TAI Query Reporter V 1.1.0.6

New functionality…

HAVING Clause

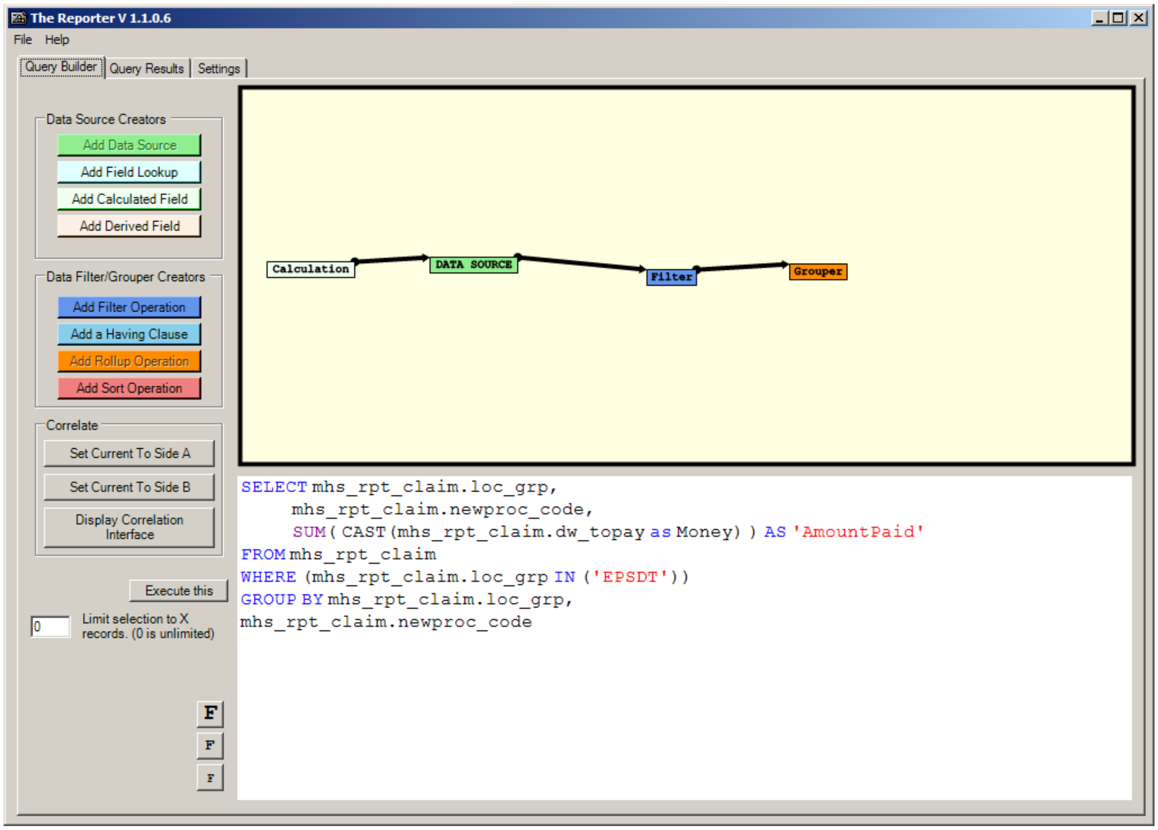
Reason:

There are often times when you have crafted a query that includes some aggregate data like SUM() and COUNT() and AVG() that are in actuality applied to whole groups of records as those records are grouped by the Grouper functionality. Meaning the SUM() or COUNT() or AVG() function is applied to more than a single record as the query gets executed in the database.

In these cases it is not possible in standard SQL syntax to place a condition on these aggregations within the where clause of the sql statement. What is necessary is to create a HAVING clause in SQL syntax and now the TAI Query Reporter can do just that.

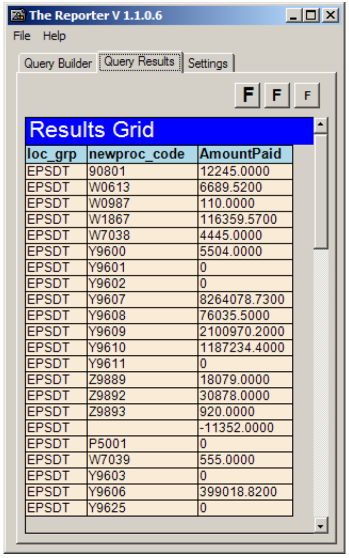
Example Usage….

Below is a sample query that has been built to sum up claims paid for in a certain timeframe, by service code. In some cases the sum of the monies paid out yields 0 dollars. This means that a claim may have been paid but then retracted. Since the financial impact of this activity is 0 it is often desirable to remove this noise from the result set. (If you are of a financial mind that is)



```
SELECT mhs_rpt_claim.loc_grp,
     mhs_rpt_claim.newproc_code,
     SUM( CAST(mhs_rpt_claim.dw_topay as Money) ) AS 'AmountPaid'
FROM mhs_rpt_claim
WHERE (mhs_rpt_claim.loc_grp IN ('EPSDT'))
GROUP BY mhs_rpt_claim.loc_grp,
mhs_rpt_claim.newproc_code
```

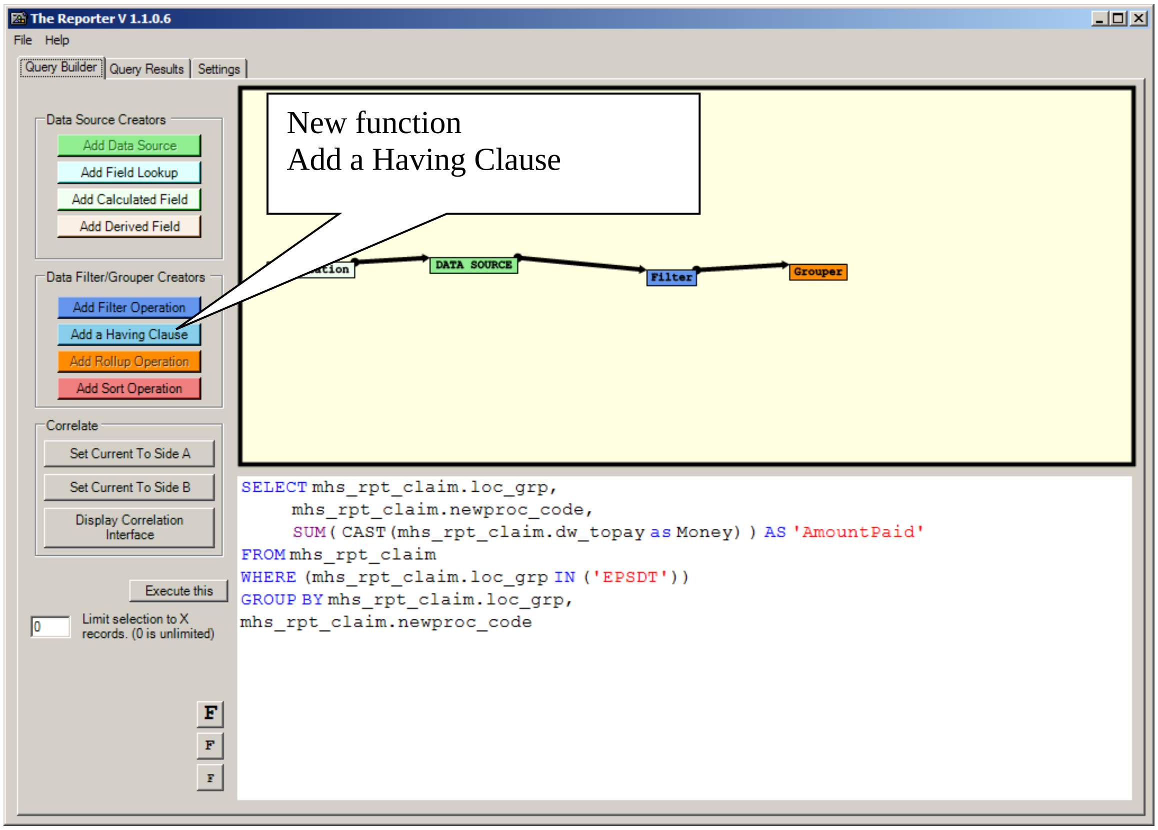
Here we are simply summing the money paid for any claim that has been classified as ‘EPSDT’ and grouping by the particular service code. The results of this query yield some values that are 0 dollars, as shown below



Results Grid

loc_grp	newproc_code	AmountPaid
EPSDT	90801	12245.0000
EPSDT	W0613	6689.5200
EPSDT	W0987	110.0000
EPSDT	W1867	116359.5700
EPSDT	W7038	4445.0000
EPSDT	Y9600	5504.0000
EPSDT	Y9601	0
EPSDT	Y9602	0
EPSDT	Y9607	8264078.7300
EPSDT	Y9608	76035.5000
EPSDT	Y9609	2100970.2000
EPSDT	Y9610	1187234.4000
EPSDT	Y9611	0
EPSDT	Z9889	18079.0000
EPSDT	Z9892	30878.0000
EPSDT	Z9893	920.0000
EPSDT		-11352.0000
EPSDT	P5001	0
EPSDT	W7039	555.0000
EPSDT	Y9603	0
EPSDT	Y9606	399018.8200
EPSDT	Y9625	0

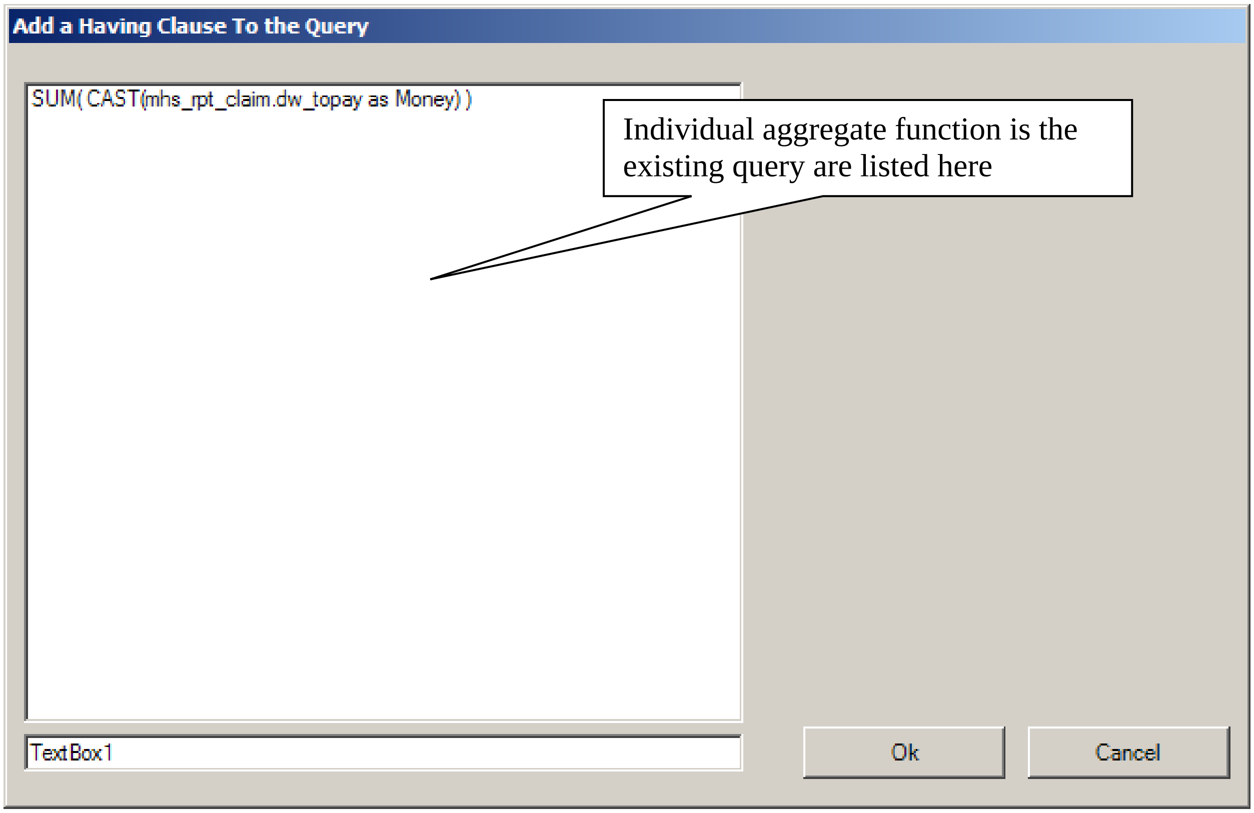
Back on the Main query screen a new button is available to Add a Having clause…



New function
Add a Having Clause

```
SELECT mhs_rpt_claim.loc_grp,
     mhs_rpt_claim.newproc_code,
     SUM( CAST (mhs_rpt_claim.dw_topay as Money) ) AS 'AmountPaid'
FROM mhs_rpt_claim
WHERE (mhs_rpt_claim.loc_grp IN ('EPSDT'))
GROUP BY mhs_rpt_claim.loc_grp,
mhs_rpt_claim.newproc_code
```

Selecting this new button will present you with the following dialog



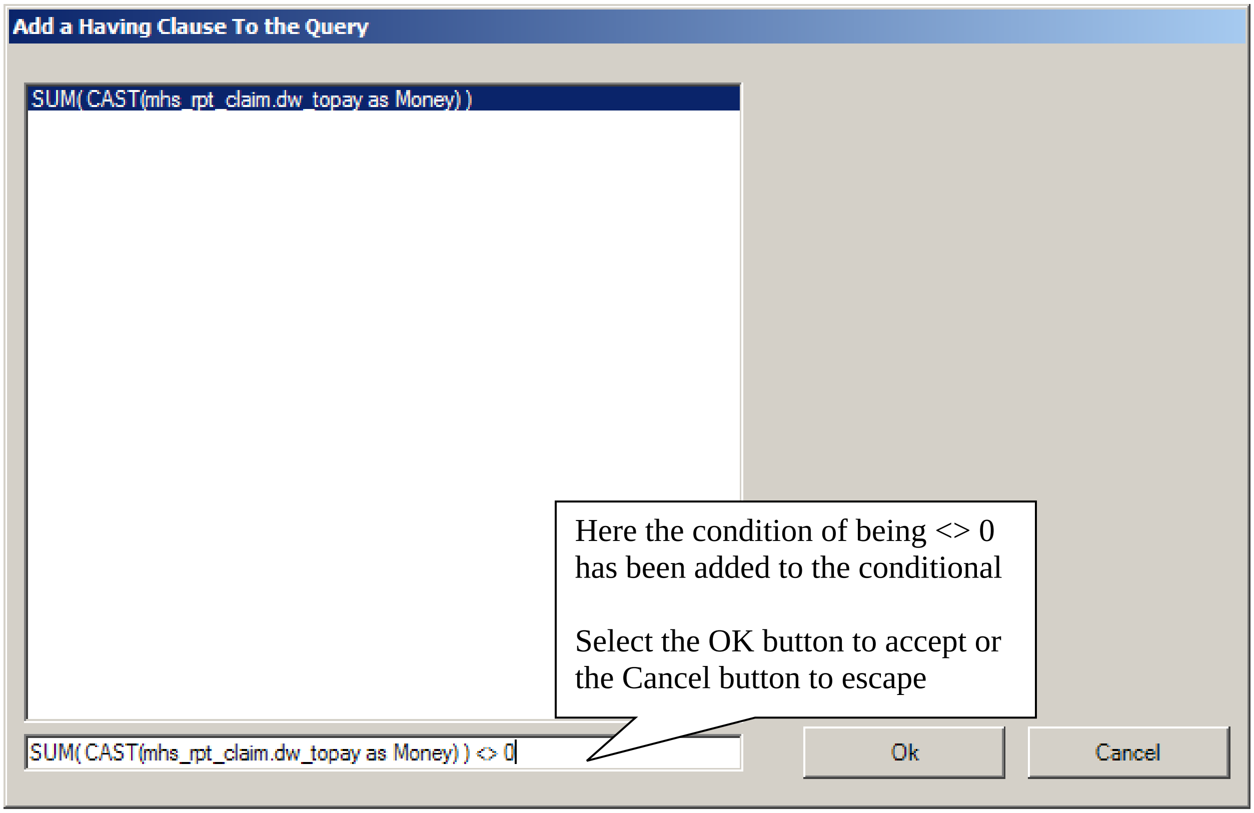
SUM(CAST(mhs_rpt_claim.dw_topay as Money))

TextBox1

Individual aggregate function is the existing query are listed here

In the above example the simple sum() aggregate is the only one in the existing query, so it’s the only one listed.

To craft a having clause simple select the desired aggregate and then type your desired conditional after the selected aggregate in the textbox below.



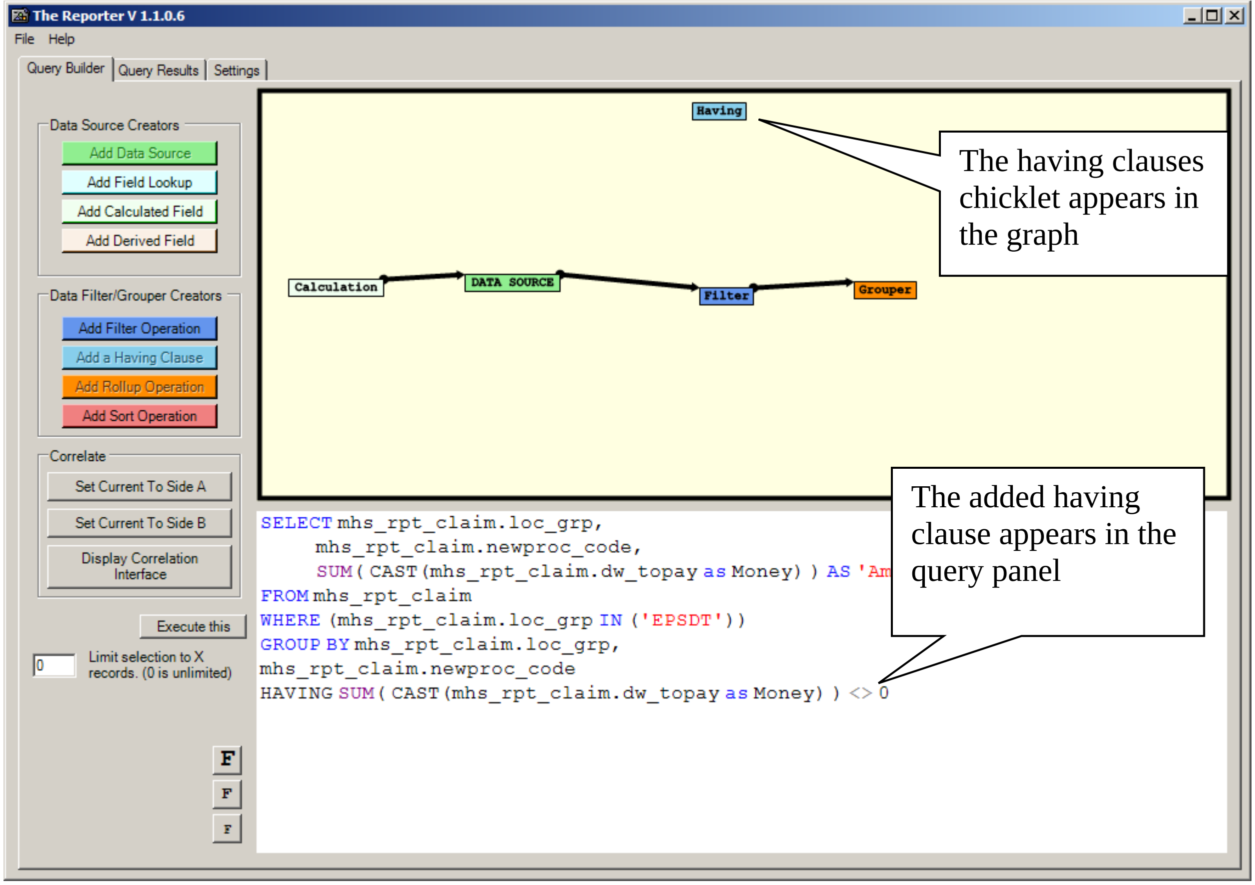
SUM(CAST(mhs_rpt_claim.dw_topay as Money))

SUM(CAST(mhs_rpt_claim.dw_topay as Money)) <> 0

Here the condition of being <> 0 has been added to the conditional

Select the OK button to accept or the Cancel button to escape

The result of the Having clause being added to the query is show below.



The having clauses chicklet appears in the graph

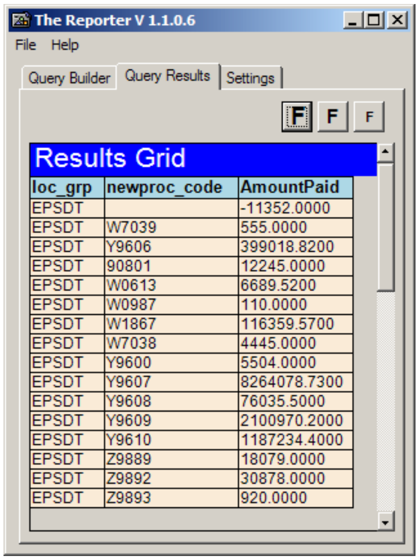
The added having clause appears in the query panel

```
SELECT mhs_rpt_claim.loc_grp,
     mhs_rpt_claim.newproc_code,
     SUM( CAST(mhs_rpt_claim.dw_topay as Money) ) AS 'Am
FROM mhs_rpt_claim
WHERE (mhs_rpt_claim.loc_grp IN ('EPSDT'))
GROUP BY mhs_rpt_claim.loc_grp,
mhs_rpt_claim.newproc_code
HAVING SUM( CAST(mhs_rpt_claim.dw_topay as Money) ) <> 0
```

Note…
Because having clauses are overriding the entire query they are not connected to the chain as the are conceptually attached to the entire query rather than a single point along the query chain.

You can only have a single having clause as of this time. If you want to alter the clause. Remove the existing clause and re-add a new one.

Results…



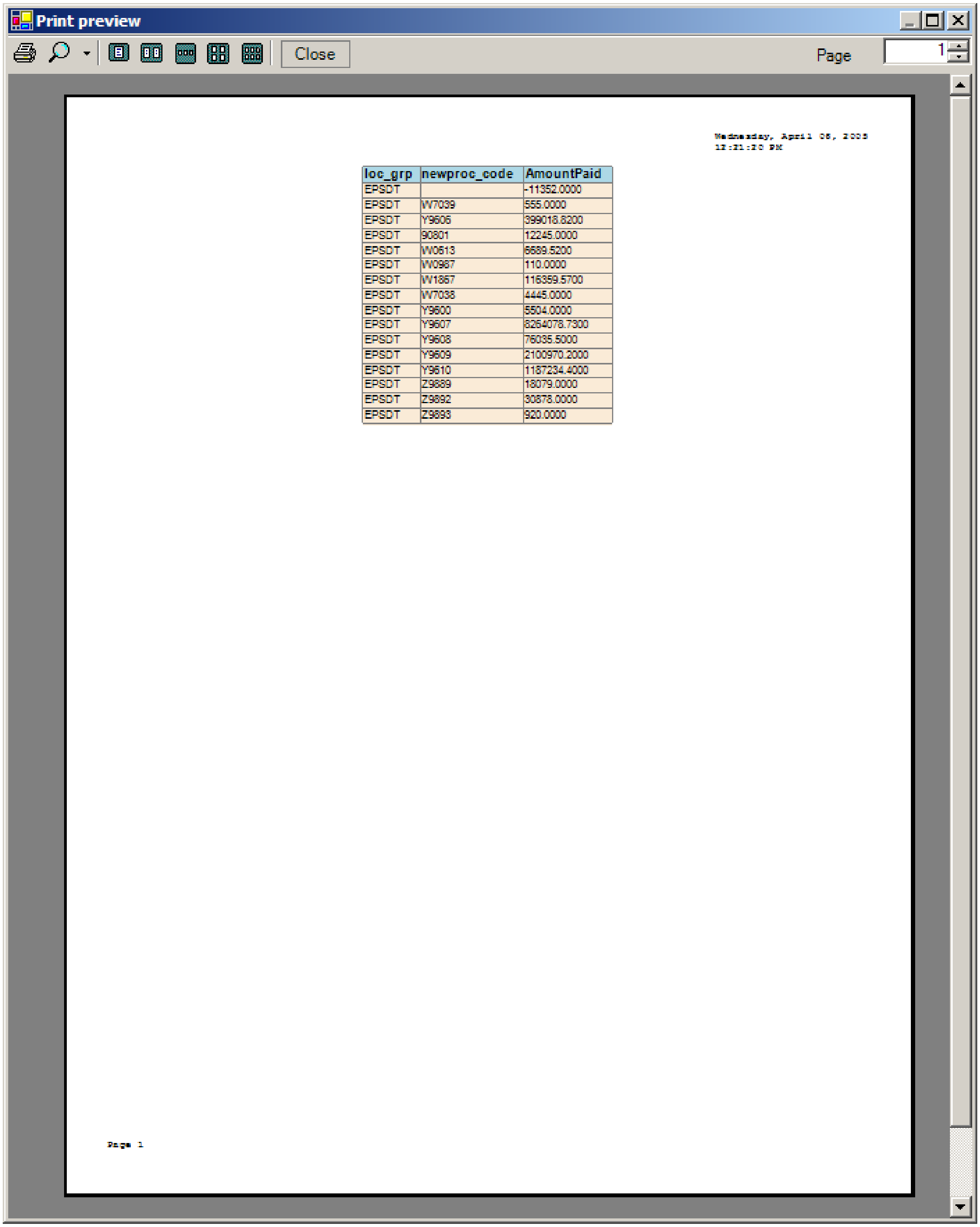
Results Grid

loc_grp	newproc_code	AmountPaid
EPSDT		-11352.0000
EPSDT	W7039	555.0000
EPSDT	Y9606	399018.8200
EPSDT	90801	12245.0000
EPSDT	W0613	6689.5200
EPSDT	W0987	110.0000
EPSDT	W1867	116359.5700
EPSDT	W7038	4445.0000
EPSDT	Y9600	5504.0000
EPSDT	Y9607	8264078.7300
EPSDT	Y9608	76035.5000
EPSDT	Y9609	2100970.2000
EPSDT	Y9610	1187234.4000
EPSDT	Z9889	18079.0000
EPSDT	Z9892	30878.0000
EPSDT	Z9893	920.0000

Note that the 0 dollar results are now missing from the end result.

Also in this new version is the ability to directly print the end result of a query by selecting Either PRINT or PRINT PREVIEW from the result grids own context menu.

Doing a print preview will show you the grids results paginated to fit onto multiple pages if the grid results are larger than one page. The date of the print and the page numbers are also printed.



Wednesday, April 06, 2005
12:21:20 PM

loc_grp	newproc_code	AmountPaid
EPSDT		-11352.0000
EPSDT	W7039	555.0000
EPSDT	Y9606	399018.8200
EPSDT	90801	12245.0000
EPSDT	W0613	6689.5200
EPSDT	W0967	110.0000
EPSDT	W1867	116359.5700
EPSDT	W7038	4445.0000
EPSDT	Y9600	5504.0000
EPSDT	Y9607	8264076.7300
EPSDT	Y9608	76035.5000
EPSDT	Y9609	2100970.2000
EPSDT	Y9610	1187234.4000
EPSDT	Z9889	18079.0000
EPSDT	Z9892	30878.0000
EPSDT	Z9893	920.0000

Page 1