README:

**NOTE:** database application was compiled on microsoft visual studio

2010 using the database headers and .dll provided by Team 23.

**NOTE ABOUT FUNCTIONALITY**: the database we were given wasn't fully functional,

query didn't work properly so information had to be retrieved from the database

manually in functions written in the main.cpp of the application. The CrossJoin

function which is supposed to allow us to join the tables doesn't work, so the program

will not be able to join tables.

**HOW TO COMPILE USING VISUAL STUDIO 2010:**

create a new project in Visual Studio, extract the header files, .txt files, and main.cpp file into the

appropriate project folder. Also extract the DatabaseTest.dll and Database.lib files into the

project folder along with the headers and main.cpp file.

in visual studio, go to project properties->linker->input and add 'Database.lib' to the list of

additional dependencies. build the project and run main.cpp.

**HOW TO COMPILE WITH OTHER IDE's**:

extract all files included in the .zip to the same folder that you desire to keep them in.

In your IDE, link the Database.lib using your IDE's tools, and then compile and run main.cpp

**LIST OF FILES THAT SHOULD BE EXTRACTED TO SAME FOLDER BEFORE COMPILATION**:

Attribute.h

Database.h

Record.h

Table.h

main.cpp

chefmozaccepts.txt

chefmozcuisine.txt

chefmozparking.txt

geoplaces2.txt

rating\_final.txt

usercuisine.txt

userpayment.txt

userprofile.txt

**USE OF THE APPLICATION:**

the application will begin by loading data into a database object, and then it

will prompt the user to enter a command through the command window. the format of

the commands is as follows:

**for commands with two parameters:**

(command): parameter1-parameter2

**for commands with one parameter:**

(command): parameter1

**Example command:**

Customer information: U1001-smoker

**several things to note**:

- commands are application specific, and can include spaces

-there MUST be a space between the ':' and parameter1

-no spaces between the '-' and parameter2 or parameter1

if you do not follow this format the application will tell the user that the command

is invalid and the user will be prompted to try again.

below is a list of commands, their function, and appropriate parameters. Commands can be written in all lower-case or all upper-case, but not a combination of upper and lower case letters.

\*= not functional due to problems with the database functions not working properly

|  |  |  |
| --- | --- | --- |
| COMMAND | FUNCTION | PARAMETERS |
| customer information | retrieve a specific characteristic about a customer from their user profile | -userID as it is listed in the userprofile.txt file  -attribute that is present in the userprofile table. |
| restaurant information | retrieve a specific characteristic about a restaurant. | - placeID of the restaurant  -desired attribute of the restaurant. Valid inputs for this parameter include “parking”,”payment”,”cuisine”, “hours”, or “ratings” |
| \*combine tables | combine tables from the database and add the new table to the database | -name of a table  -name of a table |