# CprE 419 Lab 6: Introduction to Hive

**Department of Electrical and Computer Engineering**
**Iowa State University**
**Spring 2013**

## Purpose

The goal of this Lab is to learn and use Hive, a SQL-like abstraction on top of MapReduce. Behind the scenes, Hive automatically converts statements in HiveQL (Hive's version of SQL) into MapReduce programs. At the end of the lab, you will be able to use the Hive platform and the query language HiveQL to solve large-scale data analysis problems.

## Submission

Create a zip (or tar) archive with the following and hand it in through blackboard**.**

- A write-up answering questions for each experiment in the lab.
- For each experiment, you should submit all the HiveQL statements used and also the results of the execution of the statement.

## Hive

The hive prompt can be obtained as follows:
$ hive

Hive can be run in local or in distributed mode; however, we will only be looking at the distributed mode in this lab.

Please look at the following links for better understanding of Hive.

https://cwiki.apache.org/confluence/display/Hive/GettingStarted

https://cwiki.apache.org/confluence/display/Hive/Tutorial

https://cwiki.apache.org/Hive/hiveclient.html

https://cwiki.apache.org/Hive/languagemanual-ddl.html

When working with Hive, the user views the data as a set of tables, similar to the view of an RDBMS. The metadata for these tables (metadata = data about the data) is stored separately from the data, and Hive can be configured to store metadata either as a text file or as a database or in HDFS.

## Experiment 1 (15 pts)

The first experiment is about creating a table in Hive. We will create tables independently in our own databases to avoid name conflicts. We will then load data into the tables.

**Step 1:** Create a new database with your user ID. For example, user with User ID *snt* creates a Database named *snt*.

**Step 2:** Next we will create tables in the database we created.

***Before creating any tables in the system, please issue the following command to ensure that you do write into your own Database:***
***$hive> use <Database Name>***

If you do not issue the above command, hive will create your tables in the default namespace and that may result in conflicts with other students who make the same mistake. Another way to avoid the namespace conflict is to explicitly tell hive to create the table under a certain Database as follows: <Your Database Name>.<Table Name>

**Step 3:** We will create a table to store the US Census data so that we can query it. The data can be found in the HDFS location: /datasets/Lab6/Exp1
First copy the file to your own HDFS directory: /user/<Your ID>/Lab6/Exp1/

Create a table named *Census* in Hive that can store the US census data. Then load the census data, from the HDFS file that you copied in your own directory, into the newly created table. Did this command make your HDFS file disappear? Try to find out where the file got copied by hive while loading the data into the table.

**Step 4**: Next we will create an External Table in Hive. Create an *External table* named Census_External in Hive. You can use the following location in HDFS as the location for the external table: /user/<Your ID>/Lab6/Exp1/
***You do NOT need to load any data into external tables. It gets reflected automatically.***

**Step 5:** Show and Describe all the tables created under your own Database. This is to make sure you didn't accidentally create any of the tables outside your Database.

**Step 6:** Issue a HiveQL statement to find out all the rows in the Census file that are for Iowa.

Paste the HiveQL commands that you used for all the steps for Experiment 1 with their outputs.

## Experiment 2 (20 pts)

For this experiment you will use Hive to analyze a network trace file dataset. The input file for this experiment is in the HDFS location: /datasets/Lab6/Exp2. Paste the commands and outputs in the lab submission.

A. Create an **external table** to interface the network trace file into hive.
B. Find the top 5 source-destination pairs according to number of connections.
C. Find the top 10 source IP addresses ranked according to the number of unique destination IP addresses that they connect to using the TCP protocol.

## Experiment 3 (25 pts)

For this experiment we have the input files in the following two locations:

/datasets/Lab6/Exp3/ip_trace – An IP trace file having information about connections received from different source IP addresses, along with a connection ID and time.

/datasets/Lab6/Exp3/raw_block - A file containing the connection IDs that were blocked.

Your task is to use HiveQL to perform the following data analysis tasks. *Hint:* For parts B and C, you may have to use nested HiveQL statements.

A. Create two **external tables** to load the network trace and the raw_block data into hive.
B. Generate the list of all unique source IP addresses that were blocked and the number of times that they were blocked. This list should be sorted (using HiveQL) by the number of times each IP was blocked in descending order.
C. Find top 5 most frequent ports for blocked connections.

No part of the Experiment can use multiple HiveQL statements or by creating intermediate tables. Each part (say, part B of experiment 3) must be solved using a single statement. Paste your code and the results from the Experiment into your submission.