

THE SQL QUERY WINDOW

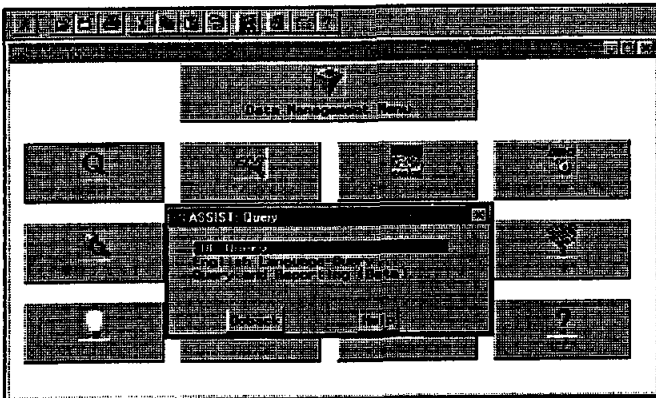
Prepared by
destiny
CORPORATION

Introduction

The SQL Query Window allows you to perform queries based on the standardized SQL language (Structured Query Language) without having any prior experience with SQL.

To enter the SQL Query Window from SAS/ASSIST, make the following selections:

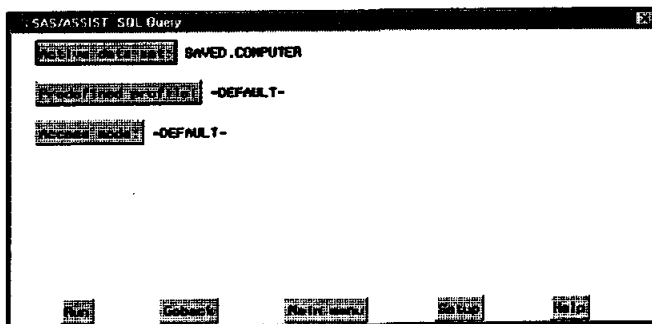
1. Select **Data Management** from the Primary Menu.
2. Select **Query** from the Data Management Menu.
3. Choose the **SQL Query** option.



Selecting Data

Selecting a Table

Select **Active data set** to choose from a list of available data sets. Once a data set has been selected, the **RUN** button will continue the query.



Selecting Columns

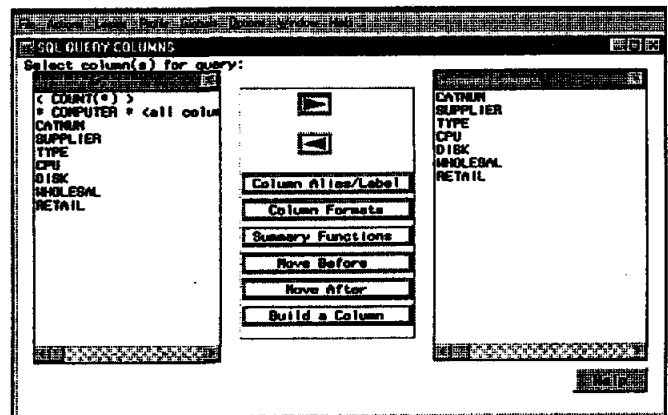
The next window to be displayed is the **SQL QUERY COLUMNS** window. Here you can select which columns you would like to be included in your query.

To select all columns, simply choose the **<all columns>** option.

By choosing the  (right arrow), all column names will be copied over to the **Selected Columns** list.

To select columns individually:

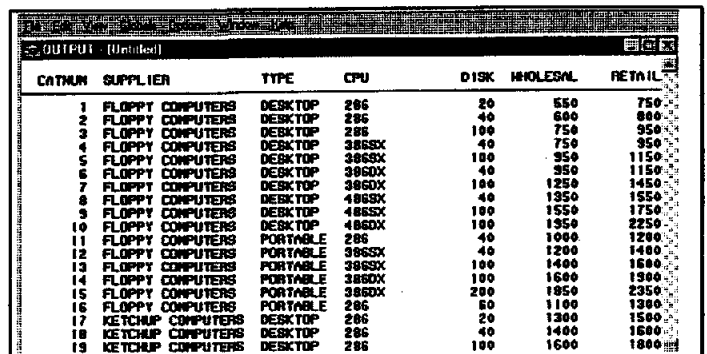
1. Select the desired column names from the **Available Columns** list and copy them to the **Selected Columns** list by choosing the  (right arrow).
2. To remove columns from the **Selected Columns** list, select the desired column names from the **Selected Columns** list and choose the  (left arrow) button. The column names will then be deleted from the current selections.



The **Actions** pulldown, **Run Query** option can be used to run the query at any time.

Once a query is run, the output is displayed immediately.

The **File** pulldown, **End** option will allow you to exit from the **OUTPUT** window and go back to the **COLUMNS** screen.



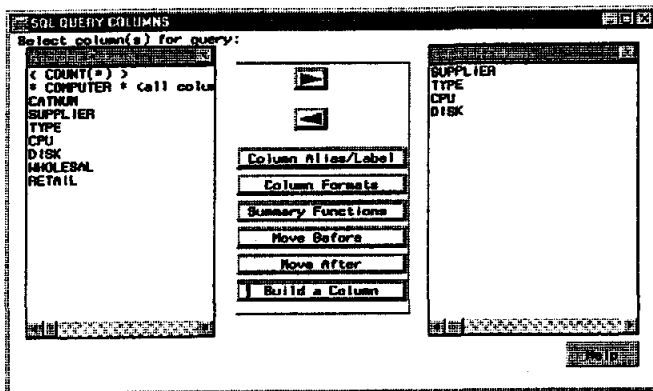
	CATNUM	SUPPLIER	TYPE	CPU	DISK	WHOLESALE	RETAIL
1	FLOPPY	COMPUTERS	DESKTOP	286	80	550	750
2	FLOPPY	COMPUTERS	DESKTOP	286	40	600	800
3	FLOPPY	COMPUTERS	DESKTOP	286	100	750	950
4	FLOPPY	COMPUTERS	DESKTOP	386SX	40	750	950
5	FLOPPY	COMPUTERS	DESKTOP	386SX	100	950	1150
6	FLOPPY	COMPUTERS	DESKTOP	386DX	40	950	1150
7	FLOPPY	COMPUTERS	DESKTOP	386DX	100	1250	1450
8	FLOPPY	COMPUTERS	DESKTOP	486SX	40	1350	1550
9	FLOPPY	COMPUTERS	DESKTOP	486SX	100	1550	1750
10	FLOPPY	COMPUTERS	DESKTOP	486DX	100	1950	2250
11	FLOPPY	COMPUTERS	PORTABLE	286	40	1000	1200
12	FLOPPY	COMPUTERS	PORTABLE	386SX	40	1200	1400
13	FLOPPY	COMPUTERS	PORTABLE	386SX	100	1400	1600
14	FLOPPY	COMPUTERS	PORTABLE	386DX	100	1600	1800
15	FLOPPY	COMPUTERS	PORTABLE	386DX	200	1850	2350
16	FLOPPY	COMPUTERS	PORTABLE	286	80	1100	1300
17	KETCHUP	COMPUTERS	DESKTOP	286	20	1300	1500
18	KETCHUP	COMPUTERS	DESKTOP	286	40	1400	1600
19	KETCHUP	COMPUTERS	DESKTOP	286	100	1600	1800

Building a Column

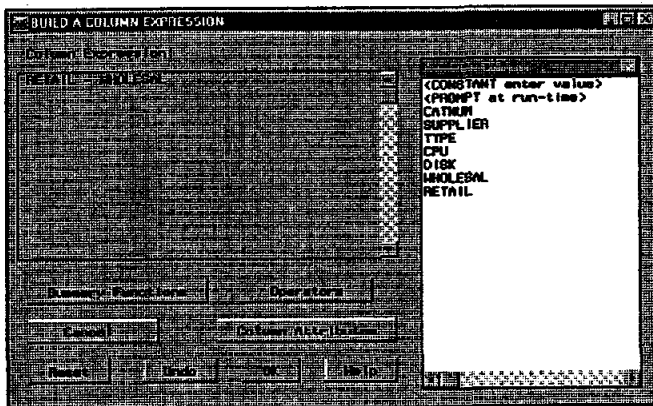
Suppose we would like to query our COMPUTER database to produce a report showing SUPPLIER, CPU, DISK, and TYPE columns, but we would also like to know a profit figure for each of these rows, determined by RETAIL - WHOLESALE.

We can add such a column to our query by building a column expression.

1. Select columns SUPPLIER, TYPE, CPU, DISK from Available Columns.



2. Select the Build a Column button. This takes us to the Build a Column Expression window.



To create the RETAIL-WHOLESALE column in the Build a Column Expression window:

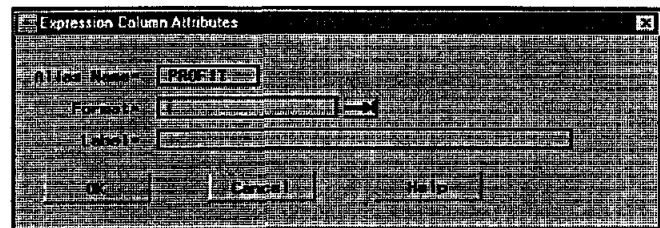
1. Select RETAIL from the Available Columns list.
2. Select the subtraction operator (-) from the operator pop-up menu.
3. Select WHOLESALE from the Available Columns list. Since this is the last entry we need for our column expression, you can ignore the operator pop-up.
4. Select OK. This will take us to the Express Column Attributes window.

Note: A column can be built from any valid SAS expression, including data step functions.

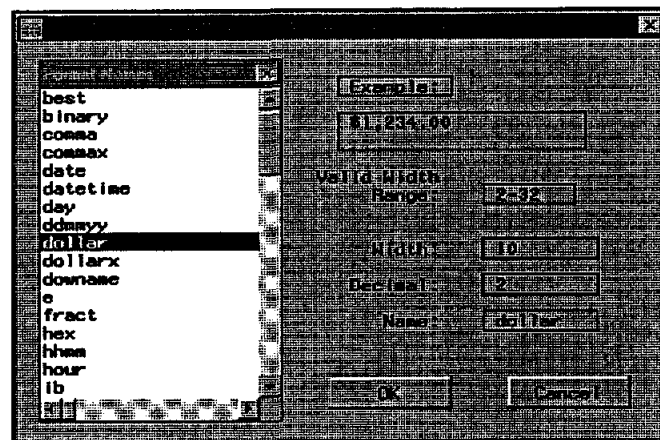
Expression Column Attributes

We can now enter the attributes of our column into the Express Column Attributes window. (The Column Attributes button on the previous screen would also take us into this window.)

1. Enter an Alias Name for the column. Here we name our column PROFIT.
2. Select the → next to the Format field to enter the Format Names window. Or, if you know the name of the format you want, you can simply enter it here.



3. Select a format from the list of Format Names. Here we select the dollar format. We also specify a width and decimal range.
4. Select OK to return to the Expression Column Attributes window.



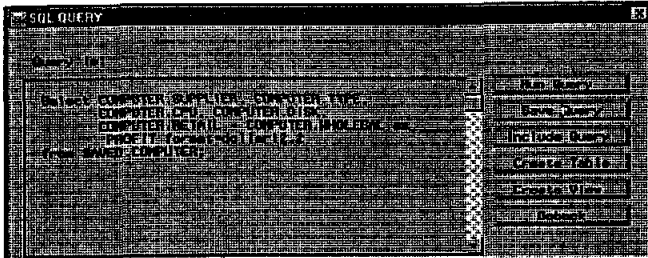
5. Enter a Label in the Expression Column Attributes window to label the PROFIT column, if desired.
6. Select OK to return to the Build a column Expression window.
7. Select OK again to return to the Columns window.

Displaying the query

1. Choose the Actions pulldown menu, Show option to view the query expression that has been built.
2. The Run Query button will execute the query from here. (Recall that the Actions pulldown, Run option will also execute the query.)

Other options available here include;

- Save Query** to save the query expression in an external file, or to a .QUERY or .SOURCE catalog entry.
- Create Table** to save the data selected by the query.
- Create View** to save the query expression as a view.
- Goback** to return to the Columns window.



The OUTPUT window shows the results of having added a PROFIT column.

Recall that we can return to the Columns window by choosing the File pulldown, End option.

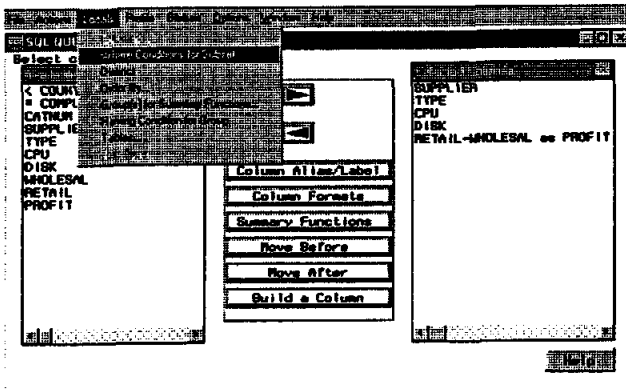
SUPPLIER	TYPE	CPU	DISK	PROFIT
FLOPPY COMPUTERS	DESKTOP	286	20	\$200.00
FLOPPY COMPUTERS	DESKTOP	286	40	\$200.00
FLOPPY COMPUTERS	DESKTOP	286	100	\$200.00
FLOPPY COMPUTERS	DESKTOP	386SX	40	\$200.00
FLOPPY COMPUTERS	DESKTOP	386SX	100	\$200.00
FLOPPY COMPUTERS	DESKTOP	386SX	40	\$200.00
FLOPPY COMPUTERS	DESKTOP	386SX	100	\$200.00
FLOPPY COMPUTERS	DESKTOP	486SX	40	\$200.00
FLOPPY COMPUTERS	DESKTOP	486SX	100	\$200.00
FLOPPY COMPUTERS	DESKTOP	486SX	100	\$200.00
FLOPPY COMPUTERS	PORTABLE	286	40	\$200.00
FLOPPY COMPUTERS	PORTABLE	386SX	40	\$200.00
FLOPPY COMPUTERS	PORTABLE	386SX	100	\$200.00
FLOPPY COMPUTERS	PORTABLE	386SX	100	\$200.00
FLOPPY COMPUTERS	PORTABLE	386SX	200	\$200.00
FLOPPY COMPUTERS	PORTABLE	286	60	\$200.00
KETCHUP COMPUTERS	DESKTOP	286	20	\$200.00
KETCHUP COMPUTERS	DESKTOP	286	40	\$200.00
KETCHUP COMPUTERS	DESKTOP	286	100	\$200.00
KETCHUP COMPUTERS	DESKTOP	386SX	20	\$200.00
KETCHUP COMPUTERS	DESKTOP	386SX	40	\$200.00

Selecting Rows

Thus far we have seen how to select and create columns for our query. We can subset our query by selecting rows.

Where Conditions for Subset - Eliminating Rows

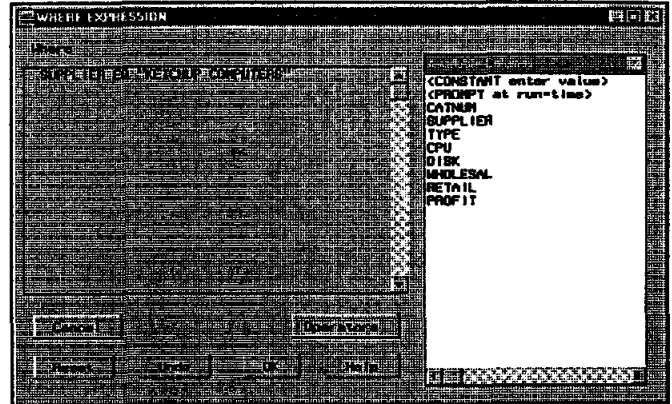
The Locals pull down, Where Conditions for Subset option takes us to the WHERE EXPRESSION window. This window allows us to set up a where expression to subset our data.



We wish to select only rows where SUPPLIER is equal to KETCHUP COMPUTERS.

To set up this expression:

1. Select SUPPLIER from the Available Columns box.
2. Select EQ from the Operator pop-up that appears on the screen.
3. Select <CONSTANT enter value> to get to the pop-up box to enter constants.
4. Enter KETCHUP COMPUTERS and then select the OK button. (You do not need to enter the quotes.)
5. When the WHERE EXPRESSION is completed, select the OK button to return to the Columns window.



Upon running the query, our output shows the subsetted data.

SUPPLIER	TYPE	CPU	DISK	PROFIT
KETCHUP COMPUTERS	DESKTOP	286	20	\$200.00
KETCHUP COMPUTERS	DESKTOP	286	40	\$200.00
KETCHUP COMPUTERS	DESKTOP	286	100	\$200.00
KETCHUP COMPUTERS	DESKTOP	386SX	20	\$200.00
KETCHUP COMPUTERS	DESKTOP	386SX	40	\$200.00
KETCHUP COMPUTERS	DESKTOP	386SX	100	\$200.00
KETCHUP COMPUTERS	DESKTOP	386SX	20	\$200.00
KETCHUP COMPUTERS	DESKTOP	386SX	40	\$200.00
KETCHUP COMPUTERS	DESKTOP	386SX	100	\$200.00
KETCHUP COMPUTERS	DESKTOP	486SX	40	\$200.00
KETCHUP COMPUTERS	DESKTOP	486SX	100	\$200.00
KETCHUP COMPUTERS	DESKTOP	486SX	200	\$200.00
KETCHUP COMPUTERS	PORTABLE	286	20	\$150.00
KETCHUP COMPUTERS	PORTABLE	386SX	20	\$200.00
KETCHUP COMPUTERS	PORTABLE	386SX	40	\$200.00
KETCHUP COMPUTERS	PORTABLE	386SX	120	\$200.00
KETCHUP COMPUTERS	PORTABLE	386SX	60	\$400.00
KETCHUP COMPUTERS	PORTABLE	386SX	200	\$200.00
KETCHUP COMPUTERS	PORTABLE	486SX	200	\$400.00

Other Enhancements

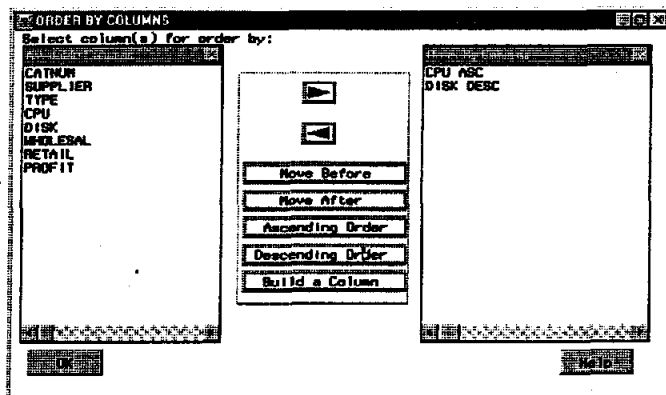
ORDER BY - ORDERING THE RESULTS

The Locals pulldown, Order By option takes us to the ORDER BY COLUMNS window. This window allows us to order our query result by selected columns.

Note that all columns are available to order by regardless of whether we have selected them as columns for our query result. Also, a Column Expression may be built with the Build A Column button, and then used to order results.

To select columns to order by:

1. Copy the column names from the Available Columns list to the Order By Columns list. Ascending Order is the default sequence.
2. To specify Descending Order, select the column to be sorted descending from the Order By Columns list, and then choose the Descending Order button.



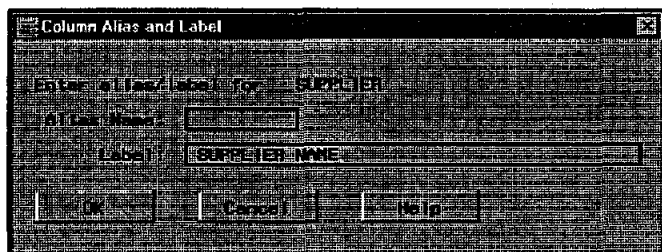
Formatting and Labelling Columns

Column formats and labels can be added to enhance our query.

To create a column label for SUPPLIER:

1. Select SUPPLIER from the Selected Columns list in the Columns window.
2. Select the Column Alias/Label button.
3. In the Column Alias and Label window, enter a label in the Label area.
4. Select OK.

Here we label our SUPPLIER column with SUPPLIER NAME.



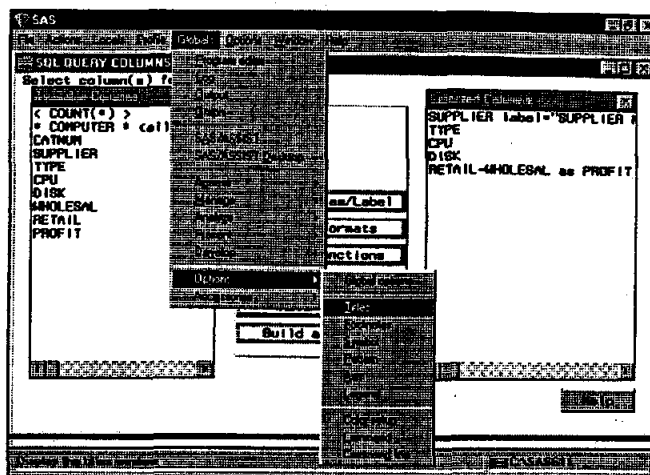
From the Columns window, the Column Formats button can be used to format a column in a similar fashion.

Titles and Footnotes

Report titles and footnotes can be modified through the Globals pulldown, Options option.

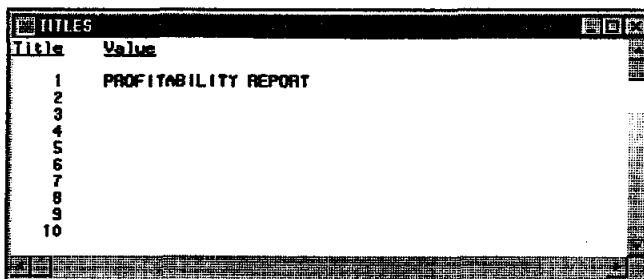
To add a title to our report:

1. Select the Globals pulldown.
2. Select Options.
3. Select Titles.

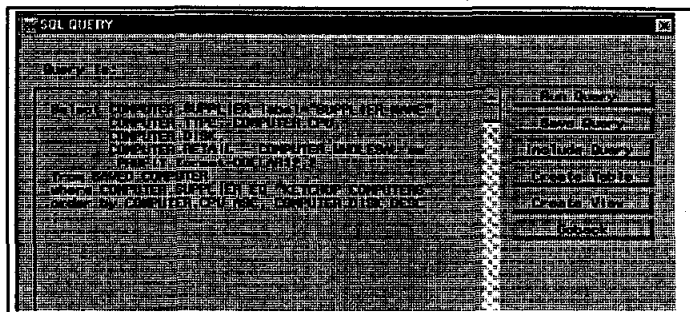


4. Enter a title in the Title window.
5. To exit the Title window, use the File pulldown, End option. (Depending on what system you are running SAS/ASSIST, you may have other options to end this window.)

Here, we enter a title of PROFITABILITY REPORT.



Viewing the query again from the Action pull down, Show query option, we can see the enhancements made to our query.

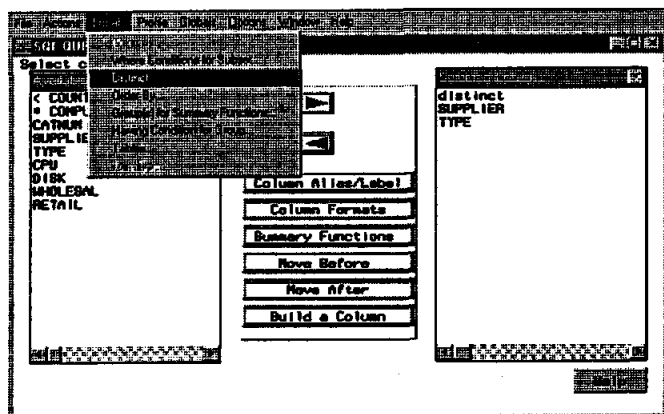


Running the query shows our output now subsetting for KETCHUP COMPUTERS, sorted by CPU type, and within each CPU type, sorted by descending values of DISK.

SUPPLIER NAME	TYPE	CPU	DISK	PROFIT
KETCHUP COMPUTERS	DESKTOP	286	100	\$200.00
KETCHUP COMPUTERS	DESKTOP	286	40	\$200.00
KETCHUP COMPUTERS	PORTABLE	286	20	\$150.00
KETCHUP COMPUTERS	DESKTOP	286	20	\$200.00
KETCHUP COMPUTERS	PORTABLE	386DX	200	\$400.00
KETCHUP COMPUTERS	PORTABLE	386DX	120	\$250.00
KETCHUP COMPUTERS	DESKTOP	386DX	100	\$200.00
KETCHUP COMPUTERS	PORTABLE	386DX	50	\$400.00
KETCHUP COMPUTERS	DESKTOP	386DX	40	\$200.00
KETCHUP COMPUTERS	DESKTOP	386DX	20	\$200.00
KETCHUP COMPUTERS	PORTABLE	386SX	120	\$200.00
KETCHUP COMPUTERS	DESKTOP	386SX	100	\$200.00
KETCHUP COMPUTERS	PORTABLE	386SX	40	\$200.00
KETCHUP COMPUTERS	PORTABLE	386SX	20	\$200.00
KETCHUP COMPUTERS	DESKTOP	386SX	20	\$200.00
KETCHUP COMPUTERS	DESKTOP	486DX	200	\$200.00
KETCHUP COMPUTERS	DESKTOP	486DX	120	\$200.00
KETCHUP COMPUTERS	DESKTOP	486DX	50	\$300.00
KETCHUP COMPUTERS	PORTABLE	486SX	200	\$400.00

DISTINCT - SELECTING DISTINCT COLUMN VALUES

Another interesting query option is the capability of pulling out all distinct values of a column or combination of columns. The Locals pull down, Distinct option allows us to specify that we want only distinct column values for all columns listed in the Selected Columns list.



Here we get all distinct combinations of SUPPLIER and TYPE in our data.

SUPPLIER	TYPE
FLOPPY COMPUTERS	DESKTOP
FLOPPY COMPUTERS	PORTABLE
KETCHUP COMPUTERS	DESKTOP
KETCHUP COMPUTERS	PORTABLE

THE TABLES WINDOW

Choosing a Table

From the Locals pulldown, Tables option we can select tables for our query. This step must be done before any other steps in formulating the query are allowed.

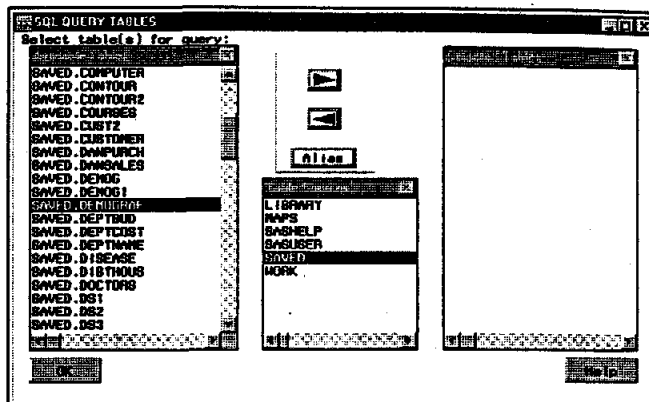
Note: If the SQL Query window was entered from SAS/ASSIST, then you already selected a table from the SAS/ASSIST: SQL Query window and therefore can bypass the TABLES window. (You will actually have been taken directly to the COLUMNS window anyway.) You can, however, come back to this window at any time to change tables.

To select a table to query:

1. Select the table from the Available Tables list.

2. Move the table to the Selected Tables list with the (right arrow) button.

NOTE: More than one table can be selected.



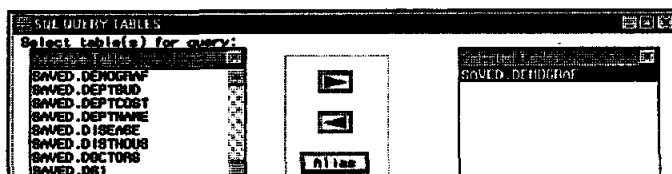
ALIASING A TABLE

It is sometimes useful to give a table a name other than the full two-level *libref.dataset* name shown. This is especially true when multiple tables with some matching variable names are being queried. By simplifying the table name, the query is easier to structure. The new name we give the table is called an alias.

To give a table an alias:

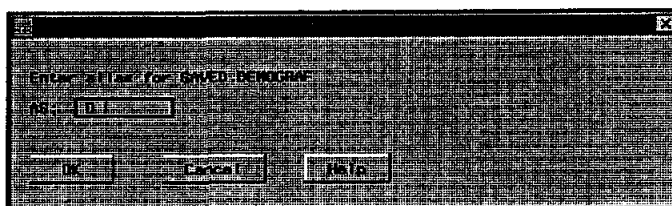
1. Select the table from the Selected Tables list.

2. Choose the Alias button. We are then prompted to enter an alias for the selected table.



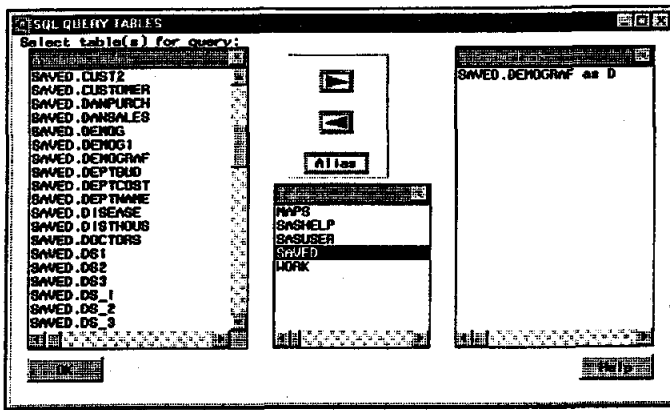
3. Enter an alias. Here we simply enter the letter D as the alias for SAVED.DEMOGRAF.

4. Select OK.



Note that once we are returned to the TABLES window, our alias is listed in the Selected Tables list alongside the original table name.

5. Select OK from the TABLES window to go directly to the COLUMNS window. You could also get to the COLUMNS window from the Locals pulldown, Columns option.



THE COLUMNS WINDOW

Applying Summary Functions

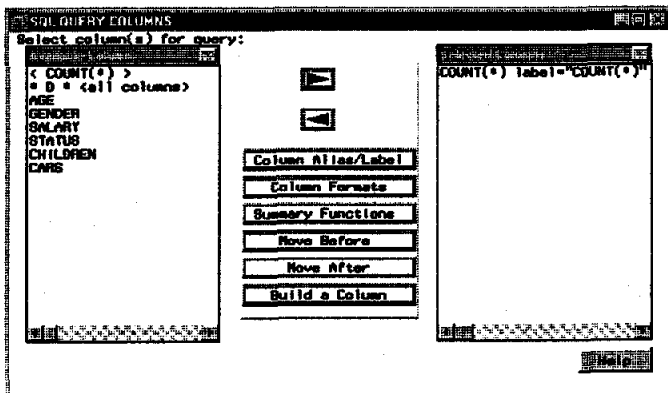
In section QW1 we saw how to use the COLUMNS window to select columns and build column expressions. Here we look at how to apply Summary Functions to a column.

The COUNT(*) Column

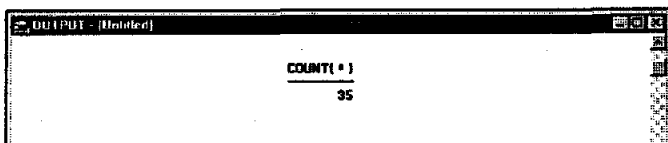
The COUNT (*) column will use the COUNT function to return a count of rows that our query has produced. In its simplest form, we get an output of one number showing a frequency count of rows in our query. The COUNT function is analogous to the FREQ or N statistic.

To select the COUNT(*) column:

Copy the COUNT(*) column from Available Columns to Selected Columns.

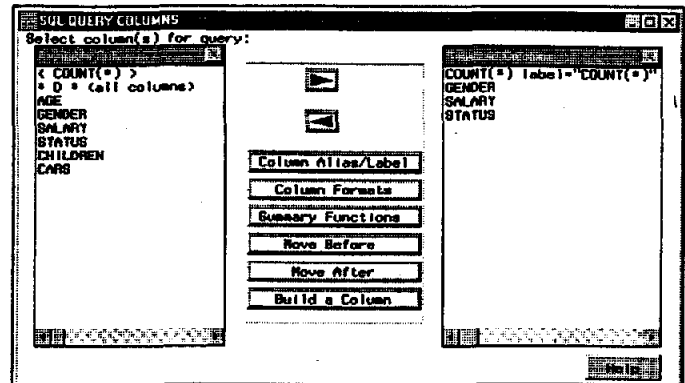


Upon running the query, our output shows the result of all rows counted.



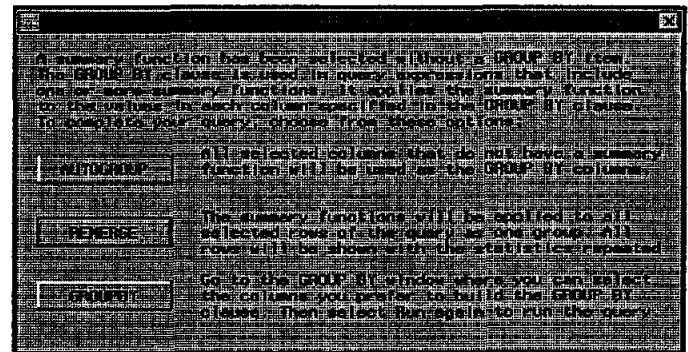
What happens when we add other columns to our query, still including the COUNT(*) function?

Here we add the GENDER, SALARY AND STATUS columns to our query.



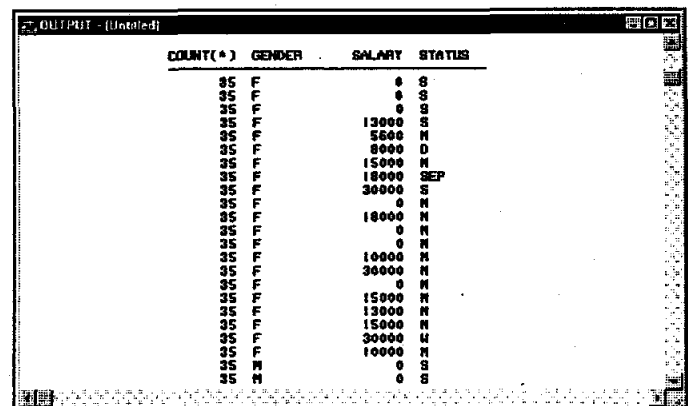
This time when we run the query, a new window pops up asking what we want to do about the summary function we have requested. Since our COUNT (*) function returns only one row as a result, but the remainder of our query will produce multiple rows, SAS doesn't know what to do with the COUNT(*) result.

Here we will select to **REMERGE** the statistic onto all rows of the query output.



NOTE: If the GROUP BY button is selected here, you are taken to the GROUP BY window, which is discussed in section QW5.4. If the AUTOGROUP button is selected, the output will be grouped by all columns that do not have a summary function specified.

Our output shows that the COUNT(*) result has been "remerged" onto every row of our query.



Other Summary Functions

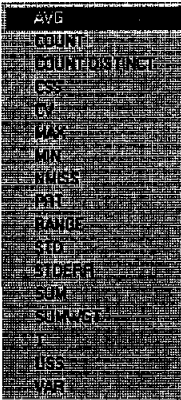
Suppose we would like to know the average salary for all rows in our DEMOGRAF table. We can apply a summary function to the SALARY column to determine this number.

To get an average salary figure:

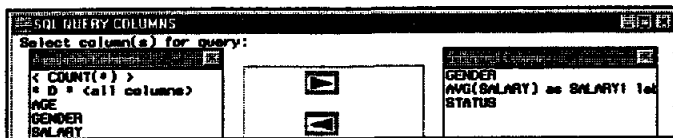
1. Select the SALARY column from the Selected Columns list.
2. Select the Summary Functions button. A pop-up list appears allowing us to select a statistic.

Summary Functions

3. Select the AVG statistic. (This is the same as the MEAN statistic.)



The Selected Columns now show that the AVG summary function will be applied to the SALARY column. We no longer are requesting the individual SALARY values, but instead an average of all salaries.



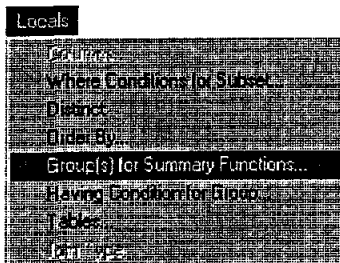
Upon running the query, we choose to "remerge" the statistic onto our result. The result shows the AVG(SALARY) column with the average salary for the table remerged onto every row.

GENDER	AVG(SALARY)	STATUS
F	11391.43	S
F	11391.43	S
F	11391.43	S
F	11391.43	S
F	11391.43	M
F	11391.43	M
F	11391.43	D
F	11391.43	D
F	11391.43	SEP
F	11391.43	SEP
F	11391.43	S
F	11391.43	S

THE GROUP BY WINDOW

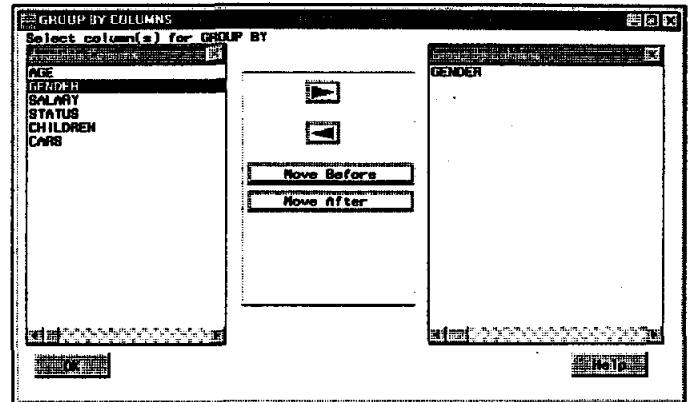
Summary functions are more commonly used to produce statistics for all values of a grouping variable. The Locals pulldown, Group(s) for Summary Functions option allows us to specify which variables to group by for statistical summarization.

What if we would like to produce an average salary for all values of GENDER, i.e. one for males and one for females? To do this we can select to group by GENDER. Any variables selected in the GROUP BY window are used to group summary function results.



To select to group by GENDER in the GROUP BY COLUMNS window:

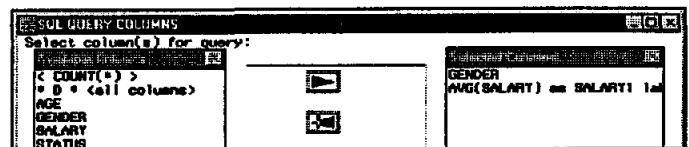
1. Select GENDER from the Available Columns list.
2. Move GENDER to the Selected Columns list with the [right arrow] button.
3. When done, select OK to return to the COLUMNS window.



Upon running the query, we now have two values of AVG(SALARY) remerged onto our output. The average salary for GENDER 'F' is remerged onto all rows for females. The average salary for GENDER 'M' is remerged onto all rows for males.

GENDER	AVG(SALARY)	STATUS
F	10980.95	S
F	10980.95	S
F	10980.95	S
F	10980.95	S
F	10980.95	M
F	10980.95	M
F	10980.95	D
F	10980.95	D
F	10980.95	SEP
F	10980.95	SEP
F	10980.95	S

If we eliminate the STATUS column from our query and simply specify that we want GENDER and AVG(SALARY) columns, we can produce a different style of report.



We now get only two rows of output, one for females and one for males. The summarization was allowed to take place without the need to remerge the results back onto other rows.

GENDER	AVG(SALARY)
F	10980.95
M	12007.14

THE HAVING EXPRESSION WINDOW

What if we want to produce a report for all rows having a salary greater than the average for their GENDER group.

With a HAVING expression, we can compare a column value to a summary function value. Remember, the summary function returns the requested statistic for each value of the GROUP BY VARIABLE.

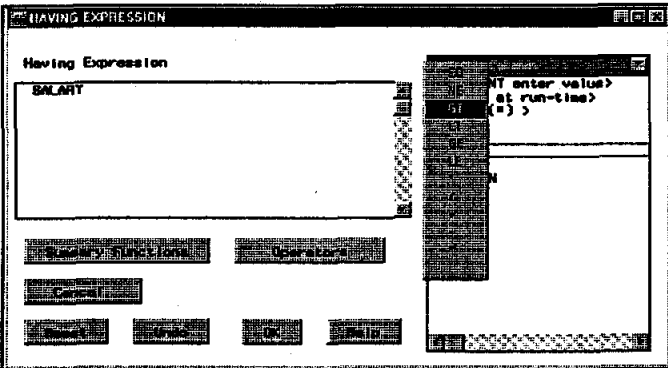
The Locals pulldown, Having Condition for Group option allows us to set up such a condition.

Note that for this query we have added the STATUS and SALARY columns to our previous query of GENDER and AVG(SALARY).

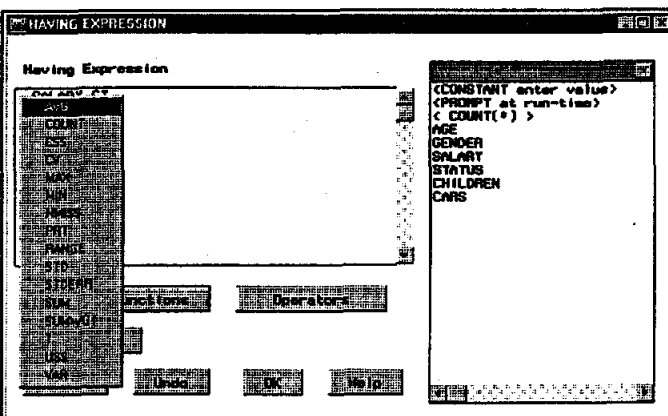
In the HAVING EXPRESSION window, we can piece together a HAVING clause. The having clause affects what rows will be chosen for output on our report.

Here we want to create a HAVING clause to select only rows for output that have a salary greater than average for their group. To put the clause together:

1. Select SALARY from the Available Columns list.
2. Select GT (for greater than) from the Operator pop-up.



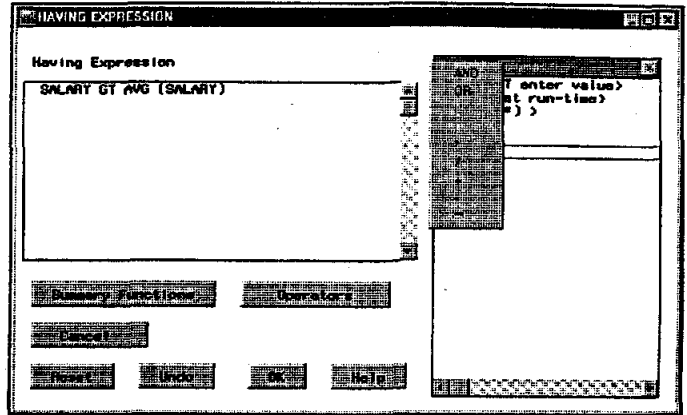
3. Select the Summary Functions button.
4. Select the AVG function from the pop-up.



5. Select SALARY from the Selected Columns list. Parentheses are automatically added, though we could have selected the Operators button to choose each parenthesis manually.
6. Another operator pop-up appears allowing us to continue adding to the HAVING clause. Since our HAVING clause is complete, we can ignore this and select OK.

The Reset button will clear the entire Having Expression.

The Undo button clears the Having Expression one token at a time from right to left.



Upon running the query, our output shows only rows that meet the HAVING condition. Only rows having a SALARY greater than AVG(SALARY) for their group have been output.

The HAVING clause:

- Considers the GROUP BY variable.
- Will create the summary statistics for comparison purposes even if the statistic is not one of the columns on the report. Therefore, if we eliminated the AVG(SALARY) column, our output would otherwise be identical.

GENDER	STATUS	SALARY	AVG(SALARY)
F	S	13000	10980.35
F	N	15000	10980.35
F	SEP	18000	10980.35
F	S	30000	10980.35
F	N	15000	10980.35
F	N	30000	10980.35
F	N	15000	10980.35
F	N	10000	10980.35
F	N	30000	10980.35
N	N	23000	12007.14
N	N	23000	12007.14
N	N	12300	12007.14
N	N	40000	12007.14

THE WHERE EXPRESSION WINDOW

Further Subsetting with the WHERE Clause

In order to contrast the WHERE clause and the HAVING clause, we will add a WHERE EXPRESSION to our previous query.

The WHERE expression subsets rows coming in to the query and cannot compare against a summary function value. The HAVING expression subset rows going out to the output, and

can calculate and compare against summary function values. Therefore, any rows that do not meet the WHERE condition are not considered in the calculation of the summary statistic on the SELECT or HAVING clauses.

We saw the <CONSTANT enter value> option earlier. Here we will consider the <LOOKUP distinct values> and <PROMPT at run-time> options to create the WHERE EXPRESSION.

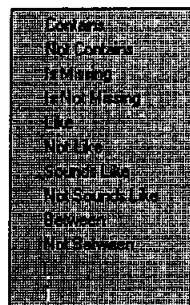
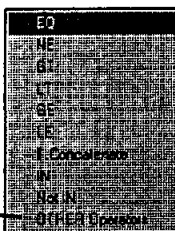
Lookup Distinct Values

The WHERE EXPRESSION window (as well as the HAVING EXPRESSION window) has the capability of looking up and listing for us all distinct values of a variable that we might want to subset by. We can then select from the list which value(s) we want included in our WHERE expression.

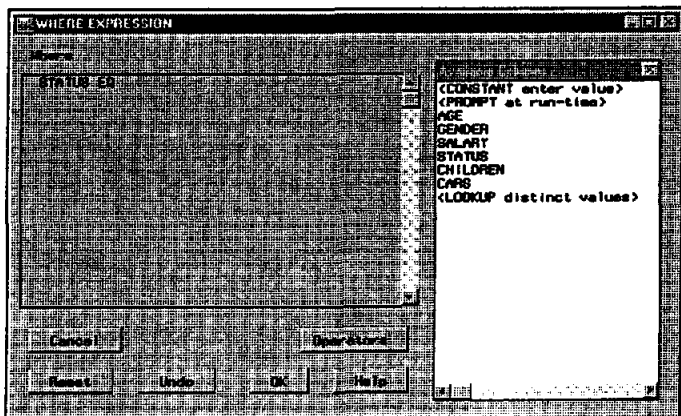
1. Begin the WHERE expression by selecting STATUS and the EQ operator. <LOOKUP distinct values> now becomes an option on the Available Columns list.

Note: You can choose from other valid WHERE operators by selecting

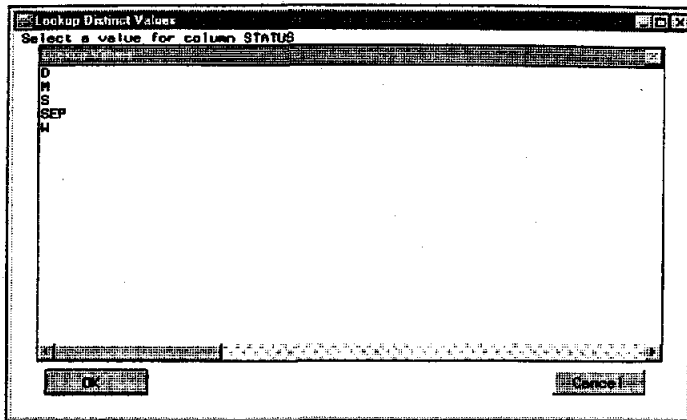
These choices are available



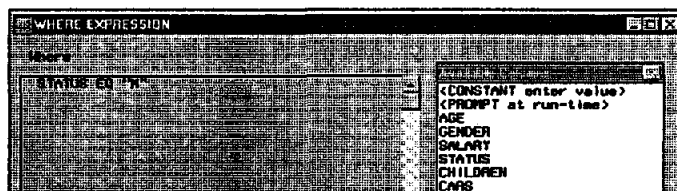
2. Select <LOOKUP distinct values>.



3. The Lookup Distinct Values window shows all values for column STATUS. Select M to include 'M'arried people only in our query.



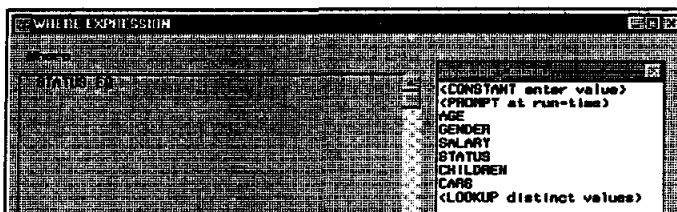
"M" has now been added to our WHERE expression.



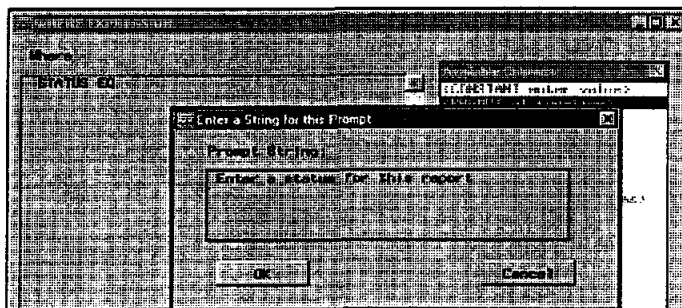
Prompt at Run Time

We can also choose to be prompted to enter a status value each time we run the query.

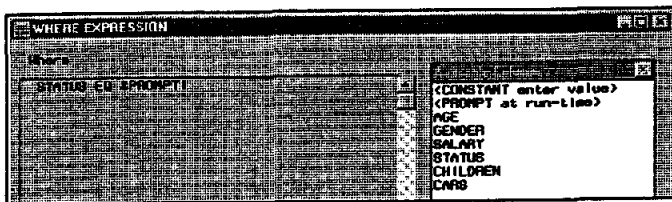
1. Again, begin the WHERE expression with the STATUS column, EQ operator.
2. Select <PROMPT at run-time>. You will be asked to Enter a String for this Prompt.



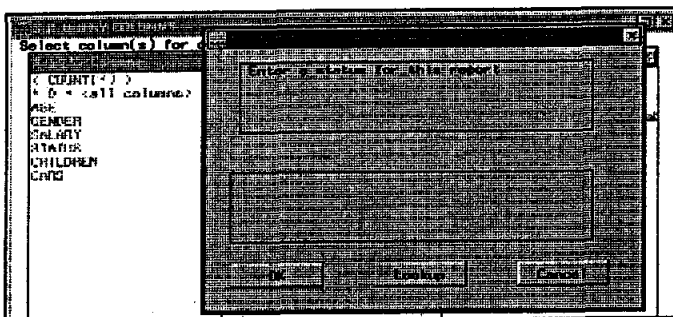
3. Here we enter a string that will appear when we run the query to prompt the user for a STATUS value. The string can be anything you like. The Prompt String here has been entered as 'Enter a status for this report'.
4. Select OK.



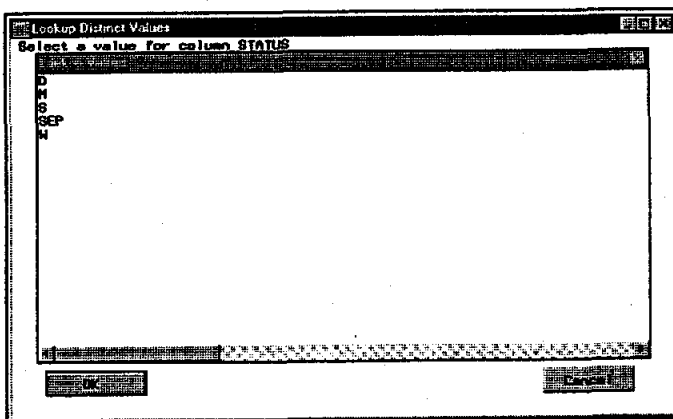
The WHERE EXPRESSION window now shows &PROMPT1 indicating that a value for this variable will be supplied at run time.



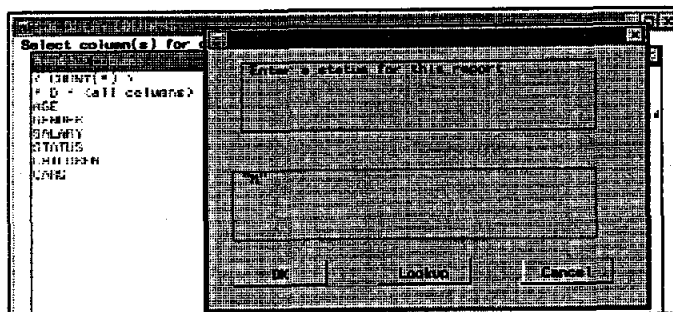
Upon running the query, we are prompted with the Prompt String we entered previously. We have the option of entering a STATUS value, or using the Lookup button to view all values of STATUS.



1. Select the Lookup button. The Lookup Distinct Values window appears.
2. Select M for 'M'arried.
3. Select OK.



4. The result of our selection is shown in the Prompt window. Select OK to continue running the query.



Our output shows that only rows for a STATUS of 'M'arried have been included in our query. Also note that the value of

our AVG(SALARY) statistic has changed since it is now based on incoming rows where STATUS is 'M'arried only. The rows on our output are those rows that :

1. Met the WHERE expression coming in, AND
2. Met the HAVING expression going out.

GENDER	STATUS	SALARY	AVG(SALARY)
F	M	15000	10123.08
F	M	18000	10123.08
F	M	30000	10123.08
F	M	15000	10123.08
F	M	13000	10123.08
F	M	15000	10123.08
N	M	23000	19716.67
N	M	23000	19716.67
N	M	40000	19716.67

If we run the query again and choose 'S'ingle for status, our result changes again.

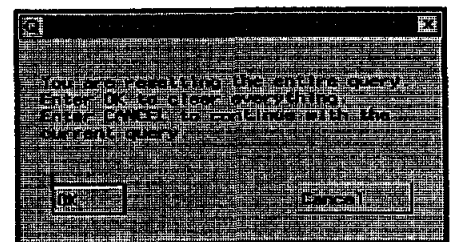
GENDER	STATUS	SALARY	AVG(SALARY)
F	S	12000	8500
F	S	30000	8500
N	S	10000	3500
N	S	7800	3500

JOINING TABLES

Resetting the Query

Before beginning a new query, we can select to reset the current query.

1. From the Actions pulldown, select Reset.
2. We are warned to make sure we know what we are doing! Select OK.



3. By default, tables from the SASUSER library are displayed in the Available Tables list. We can remove them from the list, if desired, by selecting SASUSER from the Table Sources list, and selecting Remove SASUSER from Available Tables.



Joining Tables

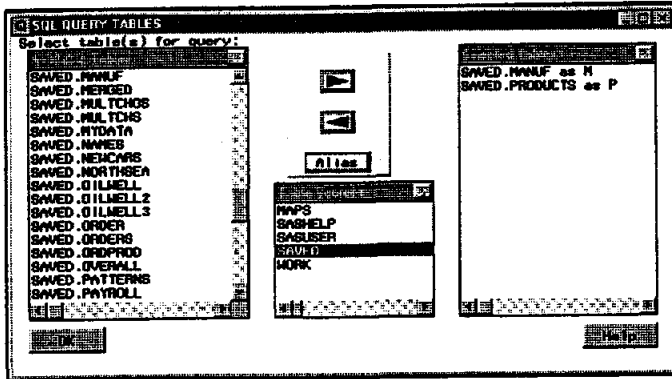
SQL allows 4 ways to join tables:

- Inner Joins
 - Left Joins
 - Right Joins
 - Full Joins
- Simply selecting more than one table implies that the tables are somehow to be joined.

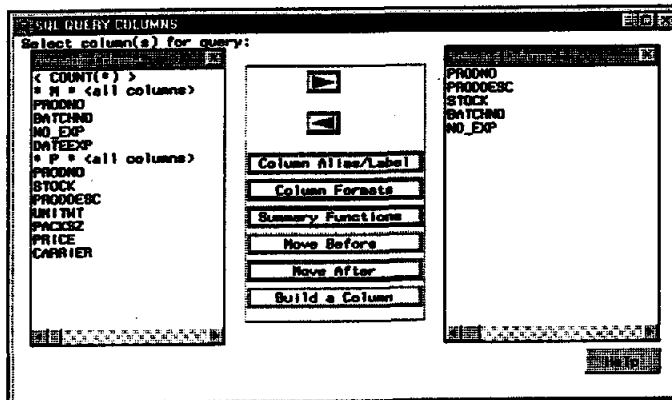
We will discuss joining the SAVED.MANUF and SAVED.PRODUCTS tables. The SAVED.PRODUCTS table contains a row for each product stocked. The SAVED.MANUF table is a subset of products for those that are currently being manufactured.

To begin the join:

1. Select **SAVED.MANUF** and **SAVED.PRODUCTS** from Available Tables.
2. Move the table names to the Selected Tables list.
3. Alias **SAVED.MANUF** as **M** and **SAVED.PRODUCTS** as **P**.



4. Columns from both tables are displayed. Select **PRODNO**, **PRODESC**, and **STOCK** from **P** (**SAVED.PRODUCTS**).
5. Select **BATCHNO** and **NO_EXP** from **M** (**SAVED.MANUF**).



If we attempt to run the query at this point, without specifying on what column the tables are to be joined, we are warned that we are about to create a Cartesian Product, which is probably not something we really want to do!

At this point, we can select the **WHERE** button to take us to the **WHERE EXPRESSION** window and create an Inner Join, or **CANCEL** and select some other join options elsewhere.

Inner Joins

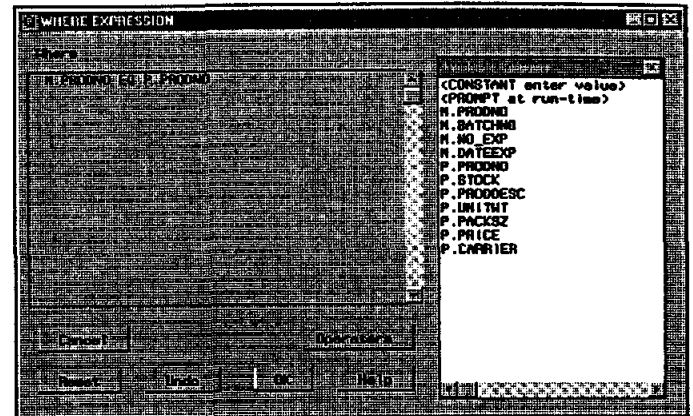
Inner joins require a **WHERE EXPRESSION** to bring rows together that have matching values for a common column or columns. An Inner Join produces an intersection of the tables being joined. The **SAVED.MANUF** and

SAVED.PRODUCTS tables have the common column **PRODNO** which is a product number. Therefore, we can join the tables by this column. The Inner Join will produce only rows where a particular **PRODNO** existed in both tables.

1. Get to the **WHERE EXPRESSION** window, either by having select the **WHERE** button from the above pop-up warning, or from the **Locals** pulldown, **Where** Conditions for Subset option.
2. Put the **WHERE EXPRESSION** together so that it says:

M.PRODNO EQ P.PRODNO.

3. Select **OK**.



Upon running this query, we get a listing of all products that exist in both tables.

PRODNO	PRODESC	STOCK	BATCHNO	NO_EXP
F40013	TABLE	0	F045	30
F40020	CHAIR	9	F098	4
F40021	LOUNGER	15	F122	10
F40022	UMBRELLA	3	F055	8
H10038	15 METER DOMESTIC	0	H045	10
H10039	30 METER DOMESTIC	2	H039	10
H20048	15 METER COMMERCIAL	0	C048	20
H20057	60 METER HEAVY DUTY	2	H025	10
T30023	BALL WHEEL SHAVEN	0	T450	20
T30024	HOE	6	T348	10
T30025	TROWEL	5	T567	10
T30026	SPADE	10	T760	15

Outer Joins

Left Joins, Right Joins, and Full Joins are all types of joins that SQL calls Outer Joins.

Left Joins join all rows from the left listed table with any matching rows from the right listed table based on some common column(s) values. Therefore, rows in the left table that have no match in the right table are still produced on the output.

Right Joins join all rows from the right listed table with any matching rows from the left listed table based on some common column(s).

Full Joins produce all rows from both tables, with matching rows matched up by the common column(s) values.

Selecting a Join Type

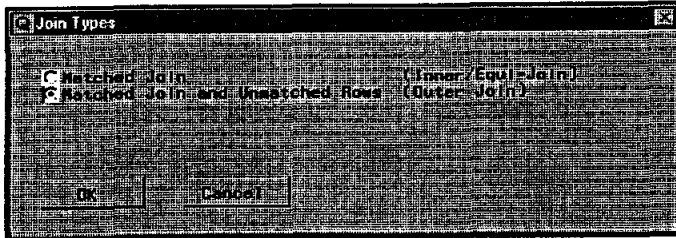
To select to join tables a method to join tables:

1. From the **Locals** pulldown, select **Join Type**.

We are prompted to choose an Inner or Outer Join.

Note: We could have created an Inner Join here instead of using the WHERE EXPRESSION window.

- Click the Matched Join and Unmatched Rows (Outer Join) selection.
- Select OK.

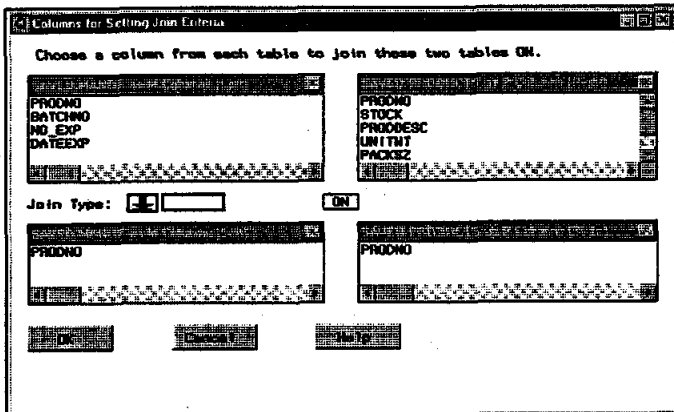


Columns for Setting Join Criteria

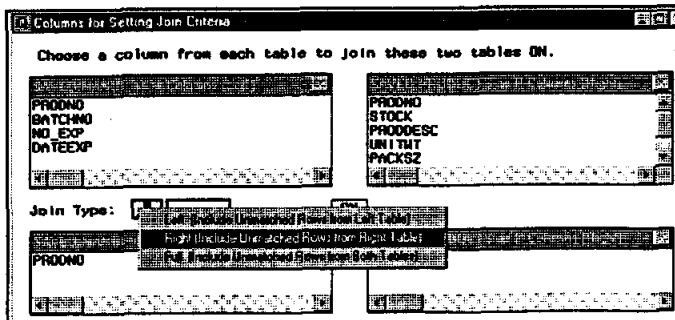
In the Columns for Setting Join Criteria window we can select which columns we want to join by. If we have chosen to produce an Outer Join, we can also select which type of Outer Join we want.

Note that SAVED.MANUF is the (Left) table, and SAVED.PRODUCTS is the (Right) table. This is because SAVED.MANUF was listed above SAVED.PRODUCTS when we selected the tables in the TABLES window, and is therefore the first, or left, table.

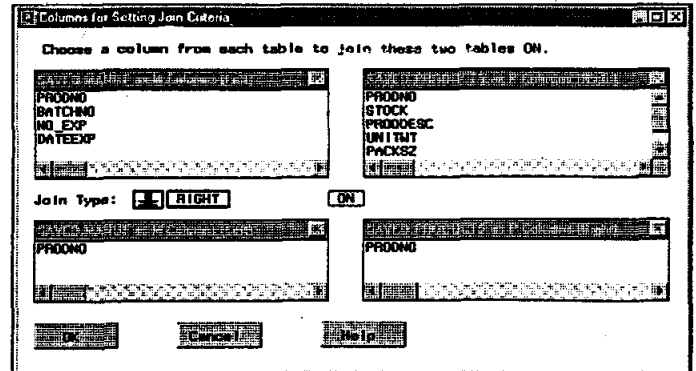
- Select PRODNO from both the left and right tables. Simply selecting it moves the column name to the bottom set of boxes. If the common columns had different names this would not be a problem.



- Select the [v] (down arrow) button next to Join Type.
- From the pop-up selection, choose Right (Include Unmatched Rows from the Right Table).



- The window now shows that we have selected a Right Join. Select OK.



By selecting a Right Join, we are saying that we want all rows from SAVED.PRODUCTS regardless of whether a product is currently being manufactured. If a product is being manufactured, information from SAVED.MANUF will be matched up.

Upon running the query, we see that we have listed all of our products, along with any related manufacturing information if it exists.

PRODNO	PRODDESC	STOCK	BATCHNO	NO_EXP
F40013	TABLE	0	F045	30
F40020	CHAIR	3	F098	4
F40021	LOUNGER	15	F122	10
F40022	UMBRELLA	3	F055	8
H10030	15 METER DOMESTIC	0	H345	10
H10039	30 METER DOMESTIC	2	H590	10
H20040	15 METER COMMERCIAL	0	C340	20
H20049	30 METER COMMERCIAL	4		
H20054	15 METER COMMERCIAL	2		
H20055	60 METER COMMERCIAL	2		
H20056	30 METER HEAVY DUTY	0		
H20057	60 METER HEAVY DUTY	2	H023	10
T30023	BALL WHEEL BARROW	0	T450	20
T30024	HOE	5	T349	10
T30025	TRIMMEL	5	T667	10
T30026	SPADE	10	T760	15
T30027	FORK	10		
T30028	RAKE	10		