

Part 1

Introduction

For the project, we will implement a schema and benchmark a relational database system. The system will then be put through a grinder. Testing how the system performs with datasets of various sizes. And testing how the system performs when we tweak its settings.

System

The system we have chosen is MySQL.

MySQL has been around for 25+ years and has been crowned as the world's most popular DBMS. [source](#)

So it would be an utmost priority for two CS students, near graduation, to gain intimate knowledge and experience with such ubiquitous system.

To host the database, we went the cloud route. Additionally, using a virtual machine would grant us that sweet sweet extra credit. As a result, our database lives in Google Cloud Platform.

```
bbeyene@cloudshell:~ (db-imp)$ gcloud sql connect db-imp --user=root
Allowlisting your IP for incoming connection for 5 minutes...done.
Connecting to database with SQL user [root].Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1667
Server version: 8.0.18-google (Google)

Copyright (c) 2000, 2021, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use Benchmark;
ERROR 1049 (42000): Unknown database 'Benchmark'
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
4 rows in set (0.01 sec)

mysql> █
```

figure 1.

Data Generation

To generate useable and scalable data, we followed the guidelines for the Winconsin benchmark.

datagen.py can generate a specified number of tuples using the following format: python3 datagen.py (no of tuples).

figure 2. generating tuples

figure 3. ONEKTUP

figure 4. TENKTUP1

figure 5. TENKTUP2

Here's a demonstration the data generation and insertion. `mysqlinsert.py` is used to insert the csv files.

figure 6. create/insert script

figure 7. changing parameters for each table

figure 8. tables loaded in

figure 9. 4/16 columns from ONEKTUP

Mike's String Struggle

Earlier versions of the script used the string format of the early Wisconsin Benchmark. If I had read just a few more pages of the document, I would have realized the string format was updated in a later version of the Benchmark.

I ended up wracking my brain trying to figure out how to implement the algorithm to generate the string pattern. I spent too many hours staring at nested while loops. Losing track of several variables and perhaps 2% of my sanity (I'm being dramatic).

Luckily I met with Bruh the next day. He pointed out the updated string format - (7 letters [A-Z] followed by 45 x's) and generation pattern.

Long story short: read the whole document before you program!