CS 587 Database Implementation
Database Benchmarking Project (Part 1)

(Punam Rani Pal, Weiwei Chen)

Github Link: https://github.com/waiwaixiaochen/CS587-Database-Benchmarking

Description:

In this project, we have followed the Wisconsin benchmark project that have been applied on the different relational database management systems to measure the performance on various sets of queries and predict its efficiency.This project is a good practice for us to increase our understanding of Database Performance. We can get some experience with a relational-style data management system of our choice as well as programming experience.

we accomplished two tasks for the first part of the project:

- **Data generation:** For the data generation, we used Python script. The data model and the schema is based on the Wisconsin Benchmark. We generated the data directly into a system.
- **System selection:** We selected MySQL for database implementation and running queries.

System used:

For this project MySQL has been chosen because we got an opportunity to use postgresql in the database introductory class (CS586) and now we planned to experience a different system since MySQL is a stable and reliable open-source database, it also features a distinct storage-engine framework that facilitates system management to configure the MySQL database server for a flawless performance.
Following are the prominent benefits:

- User friendly
- On demand high scalability
- Data security
- High performance
- Supports transactional feature
- Ease of connection with different systems

Furthermore, it also ensures high speed and unique memory caches for enhanced performance on many cloud databases.

Data sample loaded:

We followed the relational schema specified on the Wisconsin paper. There are three tables in the database: OENKTUP of size 1000, TENKTUP1 and TENKTUP2 of size 10000 respectively. All the tables share the same schema given below:

```
CREATE TABLE TENKTUP1
(   unique1         integer NOT NULL,
    unique2         integer NOT NULL PRIMARY KEY,
    two             integer NOT NULL,
    four            integer NOT NULL,
    ten             integer NOT NULL,
    twenty          integer NOT NULL,
    hundred         integer NOT NULL,
    thousand        integer NOT NULL,
    twothous        integer NOT NULL,
    fivethous       integer NOT NULL,
    tenthous        integer NOT NULL,
    odd100          integer NOT NULL,
    even100         integer NOT NULL,
    stringu1        char(52) NOT NULL,
    stringu2        char(52) NOT NULL,
    string4         char(52) NOT NULL
)
```

Following are the constraints of each attribute:

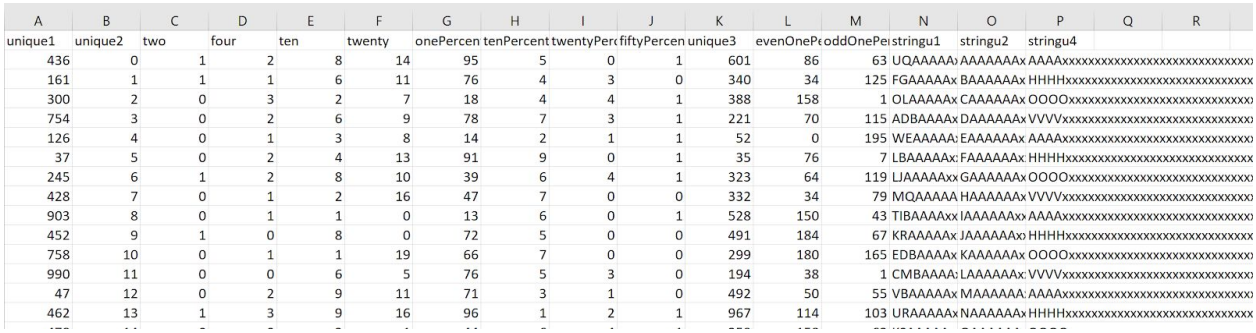
Attribute Name	Range of Values	Order	Comment
unique1	0-(MAXTUPLES-1)	random	unique, random order
unique2	0-(MAXTUPLES-1)	sequential	unique, sequential
two	0-1	random	(unique1 mod 2)
four	0-3	random	(unique1 mod 4)
ten	0-9	random	(unique1 mod 10)
twenty	0-19	random	(unique1 mod 20
onePercent	0-99	random	(unique1 mod 100)
tenPercent	0-9	random	(unique1 mod 10)
twentyPercent	0-4	random	(unique1 mod 5)
fiftyPercent	0-1	random	(unique1 mod 2)
unique3	0-(MAXTUPLES-1)	random	unique1
evenOnePercent	0,2,4,...,198	random	(onePercent * 2)
oddOnePercent	1,3,5,...,199	random	(onePercent * 2)+1
stringu1	-	random	candidate key
stringu2	-	random	candidate key
string4	-	cyclic	

Method for data generation:

- Established connection between MySQL server and the local machine Python editor.
- Populated records in each table by Python script.
- Exported the generated tables from MySQL Workbench as CSV files.

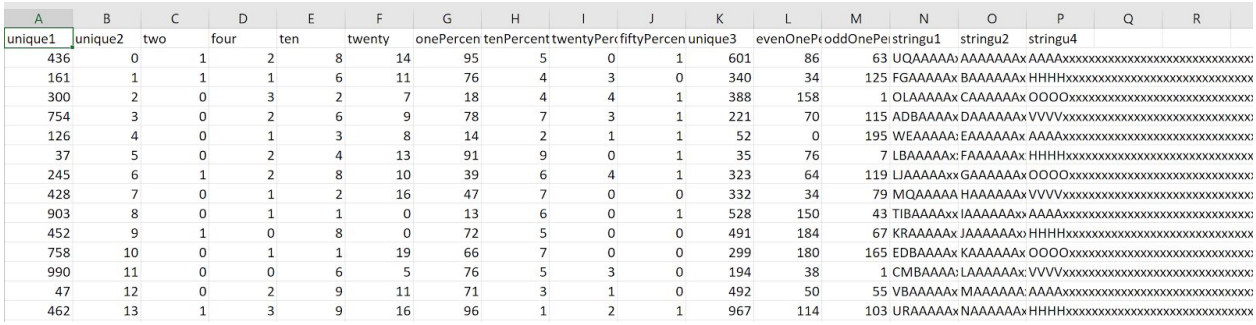
Examples of the generated tables:

Table ONEKTUP:



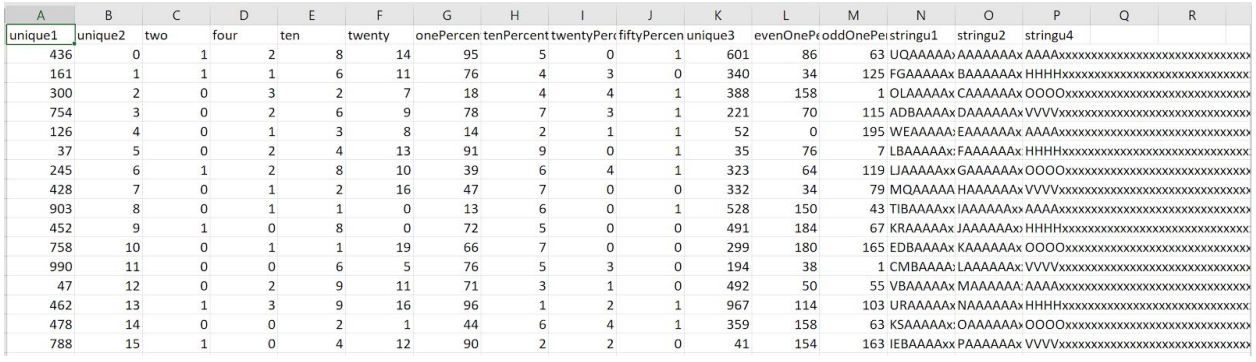
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
unique1	unique2	two	four	ten	twenty	onePercen	tenPercent	twentyPerc	fiftyPercen	unique3	evenOnePe	oddOnePe	stringu1	stringu2	stringu4		
436	0	1	2	8	14	95	5	0	1	601	86	63	UQAAAAA	AAAAAAA	AAAAxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
161	1	1	1	6	11	76	4	3	0	340	34	125	FGAAAAA	BAAAAAA	HHHHxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
300	2	0	3	2	7	18	4	4	1	388	158	1	OLAAAAA	CAAAAAA	OOOOxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
754	3	0	2	6	9	78	7	3	1	221	70	115	ADBAAAA	DAAAAAA	VVVVxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
126	4	0	1	3	8	14	2	1	1	52	0	195	WEAAAAA	EAAAAAA	AAAAxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
37	5	0	2	4	13	91	9	0	1	35	76	7	LBAAAAA	FAAAAAA	HHHHxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
245	6	1	2	8	10	39	6	4	1	323	64	119	LJAAAAA	GAAAAAA	OOOOxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
428	7	0	1	2	16	47	7	0	0	332	34	79	MQAAAAA	HAAAAAA	VVVVxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
903	8	0	1	1	0	13	6	0	1	528	150	43	TIBAAAA	IAAAAAA	AAAAxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
452	9	1	0	8	0	72	5	0	0	491	184	67	KRAAAAA	JAAAAAA	HHHHxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
758	10	0	1	1	19	66	7	0	0	299	180	165	EDBAAAA	KAAAAAA	OOOOxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
990	11	0	0	6	5	76	5	3	0	194	38	1	CMBAAAA	LAAAAAA	VVVVxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
47	12	0	2	9	11	71	3	1	0	492	50	55	VBAAAAA	MAAAAAA	AAAAxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
462	13	1	3	9	16	96	1	2	1	967	114	103	URAAAAA	NAAAAAA	HHHHxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		

Table TENKTUP1:



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
unique1	unique2	two	four	ten	twenty	onePercen	tenPercent	twentyPerc	fiftyPercen	unique3	evenOnePe	oddOnePe	stringu1	stringu2	stringu4		
436	0	1	2	8	14	95	5	0	1	601	86	63	UQAAAAA	AAAAAAA	AAAAxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
161	1	1	1	6	11	76	4	3	0	340	34	125	FGAAAAA	BAAAAAA	HHHHxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
300	2	0	3	2	7	18	4	4	1	388	158	1	OLAAAAA	CAAAAAA	OOOOxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
754	3	0	2	6	9	78	7	3	1	221	70	115	ADBAAAA	DAAAAAA	VVVVxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
126	4	0	1	3	8	14	2	1	1	52	0	195	WEAAAAA	EAAAAAA	AAAAxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
37	5	0	2	4	13	91	9	0	1	35	76	7	LBAAAAA	FAAAAAA	HHHHxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
245	6	1	2	8	10	39	6	4	1	323	64	119	LJAAAAA	GAAAAAA	OOOOxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
428	7	0	1	2	16	47	7	0	0	332	34	79	MQAAAAA	HAAAAAA	VVVVxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
903	8	0	1	1	0	13	6	0	1	528	150	43	TIBAAAA	IAAAAAA	AAAAxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
452	9	1	0	8	0	72	5	0	0	491	184	67	KRAAAAA	JAAAAAA	HHHHxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
758	10	0	1	1	19	66	7	0	0	299	180	165	EDBAAAA	KAAAAAA	OOOOxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
990	11	0	0	6	5	76	5	3	0	194	38	1	CMBAAAA	LAAAAAA	VVVVxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
47	12	0	2	9	11	71	3	1	0	492	50	55	VBAAAAA	MAAAAAA	AAAAxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
462	13	1	3	9	16	96	1	2	1	967	114	103	URAAAAA	NAAAAAA	HHHHxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		

Table TENKTUP2:



A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
unique1	unique2	two	four	ten	twenty	onePercen	tenPercent	twentyPerc	fiftyPercen	unique3	evenOnePe	oddOnePe	stringu1	stringu2	stringu4		
436	0	1	2	8	14	95	5	0	1	601	86	63	UQAAAAA	AAAAAAA	AAAAxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
161	1	1	1	6	11	76	4	3	0	340	34	125	FGAAAAA	BAAAAAA	HHHHxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
300	2	0	3	2	7	18	4	4	1	388	158	1	OLAAAAA	CAAAAAA	OOOOxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
754	3	0	2	6	9	78	7	3	1	221	70	115	ADBAAAA	DAAAAAA	VVVVxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
126	4	0	1	3	8	14	2	1	1	52	0	195	WEAAAAA	EAAAAAA	AAAAxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
37	5	0	2	4	13	91	9	0	1	35	76	7	LBAAAAA	FAAAAAA	HHHHxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
245	6	1	2	8	10	39	6	4	1	323	64	119	LJAAAAA	GAAAAAA	OOOOxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
428	7	0	1	2	16	47	7	0	0	332	34	79	MQAAAAA	HAAAAAA	VVVVxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
903	8	0	1	1	0	13	6	0	1	528	150	43	TIBAAAA	IAAAAAA	AAAAxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
452	9	1	0	8	0	72	5	0	0	491	184	67	KRAAAAA	JAAAAAA	HHHHxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
758	10	0	1	1	19	66	7	0	0	299	180	165	EDBAAAA	KAAAAAA	OOOOxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
990	11	0	0	6	5	76	5	3	0	194	38	1	CMBAAAA	LAAAAAA	VVVVxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
47	12	0	2	9	11	71	3	1	0	492	50	55	VBAAAAA	MAAAAAA	AAAAxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
462	13	1	3	9	16	96	1	2	1	967	114	103	URAAAAA	NAAAAAA	HHHHxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
478	14	0	0	2	1	44	6	4	1	359	158	63	KSAAAAA	OAAAAAA	OOOOxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		
788	15	1	0	4	12	90	2	2	0	41	154	163	IEBAAAA	PAAAAAA	VVVVxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx		

Discussion:

Through this project we gained a better understanding of how systems manages data, generating data based on a specific schema and uploading data into the system using a generated script. We also got familiar with how to establish connection from an editor to any system and populate table instead of directly uploading from a CSV file which we have done in our previous projects in database. It was a great experience learning about these technical aspects and last but not the least learning from others researches.

We have encountered some issues while installing softwares for connecting MySQL and an editor (Intellij). While uploading data, we also got some problems regarding tempering the integrity of the relational schema. For example, the attribute which is the primary key was mentioned not null and it shouldn’t be mentioned, because it is by default unique.