

## wDatabase 2.0

- Course: CS 505
- Assignment: Project 2
- Specifications: <<http://www.cs.engr.uky.edu/~marek/html/ldid.dir/proj2.505fa12.html>>
- Author: Wesley Walker
- Submitted: November 25<sup>th</sup>, 2012

## Insert Data

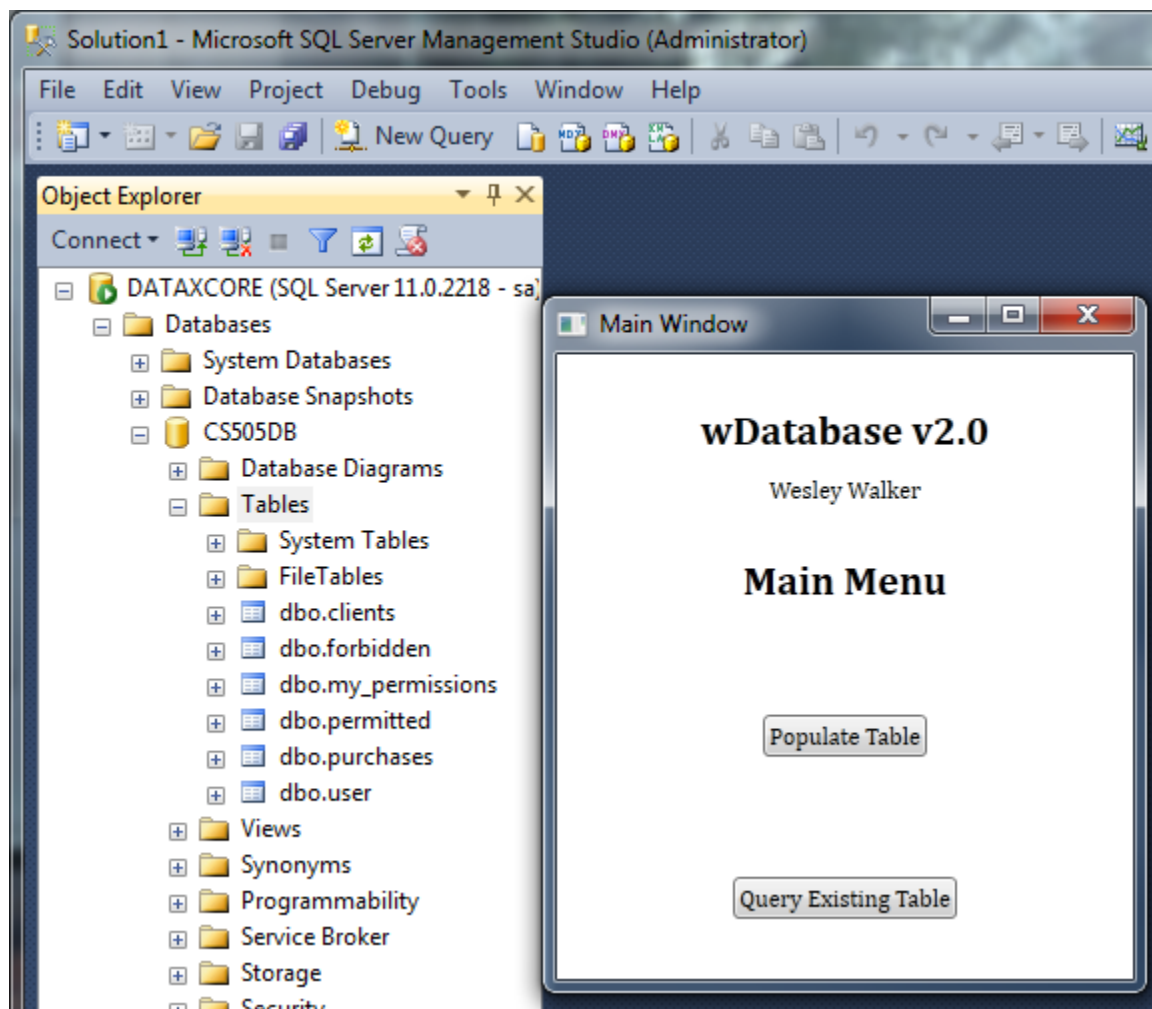


Figure 1 - Main Menu

A table may be submitted to be partitioned (as per the specifications) via the “Populate Tablets” button seen above. Once this button has been pressed, the user is presented with an *Open File Dialog Box*, where they select the desired file containing the table data and metadata (See below).

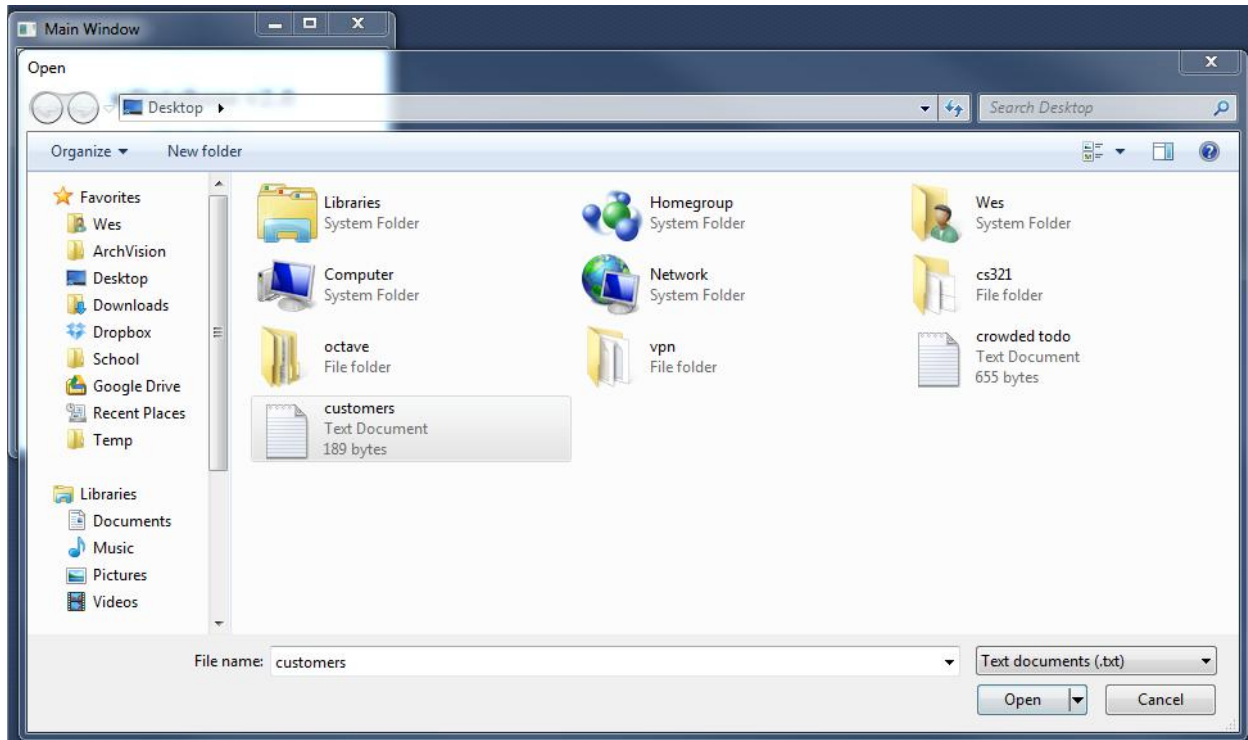


Figure 2 - Open File Dialog

The user selects the appropriate file (or inappropriate file), and are notified on the success or failure of the data import (both cases shown below)

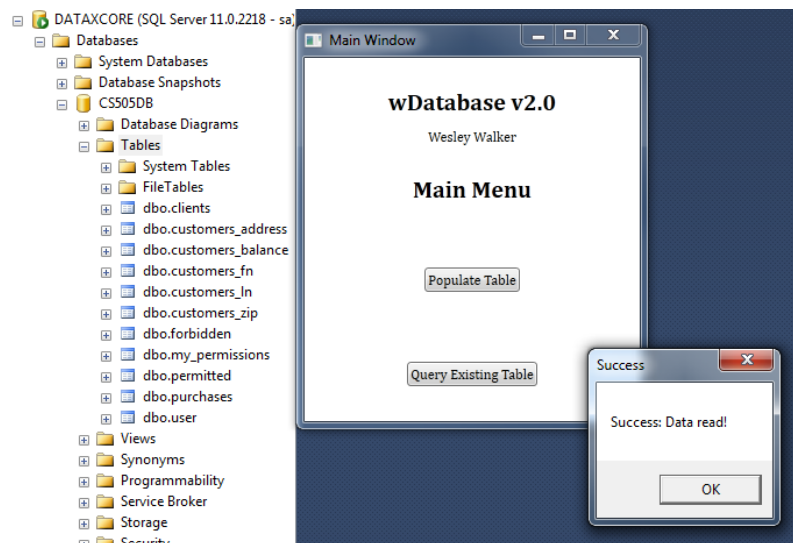


Figure 3 - Successful tablet population

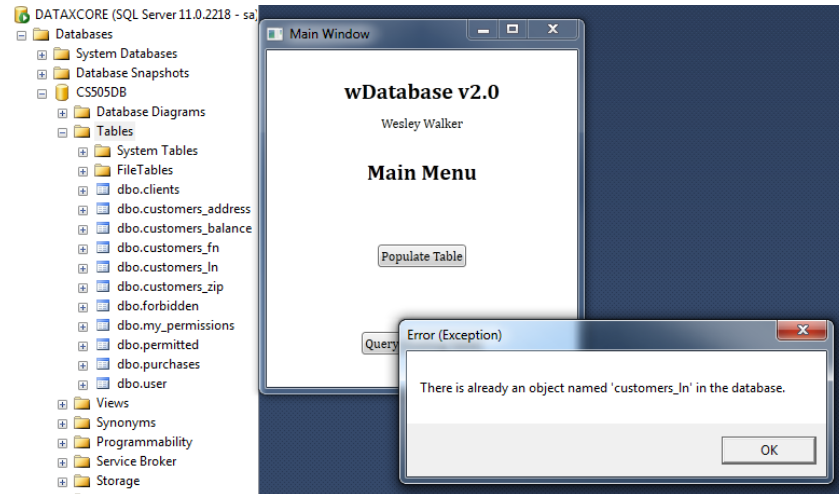


Figure 4 - Failed tablet population

(The error above would be displayed when trying to re-import an existing table).

## Columnar partitions

In the background of both figures above, you may note that on the successful reading of data, 5 new tables were created in Microsoft SQL Studio Manager with the prefix “customers\_”. These were generated by the appropriate parsing of the user’s input file:

```
ln;fn;address;zip;balance*  
char(16);char(10);char(24);integer;money*  
marek;victor;lexington;40506;149.99*  
dexter;harvey;louisville;43220;279.14*  
cashew;claudia;los angeles;93470;144.11*
```

with the filename “customers.txt”.

## Metadata

The first two rows are metadata for the table, where the first contains field names and the second contains data types. On successful “Populate Tables” functionality with customers.txt, 5 tablets are created with their respective field names and data types:

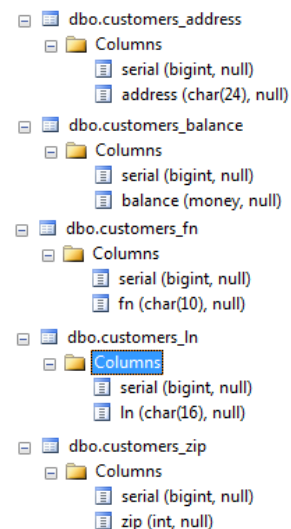
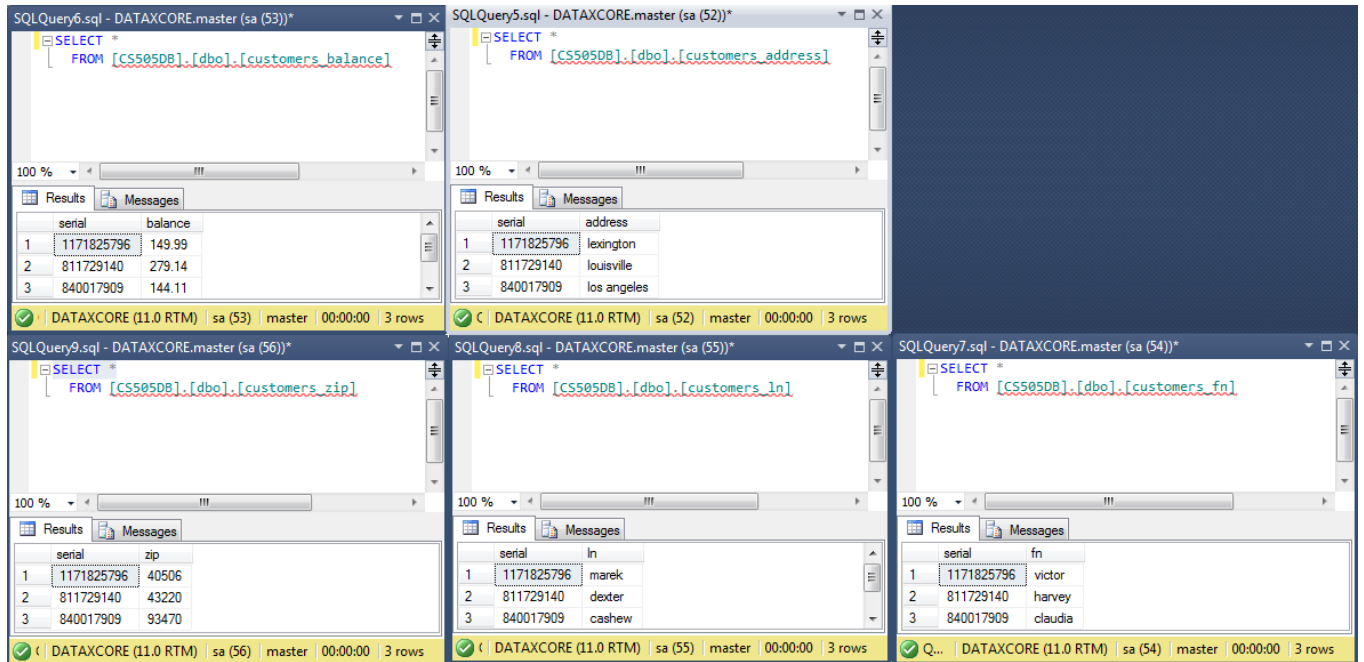


Figure 5 - Tablets, fields and data types

## Data

The data from the file is read in assuming proper ordering per record (delimited with an asterisk) and each tablet is populated with a serial key and the field data.



The screenshot displays five SQL Query windows, each showing a query and its results. The windows are titled as follows:

- SQLQuery6.sql - DATAxCORE.master (sa (53))\*
- SQLQuery5.sql - DATAxCORE.master (sa (52))\*
- SQLQuery9.sql - DATAxCORE.master (sa (56))\*
- SQLQuery8.sql - DATAxCORE.master (sa (55))\*
- SQLQuery7.sql - DATAxCORE.master (sa (54))\*

Each window shows a SELECT query and its results in a table format. The results tables are as follows:

serial	balance
1	1171825796 149.99
2	811729140 279.14
3	840017909 144.11

serial	address
1	1171825796 lexington
2	811729140 louisville
3	840017909 los angeles

serial	zip
1	1171825796 40506
2	811729140 43220
3	840017909 93470

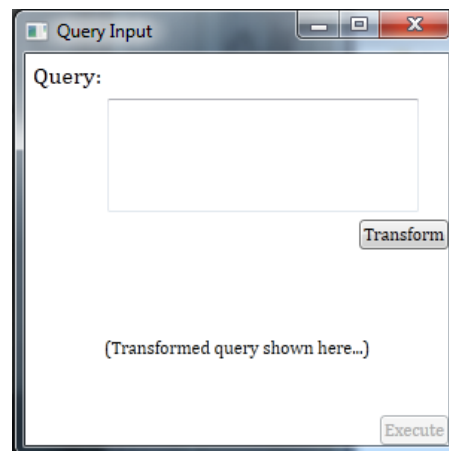
serial	ln
1	1171825796 marek
2	811729140 dexter
3	840017909 cashew

serial	fn
1	1171825796 victor
2	811729140 harvey
3	840017909 claudia

Figure 6 - Tablets, containing data

## Queries

For the user to input their desired query click the “Query Existing Table” button. Post-click, the user will be presented with the Query Input window:



Query Input

Query:

Transform

(Transformed query shown here...)

Execute

Figure 7 - Query Input window

The user can type in their SQL query as if they were querying a single table; by clicking the “Transform” button, the input query is parsed and *transformed* into a query on the tablets which will provide the desired results of the user! Once the transformed query has been generated the “Execute” button is enabled, and when clicked, will execute the transformed query on the tablets and display results.

## Examples

Captures of input queries, their transformations and the results of the query:

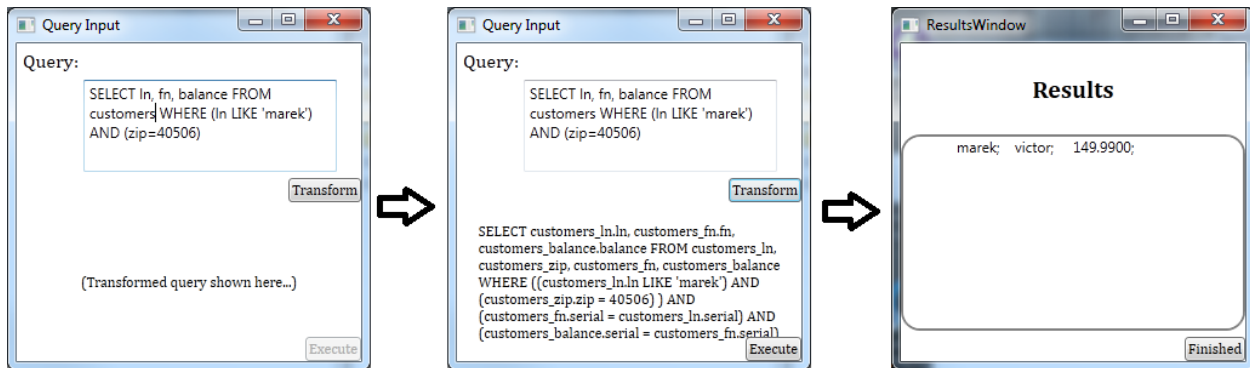


Figure 8 - Example 1

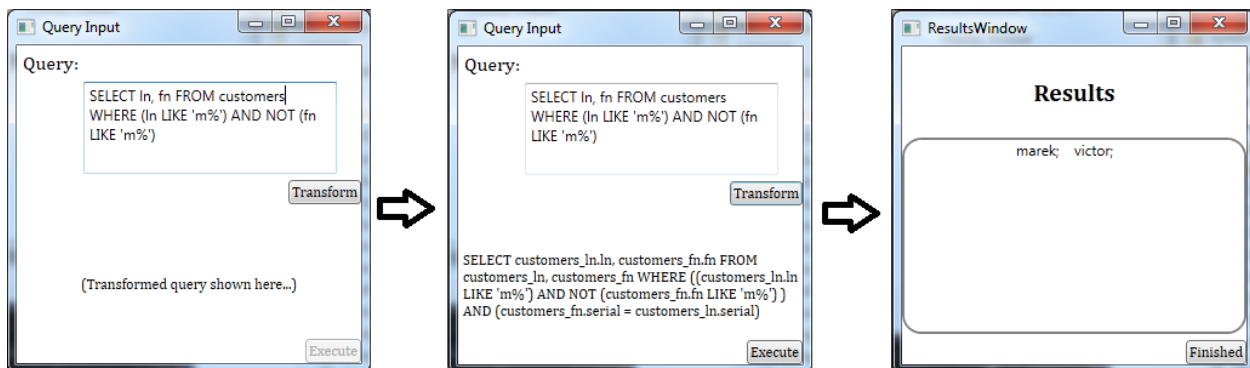


Figure 9 - Example 2

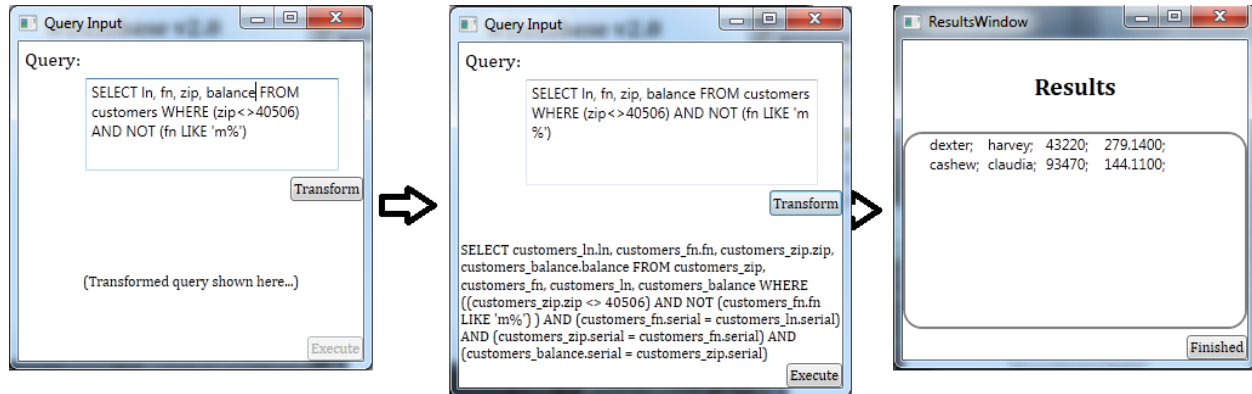


Figure 10 - Example 3

### Errors

When an error occurs (either by a bad input query, or SQL exception) the error message is displayed to the user, they then have the opportunity to retry their query if they desire to do so.



### Test Cases

- Figure 8 displays a query which contains an **equality (=)** and a **LIKE** statement.
- Figure 9 displays a query which contains another **LIKE** clause as well as the **negation** of a **LIKE** clause.
- Figure 10 displays a query which contains an **inequality (<>)** and another **negation** of a **LIKE** clause.