

The Joinless Join; *Expand the Power of SAS® Enterprise Guide® in a New Way*

Kent Phelps, Illuminator Coaching, Inc., Des Moines, Iowa
Ronda Phelps, Illuminator Coaching, Inc., Des Moines, Iowa
Kirk Paul Lafler, Software Intelligence Corporation, Spring Valley, California

ABSTRACT

SAS Enterprise Guide can easily combine data from tables or data sets by using a **Graphical User Interface (GUI) PROC SQL Join** to match on like columns or by using a **Base SAS® Program Node DATA Step Merge** to match on the same variable name. However, what do you do when tables or data sets **do not** contain like columns or the same variable name and a Join or Merge **cannot** be used?

Well, we have the answer for you! We invite you to attend our presentation on the **Joinless Join** where we **expand the power of SAS Enterprise Guide in a new way**. You will learn how to design and utilize a Joinless Join to perform Join and Merge processing using tables or data sets which do not contain like columns or the same variable name. We will briefly review the various types of Joins and Merges and then quickly delve into the detailed aspects of how the Joinless Join can advance and enhance your data manipulation and analysis.

We look forward to introducing you
to the **surprising paradox** of the
Joinless Join.

INTRODUCTION



The tagline for SAS is **The Power To Know®**, and your 'power to know' greatly expands with your ability to access, combine, and analyze important data from tables or data sets (referred to as tables going forward). **The Power To Know** sets off **The Power To Create** which leads to **The Power To Automate** – much like an intricate and fluid domino design. However, this power will quickly become disjointed if you do not know how to effectively Join or Merge tables of data which have **direct** and/or **indirect** relationships.

Here are 2 questions you need to ask yourself when analyzing 2 or more tables:

- ❖ Do the tables that I want to analyze contain like columns or the same variable name which can be utilized in a Join or Merge?
- ❖ If the tables do not contain like columns or the same variable name and a Join or Merge cannot be used, have I reached a *cavernous and insurmountable 'woe is me' research impasse* in utilizing the data?

😊 There is no need to fear, the Joinless Join is here! 😊

We are going to show you how to design and utilize a Joinless Join in SAS Enterprise Guide to:

- ❖ Access, combine, and analyze data from tables which only contain columns or variables that indirectly relate to and/or depend upon each other.
- ❖ Perform table creations, calculations, validations, and filtrations based upon indirect relationships and dependencies between the tables.
- ❖ Detect and resolve errors including hidden errors, produce a variety of reports, prevent validation process failure (Yea!), and completely... yes, completely automate your projects.

Our Presentation Demonstrates:

The Power To Know when to design and how to utilize a Joinless Join

The Power To Create tables based upon indirect relationships and dependencies

The Power To Automate projects when tables cannot be directly joined or merged

We invite you to journey with us
as we help you to
E X P A N D
the power of SAS Enterprise Guide in a new way.

Brief Overview of PROC SQL Joins and DATA Step Merges

*Just traveling along...
side by side.*

Harry Macgregor Woods

The Process of Joining and Merging

A Join or Merge is the process of combining **2** or more tables side-by-side (horizontally). Its purpose is to gather and manipulate data from across the tables for exciting insights into data relationships. The process consists of matching rows between the tables and bringing together some or all of each table's contents (see **Figure 1**).



Figure 1. The Process of Joining and Merging Tables

Joins or Merges are specified on a minimum of **2** tables where a column from each table is used for the purpose of connecting the tables. Connecting columns should have **like** values and the **same** column attributes since the success of the Join or Merge is dependent on these values. The ability to define relationships between multiple tables and to retrieve information based on these relationships is a powerful feature of the relational model.

Contrasting Joins and Merges

The differences between a Join and Merge are subtle, but differences do exist –

Join Features:

- ❖ The code conforming to ANSI guidelines is portable to other vendor databases.
- ❖ Data does not need to be sorted using BY-value.
- ❖ Does not require the same variable name.
- ❖ Duplicate matching column is not automatically overlaid.
- ❖ Results are automatically printed unless NOPRINT option is specified.

Merge Features:

- ❖ Relevant only to SAS Software and is not portable to other vendor databases.
- ❖ Data must first be sorted using BY-value.
- ❖ Requires the same variable name.
- ❖ Duplicate matching column is automatically overlaid.
- ❖ Results are not automatically printed.
- ❖ More steps are often needed than with the SQL procedure.

Symmetrical and Asymmetrical Joining and Merging

There are Syntax and Operational differences between Inner (natural) and Outer Joins or Merges –

Inner Join Features:

- ❖ An Inner Join is a Symmetrical Join
- ❖ The maximum number of tables that can be specified in a Join construct is **256**
- ❖ Uses the WHERE-clause

An **Inner Join or Merge** consists of a **symmetrical** process of relating rows in **2** or more tables and produces only matched rows from the tables. The result is illustrated by the shaded area (AB) in **Figure 2**.

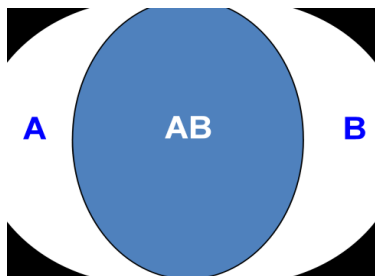


Figure 2. Venn Diagram – Inner Join or Merge

Outer Join and Merge Features

- ❖ An Outer Join or Merge is an Asymmetrical Join or Merge
- ❖ The maximum number of tables that can be specified in this construct is **2**
- ❖ Uses syntax keywords such as LEFT JOIN, RIGHT JOIN, and FULL JOIN
- ❖ Uses the ON-clause

Similar to an Inner Join or Merge, an Outer Join or Merge can match related rows in both tables. But this is where the similarities end. By using an Outer Join, the resulting set of data can also include rows with no related data from one or both tables.

Essentially the resulting set of data from an Outer Join process contains rows that **match** the ON-clause plus any **unmatched** rows from the left, right, or both tables. This approach is sometimes referred to as an **Asymmetrical Join** because its primary purpose is row preservation. This type of processing is a significant feature offered by the Outer Join construct.

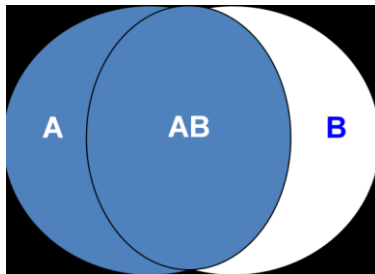


Figure 3. Venn Diagram – Left Outer Join or Merge

The result of a **Left Outer Join or Merge** produces matched rows from both tables while preserving all unmatched rows from the left table. The result is illustrated by the shaded areas (A and AB) in **Figure 3**.

The result of a **Right Outer Join or Merge** produces matched rows from both tables while preserving all unmatched rows from the right table. The result is illustrated by the shaded areas (B and AB) in **Figure 4**.

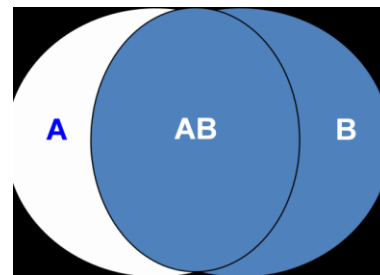
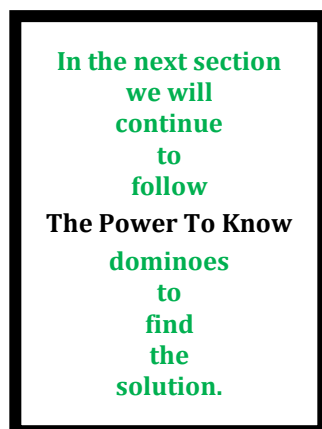


Figure 4. Venn Diagram – Right Outer Join or Merge

This special handling of matched and unmatched rows of data is the main thing that differentiates a Symmetric Inner Join from an Asymmetric Outer Join.

All of these Joins and Merges have one particular common denominator. Each of them requires a like column or the same variable name to match on. Thus, we return to the core focus of this presentation:

**What do you do when tables do not contain
like columns or the same variable name
and a Join or Merge cannot be used?**



Professor Domino will be our guide 😊

Illuminating the Paradox of the Joinless Join

*Sometimes success is seeing
what we already have
in a new light.*

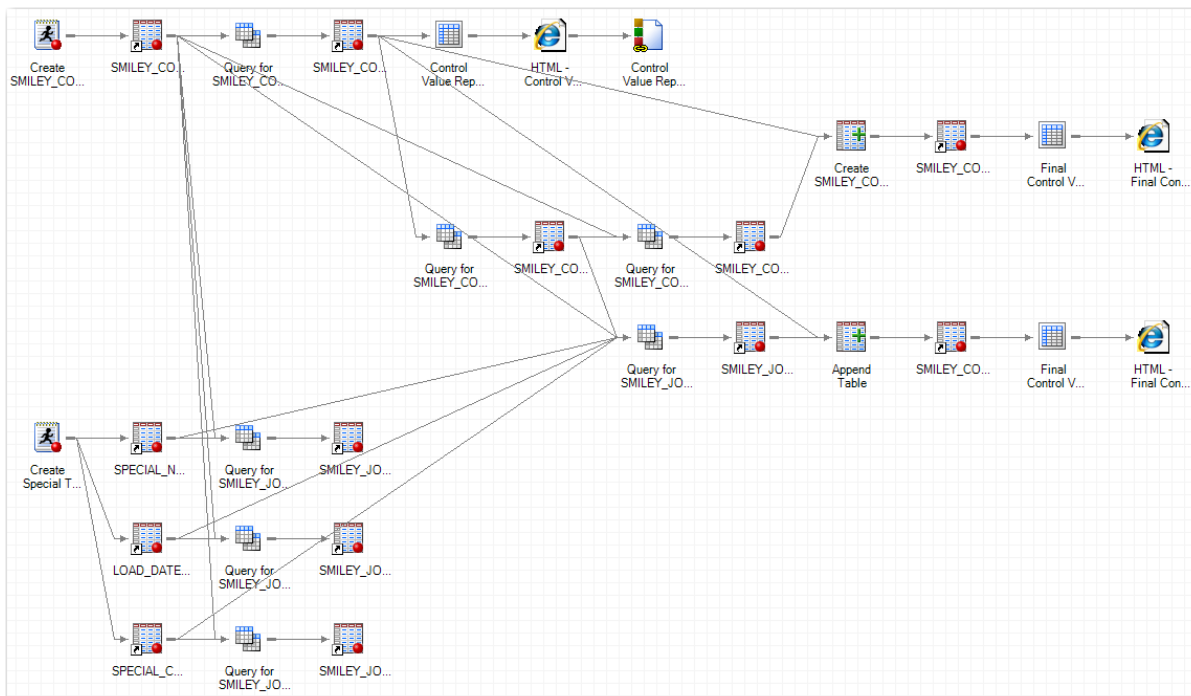
Dan Miller

The development of the **Joinless Join** came about during a recent project when the need arose to overcome the limitations of a standard Join and to resolve unforeseen issues which occurred with a **One-Way Frequency**.

SAS Highlight

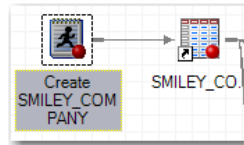
A **One-Way Frequency** contains a distribution list of values, counts, and percentages for a column.

Here is a picture of our example SAS Enterprise Guide project:



❖ Our project example demonstrates 5 ways to use a Joinless Join.

We use a Program Node to create the source table for our 1st example:



```

DATA SMILEY_COMPANY;
  LENGTH Special_Person $20 Special_Number 8 Special_Code $1 Load_Date 8;
  FORMAT Load_Date date9.;
  INFILE DATALINES DELIMITER=' ';
  INPUT Special_Person $ Special_Number Special_Code $ Load_Date;
DATALINES;
Smiley,10127911, ,19362
Smiley's Son,10173341,K,19362
Smiley's Twin,10376606,B,19362
Smiley's Wife,10927911,A,19362
Smiley's Son,11471884,E,19362
Smiley's Twin,11573691,G,19362
Smiley's Daughter,11975386,C,19362
Smiley's Son,12071884,J,19362
Smiley's Son,12871884,D,19362
Smiley's Twin,13173691,A,19362
Smiley's Wife,13771202,D,19362
Smiley's Daughter,13775498,H,19362
Smiley's Son,14171884,I,19362
Smiley's Twin,15373691,F,19362
Smiley's Son,15471884,C,19362
Smiley's Son,16074330,H,19362
Smiley's Daughter,16175498,B,19362
Smiley's Wife,16176964,I,19358
Smiley,16279111,E,19362
Smiley's Twin,16573691,K,19362;
RUN;
  
```

❖ This is the code you will need to recreate this table as you follow along.

The Program Node creates the following table:

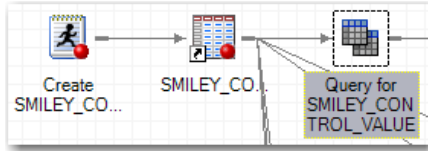
	Special_Person	Special_Number	Special_Code	Load_Date
1	Smiley	10127911		04JAN2013
2	Smiley's Son	10173341	K	04JAN2013
3	Smiley's Twin	10376606	B	04JAN2013
4	Smiley's Wife	10927911	A	04JAN2013
5	Smiley's Son	11471884	E	04JAN2013
6	Smiley's Twin	11573691	G	04JAN2013
7	Smiley's Daughter	11975386	C	04JAN2013
8	Smiley's Son	12071884	J	04JAN2013
9	Smiley's Son	12871884	D	04JAN2013
10	Smiley's Twin	13173691	A	04JAN2013
11	Smiley's Wife	13771202	D	04JAN2013
12	Smiley's Daughter	13775498	H	04JAN2013
13	Smiley's Son	14171884	I	04JAN2013
14	Smiley's Twin	15373691	F	04JAN2013
15	Smiley's Son	15471884	C	04JAN2013
16	Smiley's Son	16074330	H	04JAN2013
17	Smiley's Daughter	16175498	B	04JAN2013
18	Smiley's Wife	16176964	I	31DEC2012
19	Smiley	16279111	E	04JAN2013
20	Smiley's Twin	16573691	K	04JAN2013

❖ The SMILEY_COMPANY table is used throughout this presentation.

❖ This table contains each Special Person, Special Number, and Special Code of the 😊 Smiley Company 😊 employees.

❖ Load_Date is the date when each row was created.

This Query creates the SMILEY_CONTROL_VALUE table:



Query name: Query for SMILEY_CONTROL_VALUE

Computed Columns Prompt Manager Preview Tools Options

Add Tables X Delete

t1 (SMILEY_COMPANY)

- Special_Person
- Special_Number
- Special_Code
- Load_Date
- Computed Columns
 - Special_Person_Flag
 - Special_Number_Flag
 - Special_Code_Flag
 - Load_Date_Flag

Column Name	Identifier
Special_Person_Flag	Special_Person_Flag
Special_Number_Flag	Special_Number_Flag
Special_Code_Flag	Special_Code_Flag
Load_Date_Flag	Load_Date_Flag
Special_Person	t1.Special_Person
Special_Number	t1.Special_Number
Special_Code	t1.Special_Code
Load_Date	t1.Load_Date

A Control Value table is created in which Computed Columns are set to 1 if any data is missing in the SMILEY_COMPANY table:

Special_Person_Flag:

```

CASE
  WHEN t1.Special_Code = '' THEN 1
  ELSE 0
END
  
```

Special_Number_Flag:

```

CASE
  WHEN t1.Special_Number = 0 THEN 1
  WHEN t1.Special_Number is missing
    THEN 1
  ELSE 0
END
  
```

Special_Code_Flag:

```

CASE
  WHEN t1.Special_Code = '' THEN 1
  ELSE 0
END
  
```

Load_Date_Flag:

```

CASE
  WHEN t1.Load_Date = . THEN 1
  ELSE 0
END
  
```

The output is filtered to include only rows where a flag is set to 1:

Query name: Query for SMILEY_CONTROL_VALUE Output name: WORK.SMILEY_CONTROL_VALUE

Computed Columns Prompt Manager Preview Tools Options

Add Tables X Delete

t1 (SMILEY_COMPANY)

- Special_Person
- Special_Number
- Special_Code
- Load_Date
- Computed Columns
 - Special_Person_Flag
 - Special_Number_Flag
 - Special_Code_Flag
 - Load_Date_Flag

Filter the raw data	Operator
Where	
(CALCULATED Special_Person_Flag) = 1	OR
(CALCULATED Special_Number_Flag) = 1	OR
(CALCULATED Special_Code_Flag) = 1	OR
(CALCULATED Load_Date_Flag) = 1	

The output table contains 1 row:

```

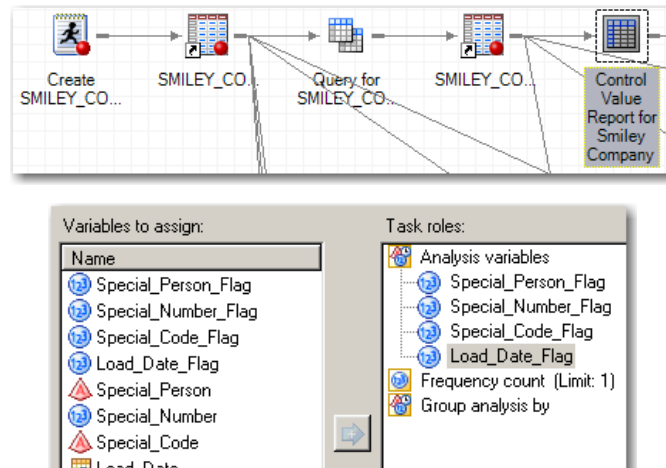
graph LR
    A[Create SMILEY_CO...] --> B[SMILEY_CO...]
    B --> C[Query for SMILEY_CO...]
    C --> D[SMILEY_CON TROL_VALUE]
  
```

Special_Person_Flag	Special_Number_Flag	Special_Code_Flag	Load_Date_Flag
1	0	0	1

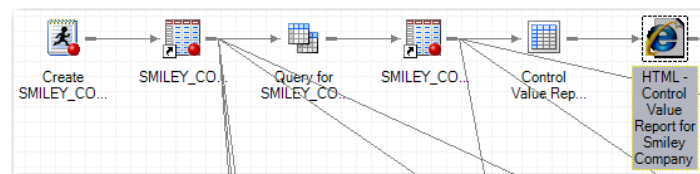
Special_Person	Special_Number	Special_Code	Load_Date
Smiley	10127911		04JAN2013

❖ Notice how the Special_Code_Flag is set to 1 because the Special_Code is missing from this row.

A One-Way Frequency is run using the 4 flags:



Here is how the One-Way Frequency output looks with the 4 flags:



Control Value Report for
Joinless_Join

The FREQ Procedure

Special_Person_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	1	100.00	1	100.00

Special_Number_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	1	100.00	1	100.00

Special_Code_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	1	100.00	1	100.00

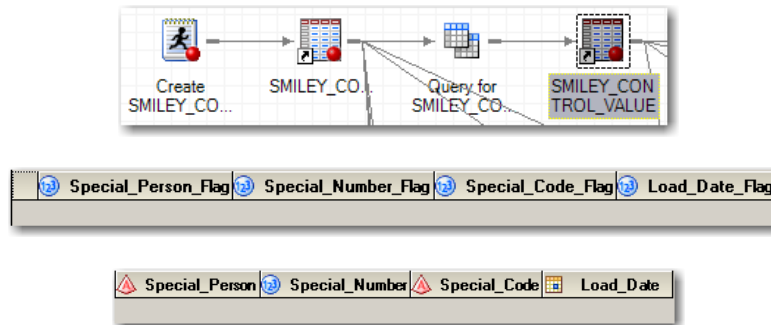
Load_Date_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	1	100.00	1	100.00

- ❖ This One-Way Frequency is setup to automatically send an email when this project is run.

Then one day NOTHING was missing from the SMILEY_COMPANY table...

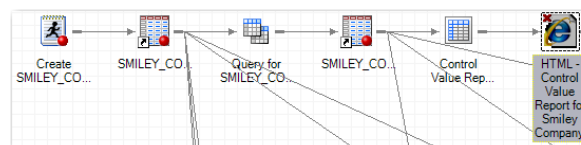
- ❖ To replicate this scenario you will need to perform the following:
 - Replace the Smiley,10127911, ,19362 DATALINE with Smiley,10127911,A,19362 in the SMILEY_COMPANY Program Node on Page 6 and rerun to have no missing data in the table.
 - Rerun the Query for SMILEY_CONTROL_VALUE and the Control Value Report One-Way Frequency.

Here is the empty SMILEY_CONTROL_VALUE table:



- ❖ Since nothing is missing from the SMILEY_COMPANY table, all of the flags are set to 0 which filters out all of the rows causing the SMILEY_CONTROL_VALUE table to be created empty.
- ❖ Do you know what happens when the SMILEY_CONTROL_VALUE table is created empty?

Note the **Red X** in the upper left corner of the One-Way Frequency output:



Control Value Report for Smiley Company

The FREQ Procedure

Special_Person_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	1	100.00	1	100.00

Special_Number_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	1	100.00	1	100.00

Special_Code_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	1	100.00	1	100.00

Load_Date_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	1	100.00	1	100.00

- ❖ At first glance, it appears the report ran just fine – but remember, the input to the Control Value Report was created empty.
- ❖ If the input is empty, then what are we seeing? Notice the **Warning Message** which appeared:

⚠ This report is from an older run of the task or program. Output from the most recent run was not available.

- ❖ This warning message unfortunately means that we are looking at the **previous** successful run of this One-Way Frequency **instead of the current results** which we are seeking.

It was ironic that the first time the Smiley_Company table processed error free (no data was missing) that the resulting empty Smiley_Control_Value table caused the One-Way Frequency to **not** run! Consequently, the previous results were generated on the monthly report instead of the current results.

Here is a review of the One-Way Frequency issue before we explore the solution:

- ❖ When data is missing in the Smiley_Company table a row is created in the Smiley_Control_Value table with the column flags set to **1**.
- ❖ When the Smiley_Control_Value table is populated with at least **1** row the One-Way Frequency runs correctly and generates current results.
- ❖ However, when data is not missing from the Smiley_Company table no rows are created in the Smiley_Control_Value table.
- ❖ When the Smiley_Control_Value table is created empty the One-Way Frequency does not run correctly and does not generate current results but instead displays the previous results.
- ❖ In summary, the One-Way Frequency runs correctly and generates current results only when the Smiley_Control_Value table is populated with at least **1** row created by missing data detected in the Smiley_Company table.

What to do, what to do...

Necessity is the mother of all inventions.
Plato / Einstein

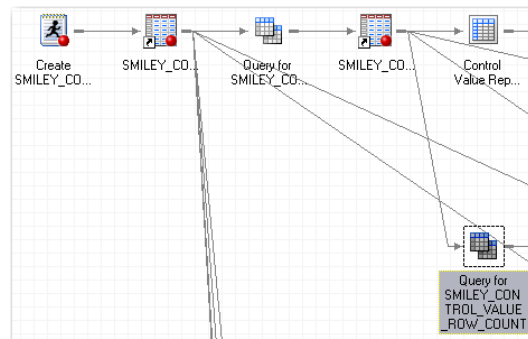
As a result of this dilemma, SAS Intuition kicked in and a quest was undertaken to find a permanent workaround solution that would enable the project to run successfully – **even if all the tables were empty**.

Here is the solution which arose during the quest to resolve this issue:

- ❖ Create a Smiley_Control_Value_Row_Count table with the row count of the Smiley_Control_Value table.
- ❖ Create a Smiley_Control_Value_Mock_Row table based upon an indirect relationship between the Smiley_Control_Value_Row_Count table and the Smiley_Company table.
- ❖ When the Smiley_Control_Value table is populated with rows, the Smiley_Control_Value_Row_Count table will contain a non-zero row count, and the Smiley_Control_Value_Mock_Row table will be created empty.
- ❖ When the Smiley_Control_Value table is empty, the Smiley_Control_Value_Row_Count table will contain a zero row count, and the Smiley_Control_Value_Mock_Row table will be created with **1** mock row of column flags set to **0**.
- ❖ Append the Smiley_Control_Value table and the Smiley_Control_Value_Mock_Row table to ensure that the appended output is always populated with either real data or mock data instead of being created empty.
- ❖ Use this appended output as the input to the One-Way Frequency to enable it to always run correctly and to generate current results.

Always Remember, It's Too Soon To Quit!
Bob Wieland (Mr. Inspiration)

This Query creates the SMILEY_CONTROL_VALUE_ROW_COUNT table with the row count of the SMILEY_CONTROL_VALUE table:



Query name: Query for SMILEY_CONTROL_VALUE_ROW_COUNT Output name: WORK.SMILEY_CONTROL_VALUE_ROW_COUNT

Computed Columns Prompt Manager Preview Tools Options

Add Tables Delete Join Tables

11 (SMILEY_CONTROL_VALUE)

- Special_Person_Flag
- Special_Number_Flag
- Special_Code_Flag
- Load_Date_Flag
- Special_Person
- Special_Number
- Special_Code
- Load_Date

Computed Columns

SMILEY_CONTROL_VALUE_ROW_COUNT

Column Name	Identifier	Summary
SMILEY_CONTROL_VALUE_ROW_COUNT	SMILEY_CONTROL_VALUE_ROW_COUNT	COUNT

Column	Details
SMILEY_CONTROL_VALUE_ROW_COUNT	COUNT(11.Special_Person)

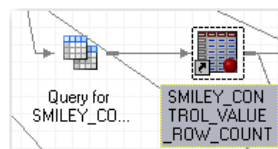
Summary groups

☒ Automatically select groups

No groups selected

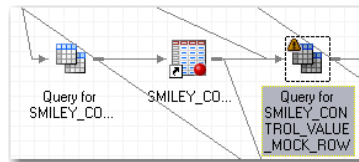
- ❖ A Count of Special_Person is used to create the SMILEY_CONTROL_VALUE_ROW_COUNT.
- ❖ Automatically Select Groups is selected and no groups are selected to count the rows.

The output table contains 1 row with 1 column:



SMILEY_CONTROL_VALUE_ROW_COUNT
1

Create a Smiley_Control_Value_Mock_Row table based upon an indirect relationship between the Smiley_Control_Value_Row_Count table and the Smiley_Company table:



Query for SMILEY_CONTROL_VALUE_MOCK_ROW for SASMain:WORK.SMILEY_COMPANY

Query name: Query for SMILEY_CONTROL_VALUE_MOCK_ROW Output name: WORK.SMILEY_CONTROL_VALUE_MOCK_ROW

Computed Columns Prompt Manager Preview Tools Options

Add Tables X Delete Join Tables

t1 (SMILEY_COMPANY)

- Special_Person
- Special_Number
- Special_Code
- Load_Date

t2 (SMILEY_CONTROL_VALUE_ROW_COUNT)

- SMILEY_CONTROL_VALUE_ROW_COUNT
- Computed Columns
- Special_Person_Flag
- Special_Number_Flag
- Special_Code_Flag
- Load_Date_Flag

Select Data Filter Data Sort Data

Column Name	Identifier	Summary	Format	Details
Special_Person_Flag	Special_Person_Flag			0
Special_Number_Flag	Special_Number_Flag			0
Special_Code_Flag	Special_Code_Flag			0
Load_Date_Flag	Load_Date_Flag			0
Special_Person	t1.Special_Person			
Special_Number	t1.Special_Number			
Special_Code	t1.Special_Code			
Load_Date	t1.Load_Date			

Computed Columns

Column	Details
Load_Date_Flag	0
Special_Code_Flag	0
Special_Number_Flag	0
Special_Person_Flag	0

Query limits

☐ Limit number of matching rows to process: 1

☒ Limit number of rows to save in output: 1

- ❖ As the mock row is created, all 4 flags are set to a 0 value meaning nothing is missing.
- ❖ Since only 1 mock row is needed, Query limits are set to create 1 output row via the Options.

Select Data Filter Data Sort Data

Filter the raw data

Where

t2.SMILEY_CONTROL_VALUE_ROW_COUNT = 0

- ❖ A filter is set to create a mock row only if the SMILEY_CONTROL_VALUE table is empty.

Notice there are no columns to Join between the two tables:

Tables and Joins

Add Tables X Delete Properties Join Order Table Options Move Up Move Down

t1 (SMILEY_COMPANY)

- Special_Person
- Special_Number
- Special_Code
- Load_Date

t2 (SMILEY_CONTROL_VALUE_ROW_COUNT)

- SMILEY_CONTROL_VALUE_ROW_COUNT

No Problem –

We can use a Joinless Join based upon an indirect relationship between the tables.

How the Joinless Join works:

	Special_Person	Special_Number	Special_Code	Load_Date
1	Smiley	10127911	A	04JAN2013
2	Smiley's Son	10173341	K	04JAN2013
3	Smiley's Twin	10376606	B	04JAN2013
4	Smiley's Wife	10927911	A	04JAN2013
5	Smiley's Son	11471884	E	04JAN2013
6	Smiley's Twin	11573691	G	04JAN2013
7	Smiley's Daughter	11975386	C	04JAN2013
8	Smiley's Son	12071884	J	04JAN2013
9	Smiley's Son	12871884	D	04JAN2013
10	Smiley's Twin	13173691	A	04JAN2013
11	Smiley's Wife	13771202	D	04JAN2013
12	Smiley's Daughter	13775498	H	04JAN2013
13	Smiley's Son	14171884	I	04JAN2013
14	Smiley's Twin	15373691	F	04JAN2013
15	Smiley's Son	15471884	C	04JAN2013
16	Smiley's Son	16074330	H	04JAN2013
17	Smiley's Daughter	16175498	B	04JAN2013
18	Smiley's Wife	16176964	I	31DEC2012
19	Smiley	16279111	E	04JAN2013
20	Smiley's Twin	16573691	K	04JAN2013

	SMILEY_CONTROL_VALUE_ROW_COUNT
1	0

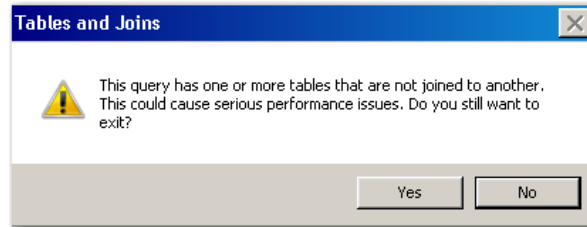
- ❖ The SMILEY_CONTROL_VALUE_ROW_COUNT table indirectly relates to the SMILEY_COMPANY table because it contains the row count of the error rows in the SMILEY_COMPANY table.
- ❖ We utilize a Joinless Join to create a **Cartesian Product** based upon this indirect relationship.

	Special_Person	Special_Number	Special_Code	Load_Date	CONTROL_VALUE_ROW_COUNT
1	Smiley	10127911	A	04JAN2013	0
2	Smiley's Son	10173341	K	04JAN2013	0
3	Smiley's Twin	10376606	B	04JAN2013	0
4	Smiley's Wife	10927911	A	04JAN2013	0
5	Smiley's Son	11471884	E	04JAN2013	0
6	Smiley's Twin	11573691	G	04JAN2013	0
7	Smiley's Daughter	11975386	C	04JAN2013	0
8	Smiley's Son	12071884	J	04JAN2013	0
9	Smiley's Son	12871884	D	04JAN2013	0
10	Smiley's Twin	13173691	A	04JAN2013	0
11	Smiley's Wife	13771202	D	04JAN2013	0
12	Smiley's Daughter	13775498	H	04JAN2013	0
13	Smiley's Son	14171884	I	04JAN2013	0
14	Smiley's Twin	15373691	F	04JAN2013	0
15	Smiley's Son	15471884	C	04JAN2013	0
16	Smiley's Son	16074330	H	04JAN2013	0
17	Smiley's Daughter	16175498	B	04JAN2013	0
18	Smiley's Wife	16176964	I	31DEC2012	0
19	Smiley	16279111	E	04JAN2013	0
20	Smiley's Twin	16573691	K	04JAN2013	0

- ❖ The **Cartesian Product** attaches the 1 column in the SMILEY_CONTROL_VALUE_ROW_COUNT table to the 20 rows of columns in the SMILEY_COMPANY table by placing it as a 5th column to the right of the Load_Date column in the SMILEY_COMPANY table.

SAS Highlight

A **Cartesian Product** is defined as a result set of all the possible rows and columns contained in 2 or more tables. The DATA Step does not easily lend itself to creating a Cartesian Product thus PROC SQL is the desired approach. Its most noticeable coding characteristic is the absence of a WHERE-clause. The resulting set of data from a Cartesian Product can be extremely large and unwieldy. Although rarely produced, a Cartesian Product Join nicely illustrates a base (or internal representation) for all Joins.

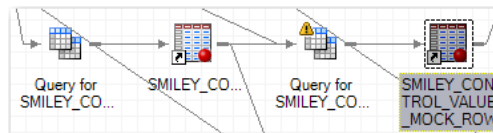


- ❖ This **Warning Message** always appears whenever 2 tables are joined with a Joinless Join because SAS knows it will create a **Cartesian Product** which can take a lot of extra resources.

Caution:

When you design your Joinless Join
make sure that one of the tables
has only **ONE** row!

Here is the complete result of the Joinless Join:

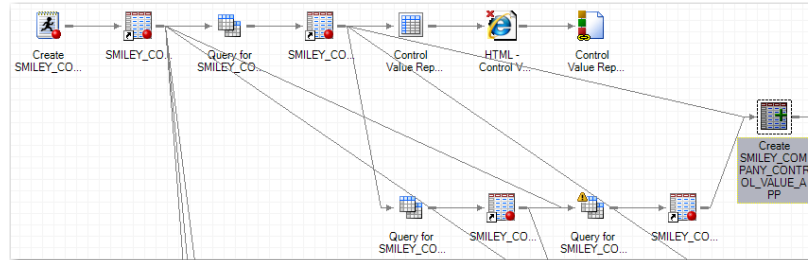


	Special_Person_Flag	Special_Number_Flag	Special_Code_Flag	Load_Date_Flag
1	0	0	0	0

Special_Person	Special_Number	Special_Code	Load_Date
Smiley	10127911	A	04JAN2013

- ❖ Notice how all 4 flags are set to 0 because no data is missing from the SMILEY_COMPANY table.

Append the Smiley_Control_Value table and the Smiley_Control_Value_Mock_Row table to ensure that the appended output is always populated with either real data or mock data instead of being created empty:



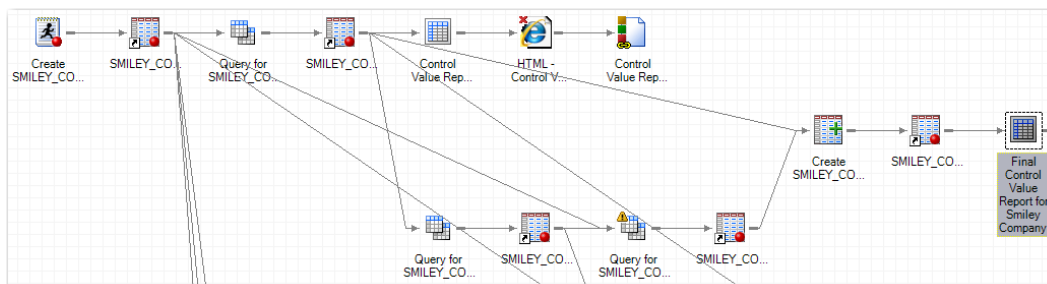
Append Table	
Tables	Results
Tables to append	
Table Name	
SMILEY_CONTROL_VALUE	
SMILEY_CONTROL_VALUE_MOCK_ROW	

Special_Person_Flag	Special_Number_Flag	Special_Code_Flag	Load_Date_Flag
1	0	0	0

Special_Person	Special_Number	Special_Code	Load_Date
Smiley	10127911	A	04/JAN2013

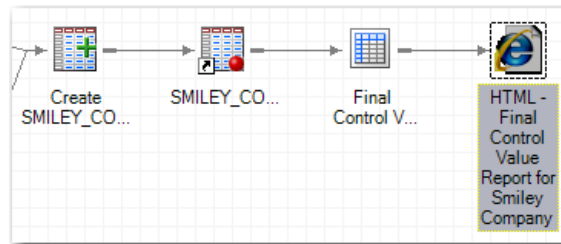
- ❖ Notice the Append result matches the Smiley_Control_Value_Mock_Row table – **Done & Done!**
- ❖ We have achieved our desired results and we have a new input to the One-Way Frequency.

The One-Way Frequency is recreated using the appended table:



Variables to assign:		Task roles:	
Name		Analysis variables	
Special_Person_Flag		Special_Person_Flag	
Special_Number_Flag		Special_Number_Flag	
Special_Code_Flag		Special_Code_Flag	
Load_Date_Flag		Load_Date_Flag	
Special_Person		Frequency count (Limit: 1)	
Special_Number		Group analysis by	
Special_Code			
Load Date			

Here is the output of the One-Way Frequency with the 4 flags:



Final Control Value Report for Smiley Company				
The FREQ Procedure				
Special_Person_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	1	100.00	1	100.00

Special_Number_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	1	100.00	1	100.00

Special_Code_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	1	100.00	1	100.00

Load_Date_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	1	100.00	1	100.00

- ❖ The One-Way Frequency correctly displays that all 4 flags are set to 0 and therefore no data is missing – thanks to the Joinless Join 😊.



Yea!!!

Strike up the Band,

Toss the Confetti,

Release the Balloons!

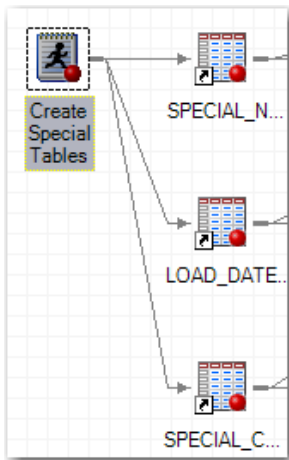
Applause... Applause... Applause...

Bring out the Treats for Everyone!



😊 Oh but wait... your new friend, the Joinless Join, is just getting started... 😊

Now we design another Program Node to create 3 additional tables:



```
DATA Special_Number_National_Average
    (KEEP=Special_Number_National_Average)

Load_Date_Check (KEEP=Load_Date_Check)

Special_Code_National_Focus
    (KEEP=Special_Code_National_Focus);

LENGTH Load_Date_Check 8;
FORMAT Load_Date_Check date9.;

Special_Number_National_Average = 12000000;
OUTPUT Special_Number_National_Average;

Load_Date_Check = '01JAN2013'd;
OUTPUT Load_Date_Check;

Special_Code_National_Focus = 'K';
OUTPUT Special_Code_National_Focus;

RUN;
```

- ❖ This is the code you will need to recreate these tables as you follow along.

Here are the 3 additional tables the Program Node creates:

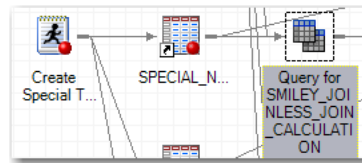
SPECIAL_NUMBER_NATIONAL_AVERAGE	
Filter and Sort	Query Builder Data
Special_Number_National_Average	
1	12000000

LOAD_DATE_CHECK	
Filter and Sort	Query
Load_Date_Check	
1	01JAN2013

SPECIAL_CODE_NATIONAL_FOCUS	
Filter and Sort	Query Builder Data
Special_Code_National_Focus	
1	K

- ❖ The Special_Number_National_Average table contains the average special number taken from all of the **Smiley Companies** nationwide. This will be used to calculate a percentage of the Special_Number column in the SMILEY_COMPANY table using a Joinless Join.
- ❖ The Load_Date_Check table contains a Load Date which is used to validate that all of the rows in the SMILEY_COMPANY table were created in 2013. This validation will be done using a Joinless Join.
- ❖ The Special_Code_National_Focus table contains a Special Code from the **Smiley Company National Headquarters** which will be used to filter the output of the SMILEY_COMPANY table using a Joinless Join.

This Query uses a Joinless Join to perform a Calculation:



Query for SMILEY_JOINLESS_JOIN_CALCULATION for SASMain:WORK.SMILEY_COMPANY

Query name: Query for SMILEY_JOINLESS_JOIN_CALCULATION Output name: WORK.SMILEY_JOINLESS_JOIN_CALCULATION

Computed Columns Prompt Manager Preview Tools Options

Add Tables X Delete Join Tables

Select Data Filter Data Sort Data

Column Name	Identifier	Summary	Format
Special_Person	t1.Special_Person		
Special_Number	t1.Special_Number		
Special_Code	t1.Special_Code		
Load_Date	t1.Load_Date		
Special_Number_Percent	Special_Number_Percent		PERCENT8.1

Tables and Joins

Add Tables X Delete Properties Join Order Table Options Move Up

t1 (SMILEY_COMPANY)	t2 (SPECIAL_NUMBER_NATIONAL_AVERAGE)
Special_Person	Special_Number_National_Average
Special_Number	
Special_Code	
Load_Date	

- ❖ The Cartesian Product is created behind the scenes since no Join exists between the 2 tables.

Computed Columns

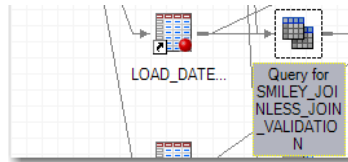
Column	Details
Special_Number_Percent	t1.Special_Number/t2.Special_Number_National_Average

- ❖ Special_Number_Percent is calculated using indirectly related columns from both tables.

	Special_Person	Special_Number	Special_Code	Load_Date	Special_Number_Percent
1	Smiley	10127911	A	04JAN2013	84.4%
2	Smiley's Son	10173341	K	04JAN2013	84.8%
3	Smiley's Twin	10376606	B	04JAN2013	86.5%
4	Smiley's Wife	10927911	A	04JAN2013	91.1%
5	Smiley's Son	11471884	E	04JAN2013	95.6%
6	Smiley's Twin	11573691	G	04JAN2013	96.4%
7	Smiley's Daughter	11975386	C	04JAN2013	99.8%
8	Smiley's Son	12071884	J	04JAN2013	100.6%
9	Smiley's Son	12871884	D	04JAN2013	107.3%
10	Smiley's Twin	13173691	A	04JAN2013	109.8%
11	Smiley's Wife	13771202	D	04JAN2013	114.8%
12	Smiley's Daughter	13775498	H	04JAN2013	114.8%
13	Smiley's Son	14171884	I	04JAN2013	118.1%
14	Smiley's Twin	15373691	F	04JAN2013	128.1%
15	Smiley's Son	15471884	C	04JAN2013	128.9%
16	Smiley's Son	16074330	H	04JAN2013	134.0%
17	Smiley's Daughter	16175498	B	04JAN2013	134.8%
18	Smiley's Wife	16176964	I	31DEC2012	134.8%
19	Smiley	16279111	E	04JAN2013	135.7%
20	Smiley's Twin	16573691	K	04JAN2013	138.1%

- ❖ Here is the final result with the Special_Number_Percent column to the right of each row.

This Query uses a Joinless Join to perform a Validation:



Query for SMILEY_JOINLESS_JOIN_VALIDATION for SASMain:WORK.SMILEY_COMPANY

Query name: Query for SMILEY_JOINLESS_JOIN_VALIDATION Output name: WORK.SMILEY_JOINLESS_JOIN_VALIDATION

Computed Columns Prompt Manager Preview Tools Options

Add Tables Delete

t1 (SMILEY_COMPANY)

- Special_Person
- Special_Number
- Special_Code
- Load_Date

t2 (LOAD_DATE_CHECK)

- Load_Date_Check
- Computed Columns
- Date_Validation

Column Name	Identifier	Summary	Format
Special_Person	t1.Special_Person		
Special_Number	t1.Special_Number		
Special_Code	t1.Special_Code		
Load_Date	t1.Load_Date		
Date_Validation	Date_Validation		

Tables and Joins

Add Tables Delete Properties Join Order Table

t1 (SMILEY_COMPANY)	t2 (LOAD_DATE_CHECK)
Special_Person	
Special_Number	
Special_Code	
Load_Date	Load_Date_Check

- ❖ The Cartesian Product is created behind the scenes since no Join exists between the 2 tables.

Column	Details
Date_Validation	case when t1.Load_Date ge t2.Load_Date_Check then 'AOK' else 'NOT_AOK' end

```

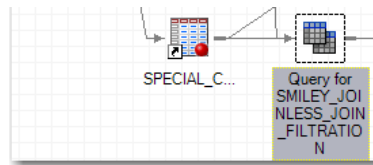
case
  when t1.Load_Date ge t2.Load_Date_Check
  then 'AOK'
  else 'NOT_AOK'
end
        
```

- ❖ Date_Validation is computed using indirectly related columns from both tables.

	Special_Person	Special_Number	Special_Code	Load_Date	Date_Validation
1	Smiley	10127911	A	04JAN2013	AOK
2	Smiley's Son	10173341	K	04JAN2013	AOK
3	Smiley's Twin	10376606	B	04JAN2013	AOK
4	Smiley's Wife	10927911	A	04JAN2013	AOK
5	Smiley's Son	11471884	E	04JAN2013	AOK
6	Smiley's Twin	11573691	G	04JAN2013	AOK
7	Smiley's Daughter	11975386	C	04JAN2013	AOK
8	Smiley's Son	12071884	J	04JAN2013	AOK
9	Smiley's Son	12871884	D	04JAN2013	AOK
10	Smiley's Twin	13173691	A	04JAN2013	AOK
11	Smiley's Wife	13771202	D	04JAN2013	AOK
12	Smiley's Daughter	13775498	H	04JAN2013	AOK
13	Smiley's Son	14171884	I	04JAN2013	AOK
14	Smiley's Twin	15373691	F	04JAN2013	AOK
15	Smiley's Son	15471884	C	04JAN2013	AOK
16	Smiley's Son	16074330	H	04JAN2013	AOK
17	Smiley's Daughter	16175498	B	04JAN2013	AOK
18	Smiley's Wife	16176964	I	31DEC2012	NOT_AOK
19	Smiley	16279111	E	04JAN2013	AOK
20	Smiley's Twin	16573691	K	04JAN2013	AOK

- ❖ Here is the final result with the Date_Validation column to the right of each row.

This Query uses a Joinless Join to perform a Filtration:



Query for SMILEY_JOINLESS_JOIN_FILTRATION for SASMain:WORK.SMILEY_COMPANY

Query name: Query for SMILEY_JOINLESS_JOIN_FILTRATION Output name: WORK.SMILEY_JOINLESS_JOIN_FILTRATION

Computed Columns Prompt Manager Preview Tools Options

Add Tables X Delete Join Tables

t1 (SMILEY_COMPANY)

- Special_Person
- Special_Number
- Special_Code
- Load_Date

t2 (SPECIAL_CODE_NATIONAL_FOCUS)

- Special_Code_National_Focus

Select Data Filter Data Sort Data

Column Name	Identifier	Summary	Format
Special_Person	t1.Special_Person		
Special_Number	t1.Special_Number		
Special_Code	t1.Special_Code		
Load_Date	t1.Load_Date		

Tables and Joins

Add Tables X Delete Properties Join Order Table Options Move

t1 (SMILEY_COMPANY)	t2 (SPECIAL_CODE_NATIONAL_FOCUS)
Special_Person	Special_Code_National_Focus
Special_Number	
Special_Code	
Load_Date	

- ❖ The Cartesian Product is created behind the scenes since no Join exists between the 2 tables.

Select Data Filter Data Sort Data

Filter the raw data

Where

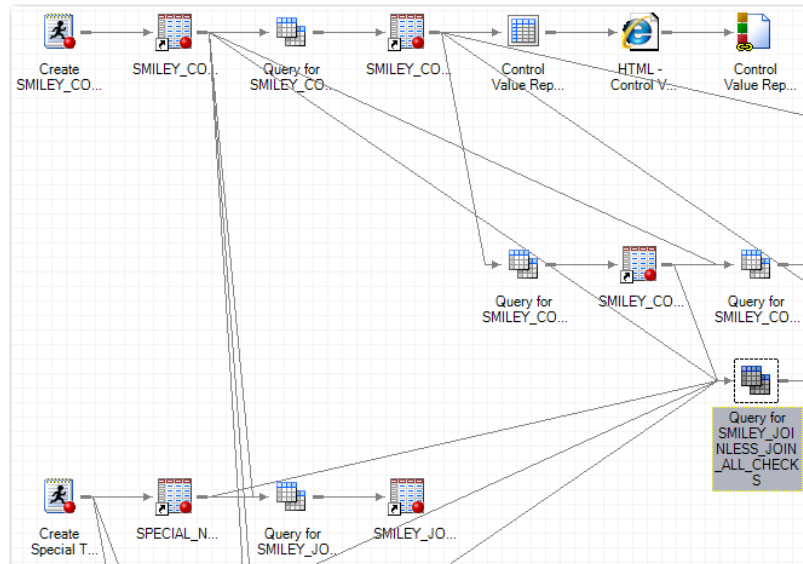
t1.Special_Code = t2.Special_Code_National_Focus

- ❖ A filter is created based upon indirectly related columns from both tables.

	Special_Person	Special_Number	Special_Code	Load_Date
1	Smiley's Son	10173341	K	04JAN2013
2	Smiley's Twin	16573691	K	04JAN2013

- ❖ Here is the final result with the Smiley_Company table filtered indirectly based upon the Special_Code_National_Focus table.

This Query uses 4 Joinless Joins to perform a Mock Row Creation, a Calculation, a Validation, and a Filtration:



Query for SMILEY_JOINLESS_JOIN_ALL_CHECKS for SASMain:WORK.SMILEY_COMPANY

Query name: Query for SMILEY_JOINLESS_JOIN_ALL_CHECKS Output name: WORK.SMILEY_JOINLESS_JOIN_ALL_CHECKS

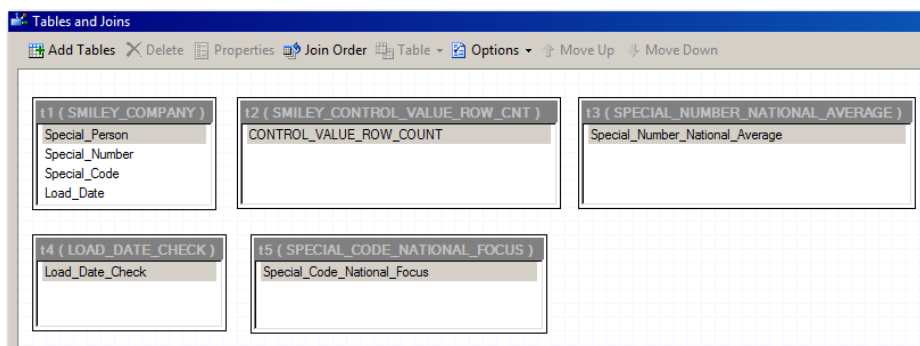
Computed Columns Prompt Manager Preview Tools Options

Add Tables X Delete Join Tables

Column Name	Identifier	Summary	Format
Special_Person_Flag	Special_Person_Flag		
Special_Number_Flag	Special_Number_Flag		
Special_Code_Flag	Special_Code_Flag		
Load_Date_Flag	Load_Date_Flag		
Special_Person	t1.Special_Person		
Special_Number	t1.Special_Number		
Special_Code	t1.Special_Code		
Load_Date	t1.Load_Date		
Special_Number_Percent	Special_Number_Percent		PERCENTN8.1
Date_Validation	Date_Validation		
Special_Code_Match	Special_Code_Match		

Computed Columns

- Special_Person_Flag
- Special_Number_Flag
- Special_Code_Flag
- Load_Date_Flag
- Special_Number_Percent
- Date_Validation
- Special_Code_Match



❖ The Cartesian Product is created behind the scenes since no Join exists between the 5 tables.

The Mock Row Creation, Calculation, Validation, and Filtration are represented by Computed Columns:

Computed Columns	
Column	Details
Date_Validation	case when t1.Load_Date get t4.Load_Date_Check then 'AOK' else 'NOT_AOK' end
Load_Date_Flag	case t2.CONTROL_VALUE_ROW_COUNT when 0 then 0 else . end
Special_Code_Flag	case t2.CONTROL_VALUE_ROW_COUNT when 0 then 0 else . end
Special_Code_Match	case when t1.Special_Code = t5.Special_Code_National_Focus then 'MATCH' else 'NO MATCH' end
Special_Number_Flag	case t2.CONTROL_VALUE_ROW_COUNT when 0 then 0 else . end
Special_Number_Percent	t1.Special_Number/t3.Special_Number_National_Average
Special_Person_Flag	case t2.CONTROL_VALUE_ROW_COUNT when 0 then 0 else . end

- ❖ The Computed Columns are derived in the same way as shown in the previous examples along with one new Special_Code_Match Computed Column representing Filtration.

	Special_Person_Flag	Special_Number_Flag	Special_Code_Flag	Load_Date_Flag	Special_Person	Special_Number	Special_Code	Load_Date	Special_Number_Percent	Date_Validation	Special_Code_Match
1	0	0	0	0	Smiley	10127911	A	04JAN2013	84.4%	AOK	NO MATCH
2	0	0	0	0	Smiley's Son	10173341	K	04JAN2013	84.8%	AOK	MATCH
3	0	0	0	0	Smiley's Twin	10376606	B	04JAN2013	86.5%	AOK	NO MATCH
4	0	0	0	0	Smiley's Wife	10927911	A	04JAN2013	91.1%	AOK	NO MATCH
5	0	0	0	0	Smiley's Son	11471884	E	04JAN2013	95.6%	AOK	NO MATCH
6	0	0	0	0	Smiley's Twin	11573691	G	04JAN2013	96.4%	AOK	NO MATCH
7	0	0	0	0	Smiley's Daughter	11975386	C	04JAN2013	99.8%	AOK	NO MATCH
8	0	0	0	0	Smiley's Son	12071884	J	04JAN2013	100.6%	AOK	NO MATCH
9	0	0	0	0	Smiley's Son	12871884	D	04JAN2013	107.3%	AOK	NO MATCH
10	0	0	0	0	Smiley's Twin	13173691	A	04JAN2013	109.8%	AOK	NO MATCH
11	0	0	0	0	Smiley's Wife	13771202	D	04JAN2013	114.8%	AOK	NO MATCH
12	0	0	0	0	Smiley's Daughter	13775498	H	04JAN2013	114.8%	AOK	NO MATCH
13	0	0	0	0	Smiley's Son	14171884	I	04JAN2013	118.1%	AOK	NO MATCH
14	0	0	0	0	Smiley's Twin	15373691	F	04JAN2013	128.1%	AOK	NO MATCH
15	0	0	0	0	Smiley's Son	15471884	C	04JAN2013	128.9%	AOK	NO MATCH
16	0	0	0	0	Smiley's Son	16074330	H	04JAN2013	134.0%	AOK	NO MATCH
17	0	0	0	0	Smiley's Daughter	16175498	B	04JAN2013	134.8%	AOK	NO MATCH
18	0	0	0	0	Smiley's Wife	16176964	I	31DEC2012	134.8%	NOT_AOK	NO MATCH
19	0	0	0	0	Smiley	16279111	E	04JAN2013	135.7%	AOK	NO MATCH
20	0	0	0	0	Smiley's Twin	16573691	K	04JAN2013	138.1%	AOK	MATCH

- ❖ Here is the final result with the Flags to the left and the Calculation, Validation, and Filtration Computed Columns to the right of each row.

Control Value Report for Smiley Company All Joinless Joins				
The FREQ Procedure				
Special_Person_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	20	100.00	20	100.00
Special_Number_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	20	100.00	20	100.00
Special_Code_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	20	100.00	20	100.00
Load_Date_Flag	Frequency	Percent	Cumulative Frequency	Cumulative Percent
0	20	100.00	20	100.00

- ❖ The One-Way Frequency correctly displays that all 4 flags are set to 0 and therefore no data is missing – thanks to the Joinless Join 😊.

CONCLUSION

Our presentation has shown you how a **Joinless Join** can bridge the research impasse you experience when needing to combine data from SAS tables which do not contain like columns or the same variable name. You have learned how to perform table creations, calculations, validations, and filtrations based upon indirect relationships and dependencies. The Joinless Join empowers you to creatively overcome the limits of a standard Join or Merge and enables you to expand the power of SAS Enterprise Guide in a new way.

The Power To Know when to design and how to utilize a Joinless Join sets off **The Power To Create** tables based upon indirect relationships and dependencies which leads to **The Power To Automate** projects when tables cannot be directly joined or merged. (Say that statement really fast for fun 😊.) Begin thinking about how you can use the powerful and paradoxical Joinless Join to advance and enhance your programming.



*How wonderful it is that we need not wait a single minute
before starting to improve ourselves and our world!*

Anne Frank

SAS Programming is like a series of intricate and fluid domino designs and **you** are the designer. Your desire to design a quality program fuels your thoroughness and attention to detail. As a **SAS Professional**, your inquisitive nature, research oriented mindset, and solution driven focus are among your greatest assets.



*Your life is like a campfire at night -
You never know how many people will see it
and be comforted and guided by your light.*

Claire Draper

Rule #6: Study hard and learn all you can.

😊 Roy Rogers Riders Club Rules 😊



Always remember, *It's not what the world holds for you, it's what **YOU** bring to it!* (Anne of Green Gables); so keep learning, growing, and sharing. Continue to develop and build on your many skills and talents. Keep looking for different ways to express your God-given abilities and ideas. You will soon discover new and creative ways to program in SAS. Plan on coming back next year to the MWSUG Conference to shed some light on the exciting things you are learning. We look forward to your teaching sessions in the future.

As we conclude our presentation, we want to introduce you to our **SAS Mascot, Smiley**. Smiley represents the **SAS Joy** which each of us experience when we find better ways to accomplish mighty and worthy deeds using SAS. The four of us hope your time with us today has expanded and enriched your SAS knowledge.

Thank You For Honoring Us With Your Participation

😊 **Happy SAS Trails To You... Until We Meet Again** 😊



MEET THE AUTHORS

Writing is a permanent legacy.

John C. Maxwell

Kent Phelps (*Co-Founder/President, Illuminator Coaching, Inc.*) is a Senior Data Governance Analyst and has worked in IT and Data Governance since 1990. He has programmed in SAS since 2007, is a SAS Certified Professional specializing in combining and automating the best of SAS Enterprise Guide with Base SAS, and has Co-Created and Co-Led *Intro To SAS EG* classes. Kent has a B.S. in Electrical Engineering, has studied Transformational Leadership, Dynamic Teamwork, and Personal Growth since 1994, and is a *48 Days To The Work You Love* Coach and a *John Maxwell Team* Coach. His hope is to encourage you to pursue your unique destiny and to equip you to navigate your journey with purpose and passion.

Ronda Phelps (*Co-Founder/President, Illuminator Coaching, Inc.*) is a Writer, Communicator, Teacher, and Coach. Formerly, she worked in the Banking and Insurance industries for 19 years. Current endeavors include co-writing SAS White Papers, writing her first children's book, and developing content for the ICI website. Ronda has studied Transformational Leadership, Dynamic Teamwork, and Personal Growth since 1994 and is a *48 Days To The Work You Love* Coach and a *John Maxwell Team* Coach. She believes YOU are a gift that the world is waiting to receive. Her hope is to encourage you to reach your life, work, and leadership potential and to equip you to build an enduring legacy of inspiration, excellence, and honor.

Kirk Paul Lafler (*Founder/Senior Consultant, Software Intelligence Corporation*) has programmed in SAS since 1979. He is a SAS Certified Professional, provides IT Consulting Services, is a SAScommunity.org Emeritus Advisory Board member, and trains/mentors SAS users worldwide. Kirk has authored 5 books including *PROC SQL: Beyond the Basics Using SAS* (2nd Edition, SAS Institute 2013), has written over 500 papers, has been invited to speak/train at over 300 SAS international, regional, local, special-interest and in-house user group conferences/meetings, and has received 22 BEST Contributed Paper, Hands-On Workshop (HOW), and Poster Awards. His popular SAS Tips column *Kirk's Korner of Quick and Simple Tips* and his fun *SASword Puzzles* appear on various SAS websites and in several SAS User Group newsletters.

We invite you to share your valued comments with us:

Kent Phelps

Co-President, Senior Data Governance Analyst, Writer, Communicator, Teacher, and Coach
Illuminator Coaching, Inc.

E-mail: SASketeers@IlluminatorCoaching.com

Ronda Phelps

Co-President, Writer, Communicator, Teacher, and Coach
Illuminator Coaching, Inc.

E-mail: SASketeers@IlluminatorCoaching.com

Kirk Paul Lafler

Senior Consultant, Application Developer, Trainer, Mentor, and Author
Software Intelligence Corporation

E-mail: KirkLafler@cs.com

LinkedIn: <http://www.linkedin.com/in/KirkPaulLafler>

Twitter: @sasNerd

😊 **We Look Forward To Connecting With You In The Future** 😊

ACKNOWLEDGMENTS

We want to thank **Brian Varney** and **Swati Agarwal**, MWSUG 2013 *Beyond the Basics* Section Chairs, for graciously accepting our abstract and paper. We also want to express our gratitude to the MWSUG 2013 Conference Co-Chairs, **George J. Hurley** (Operations Chair) and **Matthew Karafa** (Academic Chair), the MWSUG Executive Committee, SAS Institute, and Conference Leaders for their diligent efforts in organizing and hosting this illuminating, empowering, and energizing conference. In addition, we offer our appreciation to **Denise Lyon**, Director Data Governance, for her inspiring leadership and empowering support in SAS; and to **Mike McCullough**, Senior Data Governance Analyst, for introducing Kent to SAS Enterprise Guide.

REFERENCES

Celko, Joe (2010), *Joe Celko's SQL for Smarties, Fourth Edition: Advanced SQL Programming (The Morgan Kaufmann Series in Data Management Systems)*; November 10, 2010; ISBN-10: 0123820227; ISBN-13: 978-0123820228.

Foley, Malachy J. (2005), *Merging vs. Joining: Comparing the DATA Step with SQL*, Proceedings of the 30th Annual SAS Users Group International (SUGI) 2005 Conference, University of North Carolina, Chapel Hill, NC, USA. http://www.scsug.org/SCSUGProceedings/2005/Foley_Merging%20vs%20Joining%20-%20184.pdf

Kent, Paul, *SQL Joins -- The Long and The Short of It*; SAS Institute Inc., Cary, NC, USA.

Lafler, Kirk Paul (2013), *PROC SQL: Beyond the Basics Using SAS, Second Edition*; SAS Press.
<http://support.sas.com/publishing/authors/lafler.html>

Lafler, Kirk Paul and Mira Shapiro (2013), *Point-and-Click Programming Using SAS® Enterprise Guide®*, NorthEast SAS Users Group (NESUG) 2013 Conference.

Lafler, Kirk Paul (2012), *Exploring DATA Step Merges and PROC SQL Joins*, Proceedings of the 6th Annual SAS Global Forum (SGF) 2012 Conference, Software Intelligence Corporation, Spring Valley, CA, USA.
<http://support.sas.com/resources/papers/proceedings12/251-2012.pdf>

Lafler, Kirk Paul (2012), *Exploring DATA Step Merges and PROC SQL Joins*, Proceedings of the 14th Annual Pharmaceutical SAS Users Group (PharmaSUG) 2012 Conference, Software Intelligence Corporation, Spring Valley, CA, USA. <http://pharmasug.org/proceedings/2012/TA/PharmaSUG-2012-TA02.pdf>

Lafler, Kirk Paul (2012), *You Could Be a SAS® Nerd If. . .*, Proceedings of the 23rd Annual MidWest SAS Users Group (MWSUG) 2012 Conference, Software Intelligence Corporation, Spring Valley, CA, USA.
<http://www.mwsug.org/proceedings/2012/S1/MWSUG-2012-S103.pdf>

Phelps, Kent ♥ Ronda Team Phelps and Kirk Paul Lafler (2013), *SAS® Commands PIPE and CALL EXECUTE; Dynamically Advancing From Strangers to Best Friends*, Proceedings of the 24th Annual MidWest SAS Users Group (MWSUG) 2013 Conference, Illuminator Coaching, Inc., Des Moines, IA, USA, and Software Intelligence Corporation, Spring Valley, CA, USA.

TRADEMARK CITATIONS

SAS and all other SAS Institute, Inc., product or service names are registered trademarks or trademarks of SAS Institute, Inc., in the USA and other countries. The symbol, ®, indicates USA registration. Other brand and product names are registered trademarks or trademarks of their respective companies.