Creating 1,000 objects with a basic Dublin Core metadata record
2. Listing 1,000 objects
3. Retrieving the RDF for the 1,000 objects

Each step was timed, and graphs of the times are shown below. The test scripts are available to view/download.

Using Client-Supplied Identifiers (Single Container)

Using client-supplied identifiers, placing all child objects in a single container, the performance of MySQL and LevelDB was very similar, and PostgreSQL performance was a little better:

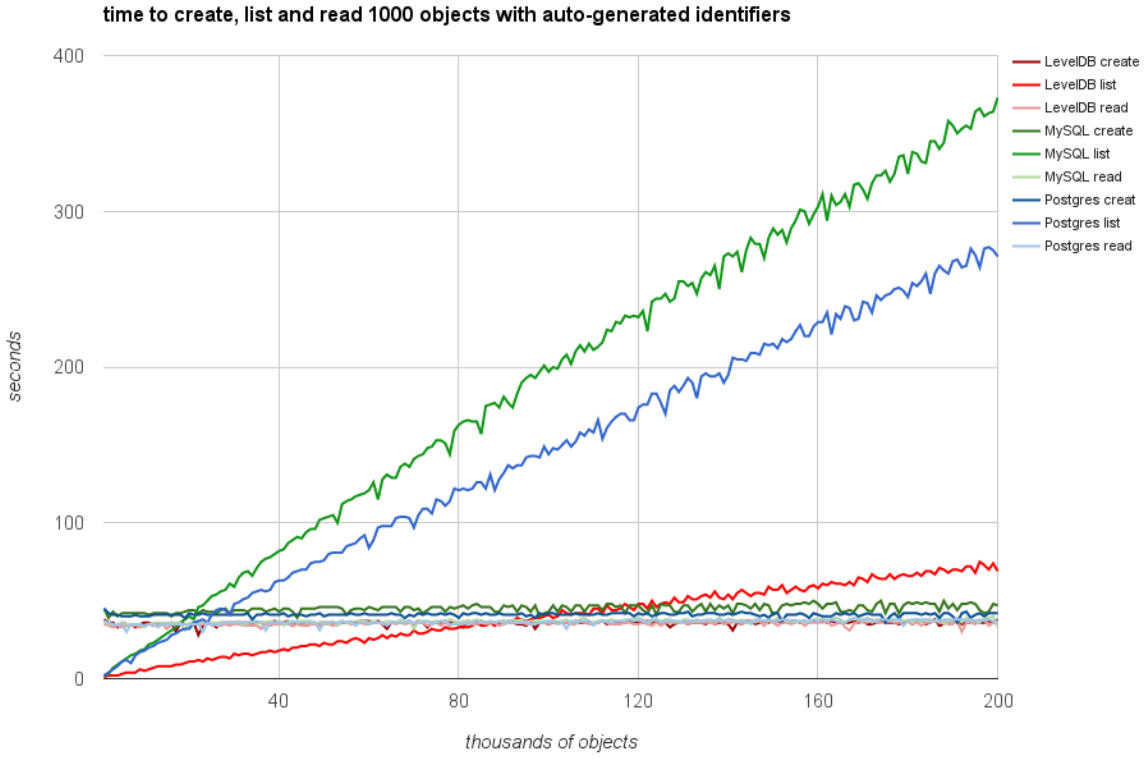
- All three had list and read times that were essentially flat as the number of child objects grew
- LevelDB and MySQL had roughly the same performance for creating objects, but PostgreSQL had better and more consistent performance
- LevelDB and MySQL both had increasingly erratic performance after around 45,000 child objects were created, while creating objects in PostgreSQL scaled linearly.



Using Auto-Generated Identifiers

Using auto-generated identifiers, placing child objects in a 4-level hierarchy, the performance was similar for create and read times, but significantly different for list operations:

- All three had create and read times that were essentially flat as the number of child objects grew
- All three had list times that grew as the number of child objects grew, but the PostgreSQL list operations grew more quickly, and the MySQL list operations grew even more quickly


time to create, list and read 1000 objects with auto-generated identifiers
400
300
200
100
0
seconds
40
80
120
160
200
thousands of objects
LevelDB create
LevelDB list
LevelDB read
MySQL create
MySQL list
MySQL read
Postgres creat
Postgres list
Postgres read