**Western Washington University – CSCI Department**

**CSCI 330 Database Systems**

# SURLY { II } Report

Parker Ford, Elliot Trienen

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

Parker: Delete Where, Select Where, Project, Temp Relations

Elliot: Join, Qualify Attributes, Temp Relations, Printing

## What programming language did you select and why?

We used java for this project. It is the language we are both most familiar with and we had an object-oriented design in mind, so Java was a perfect fit.

## List libraries or programming language features you made use of?

* Linked Lists: Linked Lists’ were crucial in the over all structure of our program. Our database as a whole was comprised of a linked list of relations, which are a linked list of tuples, which are a linked list of attribute values. Without the functionality being able to add and delete objects from the list on a whim, it would have been extremely difficult for the program to function. Also, the ability to quickly search a linked list was helpful as well.
* Scanner: Scanner was essential for parsing the text file. It is what allowed us to determine how to interpret text. The use of regex was also extremely important in splitting the strings we scanned in order to distribute the data into its respective variables.

## Deliverables

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

* \* include at top level a file titled README.TXT that provides *Installation and Demo Instructions* containing instructions on how to install and demo your program

## Coverage - Did you complete all of SURLY Part I/II - what is missing?

|  |  |  |
| --- | --- | --- |
| **version** | **Feature** | **Covered/Comment** |
| I | Relation | X |
| I | Insert | X |
| I | Print | X |
| I | Heap Storage | X |
| I | Catalog | X |
| I | Destroy | X |
| II | Delete where … AND/OR | X |
| II | Select where … AND/OR | X |
| II | Project | X |
| II | Join | X |
| other | Import/Export, GUI, … |  |

## How did you implement

* **Relations** – Relations are linked list of tuples.
* **Tuples** – Tuples are a linked list of attributes.
* **Attributes** – Attributes contain a name, a value, and a value type.
* **Insert** – Insert will add a new tuple in the specified linked list of tuples.
* **Catalog** – Catalog is a relation that can not be destroyed that holds the info about all created relations other than itself.
* **Destroy** – Destroy will located a given relation and completely remove it from the database linked list.
* **Delete where** – Delete where will completely remove all tuples from a relation linked list if no conditions are specified. If conditions are specified it will loop through the given relation, finding all the tuples that match the conditions given and storing the indexes in an array. It then will remove all the tuples at the stored indexes.
* **Select where** – Select where will create a temp relation that is a copy of the given relation if no conditions are given. If conditions are given it will search the given relation, putting all of the indexes of matching tuples into an array. It then creates a temp array with only the tuples at the stored indexes.
* **Project –** Project will loop through the given relation, creating a new tuple of only the given attributes. After each tuple is created it is inserted into a temp array.
* **Join-**Join will search two relations. It first locates that attributes that are being used as the join condition, then, when a match is found, it creates a new tuple that contains the data from both relations. These tuples are then added to a new temp relation.

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

### Limitations of the current release.

Currently there is not much a GUI, as it is limited to only reading text documents through command line. It also does not have any way to do outer joins.

### Extra features you added - e.g., going beyond the SURLY I/II spec

No extra features added.

### Things you are especially proud of

The way our program outlines the tables in the command line looks really nice!

## Recommendations

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

If I could start over, I would spend more time actually planning out how the classes would function and interact with each other. We were adding functionality on the spot and it led to our code being unorganized and not super-efficient. Having all that figured out beforehand would have made things go a little smoother overall.

### What did you learn about databases from SURLY?

SURLY taught me how powerful databases can be. Having an easy way to store and search and organize data is so extremely valuable for things like storing user info and performing statistical analysis.

### Any other comments?