**Southeastern Louisiana University – CSIT Department**

**CMPS 439 Database Systems**

# SURLY I Report

**William Williamson**

## Who is on your team and what's the division of labor?

I am implementing SURLY on my own.

## Programming Language

The language chosen for this implementation of SURLY is Python 3.6. The versatility of Python makes it a relevant and useful technology in many aspects of computer science and information technology. It is especially powerful when handling data partly due to the ability it gives the user to perform tasks cleanly, as well as third-party libraries that have grown up around the language.

The primary reason Python was chosen was for the opportunity to learn and gain experience with some of the aspects of the language that have made it the most in-demand language, aside from SQL, in the data science job market, according to an analysis of data science job listings on Indeed.com. [1]

## List of libraries or programming language features

* **Dictionaries -** A key/value data structure with O(1) time complexity for “get”, “set”, and “delete” operations.[2] Dictionaries were used in Surly as a way to store and organize the different types of data associated with a relation.
* **NamedTuples -** Part of the Collections library, a namedtuple is similar to a dictionary in that values can be accessed using a key, but can also be accessed using an index or position. NamedTuples are immutable like other tuples. Using namedtuples provides an easy way to check new records being inserted into a relation against the attribute definitions for that relation.
* **Dataframes -** A dataframe is a 2 dimensional data structure from the Pandas library, a third party library used for data analysis. It is currently being used in Surly to format output, but will provide more functionality in future releases of Surly

## Deliverables

|  |  |
| --- | --- |
| **Checklist of deliverables** |  |
| Hardcopy of | I |
| This writeup | x |
|  |  |
| Zip file containing | I |
| This writeup | x |
| Test cases showing input/output | x |
| Source code | x |
| README.TXT \* | x |

## Coverage - SURLY Part I

|  |  |  |
| --- | --- | --- |
| **version** | **Feature** | **Covered/Comment** |
| I | Relation | Yes |
| I | Insert | Yes |
| I | Print | Yes |
| I | Index - Heap | No |
| I | CATALOG | Yes |
| II | Destroy | n/a |
| II | Delete | n/a |
| II | Project | n/a |
| III | Join | n/a |
| III | Select where … AND/OR | n/a |
| III | Delete where … AND/OR | n/a |
| III | Import/Export in XML | n/a |
| optional | View, Constraint, … | n/a |

## Implementation

### SURLY I

**Overview:**

The main entry point into the program is a user interface that tokenizes the input data and creates an instance of Surly. The Surly class is the center of operations for the program. This class handles the creation of the database, creation of relations, and printing, while non-class methods handle certain parsing and tokenization tasks.

* **Database** – Once the Surly object is instantiated, a Database object is initialized with it’s own Catalog. Surly currently does not support more than one database object, but was built in a way to enable multiple databases in the future.
* **Relations** – New relations are created by the Surly instance as a dictionary. The relation name is the key and the value is an array of named tuples which store the attribute definitions. A reference to a relation is stored in the database’s catalog.
* **Tuples** – The creation of tuples is passed to the Relation class and checked against the attribute definitions before being stored in a records dictionary. Tuples are called records to disambiguate from Python’s tuple data type.
* **Attributes** – Attributes are created in the Relation class as an Attribute object and stored in an array belonging to the relation.

## Things you did differently (e.g., than the SURLY spec)

### Started to add a command-line interface, but it was not going to be complete by the due date. Future releases will have a command-line interface as well as a graphical user interface.

## Recommendations

### Things you would do differently if starting over now.

Meet with the instructor more frequently for guidance and feedback.

### Did SURLY meet your objectives for this course?

Yes

### Suggestions on how to improve SURLY I assignment

It would be helpful for students if there was a homework assignment to write an outline or a plan for implementing Surly. This would help students realize what the project requires and how to approach it before it’s too late.

### Any other comments?

Surly is a great project. It really enforces the need to think and plan out the structure of data.

**References:**

[1] <http://r4stats.com/2017/02/28/r-passes-sas/>

[2] <https://wiki.python.org/moin/TimeComplexity>