Join 2 Tables and Create a Subset

Ingredients:

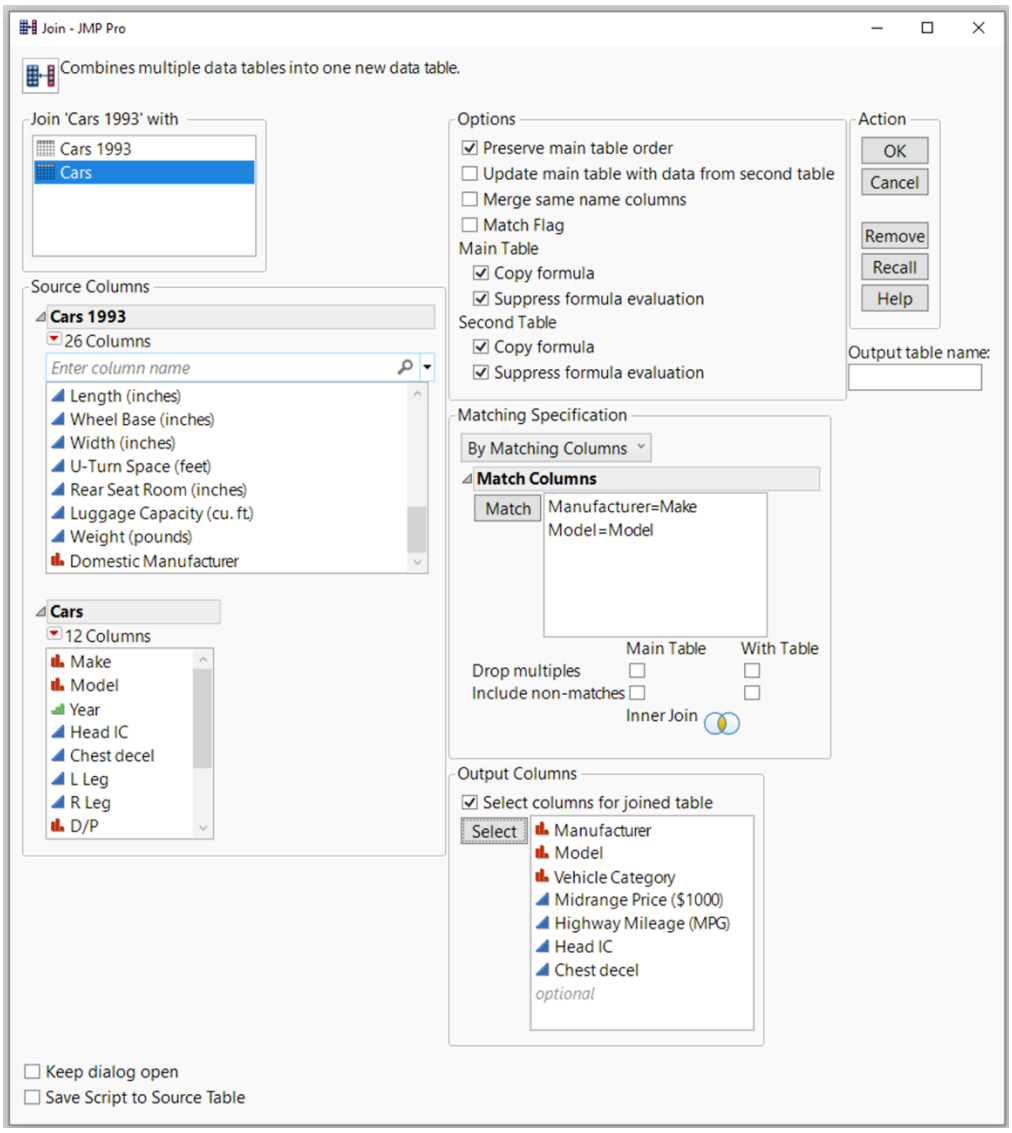
Data Table Objects
Enhanced Log

Sample Data Tables: Cars, Cars 1993
Difficulty – Medium
Video Length – 4:36

In this simple recipe we see how to work with operations from the Tables menu. The JSL code behind many common interactive operations to JMP objects is saved to the Action Recorder and we will want to use it here. Only a few slight tweaks to the prepackaged code are needed to make the results more general and robust. We will start by performing all operations interactively then copy the code from the Action Recorder to a script window. From there we will make our alterations. Working with Data Table columns and row selection will also be shown.

Steps:

1. Start by opening the Log window (under the Windows menu for a Mac and View for Windows). Using the red hotspot at the top left clear the log.
2. Open the Sample Data tables Cars and Cars 1993. Select the later Data Table to make it the current selection.
3. Join the two tables as shown in the dialog below. We will only need a subset of the columns.



4. Rename the columns Midrange Price ($1000) and Highway Milage (MPG) to Price and MPG, respectively
5. Select all the Small and Compact cars. Start by making sure no row or column is selected. Select one Small and one Compact observation from the Data Table. Hover the cursor over one of the selected cells. Right click and choose Select Matching Cells. Alternatively, you can use Rows > Row Selection > Select Where.
6. Using Subset under the Tables menu, subset the selected rows.
7. Close the table created by Join. You don't have to save.
8. Rename the subset table My Cars.
9. Copy the code from the Log. Select the items at the top and from the hot spot choose Save Script > To Script Window. Alternatively, you can copy the material at the bottom of the log and paste it into a script window.

We will be making some alterations to this code, so we don't have to rely on hard coded table names.

10. Add `Names Default to Here(1);` to the top of the script. This is a good programming practice. It ensures that the variables created in the script are scoped locally.
11. Assign variable names to the two Data Tables we started with. We will use `dtCars1` for Cars1993 and `dtCars` for Cars.
12. In the Join command replace `Data Table("Cars 1993")` with `dtCars1993` and `Data Table("Cars")` with `dtCars2`.
13. The Join command returns a reference to the newly created joined table. We can access it by adding a variable name and equal sign in front of the dtCars1 reference:

    ```
    dtJoined = dtCars1 << Join(…
    ```

    The double less than sign (<<) is the `Send` operator. We will use this syntax frequently when performing an operation with a JMP object. Most, but not all, JMP object return a reference when sent an operation (also called a message).
14. Replace the named reference to the joined table with `dtJoined`. This reference will likely have a name similar to Data Table("untitled").
15. Like `Join`, `Subset` returns a reference to the newly created table. Use the name `dtSubset` to get this reference

    ```
    dtSubset = dtJoined << Select Where(…
    ```

16. Replace the hard coded name in the `Close` function with `dtJoined`, its reference variable. Add a second argument to the function, `No Save`, to stop JMP from prompting you to save the table before closing it. The `No Save` argument can be used with or without quotes

    `Close(dtJoined, No Save)` and `Close(dtJoined,"No Save")` work the same way.
17. Finally, replace the hard coded reference to the subset with its variable reference `dtSubset`.

Hints for Success:

- Perform as much of the task interactively and use the code captured in the Enhanced Log.
- Replace hard coded Data Table names with variable references.
- Check JSL functions for optional arguments that alter standard behavior.