CS386D
3/12/20
**V1.0**

# Homework A
# Query Optimization, Database 2

**Required Reading:**

1. The original paper defining what has subsequently been named, "The Wisconsin Benchmark": "Benchmarking Database Systems , A Systematic Approach" http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.54.7764

2. A relatively recent retrospective paper: "The Wisconsin Benchmark: Past, Present, and Future" http://firebird.sourceforge.net/download/test/wisconsin_benchmark_chapter4.pdf

**Introduction:**

A first goal of this lab is to observe that query optimizers do consider the statistical properties of the data and respond with different query plans. The pedagogical goal is that between this lab and the previous lab, is that you will finish the semester with experience with tools that are used to assess a database workload and optimize the execution of individual queries and the physical organization (partitioning and indexing data) of a database.

**Objective, Part 1:**

The Wisconsin database benchmark is a foundational piece of work concerning the benchmarking and performance analysis of relational database system. Once you read the first required paper it should be apparent to you that the test database is created to facilitate the performance analysis of individual operators and access paths with respect to the statistical character of the arguments.

**COVID Context:**

Just as the empirical assignment, 5a, 6a, was broken into two parts, 1) the creation of the database, 2) performance measurement, this assignment is done in the same way. This write-up concerns only the creation of the database, ie. Part 1. The actual work requires reading and a modest amount of programming. It is also assignable without coordination with other parts of the class. → I can get push this out now, and if you, proactively get on top of it, it is assured that your time is well spent.

**Test data:** The test data will comprise the same data table replicated 6 times. Let that table be defined as *TestData*(pk, ht, tt, ot, hund, ten, filler) Let there be at least 5,000,000 rows. Recall you don't have to execute the queries. The columns are defined:
a unique primary key, pk.
Column ht should contain uniformly distributed values from 0 to 99,999 (i.e. **h**undred **t**housand),
Column tt, should contain uniformly distributed values from 0 to 9,999 (i.e. **t**en **t**housand),
Column ot, should contain uniformly distributed values from 0 to 999 (i.e. **o**ne **t**housand),
Column hund, should contain uniformly distributed values from 0 to 99 (i.e. etc.),
Column ten, should contain uniformly distributed values from 0 to 9

Column filler – same as before to make a row at least, but not much larger than 256 bytes.

The values should be generated at random, with replacement. Each column's contents should be generated independent of the other columns (i.e. → the easiest possible way of going about this).

Note this schema is modeled after the schema in a famous database benchmark, *The Wisconsin Benchmark*, Per above, a pair of papers are required reading for this assignment. The original paper: "Benchmarking Database Systems , A Systematic Approach", and a retrospective paper are assigned reading: "The Wisconsin Benchmark: Past, Present, and Future"
http://firebird.sourceforge.net/download/test/wisconsin_benchmark_chapter4.pdf

**Physical Schema:** Create three copies of the test data table. Call the tables A, B and C. The contents should be identical. Only the names of the tables are different.

Create three more copies of the test data table. Call them A', B', C', (or Aprime, Bprime, Cprime). Build secondary indices on all eligible columns of A',B', and C'