School of Computer Science

COMP47470

Project 1
Performance Testing of Two Different Database Management Systems

Teaching Assistant:	Leandro Almeida
Coordinator:	Dr Anthony Ventresque
Date:	Tuesday 14th February, 2017
Total Number of Pages:	3

General Instructions

- This project has four main objectives: to make you install and use a relational data base (PostgresSQL or MySQL), to make you write programs using database connectors (e.g., JDBC), to make you install and use a NoSQL database (e.g., MongoDB), and finally we want you to design and run some performance tests on two different database systems: relational and NoSQL (e.g., document-based).
- You are encouraged to collaborate with your peers on this project, but all written work must be your own. In particular we expect you to be able to explain every aspect of your solution if asked.
- We ask you to hand in an archive (zip or tar.gz) of your solution: code/scripts and a 5-10 page pdf report of your work (no need to include code in it).
- We also ask you to give us the url of a VM (that we could download and run with VirtualBox, or in a Cloud instance, etc.) that would contain all your data and programs. Running your code should be as simple and easy as possible so make sure this is explained well in a README.txt file describing how to run your programs.
- The report should include the following sections:
  1. a short introduction
  2. a requirement section that answers the question *what* is the system supposed to do
  3. an architecture/design section that answers the question *how* your solution has been designed to address the requirements described in the previous section
  4. a series of sections that describe the different challenges you faced and your solutions. For instance, take one of the script, describe the difficulty you faced and your solution. These sections can be short – the objective here is to show how you crafted the solutions with the tools you have learnt so far.
  5. a short conclusion
- The project is worth **10% of the total grade** for this module. The breakdown of marks for the project will be as follows:
  - Import IMDB in to a RDBMS: 15%
  - Extraction of IMDB from your RDBMS database into a NoSQL database: 30%
  - performance/load testing (design of some scenarios and analysis of the results): 30%
  - Report: 25%
- **Due date: 10/03/2017**

1 Import IMDB in a RDBMS

IMDB (the Internet Movie Database) ”is an online database of information related to films, television programs and video games, including cast, production crew, fictional characters, biographies, plot summaries, trivia and reviews.” IMDB’s database is not huge but still contains millions of films, series and individuals[1]. This will be enough for you to test the (relative) performance of different platforms and tools.

There are various ways to collect and store IMDB’s database into a Relational Database, for instance this for Java: `http://www.jmdb.de` or this for Python: `http://imdbpy.sourceforge.net`.

Your first task will be to use one of those tools to import the IMDB dataset into a RDBMS (Relational Database Management System): MySQL or PostgreSQL.

1. install MySQL or PostgreSQL
2. use one of the import tools to extract IMDB’s dataset and populate a database in your system
3. in the report, describe in details the schema of the database

2 Import IMDB in a NoSQL System

The second part of the project consists in writing your own program or script to populate a NoSQL database with the information stored in your RDBMS.

For instance (suggestion) you could read the data in your MySQL database and populate a MongoDB database with documents corresponding to movies (or actors, or both, etc.).

Here you are asked to come up with your own program (in Java or Python) that will use a database connector (e.g., JDBC for Java) to collect the data on one side and write it in the other side.

1. use a database connector (e.g., JDBC for Java or MySQL Connector for Python etc.) to read the information from your local IMDB database in your application
2. design document(s) for your document database (if, for instance, you use MongoDB) – this can be done with any type of NoSQL database management system.
3. populate the NoSQL database (e.g., MongoDB) using your program (e.g., `https://docs.mongodb.com/ecosystem/drivers/java/` for Java)
4. write some tests that verify that the transfer is correct: define some test scenarios/use cases and write some unit tests to check that a certain number of facts are equivalent in both databases (e.g., MySQL and MongoDB).

Note that the structure of the NoSQL database has an impact on the performance of the queries[2] you will be sending to the NoSQL database management system.

[1] `http://www.imdb.com/stats`

[2] See for instance this research paper: `https://hal.archives-ouvertes.fr/file/index/docid/1018129/filename/isstaws13ttc-id2-p-18133-final.pdf` for a description of this problem.

3 Performance Testing of both Database Systems

Now we want to perform some load testing of the two database systems, to evaluate their relative performance.

JMeter[3] is a very famous performance testing tool. Often, you would use JMeter to load/stress/spike test a web application or a web server – for instance to see whether your web application or web site can support a certain load and still give your users a good quality of service.

Here we want to apply the same techniques to database load/performance testing. This page: `http://jmeter.apache.org/usermanual/build-db-test-plan.html` is a good introduction to what JMeter can do for you here.

1. install JMeter and get familiar with its interface
2. run some simple testing plans for both your database management systems (i.e., define a number of users, number of queries, type of queries etc.)
3. create your own scenarios for performance testing of the database management systems. Describe well your scenarios (e.g., what type of application or users could have certain behaviour or pattern etc.) in the report
4. identify certain metrics that you want to record (e.g., response time, number/size of DB queries, number of failed transaction, CPU/Memory etc.). Some tools from the JMeter galaxy (e.g., `https://jmeter-plugins.org/wiki/PerfMon/`) can help you to collect information that JMeter does not provide natively
5. report your findings (we prefer images/graphs/interpretations of the results rather than just table after table of numbers)

4 Conclusion

Obviously we expect you to find something from this study of two different Database Management Systems, a Relational and a NoSQL systems. Do not hesitate to play with different parameters and to run as many tests as you can.

You won't be marked only on the findings you get, but on the whole process and engineering: ability to ingest and populate data, design schemas and experiments, analyse results, etc.

This assignment is your opportunity to think and be creative (not just apply a list of instructions). Don't be shy, there is no best solution: only good solutions – basically most solutions you'll get will be good as long as you can justify them.

Good luck!

[3]`http://jmeter.apache.org`