Fall 2018
Due: Friday 30 November, 2018 at 11:59 p.m.

This assignment is in 4 parts.

Part 1: Convert the ER Model below into a relational database.

Part 2: Create a Postgres database schema containing tables and attributes created in Part 1.
NOTE: It must be emphasized it is part of the project that the tables be part of a Postgres *schema*. The name of the schema should be the same as your zID number.

Make sure to enforce primary key and foreign key constraints if they exist.

Place your commands for creating the schema into a file which should be e-mailed according to the instructions below. The pg_dump command could be very useful here. The script will be graded for functionality by sending it as input to psql, i.e.
`psql < scriptfile`.

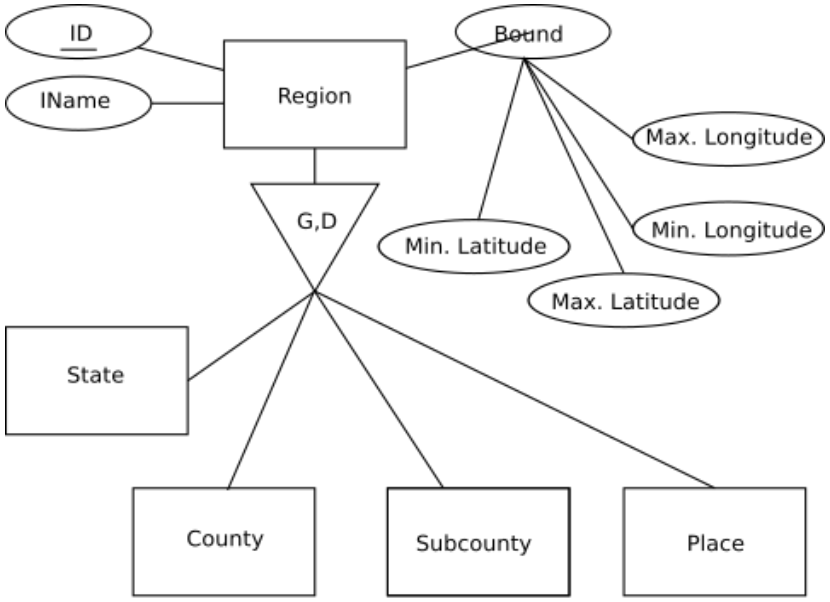
Part 3: Write a C++ program (appropriately documented) to read some data from two types of data files and load the data into the Postgres database that you created in Part 2.

Part 4: Write a program which will query the user for the name of a region . The program should query the database and print out information about the region. This program is described more fully below.

Submission Instructions: Place your schema creation script, your program source code files (and associated files) along with your data files into one directory on your turing account. Do NOT send data files. Move up from that directory and type the following magic incantation (assuming the directory in question is called assign5)

```
tar -cvf - assign5 | gzip -c9 > assign5.tgz
```

Spaces are important. This will create a file, assign5.tgz that should be e-mailed to `cs511_2018_3@cs.niu.edu`.



Geographic Bounds ER Model – Entities, Relationships, and Attributes

The ER model helps maintain a multilingual database of terms associated with geographical forms.

Each entity and relationship below contains a description of its role in the project, along with necessary attributes.

Entities

Region
The basic entity of the project. Each region can be thought of as a place on the Earth that takes up space. The representation of the region is not stored here. All regions in this project pertain to the United States.

ID *** Key ***

State
The states of the United States.
ID *** Key ***

County
States are divided into counties. Well, some states are subdivided into units called something else, like parishes, but for this project, the state subdivisions will all be called counties. County names are not unique across states. DeKalb County, Illinois; DeKalb County, Georgia; and DeKalb County , Indiana are all very different.

ID *** Key ***

Subcounty
Counties are often divided into subcounty units. Usually these are called townships, but many other names exist. In this project, they will all be referred to as subcounties.

ID *** Key ***

Place
Cities, villages and towns. Place names are not unique. It is possible in reality for a city name to be associated with different cities, hence different regions. For example, Paris, France, and Paris, Idaho are

two very different cities. As are Cairo, Illinois, and Cairo, Egypt. As are DeKalb County, Illinois, DeKalb County, Georgia, and DeKalb County , Indiana.

ID *** Key ***

Attributes

ID States have a two digit unique identifier. Counties have a 5 digit unique identifier. The first two digits of a county ID are the state ID in which the county is found. The subcounties and places have a 7 digit unique identifier. The first two digits in the subcounty and place IDs are the state ID in which the subcounty or place is found.

Name Region names are all in English.

Bound Each region has a geographical extent. These are given as a minimum and maximum in both longitude and latitude. The longitude and latitude values are represented as scaled integers. For example, the latitude 41.234567 degrees would be represented as the integer 41234567. The scale factor is always 1 million, or 10^6.

Data Files

Bound Files

Some data files will be provided for this assignment. Like most real world data, the files are not in an ideal format for the data base. Your loading program will have to read and process the data file records. There are two types of data files.

The first set of files (.bnd files) contain the bounding information for regions. Each line of the file corresponds to a region and its bounds. Below is an example of some lines from a bound file:

0000045-083353958+32033472-078499288+35215473
0000046-104057779+42479667-096436614+45945425
1719083-088129502+42152779-088038874+42185369
1719161-088797063+41886483-088700876+41971691
1719200-088650103+40115458-088631589+40127039

- Characters 1-7 are the ID number of the region.
- Characters 8-17 are the scaled minimum longitude for the region.
- Characters 18-26 are the scaled minimum latitude for the region.
- Characters 27-36 are the scaled maximum longitude for the region.
- Characters 37-45 are the scaled maximum latitude for the region.

Positive latitudes are degrees north of the equator. Negative latitudes are south of the equator. Positive longitudes are east of the prime meridian. Negative longitudes are west of the prime meridian.

Name Files

These files come directly from the US Census Bureau's TIGERLINE data set, used to create maps for census purposes. These files have the suffix RTC. The format is one record per line. There are many fields in each record, but many fields can be ignored. Other fields have different meanings depending on context. Below are some example lines from a TigerLine Record C name file.

```
C0302     2000        71J1602                     Chicago--Gary--Kenosha, IL--IN--WI
C0302     2000        73J1600                     Chicago, IL
C0302     2000        75O                 22960   DeKalb, IL
C0302     1990        75U                 0619    Aurora, IL
C0302     2000        76O                 32869   Genoa, IL
C0302     2000        76O                 78796   Sandwich, IL
C030217                01S                         Illinois
C030217   2000        S15    11220                CRESTON COMMUNITY CONSOLIDATED SCHOOL DISTRICT 161
C030217   2000        S15    14410                ESWOOD COMMUNITY CONSOLIDATED DISTRICT 269
C030217   200016470C1060P                         Cortland
C030217   199016470C1460P                         Cortland
C030217   200019161C1258P                         DeKalb
C030217   199019161C1458P                         De Kalb
C030217   199028898C1458P                         Genoa
C030217   200028898C1858P                         Genoa
C030217   200035268C1061P                         Hinckley
C030217   199035268C1461P                         Hinckley
C030217   200040065C1061P                         Kingston
C03021703720004007 8T1044M                        Kingston
C03021703719904629 2T1 44M                        Malta
C03021703720004629 2T1044M                        Malta
C03021703719904769 6T1 44M                        Mayfield
C03021703720004769 6T1044M                        Mayfield
C03021703719904898 3T1 44M                        Milan
C03021703720004898 3T1044M                        Milan
C03021703719905820 0T1 44M                        Paw Paw
C03021703720005820 0T1044M                        Paw Paw
C03021703719905967 6T1 44M                        Pierce
C03021703720005967 6T1044M                        Pierce
```

A description of the file format can be found below:

Start	End	Description
1	5	Ignored.
6	7	State – two digit code. May be blank in some records, but these records should be ignored.
8	10	County code. Combined with state code, makes the unique identifier for counties. Blank for states.
11	14	Year of data set. Records that have 1990 for this field should be ignored. Records that have 2000 or blank in this field are of potential interest.
15	19	Place code. Combined with the state code, this makes the unique identifier for places and for county subdivisions.
20	24	Ignored.
25	25	Region type. States have the value 'S'. Counties have the value 'C'. Subcounties have the value 'M'. Places have the value 'P'. Records with any other value in this field should be ignored.
26	52	Ignored.
53	112	Name of Region

Program(s)

Your program to load data files should not be hardwired to a specific set of files. The user should be allowed to determine (either through command line arguments, or interactive prompts, or data file entries) what files are to be loaded. Each entry in a data file should be examined to see if it is already in the database. If it is, it should be ignored. It should be possible to load in files multiple times.

A separate program should ask the user for a region name. Names may have spaces. After looking up this name, all regions having this name should be printed. The bounds for each region should be printed in Degree Minute Second notation.

To convert a decimal degree value to Degree Minute Seconds, take the decimal degree value. The integer part is the degrees. Take the fractional part and multiply by 60. The integer of this result is the number of minutes. Take the fractional part of this result and multiply again by 60. This is the number of seconds.

For example, the place of DeKalb, Illinois (ID: 1719161) has the bounds
88 degrees, 47 minutes, 49.43 seconds to 88 degrees, 42 minutes, 3.15 seconds west in longitude; and from 41 degrees, 53 minutes, 11.34 seconds to 41 degrees, 58 minutes, 18.09 seconds north in latitude.

Additional Implementation Points

- ID values should be stored as integers, not strings.
- The database should be implemented as a schema.
- Do NOT submit data files with your assignment.