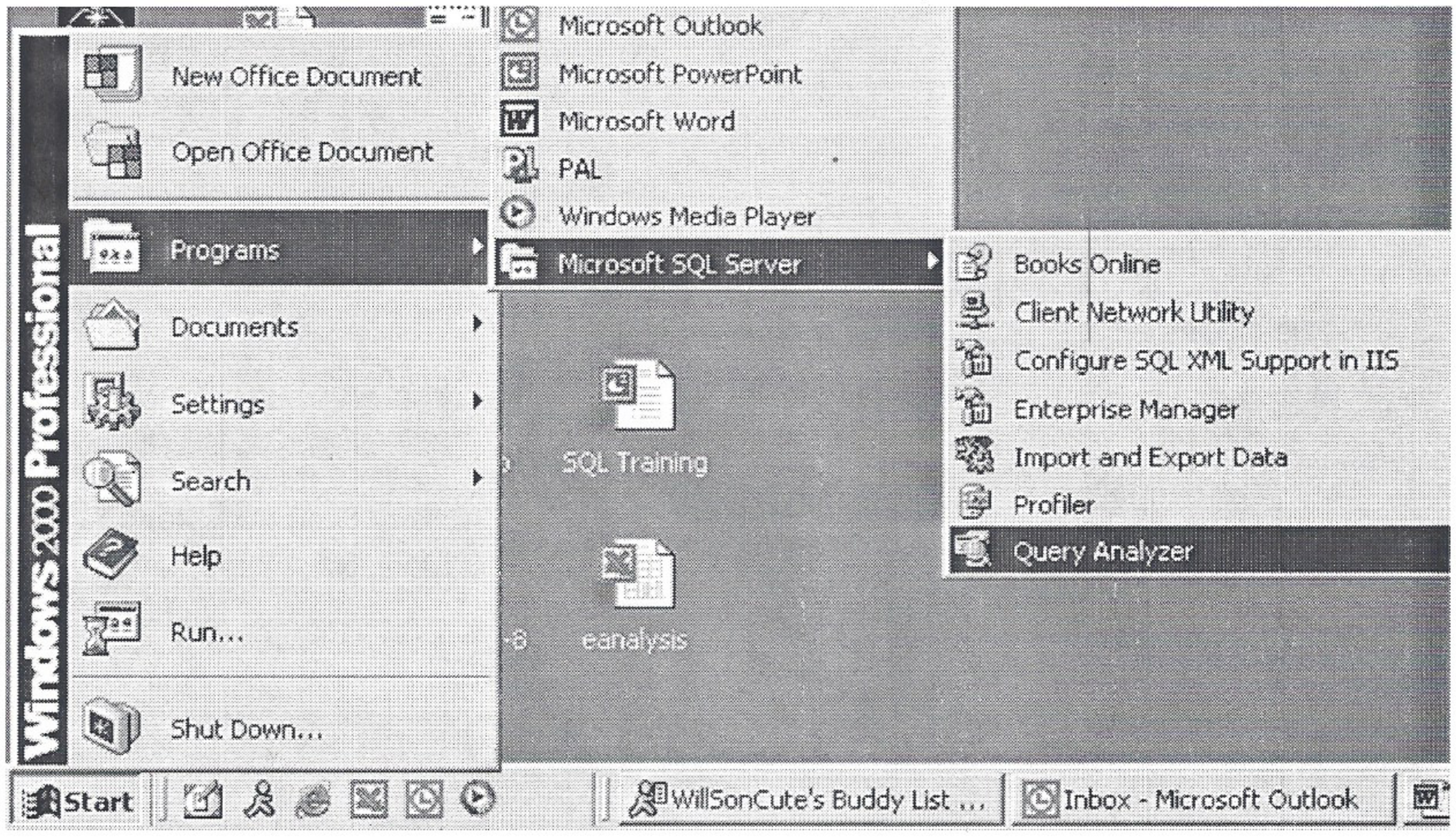
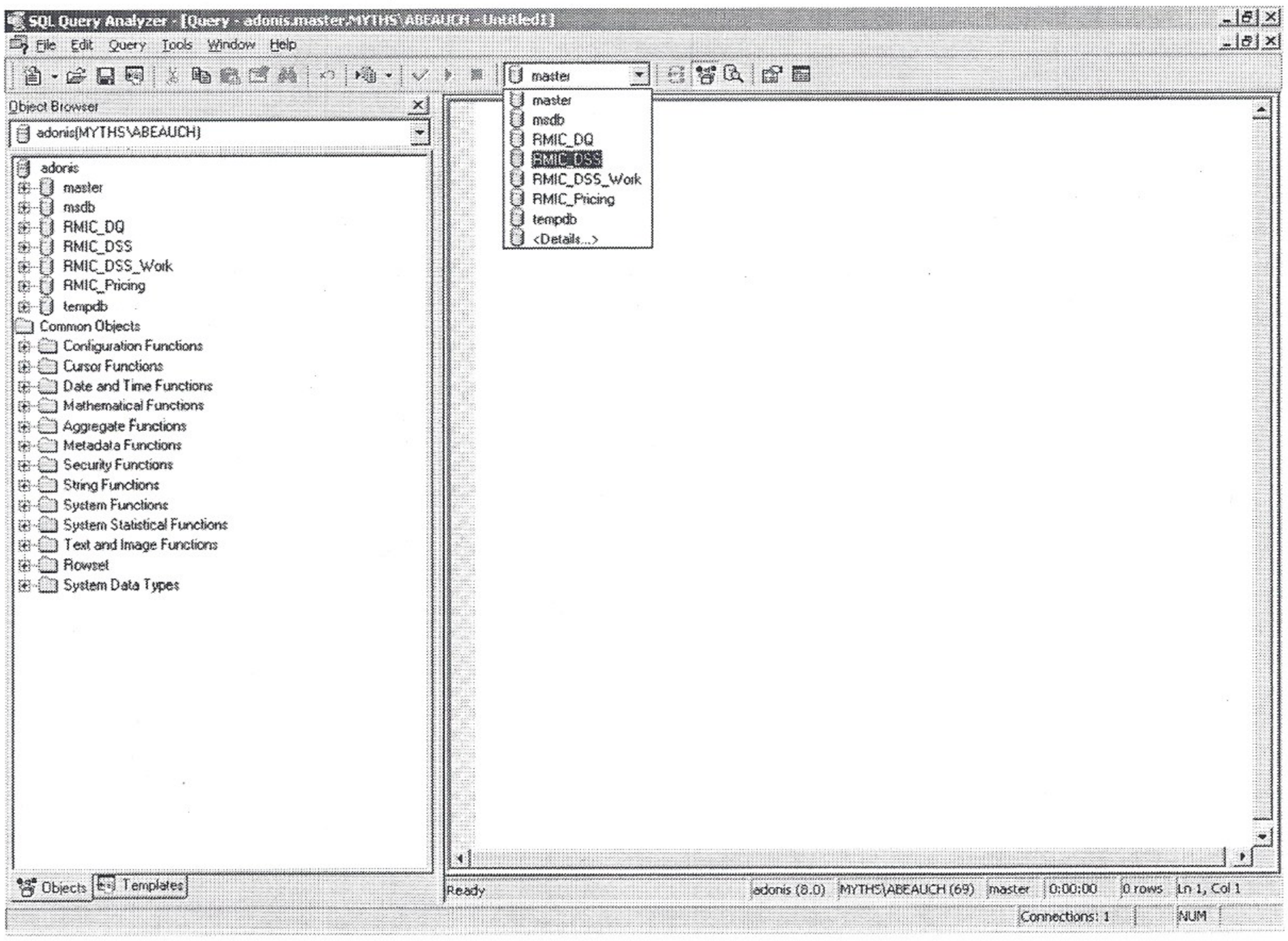
Navigating within MS Query Analyzer

To log into MS Query Analyzer, click on Start >Programs >Microsoft SQL Server >Query Analyzer.

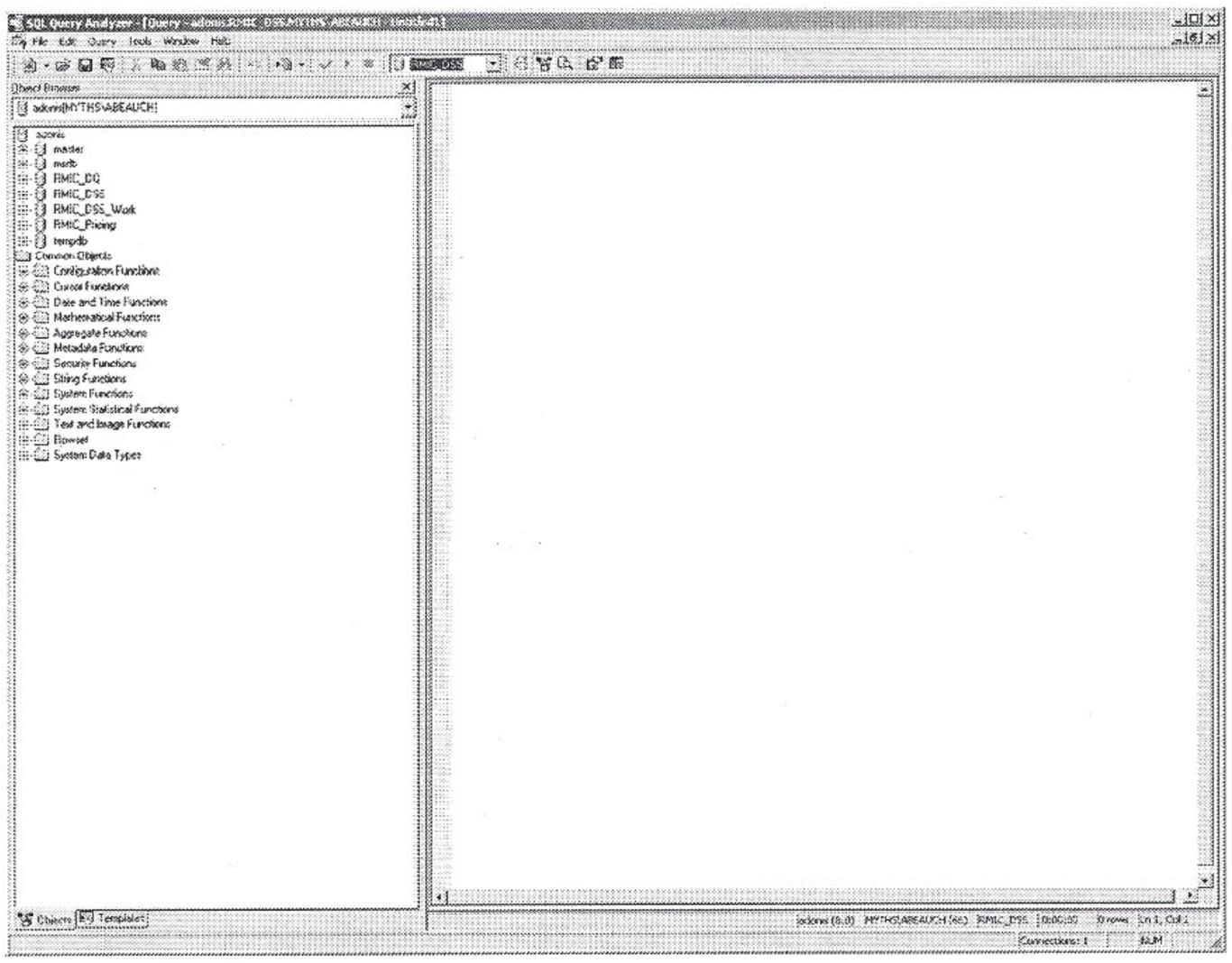


In the "Connect to" window, enter Adonis in the SQL Server field and select Windows Authentication. (see below)

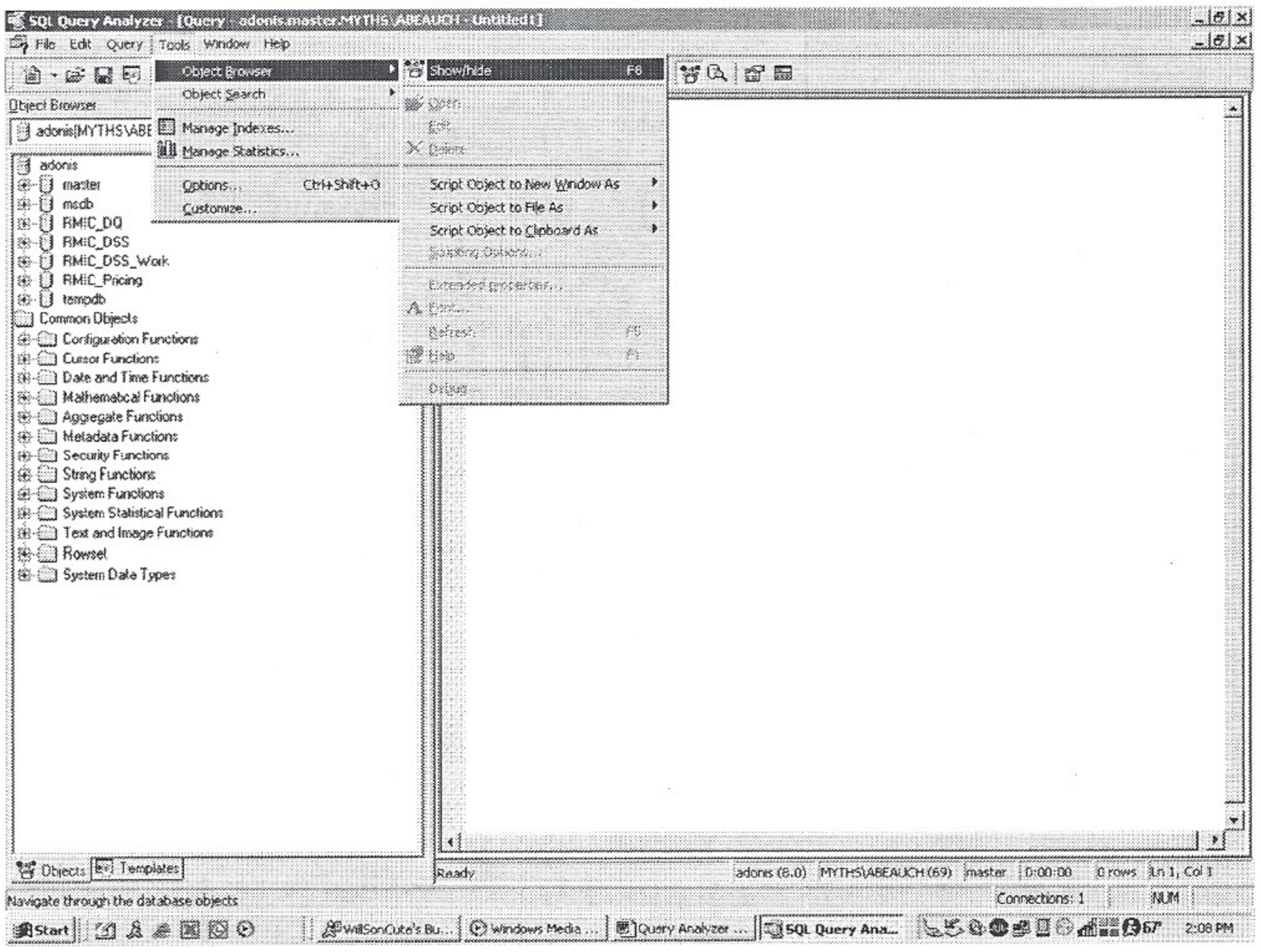
Once Query Analyzer opens, in your tool bar at the top of the screen, click on the arrow in the drop-down window and select RMIC_DSS.



Below is what your screen should look like. An Object Browser will be displayed on the left side of the screen, while a blank query area is displayed on the right.



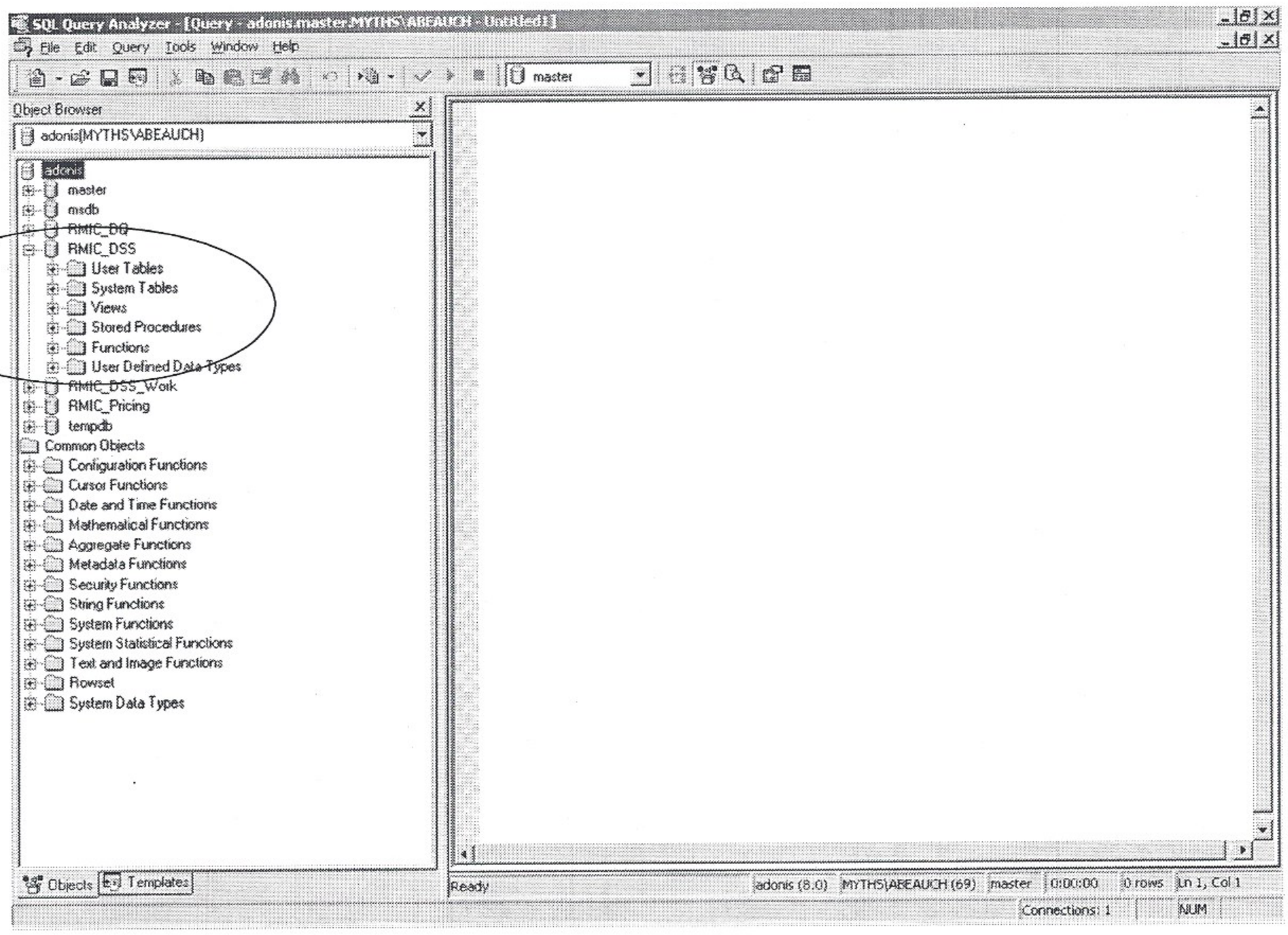
If you don't see the Object Browser, click on Tools > Object Browser > Show/Hide.



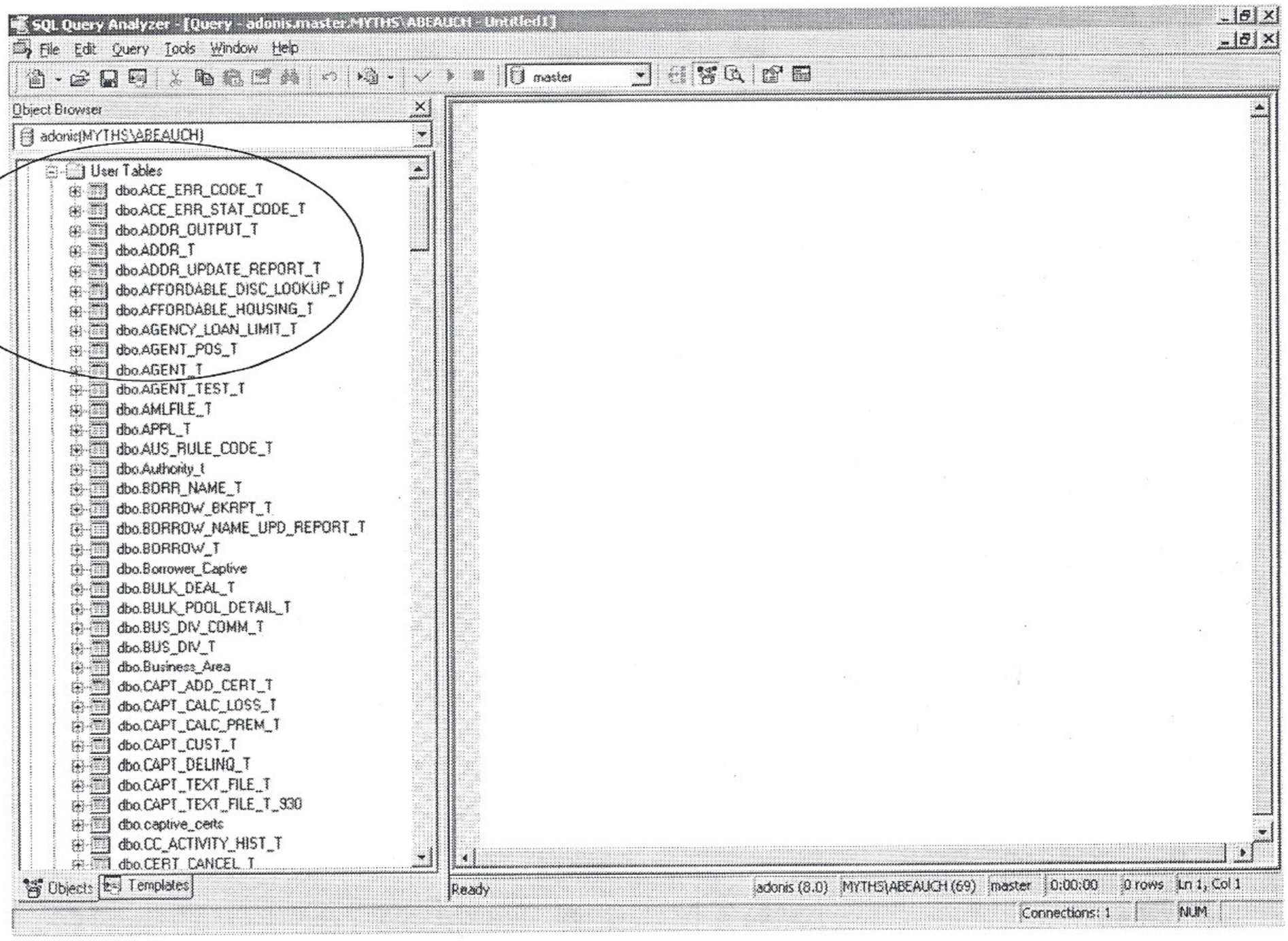
Viewing DSS Tables and Columns

There are many tables that make up DSS. Using the Object Browser, users can view all tables in DSS along with the columns that make up a table.

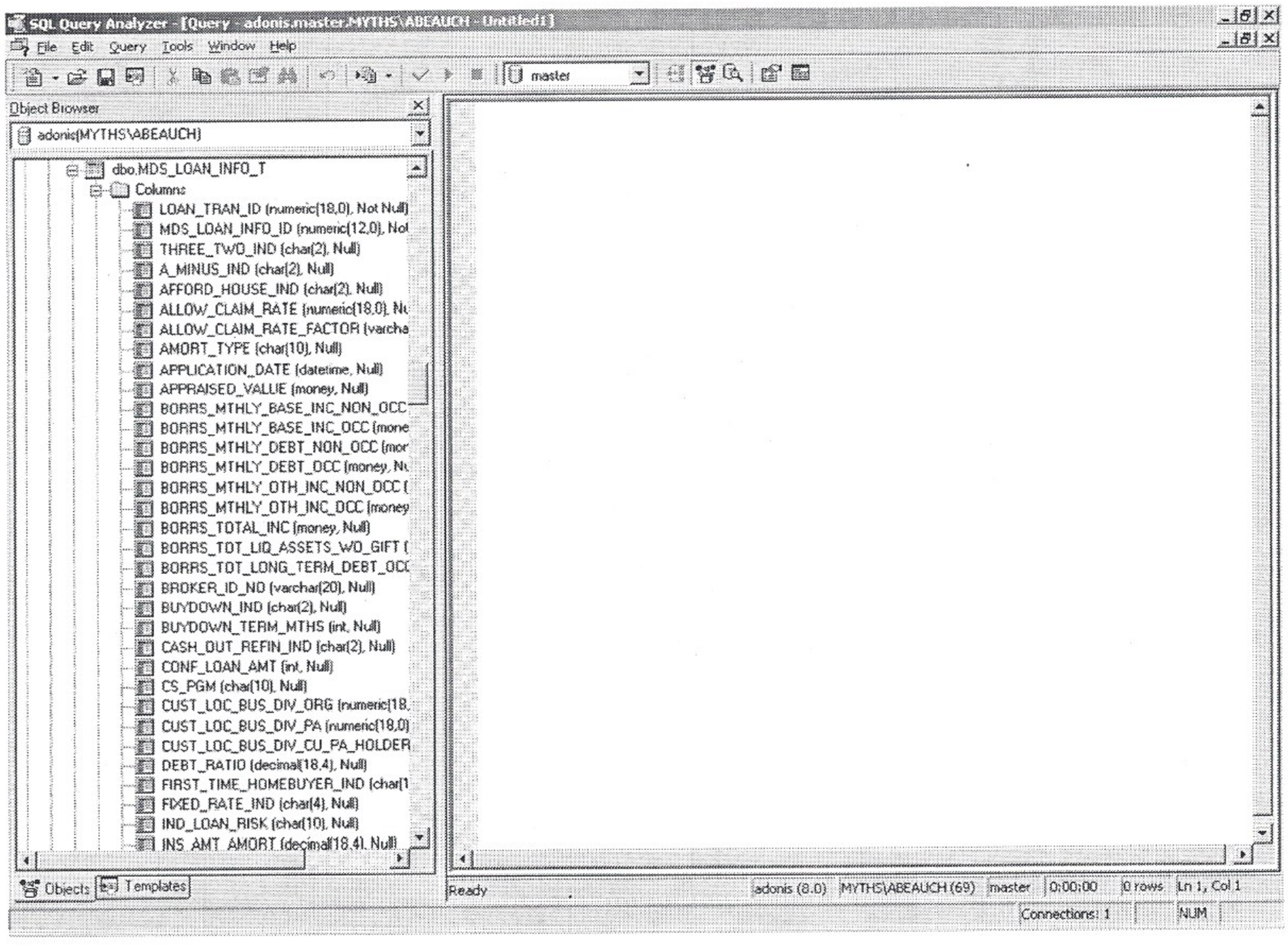
To view tables in DSS use the Object Browser and expand RMIC_DSS by clicking on the + sign.



Expand "User Tables" by clicking on the + sign.



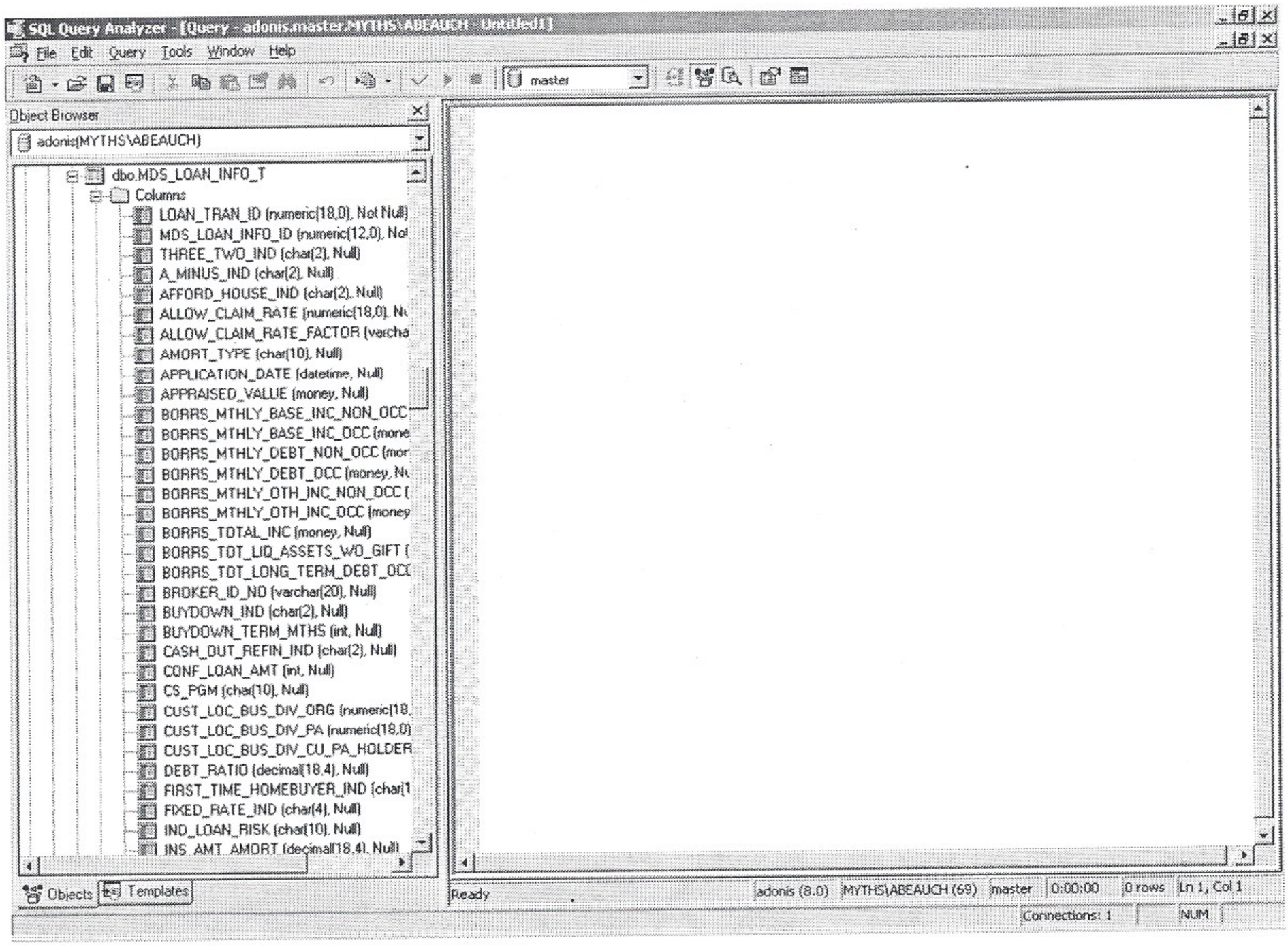
Users can scroll up and down to see all of the tables that make up our DSS Reporting environment. The tables tend to be listed in alphabetical order by prefix, then by table name.
To see the columns that make up a particular table, click on the + sign beside the table name to expand the table. For this example we will look at dbo.MDS_LOAN_INFO_T.



Changing Databases in Query Analyzer

Sometimes we need access to other databases outside of DSS to run the SQL Statements that we need to run. **Note: You have to be given access rights to these databases outside of DSS in order to use them.** Users can only see the databases they have access to in the object browser and the drop down field on the tool bar.

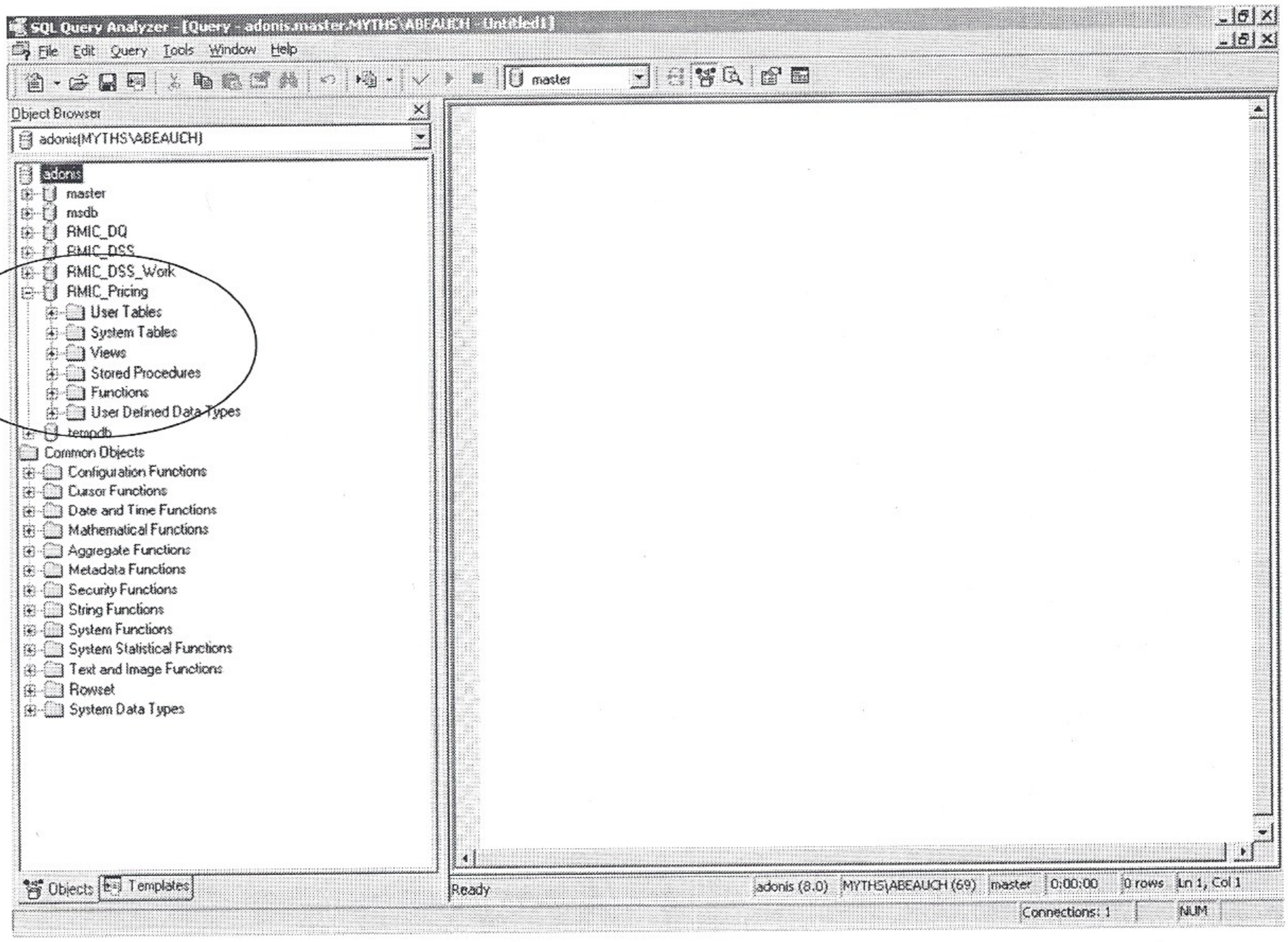
If you already have access to the database needed and you know it is located on Adonis, you can use the object browser window to open up the database that you need to connect to. Collapse the database that is currently open and locate the one that is needed. For example, below I collapsed RMIC_DSS and expanded RMIC_Pricing.



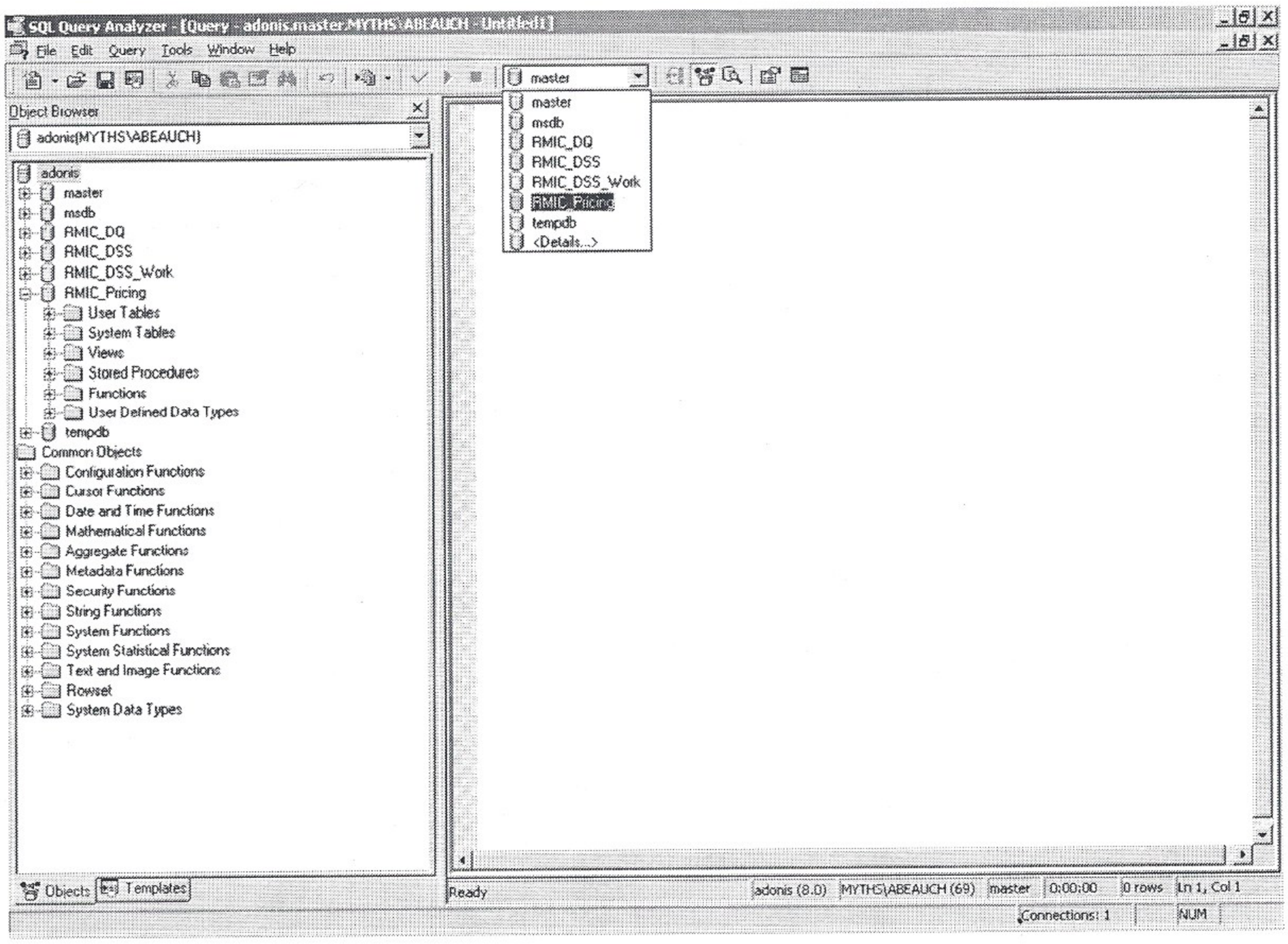
Changing Databases in Query Analyzer

Sometimes we need access to other databases outside of DSS to run the SQL Statements that we need to run. **Note: You have to be given access rights to these databases outside of DSS in order to use them.** Users can only see the databases they have access to in the object browser and the drop down field on the tool bar.

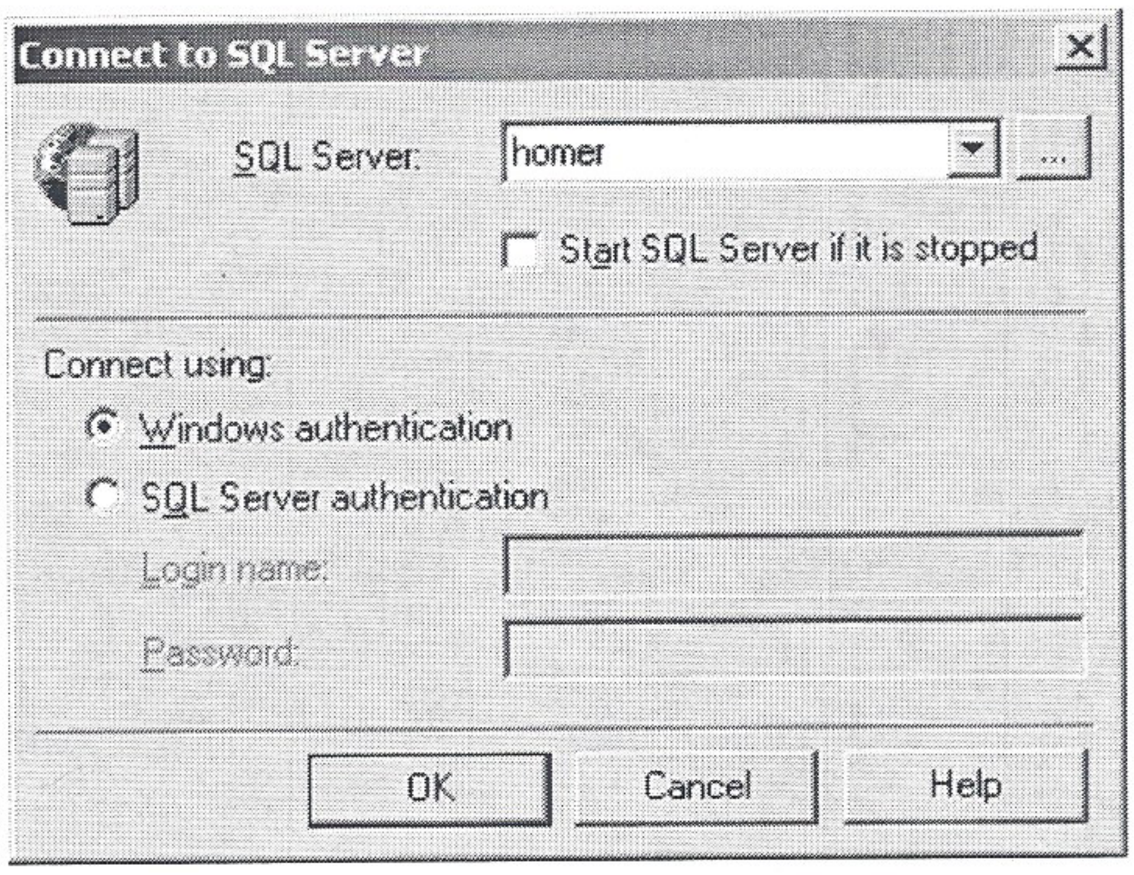
If you already have access to the database needed and you know it is located on Adonis, you can use the object browser window to open up the database that you need to connect to. Collapse the database that is currently open and locate the one that is needed. For example, below I collapsed RMIC_DSS and expanded RMIC_Pricing.



Users will also need to change the drop down window in the tool bar at the top of your screen to point to RMIC_Pricing.



If the database needed is located on a different server, users will need to connect to the appropriate server. If you are logging into Query Analyzer, users can enter the server name in the "Connect to" window upon logging in. If the user is already in Query Analyzer, click on File > Connect and enter the name of the server needed.

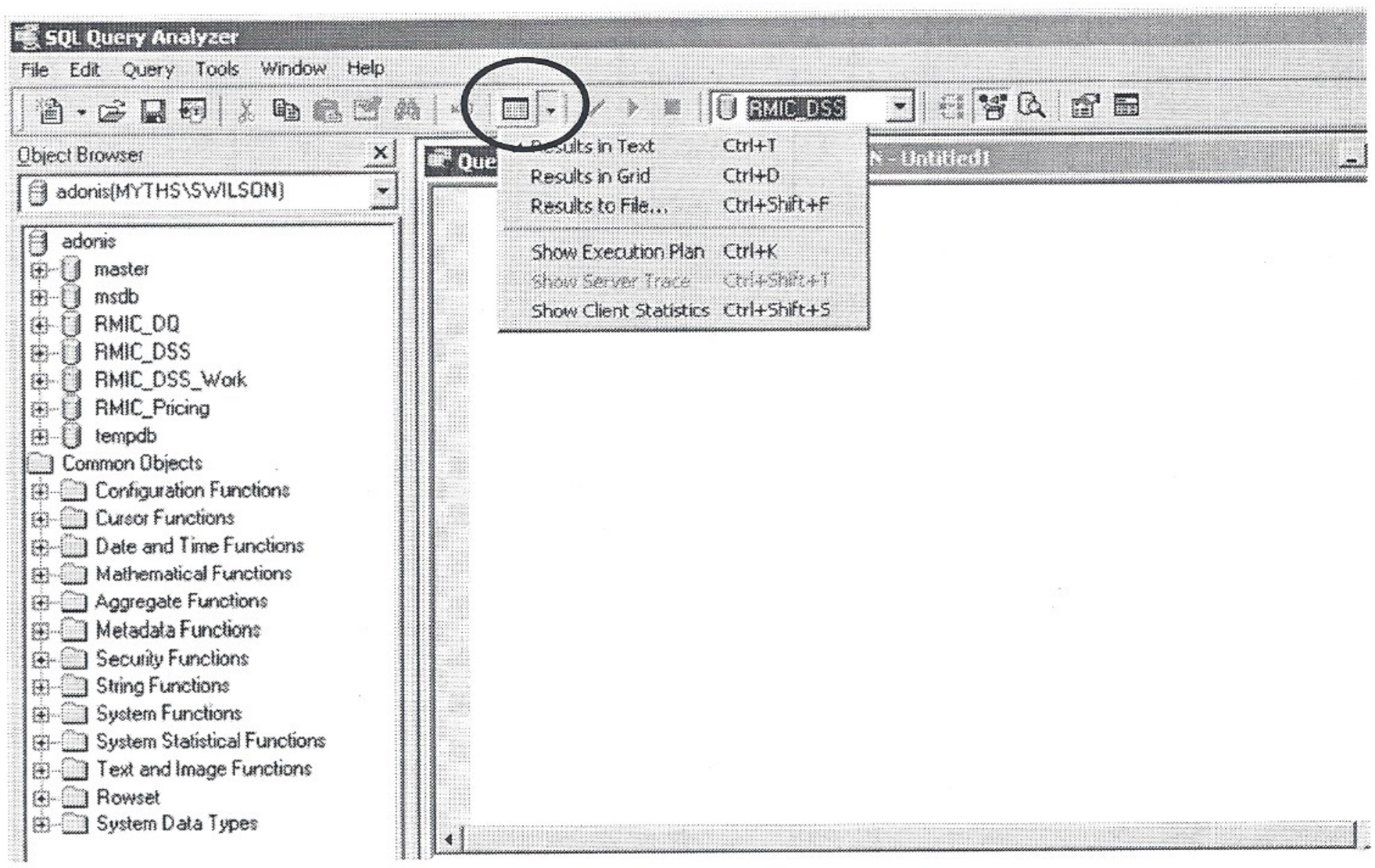


Then the user follows the same steps to make sure the correct database is showing in the drop down menu on the tool bar and uses the object browser to navigate to the appropriate table(s) and columns.

Viewing and Saving Query Results

Depending on the user's needs, there are different options for utilizing the results of a query. Users can view the results or save the results to a file. If you are working on something as a spot check and really don't need to use the data afterwards, then there are a couple of ways to view the data.

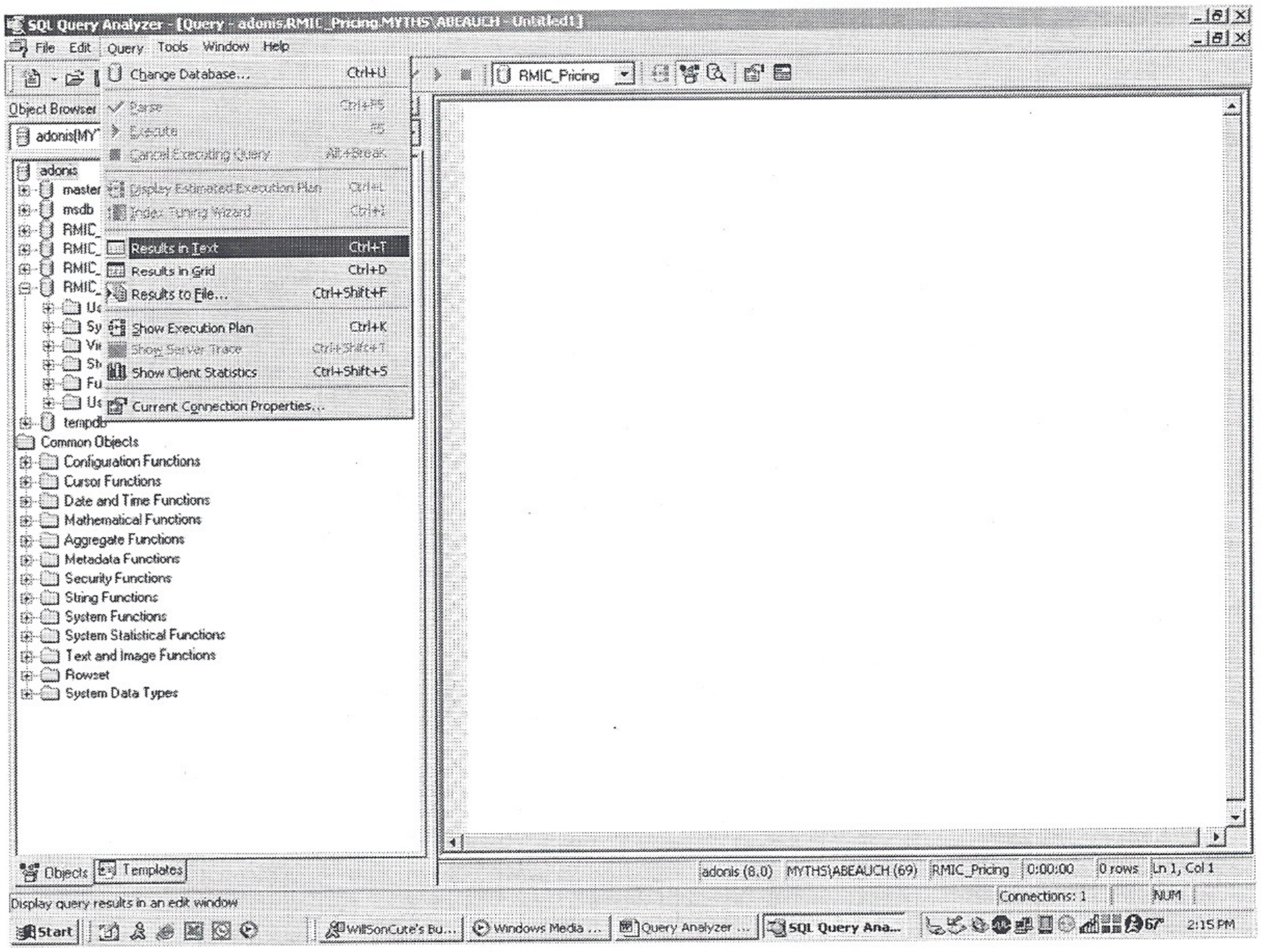
You can click the arrow next to "Execute Mode" icon on the toolbar to select how you would like to review your results.



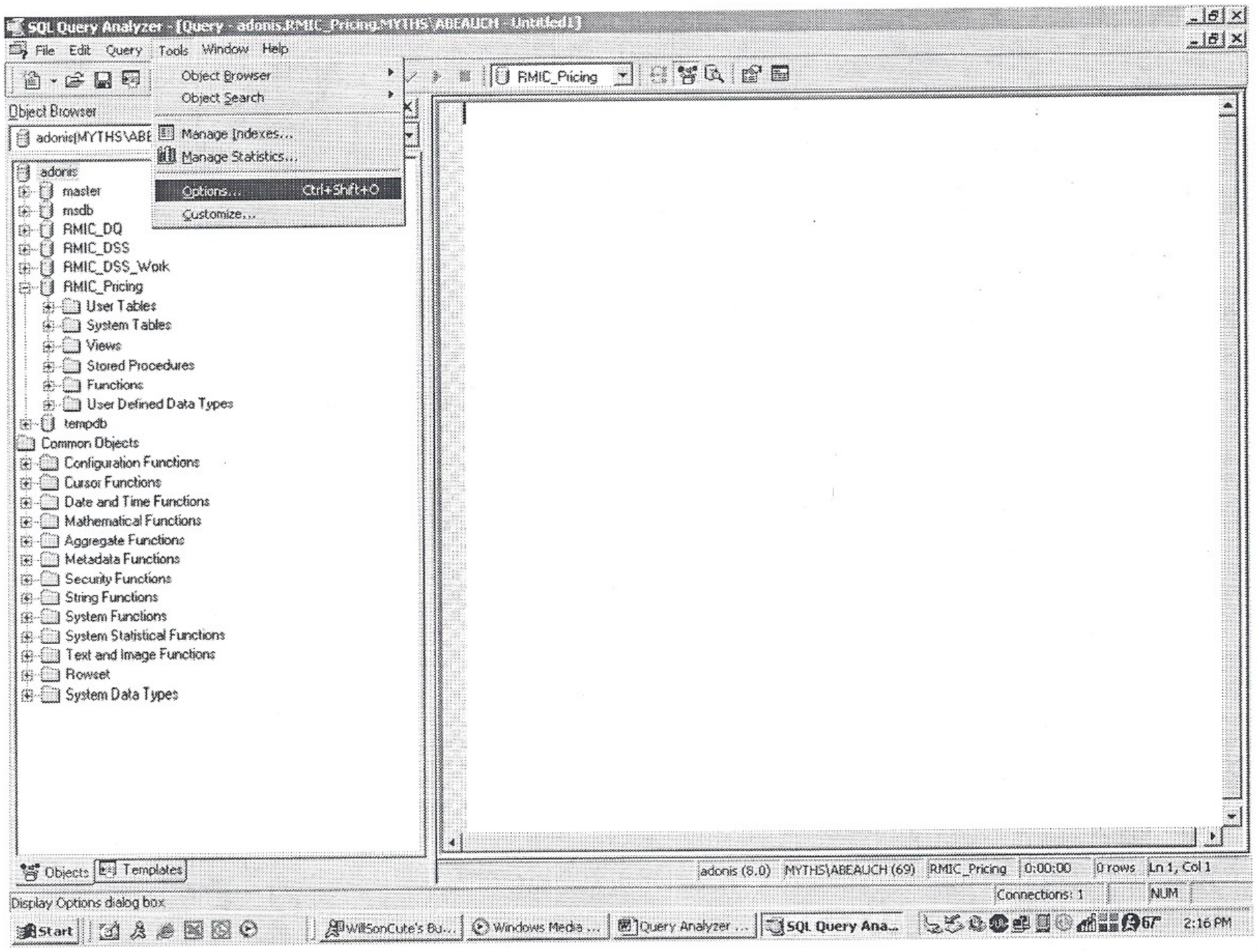
Or you can follow the steps outlined in the next section to utilize the Options menu to format your results.

Results in Text

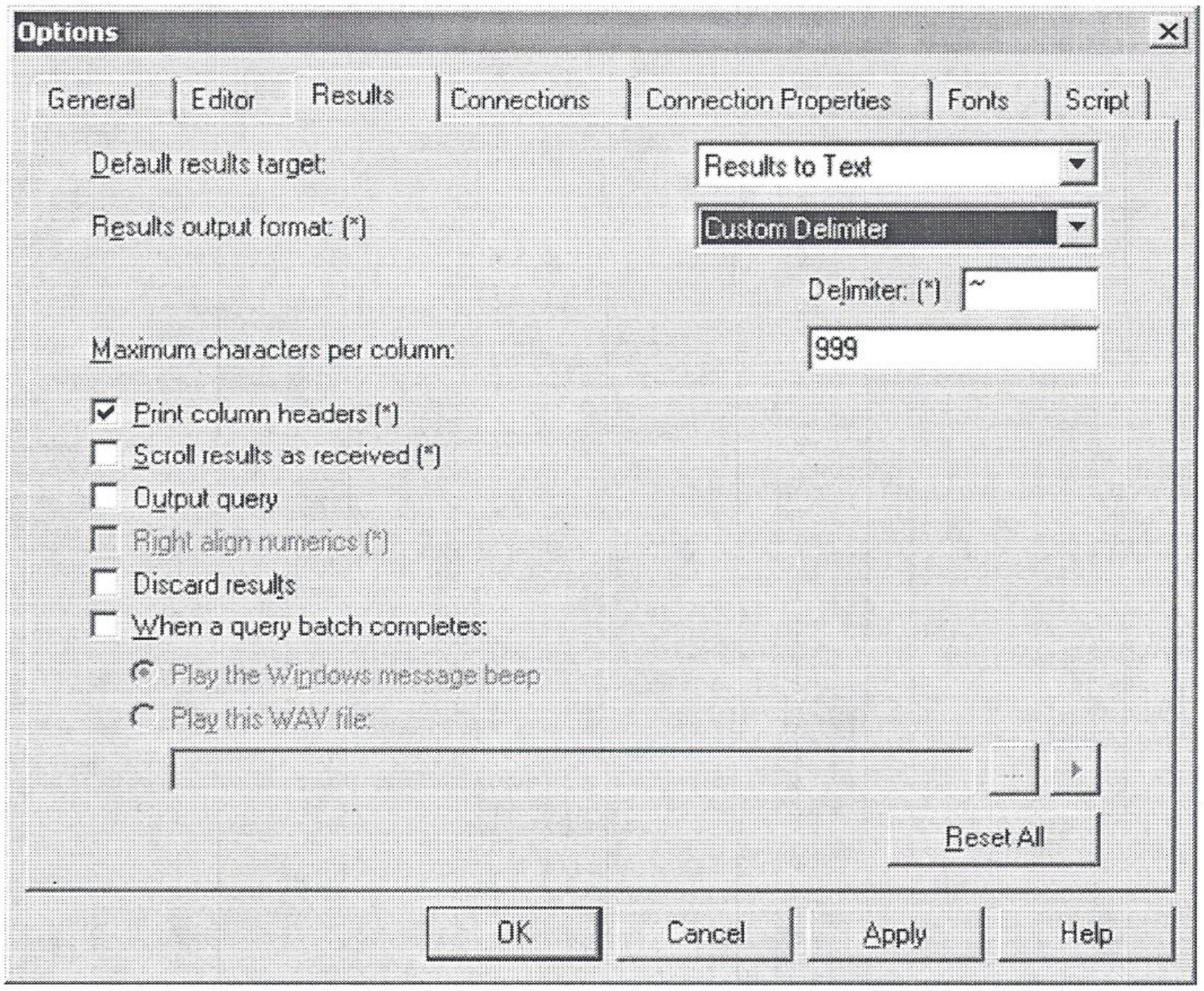
Click on Query in the menu bar and scroll down to Results in Text.



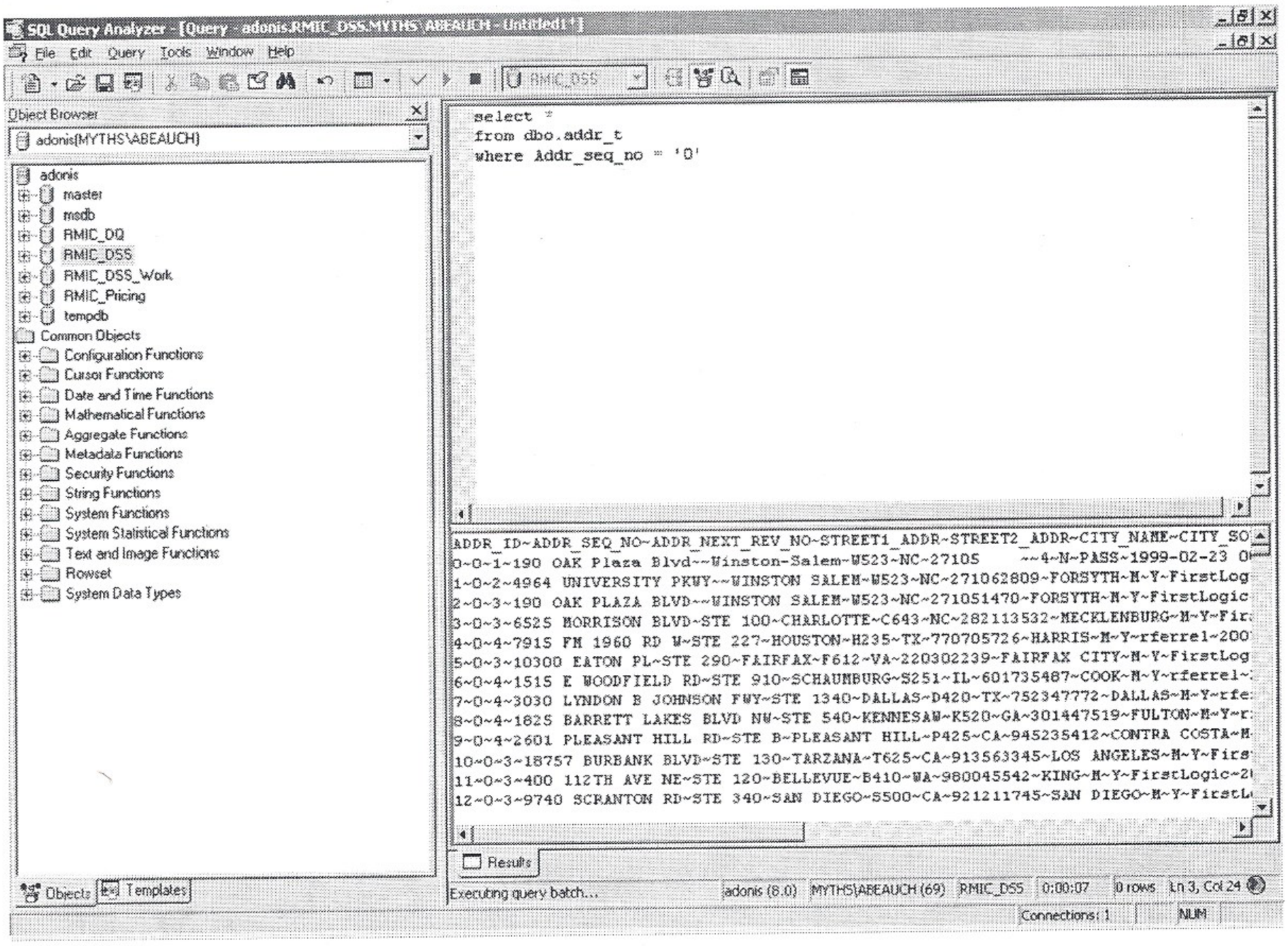
Then go to Tools > Options



Your text will need to be "delimted" by some character. To designate the character, click on the Results Tab. We highly recommend using the tilde as the delimiter, because it is not a character in any DSS database field. Using maximum characters of 999 also prevents any data from getting "chopped off". Users should select print column headers so they will know what data they are looking at in the results.

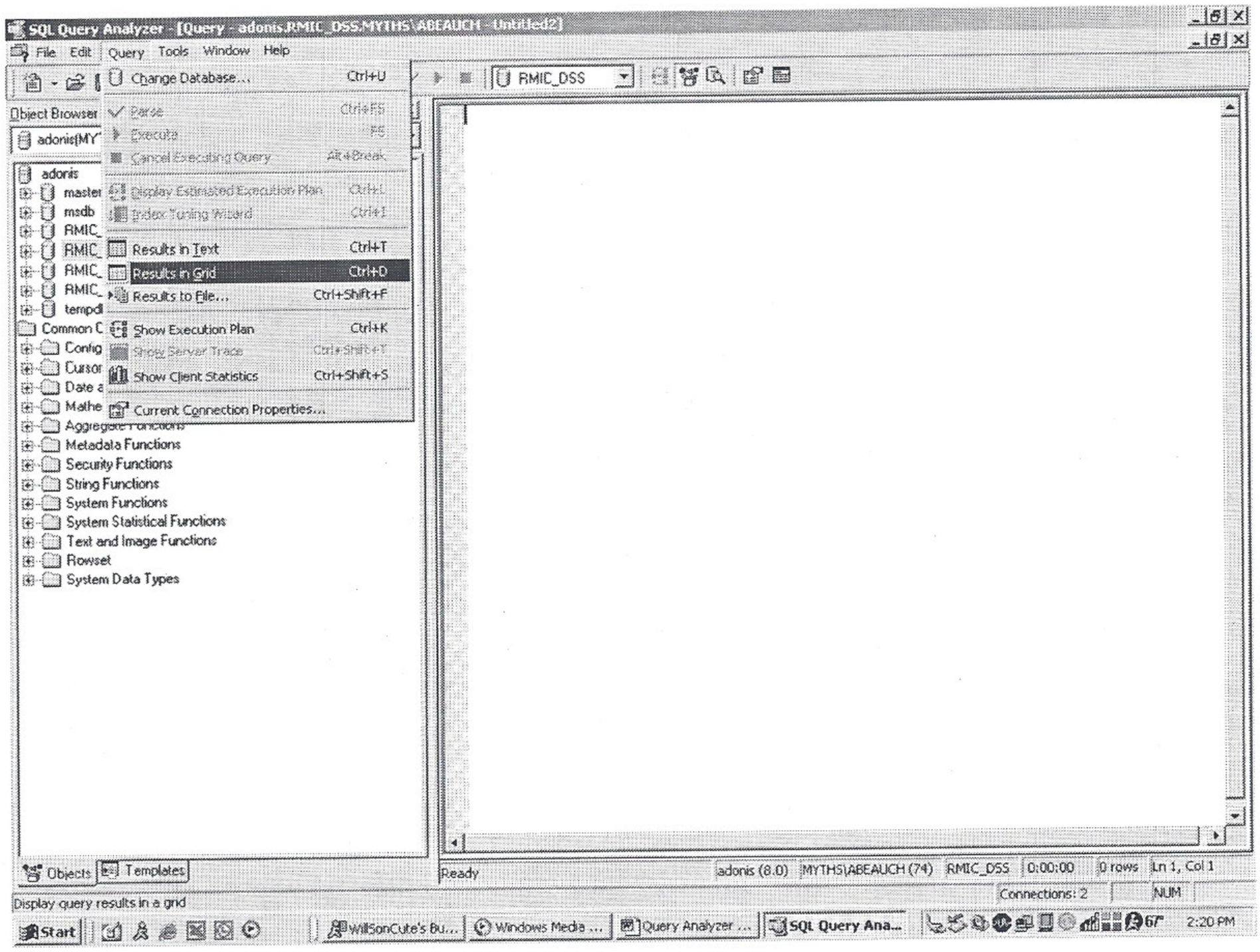


After setting up the parameters for Results to Text, when a query is ran, the results will appear as shown below. It is in a text format delimited by the character the user selected.

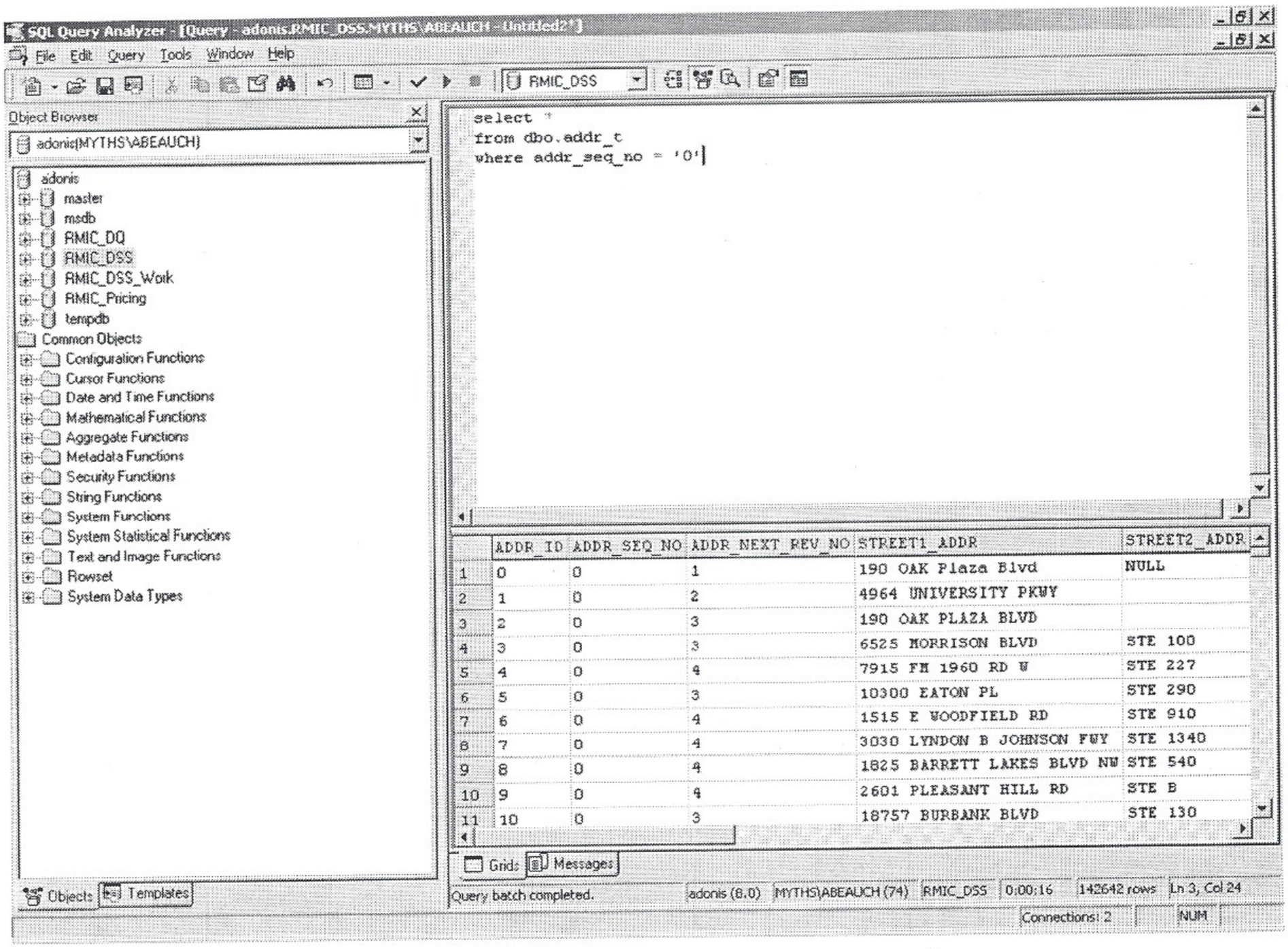


Results to Grid

To use the Results to Grid option, click on Query > Results to Grid.

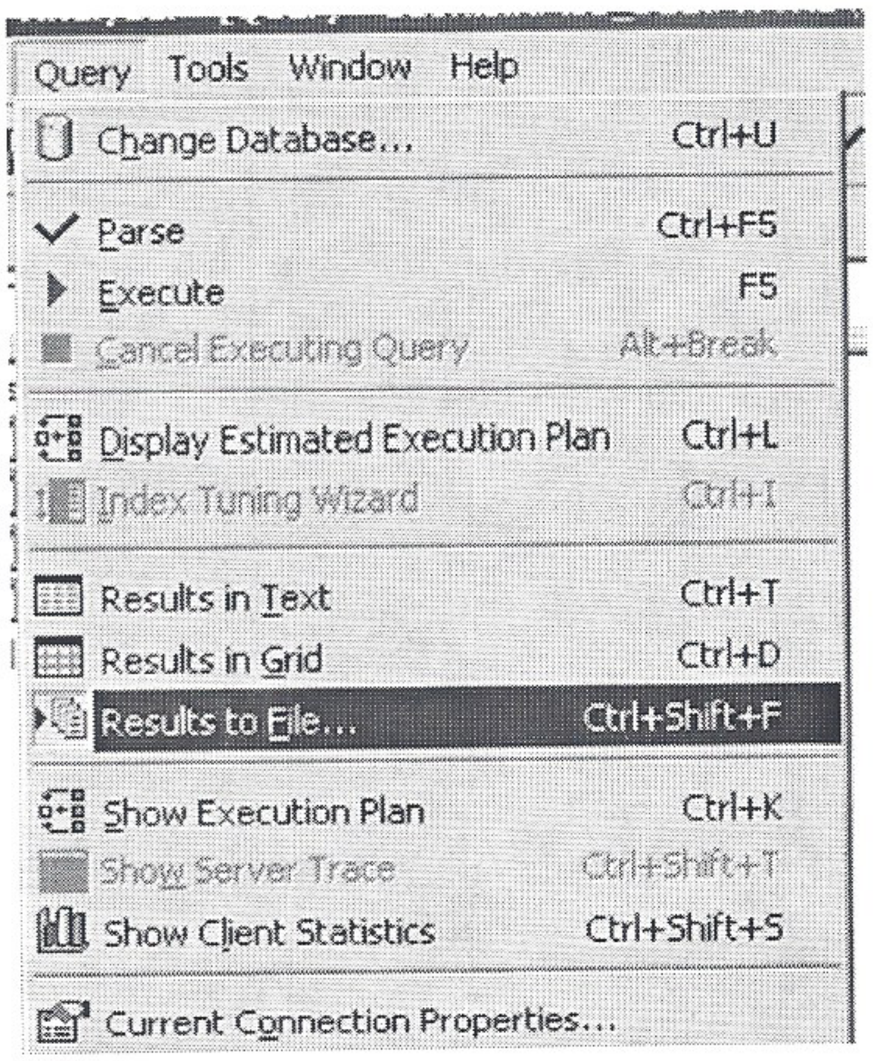


Once the query has completed running the results will be shown in a grid fashion, like below.

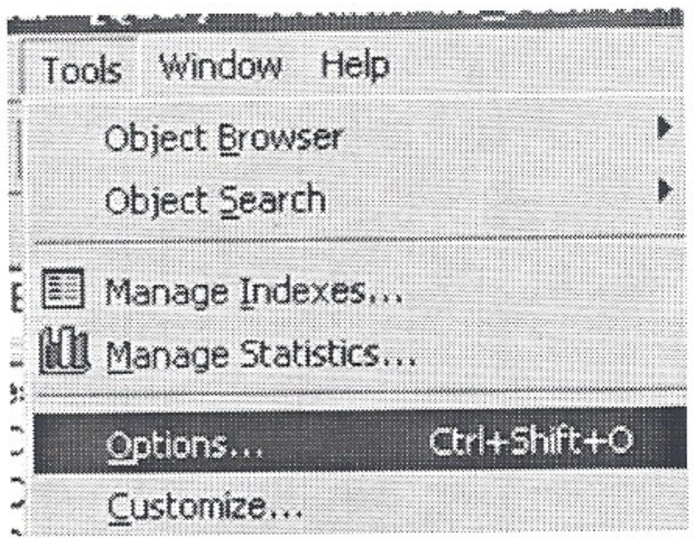


Results to File

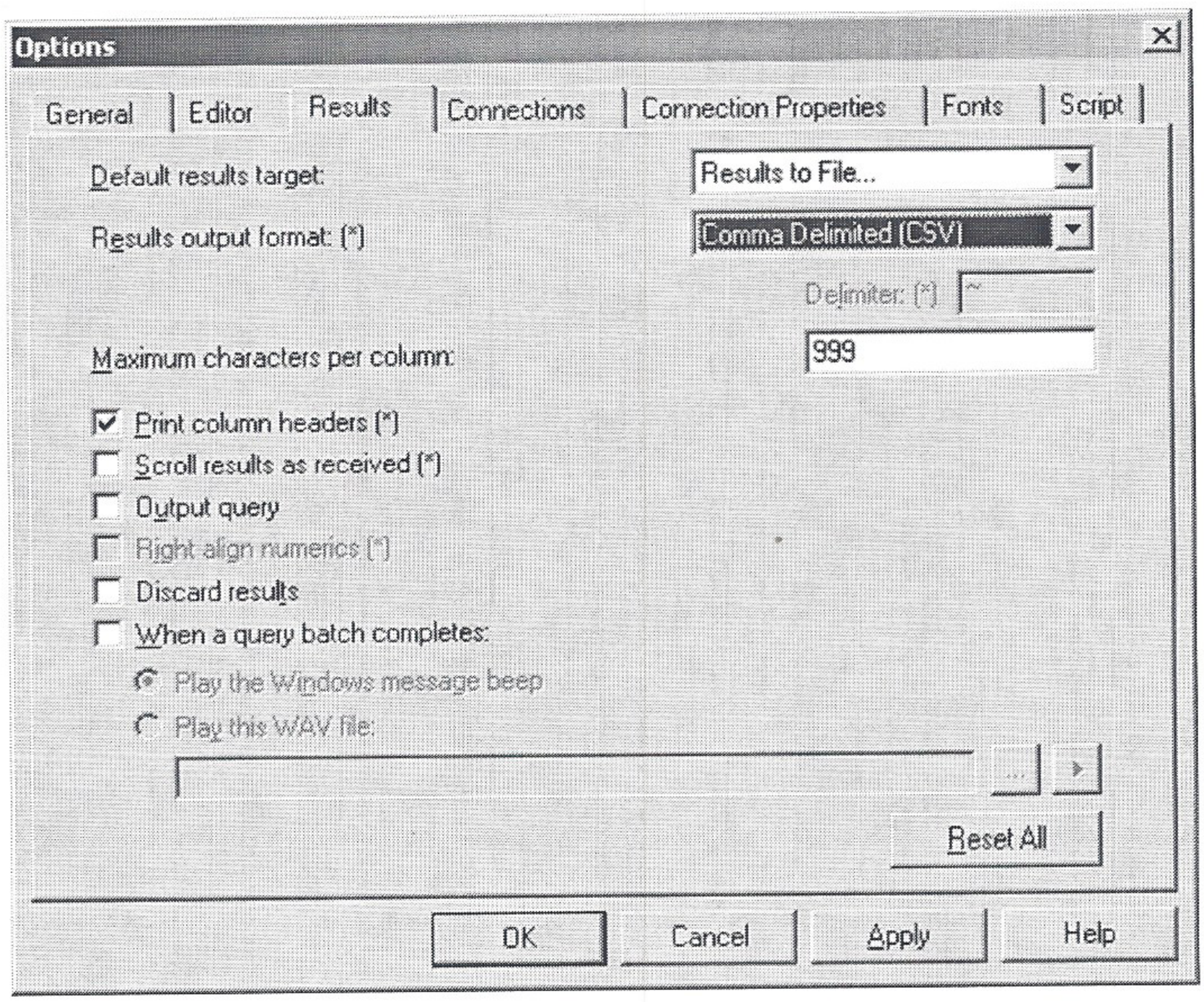
Using the Results to File option allows the user to have the SQL automatically write the results to a designated file, format and location. To use this option, click on Query >Results to File



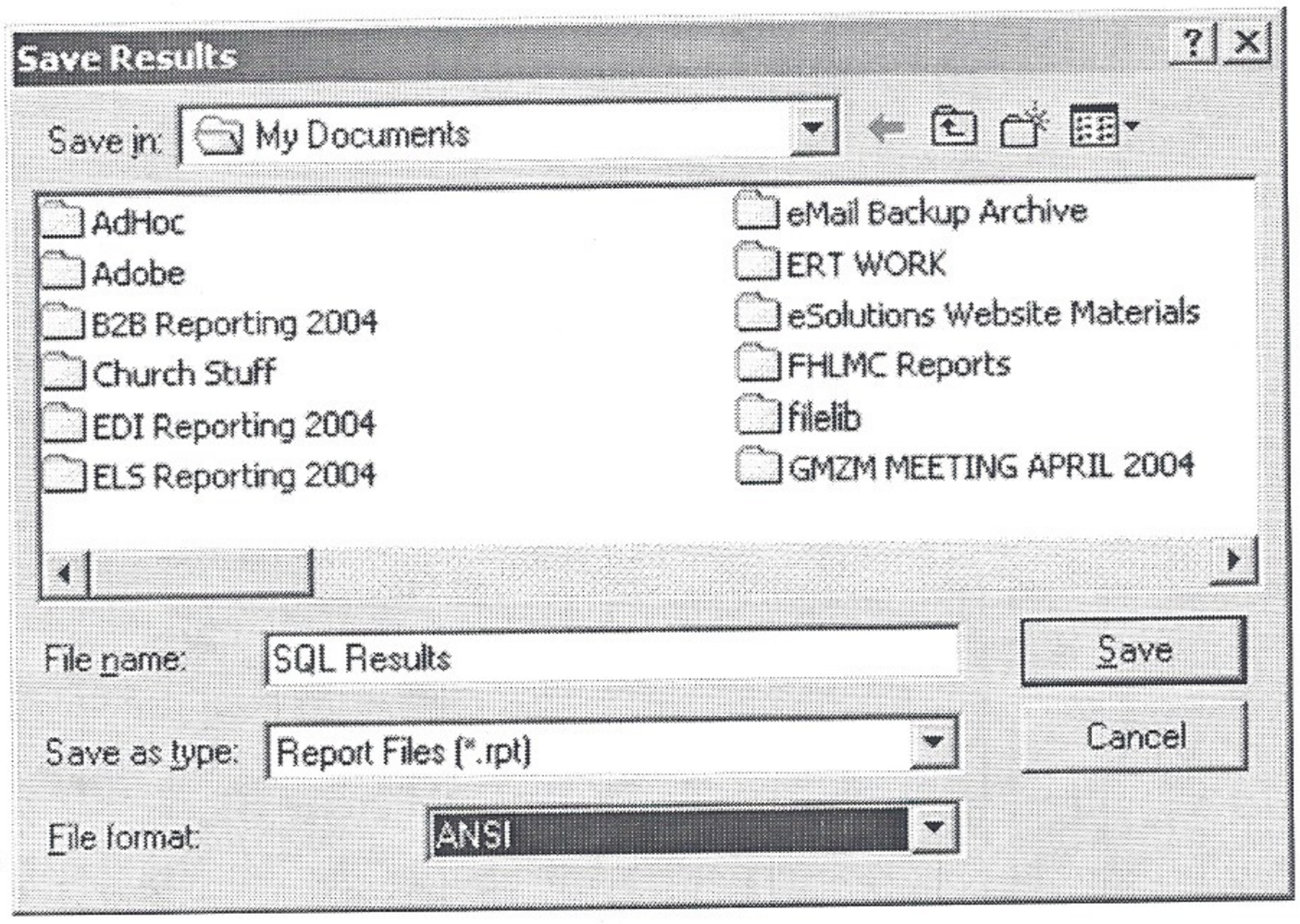
Then go to Tools > Options



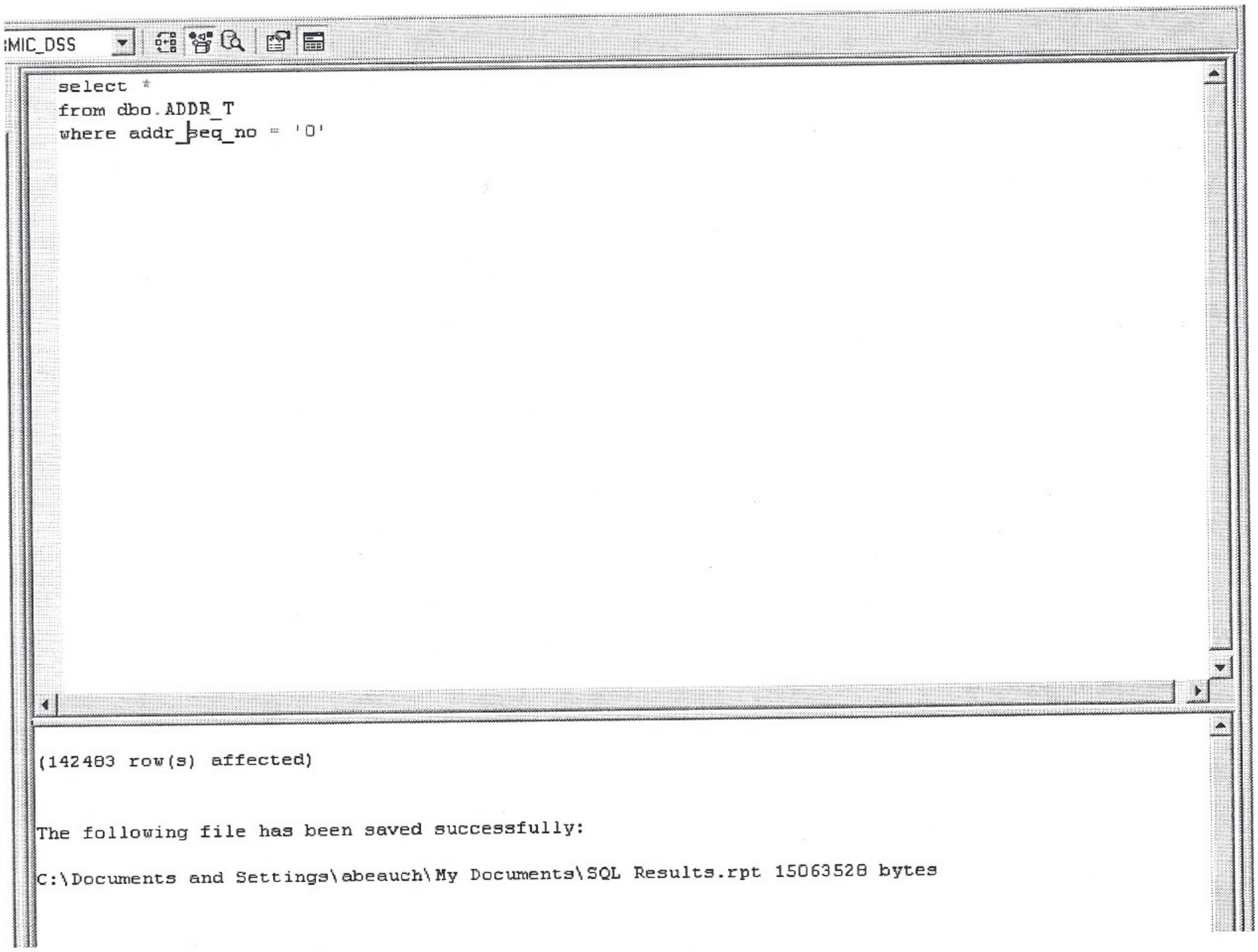
Click on the Results Tab. Default results target should read "Results to File...". Click on the drop down arrow beside "Results Output Format" to specify the output format that is right for you. In this example we will use Comma Delimeted.



Run the SQL Statement. Upon starting the "Save Results" window will appear where the user can designate the location of the results to be saved.



When the statement completes, the user will see the message shown below, confirming completion of the sql statement along with the path where the results have been saved.



The user can then go get their results and open the file. Results can also be imported into tools such as Microsoft Excel and Access for further analysis.

Note that whenever you change the mode in which your results are displayed or saved, you will have to click the "Execute Query" icon to update the screen view.

Class Exercises: Navigating in SQL Query Analyzer

1. What is the name of the Server where DSS resides?

2. In RMIC_DSS database, on the dbo.cert_t table what are the names of the 10^{th} and 18^{th} columns?

3. In RMIC_Pricing Database, how many tables are there?

4. In RMIC_Pricing Databse, in the dbo.MUNITAX table, what is the name of the last column?

5. Enter the following SQL Statement and use the three different options for viewing results:

```
select *
from dbo.AFFORDABLE_HOUSING_T
```

   a. Results to Text
   b. Results to Grid
   c. Results to File

Learning to Write SQL Statements

Note: where users see "Top x" in a sql statement, this has only been included for class to limit the number of rows returned by the sql and therefore limit the time it takes for the sql statement to run.

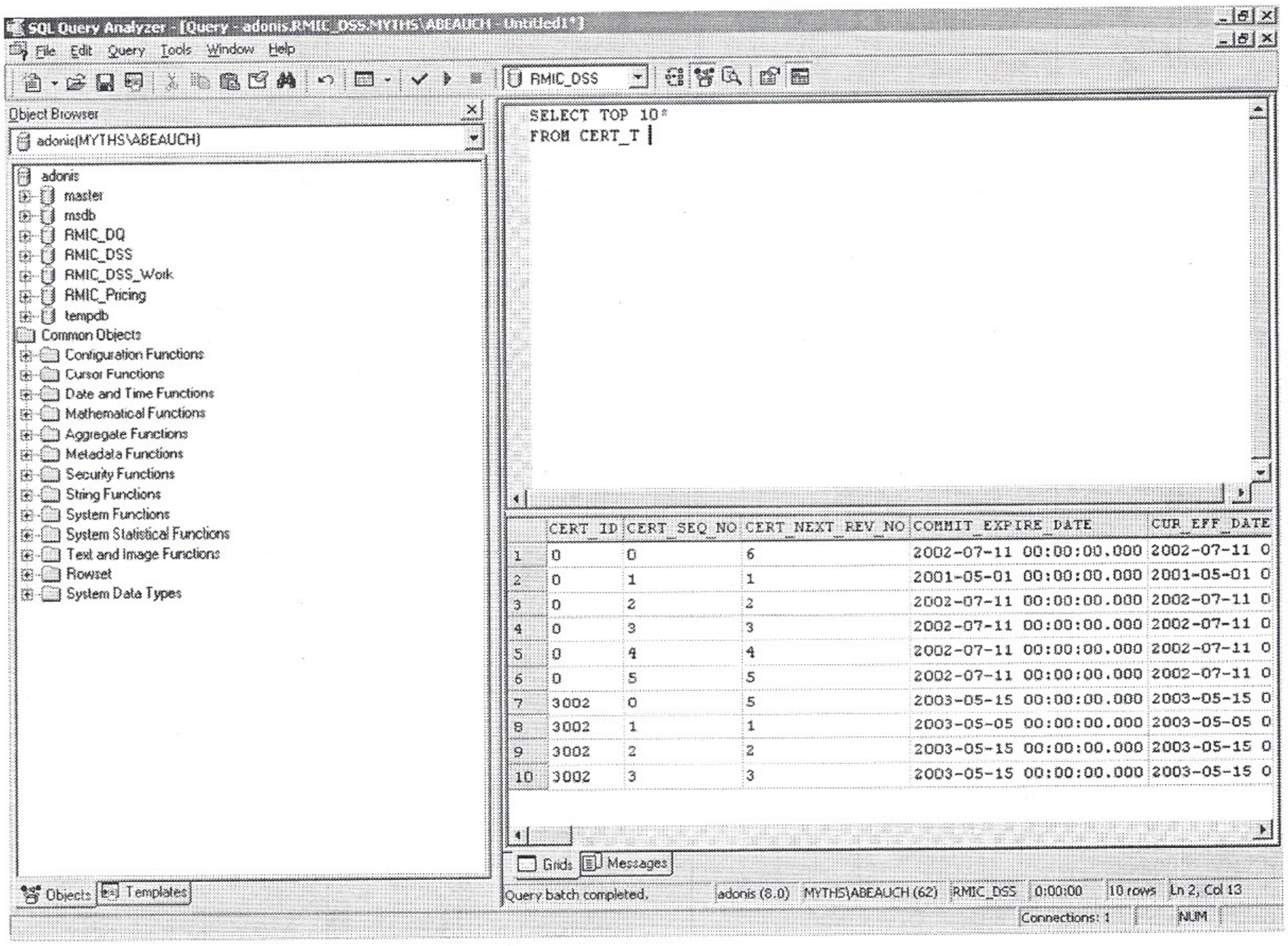
1) Basic select statement

The basic select statement contains two parts: what to select, and from where to select. These two elements are the foundation of a sql select statement. The example used below uses an * for the selection criteria. An * translates into "give me everything". This type of statement is good if you want to see what the data/fields look like in a table.

The statement below will return, for the first 10 rows, all of the fields in the cert_t table.

```
Select top 10 * from cert_t
```

Click the Execute Query icon  or hit F5 on the keyboard to execute your query.



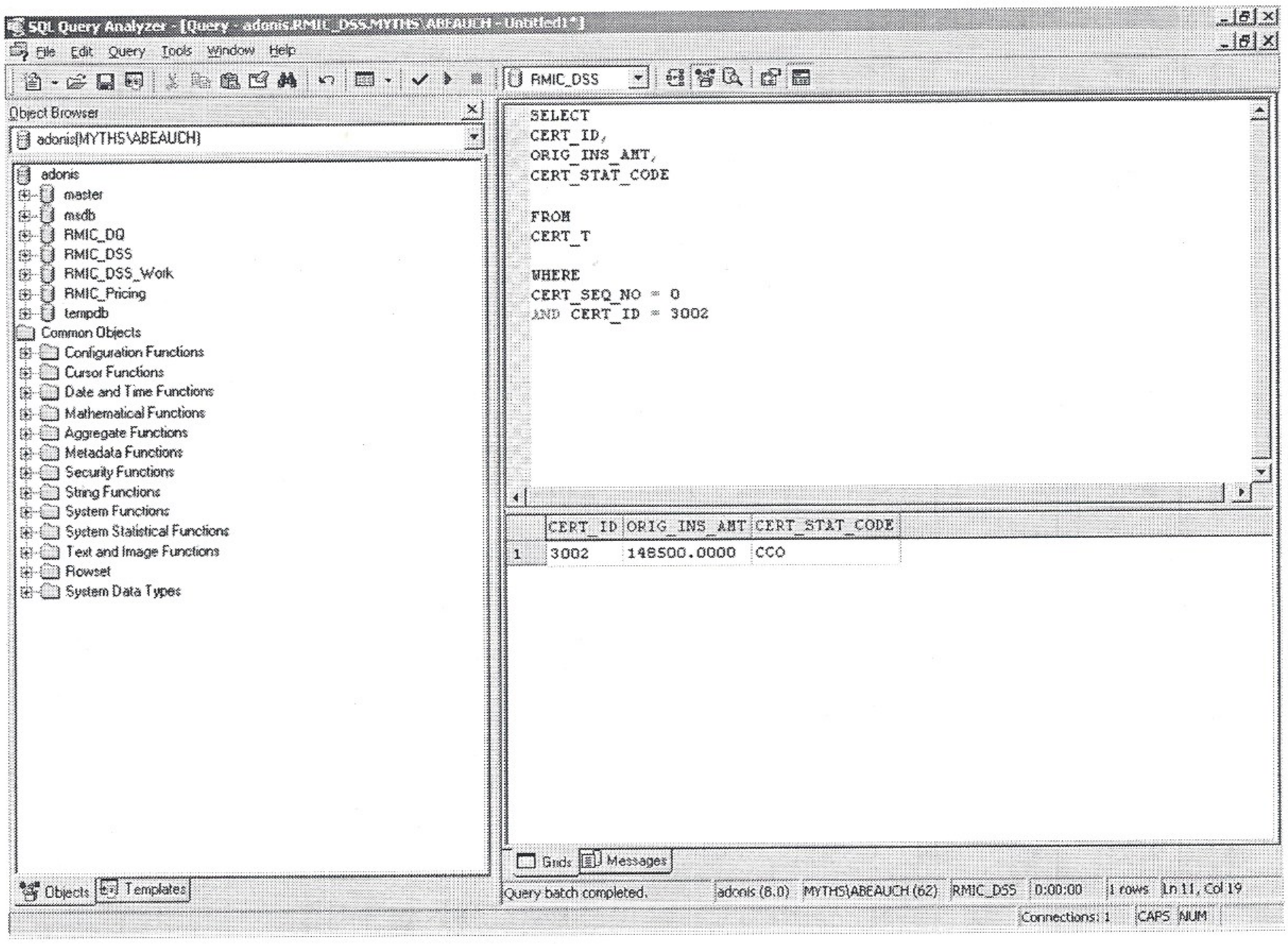
2) Where conditions and current record indicators

In addition to telling sql what to select and from where to select, users can require that certain criteria be met in for data that gets returned. This can be compared to on-line vehicle purchase sites where the user limits their search to what they want. "I want a truck". "It has to be red". "It should also have a sunroof". This limits the vehicle search to only those meeting the criteria given.

In sql, this functionality is referred to as a Where condition. This select statement example can be translated into the following:
Give me the certificate id, original insured amount, certificate status from cert_t table for certificate 3002 – current record only. The statement also requires that the data be the most current for that certificate. There are several tables in DSS that contain history. It is important that users look for fields such as current record indicators, or sequence numbers if they want to see the most recent version of a record.

Select cert_id, orig_ins_amt, cert_stat_code

From cert_t
Where cert_seq_no = 0 ---current record indicator and filter condition
And cert_id = 3002

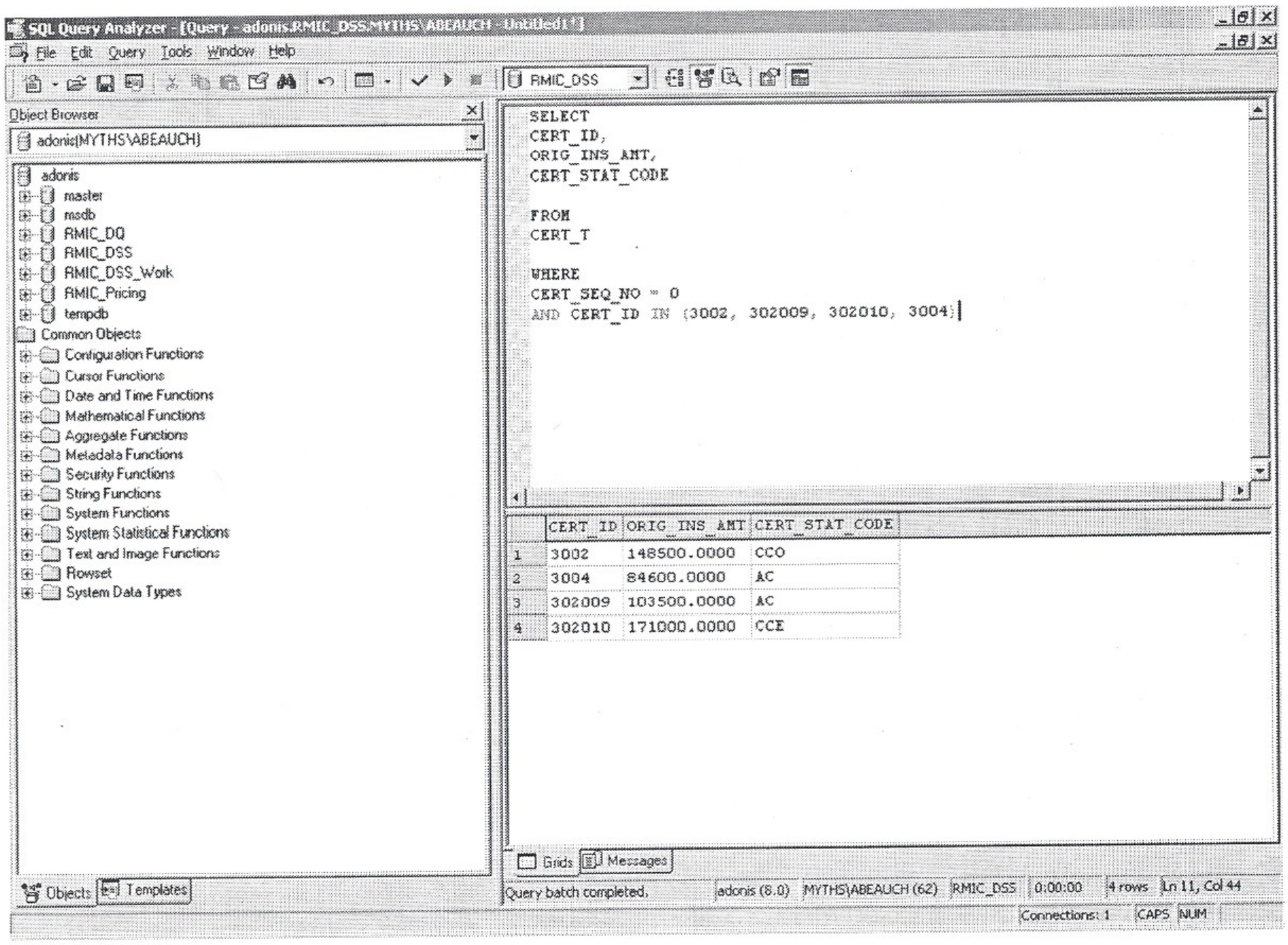


3) IN statements

The IN statement is used within a Where condition. IN statements are useful when you want data for specific records. For example if you wanted data returned for 10 location business divisions (LBD) of a customer and you had the LBD IDs for these customers, you would include the specific LBD IDs within the IN statement.

The sql statement below uses the functionality of an IN statement to return the certificate id, original insured amount, certificate status from cert_t table for certificate 3002, 302009, 302010, 3004 – current record only.

Select cert_id, orig_ins_amt, cert_stat_code
From cert_t
Where cert_seq_no = 0 ---current record indicator and filter condition
And cert_id **IN** (3002, 302009, 302010, 3004)

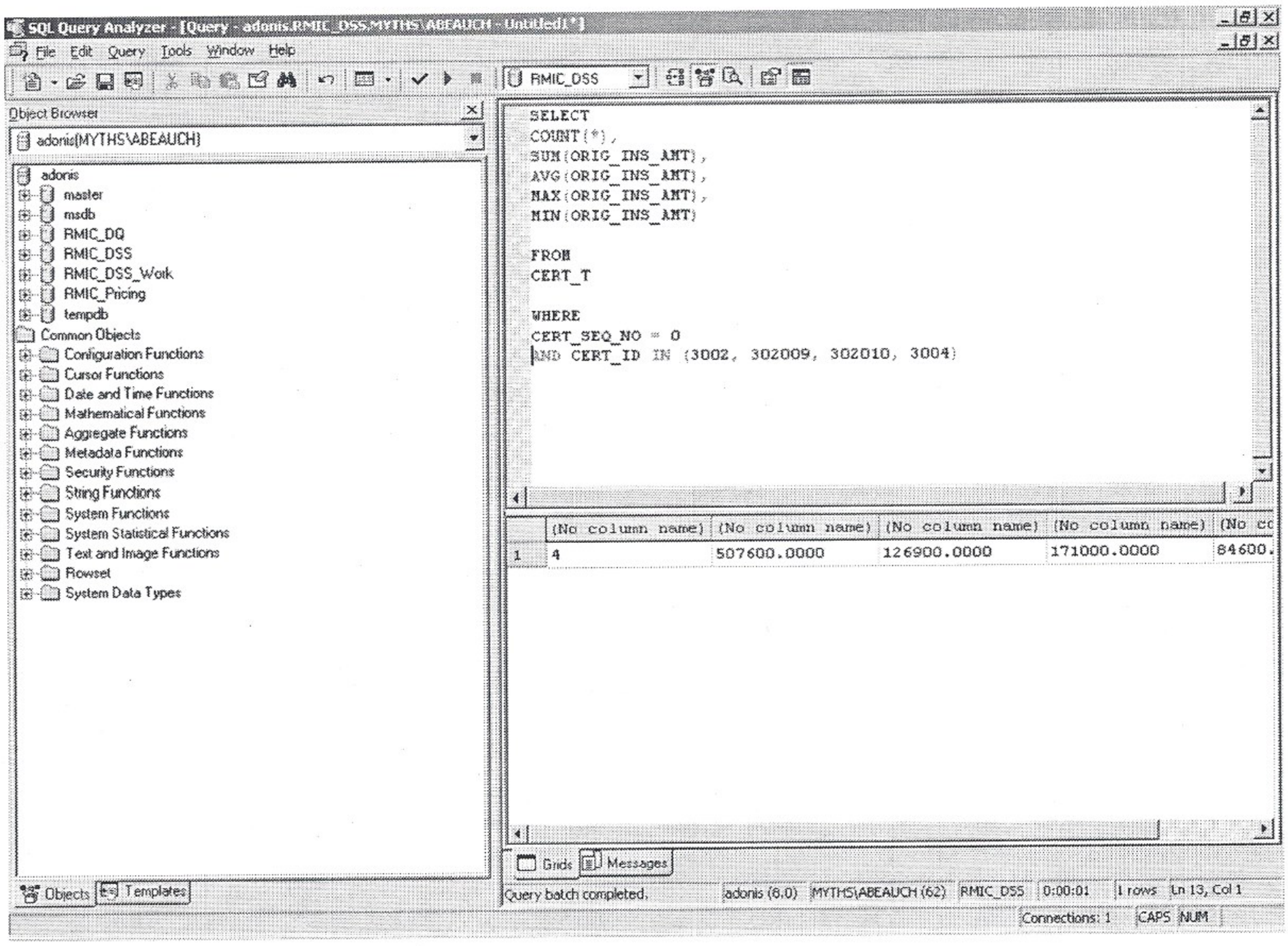


4) Aggregate functions

Sql will also provide some mathematical functions to summarize the data users are looking at. Sql will count, add and average data. It can also return the maximum and minimum values for a data element. This is useful when you don't need a lot of data to answer a question.

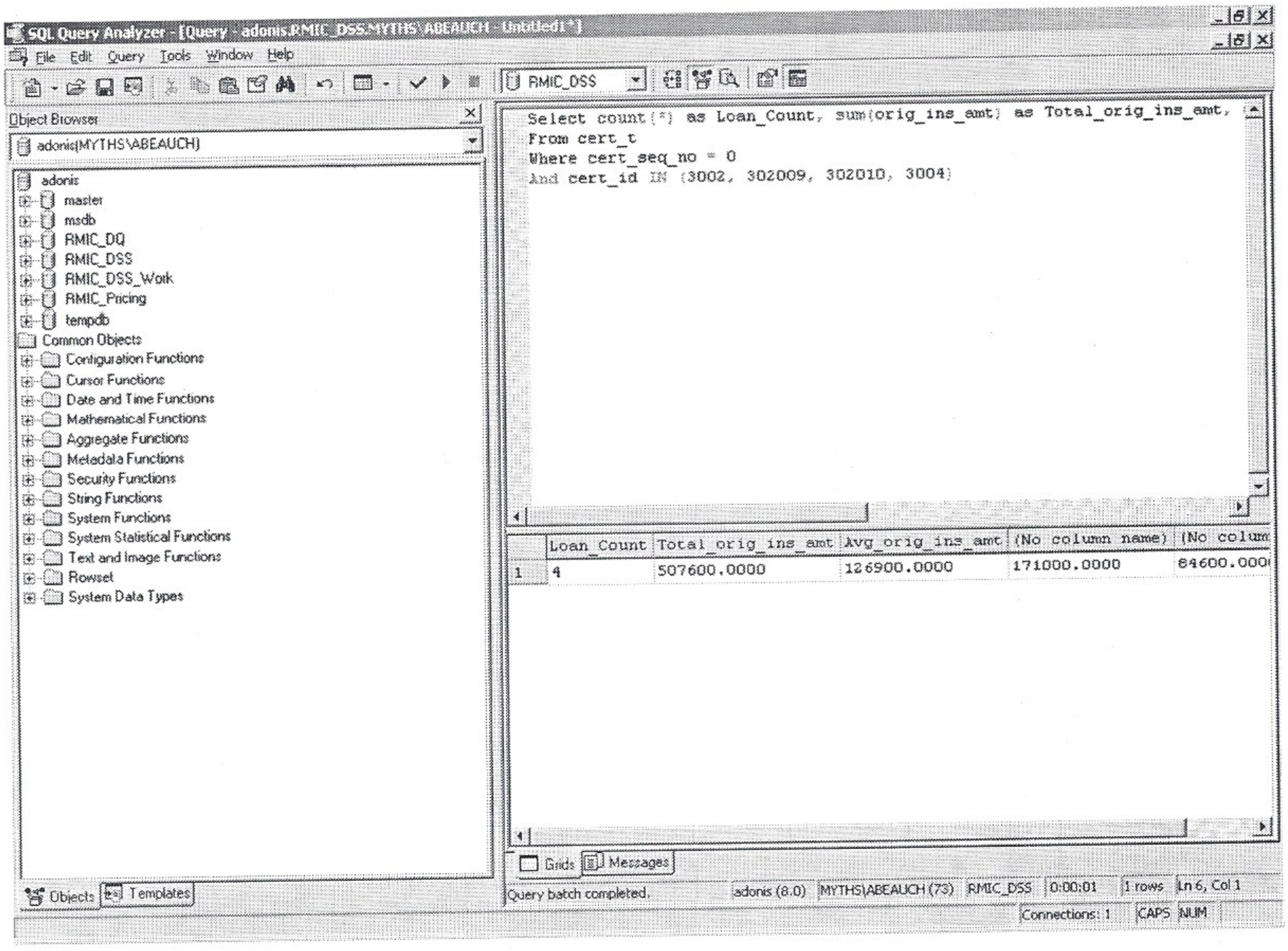
This sql statement combines all of the aggregate functions for the certificate numbers indicated. It provides the count of loans, the sum and average of the original insured amount. It also shows the smallest and largest insured amount.

```
Select count(*), sum(orig_ins_amt), avg(orig_ins_amt),
max(orig_ins_amt), min(orig_ins_amt)
From cert_t
Where cert_seq_no = 0
And cert_id IN (3002, 302009, 302010, 3004)
```



To give column names to the output, users can use aliases for the column names. Count(*) as Loan_Count will name the column Loan Count. The sql below is the same as above with the addition of three column names.

Select count(*) as Loan_Count, **sum**(orig_ins_amt) as Total_orig_ins_amt, **avg**(orig_ins_amt) as Avg_orig_ins_amt, **max**(orig_ins_amt), **min**(orig_ins_amt)
From cert_t
Where cert_seq_no = 0
And cert_id **IN** (3002, 302009, 302010, 3004)

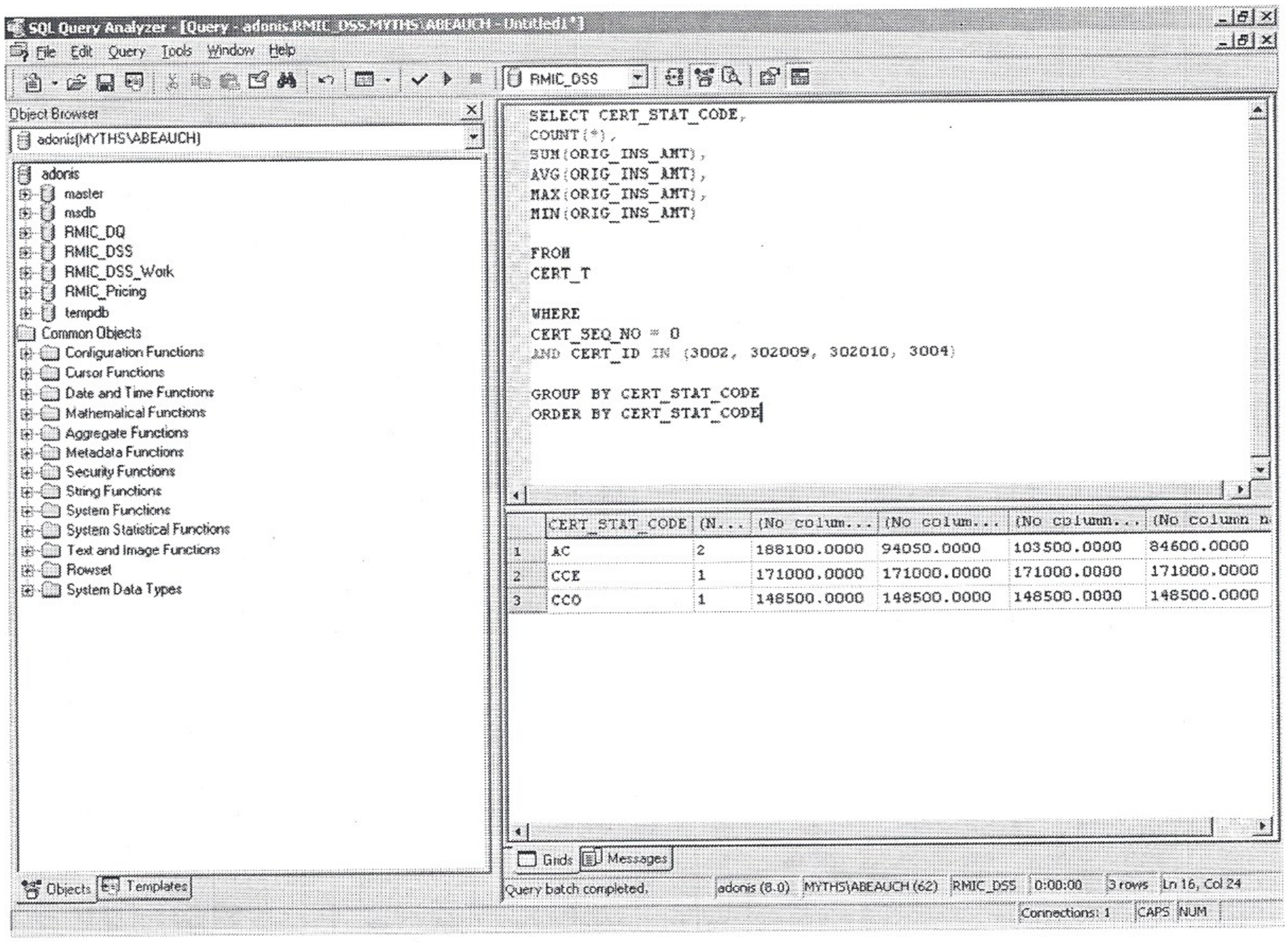


5) Group By and Order By conditions

Sql allows the user to "organize" the output of a statement. A good way to do this is to use Group By and Order By conditions. Group By tells sql how to group the data. Order By tells sql how to sort the data.

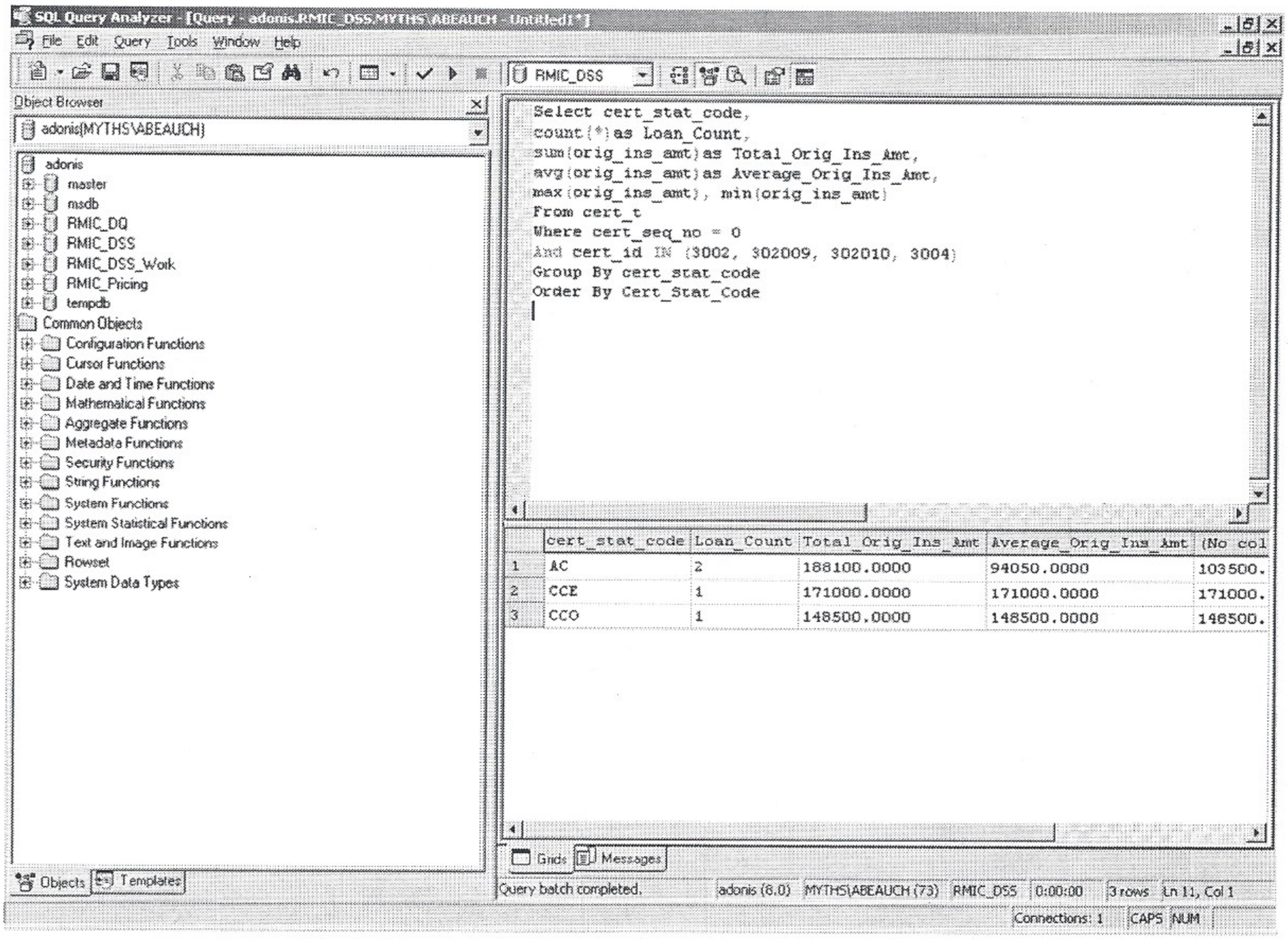
In this example the sql provides (for the loans indicated) the count of loans, the sum and average of the original insured amount. It also shows the smallest and largest insured amount. It then goes another step and groups the data by certificate status and sorts by certificate status.

```
Select cert_stat_code, count(*), sum(orig_ins_amt), avg(orig_ins_amt),
max(orig_ins_amt), min(orig_ins_amt)
From cert_t
Where cert_seq_no = 0
And cert_id IN (3002, 302009, 302010, 3004)
Group By cert_stat_code
Order By Cert_Stat_Code
```



To give column names to the output, users can use aliases for the column names. Count(*) as Loan_Count will name the column Loan Count. The sql below is the same as above with the addition of three column names.

```
Select cert_stat_code,
count(*)as Loan_Count,
sum(orig_ins_amt)as Total_Orig_Ins_Amt,
avg(orig_ins_amt)as Average_Orig_Ins_Amt,
max(orig_ins_amt), min(orig_ins_amt)
From cert_t
Where cert_seq_no = 0
And cert_id IN (3002, 302009, 302010, 3004)
Group By cert_stat_code
Order By Cert_Stat_Code
```

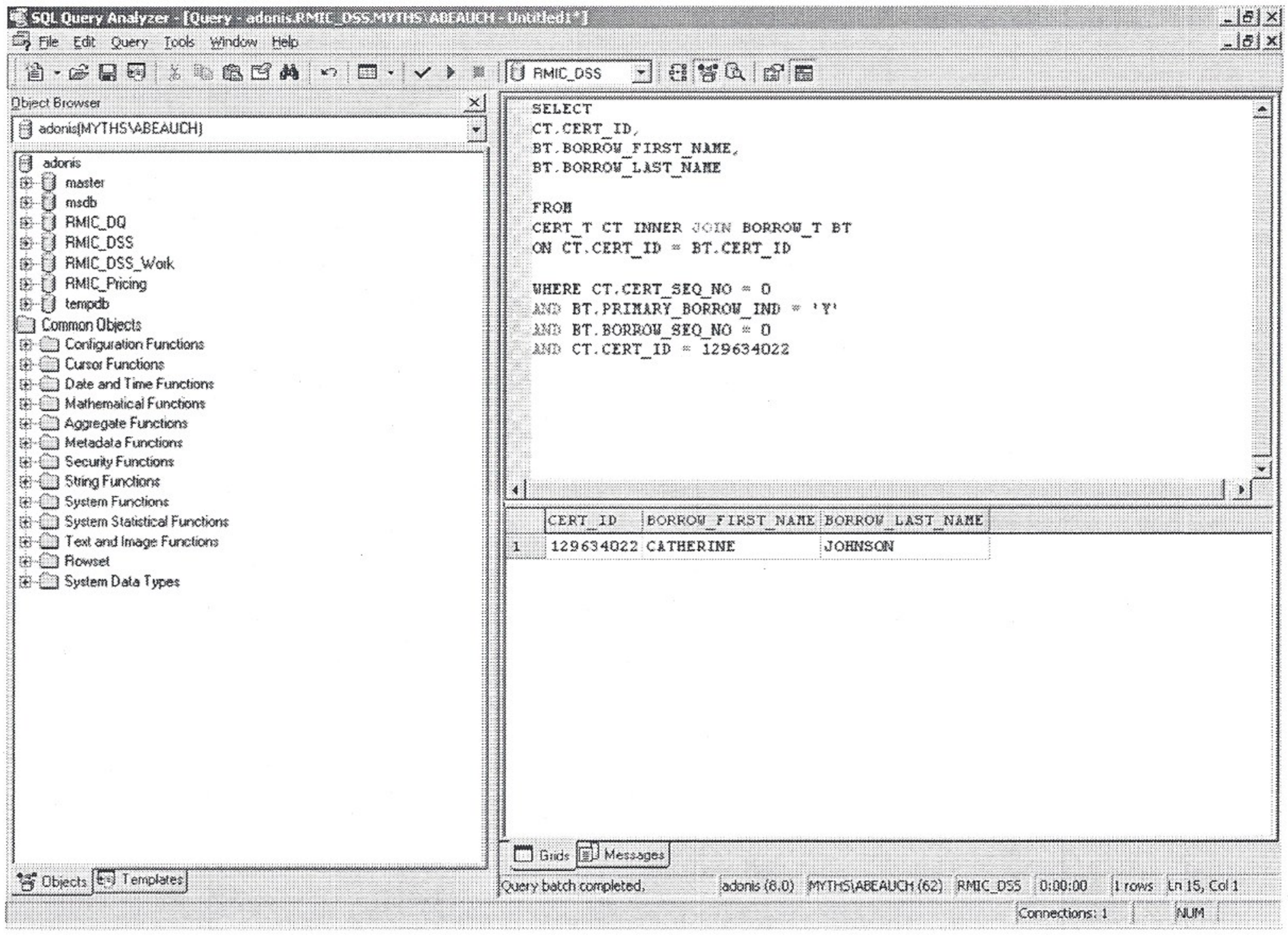


6) Table Joins and Aliases

Sometimes we need to get data from more than one table. Users can "join" tables in order to meet this need. Table aliases allow the user to give a shorter table name. This shortens the amount of text to be entered for each field in each table. For example if you are going to pull 10 fields from the Loan_Case_Info_t table and join this to fields from another table, instead of typing Loan_Case_Info_t in front of all ten fields, use an alias (LC) and just type "LC." in front of the fields. Aliases are time-savers. In more complex table joins aliases may be required to identify fields properly.

The sql statement below joins two tables in order to return the certificate id, primary borrower first name and last name for certificate 129634022.

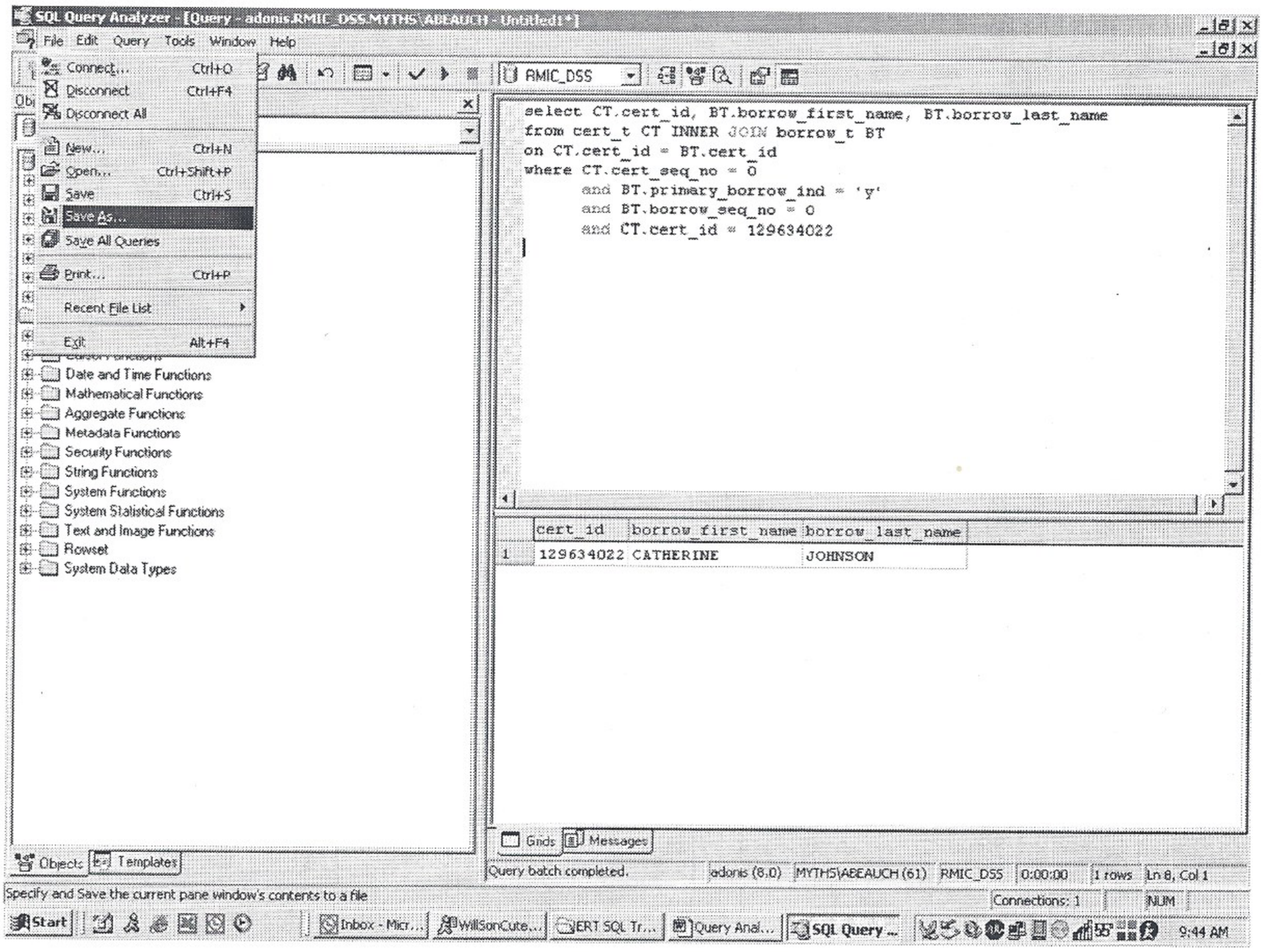
```
select CT.cert_id, BT.borrow_first_name, BT.borrow_last_name
from cert_t CT INNER JOIN borrow_t BT
on CT.cert_id = BT.cert_id
where CT.cert_seq_no = 0
      and BT.primary_borrow_ind = 'y'
      and BT.borrow_seq_no = 0
      and CT.cert_id = 129634022
```


SQL Query Analyzer - [Query - adonis.RMIC_DSS.MYTHS\ABEAUCH - Untitled1*]
File Edit Query Tools Window Help
RMIC_DSS
Object Browser
adonis(MYTHS\ABEAUCH)
adonis
master
msdb
RMIC_DQ
RMIC_DSS
RMIC_DSS_Work
RMIC_Pricing
tempdb
Common Objects
Configuration Functions
Cursor Functions
Date and Time Functions
Mathematical Functions
Aggregate Functions
Metadata Functions
Security Functions
String Functions
System Functions
System Statistical Functions
Text and Image Functions
Rowset
System Data Types
Objects
Templates
SELECT
CT.CERT_ID,
BT.BORROW_FIRST_NAME,
BT.BORROW_LAST_NAME
FROM
CERT_T CT INNER JOIN BORROW_T BT
ON CT.CERT_ID = BT.CERT_ID
WHERE CT.CERT_SEQ_NO = 0
AND BT.PRIMARY_BORROW_IND = 'Y'
AND BT.BORROW_SEQ_NO = 0
AND CT.CERT_ID = 129634022
CERT_ID BORROW_FIRST_NAME BORROW_LAST_NAME
1 129634022 CATHERINE JOHNSON
Grids
Messages
Query batch completed.
adonis (8.0)
MYTHS\ABEAUCH (62)
RMIC_DSS
0:00:00
1 rows
Ln 15, Col 1
Connections: 1
NUM

Saving your SQL Statements

To save a sql statement for use later, users need to do the following:

- Click once in the portion of the screen containing the sql statement
- Click on File Save As



- Give the sql statement a descriptive name and pick a location on your hard drive where you can easily locate it.