# CS236: DATABASE MANAGEMENT SYSTEMS
# Spring 2019

## PROJECT DESCRIPTION

The goal of this project is to generate the skyline set for a given input dataset using the Hadoop's MapReduce framework. You will be implementing two of the algorithms discussed in the lectures, mainly the Block Nested Loop (BNL) algorithm and Sort Filter Skyline (SFS) algorithm. You will be comparing the performance of these algorithms for varying input size and data distributions.

## ALGORITHM DETAILS

Regardless of the chosen skyline algorithm, there exist two phases in calculating the skyline set on top of Hadoop's MapReduce framework. These phases are:

(1) Partition the data and calculate a collection of local skylines.

(2) Compute the global skyline set by merging the previously calculated local skylines.

**Extra Credit (10%):**

Choosing the right partitioning strategy when calculating the skyline set is very important. By default, Hadoop employs a random partitioning mechanism when splitting the data across reducers. Random partitioning generated large intermediate results thus merging is the most expensive part of the computation. One can use a different partitioning strategy to avoid the negative effects of merging the local. Implement a custom MapReduce partitioner following a different partitioning strategy (such as angle-space partitioning).

## EXPERIMENTAL DATASET DETAILS

You will measure the performance of your implementation using synthetic data. We provide a script (gendata.sh) that you must use to generate data having different characteristics (e.g. varying tuple number, attribute number and distribution). You will have to adjust the script accordingly for your experiments. In the following figure, we indicate the script's contents:

```
#!/bin/bash

if [ ! -d randdataset-1.1.0/ ]
then
    echo "Extracting randdataset..."
    tar -zxvf randdataset-1.1.0.tar.gz
fi

if [ ! -f randdataset-1.1.0/src/randdataset ]
then
    echo "Configuring environment..."
    (cd randdataset-1.1.0/ && ./configure && make)
fi

#MULTIPLICATION FACTOR TO GENERATE TUPLES
FACTOR=$1
N=$((FACTOR*1024*1024))
#NUMBER OF ATTRIBUTES PER TUPLE
D=$2
#DATASET DISTRIBUTION: (c) for correlated, (i) for independent, (a) for anticorrelated data
DISTR=$3
#TUPLE PADDING TO EMULATE OTHER TUPLE INFORMATION
PADDING=64

./randdataset-1.1.0/src/randdataset -I -$DISTR -d $D -n $N -p $PADDING > $DISTR"_"$N"_"$D.csv
```

In the script we use *N* to indicate the number of tuples, *D* the number of attributes and *DISTR* the data distribution. In order to generate a dataset consisting of 2097152 tuples, 3 attributes following and independent distribution, we execute the following:

```
source gendata.sh 2 3 i
```

The expected output after executing the above command is a csv file named i_2097152_3.csv, which contains in addition to three tuples per tuples, a tuple id and a tuple string used for padding. Using the *head* command, we can see that each line of the corresponding csv file looks similar to the following:

```
1,8.401877171547095e-01,3.943829268190930e-01,7.830992237586059e-01,'abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijkl'
2,7.984400334760733e-01,9.116473579367843e-01,1.975513692933840e-01,'bcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklm'
3,3.352227557148890e-01,7.682295948119040e-01,2.777747108031878e-01,'cdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmn'
4,5.539699557954305e-01,4.773970518621602e-01,6.288709247619244e-01,'defghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmno'
5,3.647844727918433e-01,5.134009101956155e-01,9.522297251747128e-01,'efghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnop'
6,9.161950680037007e-01,6.357117279599009e-01,7.172969294326831e-01,'fghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopq'
7,1.416025553558034e-01,6.069688762570586e-01,1.630057162432958e-02,'ghijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqr'
8,2.428867706297370e-01,1.372315767860187e-01,8.041767542269904e-01,'hijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrs'
9,1.566790892540846e-01,4.009443942461835e-01,1.297904467814557e-01,'ijklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrst'
10,1.088088020257693e-01,9.989245180035590e-01,2.182569053109069e-01,'jklmnopqrstuvwxyzabcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstu'
```

For your experiments, you should generate the following dataset:

| | Tuple Factor | Attribute number |
|---|---|---|
| **Correlated data** | 16,32,64 | 2,3 |
| **Independent data** | 16,32,64 | 2,3 |
| **Anticorrelated data** | 16,32,64 | 2,3 |

Measure the execution time for each algorithm that you developed using the generated data while varying the number of reducers used by your implementation. Create plots using your measurements and explain the results.

**Notes:**

1) Experiment with small data to ensure the correctness of your algorithms.
2) Due to limited disk space, you should avoid generating all the data at once.

**Deliverables:**

1) A report with a brief description of your implementation and explanation of your results.
2) Your code.

**References & Reading Material:**

Chomicki, Jan, et al. "Skyline with presorting: Theory and optimizations." Intelligent Information Processing and Web Mining. Springer, Berlin, Heidelberg, 2005. 595-604.

Vlachou, Akrivi, Christos Doulkeridis, and Yannis Kotidis. "Angle-based space partitioning for efficient parallel skyline computation." Proceedings of the 2008 ACM SIGMOD international conference on Management of data. ACM, 2008

https://hadoop.apache.org/docs/stable/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html

https://acadgild.com/blog/mapreduce-custom-partitioner