Andrew Libby, Ryan Potter, Jackson Tolliday, Reuben Trachtenberg

Team 8

02/03/2024

**Here are our query commands:**

INPUT RETURN

state == X all stats for state X

median age of X median age for state X

median age >/</== X all states with median age over/under/equal to X

obesity rate of X obesity rate for state X

obesity rate >/</== X all states with obesity rate over/under/equal to X

cow-human ratio of X cow-human ratio for state X

cow-human ratio >/< /==X all states with cow-human ratio over/under/equal to X

life expectancy of X life expectancy for state X

life expectancy >/</== X all states with life expectancy over/under/equal to X

ski resorts of X number of ski resorts, if any

**Keywords**: [state, median age, obesity rate, cow-human ratio, life expectancy, ski resorts]

Additionally, we will provide functionality for logical AND. **Below are some example queries:**

median age > 38 AND cow-human ratio < 1

state == Alabama

ski resorts of Texas

As far as communication between the parser and a datastore, we will be using a dictionary to hold the data drawn from Firebase. The parser will return a list that includes a keyword (or keywords), the corresponding symbol(s), and a double or string value that represents a state or a ratio/age/rate. This list will be checked against the dictionary and our query program will return the correct entries from the dictionary based on the parser’s return value.

**Functions**:

parse\_string(userInput) will take a string that represents the user’s query and will return a list consisting of attributes, symbols, and state names/doubles to be checked against the firebase dictionary

create\_dictionary() has no parameters and will return a dictionary filled with state data loaded from firebase (via a JSON file)

format\_dictionary(states\_dict) will take a dictionary of states and format its output to be readable and organized

**Additional Information**:

We have considered creating additional functions to handle individual aspects of the parsing. For example, we may create a greater\_than() function that would parse a chunk of the query and append a ‘>’ symbol to the list to be returned in parse\_string. That said, we have not found this to be necessary yet.

If time allows, we also thought it would be neat to add logical OR to our query language, but this would require some extra steps and is not yet accommodated for.