Query Performance

Jeremiah J. Flowers

Abstract—This project is ment to explore the performance difference between the difference relational database management systems, along with the different server-side scripts used to by websites. The databases will be similar across mulitples versions and will be tested against the same queries to be judged on time of completion. Server-side scripts will be used to execute the queries and will be looked at for any annomolies, such as compatability issues between script and database.

Index Terms—Relation Database Management System, Web Server, Query, Server-side Script

1 Introduction

THIS project will show which relational database management system (RDBMS) and server-side script combination are the fastest. The results should help developers build faster systems for clients.

The experiment is held online. There is a web server that will execute the queries. There are a couple databases setup separetly, with identical datasets, to process any query. The website will host multiple scripts that will run multiple queries to the multiple databases.

2 Setup

To host everything online, I used Amazon Web Service (AWS) to create the web server and databases. AWS is a cloud computing service that lets you allocate resources from Amazon to use for your virtual space.

2.1 Web Server

The web server is setup to host the website for the project. Using Windows 2012 R2 and Internet Information Services (IIS) I create a simple website that has PHP and ASP.NET 4.5 setup and ready to use. The default versions of these scripts do not come with connectors to other databases, so having to load the drivers was needed to get the scripts to connect properly to the databases.

The Oracle drivers for PHP and ASP.NET were not able to be installed. So no tests for Oracle database were able to happen.

2.2 Databases

There were two databases that were setup; MySQL and Microsoft SQL Server. The databases were populated with 4 tables. Three tables contained 5 millions rows and the last table contained million rows. A script with nested loops of insert statements was made in T-SQL. This script took 6+ hours to complete in SQL Server. Next I used a migrating data tool for MySQL to transfer the data from SQL Server to MySQL, which took even longer

I tried to get Oracle filled with data but the migration tool for Oracle would not transfer any of the data over. Next I tried to write a similar script in PL/SQL for Oracle but the script could not complete. The tool I was using for Oracle would timeout and crash. The only solution was to run the script in small increments.

2.3 Query

The queries I ran against the databases were all SELECT statements. There were 8 at first, then an additional 5 more later. The first four queries are simple “how fast can you retrieve the data from each of the tables?” The next two use the WHERE clause to limit the result set, and the last two use JOIN to combine three tables together.

After looking at the first sets of query times, I added 5 more that used JOIN and WHERE clause to show a growth of escalation. The five new queries are the same query as 8th query that joins the three tables, except for the added where CLAUSE. The 5 query ask to retrieve 100, 1000, 10000, 100000, and 1000000 rows.

3 RESULTS

Out of two scripts and two databases, there are 4 datasets to be looked out. All times are in seconds. Things to know; query 3 retrieves only 100,000 rows and queries 8 and beyond are retrieving 8 columns

3.1 PHP

MySQL			MS SQL	
1	6.026		1	0.153
2	5.990		2	0.002
3	0.098		3	0.002
4	2.088		4	0.002
5	2.044		5	0.006
6	5.141		6	0.002
7	---		7	1.141
8	---		8	---
9	---		9	.003
10	---		10	.003
11	---		11	0.073
12	---		12	0.718
13	---		13	1.575

As you can see, MS SQL times are a lot quicker than MySQL. The dashes represent the queries could not be completed because of a time out. Using AWS monitoring I can see that the connection is dropped even though the query looks like it may still be processing.

For MS SQL the last 5 querys show an escalation in time as the queries are retrieving more and more rows. The slope from query 11 to 12 seems to 1 to 1(time to row), while slope of query 12 to 13 is less steep with 1 to 5. Somewhere after 1 million rows, the queries time out and are unable to handle 5 millions rows (8^{th} query).

3.2 ASP.NET

MySQL			MS SQL	
1	5.859		1	2.125
2	5.281		2	3.078
3	0.047		3	0.047
4	1.234		4	19.594
5	2.016		5	1.219
6	4.906		6	3.531
7	---		7	2.5
8	---		8	---
9	128.375		9	0
10	---		10	0.016
11	---		11	0.141
12	---		12	1.469
13	---		13	14.344

ASP times for MySQL are very close to what PHP had. The only exception being query 9 being able to run, even though it took 2 minutes to finish. Query 9 retrieves only 100 rows from 3 tables that have been joined; this shows that MySQL doesn't optimize well when joining tables together.

For MS SQL the times for completion are not at all as quick as times for the PHP script. Only 1 out of the first 6 query hit under a second. Query 12 and 13 show the same huge step in query time.

4 DISCUSSION AND CONCLUSIONS

The the best performance for this project would be Microsoft SQL Server paired with PHP. MySQL can only handle the small basic task but when it is required to handle. The time outs on half the queries show that MySQL is only ment for small projects.

Oracle was unfortunately unable to be tested. I was unable to get PHP or ASP.NET to connect to the database. Even still so, getting the data to load into the database was a verly slow process. Weather it was the tool I was using or not that was causing the problem; this could be a sign that Oracle doesn't handle large amounts of data similairly to MySQL.

REFERENCES

[1] Vicknair, C., Macias, M., Zhao, Z., Nan, X., Chen, Y., & Wilkins, D. (2010, April). A comparison of a graph database and a relational database: a data provenance perspective. In *Proceedings of the 48th annual Southeast regional conference* (p. 42). ACM. Retrieved from: http://www.cs.olemiss.edu/~ychen/publications/conference/vicknair_acmse10.pdf