MWSUG 2024 ~ Paper BL-103

Simplify Complex SAS® Logic through the Power of Bit Level Data Comparison

Kent & Ronda Team Phelps ~ *The SASketeers ~ All for SAS & SAS for All!*
Illuminator Coaching, Inc.

ABSTRACT

Business requirements are built on a foundation of business rules that guide the business logic in your SAS® programs. Business logic often becomes more complex than it needs to be ~ making it challenging to design simple and efficient programs. Our presentation highlights a powerful data transformation and analysis tool called *Bit Level Data Comparison*. This tool was designed in SAS and other languages and successfully used in a variety of industries.

Join us for a great opportunity to learn how to replace long lines of repetitive complex business logic with simple concatenations and SELECT WHEN statements. You will be empowered to break down and map business logic into *Bit Levels* and groups of flags, codes, values, and indicators in various combinations. This session on *Bit Level Data Comparison* will expand your toolkit and enable you to design programs that are more enjoyable and easier to code, read, update, and maintain.

INTRODUCTION

The tagline for SAS is *The Power To Know*® and your 'power to know' will greatly expand with your increasing ability to simplify complex SAS® logic through the power of *Bit Level Data Comparison*. Your **Power To Know** enables the **Power To Transform** which leads to the **Power To Build** more efficient programs ~ but these powers will quickly lose momentum if you do not learn how to design programs that are more enjoyable and easier to code, read, update, and maintain.

Three questions to ask yourself when designing your program:

- What parts of my program will have repetitive logic?
- What parts of my program will have complicated logic?
- What parts of my program will have key logic?

This presentation demonstrates the:

- **Power To Know** how to create *Bit Level Data Comparisons*.
- **Power To Transform** complex business rules, requirements, and logic into simplified code.
- **Power To Build** more efficient programs that are easier to code, read, update, and maintain.

THE POWER TO KNOW

Your power to know how to create *Bit Level Data Comparisons* begins with these definitions:

- **Bit ~** one piece of information needed to help build business logic.
- **Bit Level Data ~** combination of *Bits* needed to help build business logic.
- **Bit Level Data Comparison ~** comparison of *Bit Level Data* within the business logic.

We will apply these definitions to the two most commonly used *Truth Tables*, the *OR Logical Operator* and the *AND Logical Operator*.

Here is the standard mapping of the OR Logical Operator Truth Table:

X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

- X and Y are the two *Bits* used to build the business logic for Z.
- X and Y concatenated together are the *Bit Level Data* used to build the business logic for Z.
- When the *Bit Level Data Comparison* matches '00' ~ the business logic for Z resolves to '0'.
- When the *Bit Level Data Comparison* matches '01' ~ the business logic for Z resolves to '1'.
- When the *Bit Level Data Comparison* matches '10' ~ the business logic for Z resolves to '1'.
- When the *Bit Level Data Comparison* matches '11' ~ the business logic for Z resolves to '1'.
- Please note the repetitive resolutions of the last three business logic statements all resolving to '1' ~ and the key resolution of the first business logic statement being the only one resolving to '0'.

Here is the standard mapping of the AND Logical Operator Truth Table:

X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

- X and Y are the two *Bits* used to build the business logic for Z.
- X and Y concatenated together are the *Bit Level Data* used to build the business logic for Z.
- When the *Bit Level Data Comparison* matches '00' ~ the business logic for Z resolves to '0'.
- When the *Bit Level Data Comparison* matches '01' ~ the business logic for Z resolves to '0'.
- When the *Bit Level Data Comparison* matches '10' ~ the business logic for Z resolves to '0'.
- When the *Bit Level Data Comparison* matches '11' ~ the business logic for Z resolves to '1'.
- Please note the repetitive resolutions of the first three business logic statements ~ and the key resolution of the last business logic statement being the only one resolving to '1'.

THE POWER TO TRANSFORM

Your power to transform complex business rules, requirements, and logic into simplified code using the *OR Logical Operator* and the *AND Logical Operator* starts now.

Business Logic Scenario using the OR Logical Operator:

- A customer receives a 10% Discount if ***Policy Age*** is at least 3 years ~ or ~ they are ***Claim Free*** for at least 2 years.

Option 1

Here is the complete business logic before we apply the *Bit Level Data Comparison*:

```
/* Determine the Policy Age and if it is greater than or equal to 3 years */
IF INTCK('year',Original_Policy_Date,Policy_Renewal_Date,'continuous') >=3
OR
/* Determine if the policy is Claim Free for greater than or equal to 2 years when claim exists */
((Claim_Exists = 'Y' AND INTCK('year',Last_Claim_Date,Policy_Renewal_Date,'continuous') >=2)
OR
/* Determine if the policy is Claim Free for greater than or equal to 2 years when no claims */
(Claim_Exists = 'N' AND INTCK('year',Original_Policy_Date,Policy_Renewal_Date,'continuous') >=2))
THEN Discount _Percent = 10;
ELSE Discount_Percent = 0;
```

How to determine the *Discount Percent* using *Bit Level Data Comparison*:

Questions to ask yourself ~

What are the *Bits*?

- Policy Age
- Claim Free

How are the *Bits* populated?

- Policy Age
  - Yes/No flag
    - 'Y' for 3 years or older.
    - 'N' for less than 3 years.
  - Number of years ~ 2-digit integer with leading zero and no decimal.
- Claim Free
  - Yes/No flag
    - 'Y' for 2 years or older.
    - 'N' for less than 2 years.
  - Number of years ~ 2-digit integer with leading zero and no decimal.

Tip for Success

Only focus on one *Bit* at a time in order to simplify intermediate logic.

Option 2

- Name the first *Bit* ~ Policy_Age.
- Name the second *Bit* ~ Claim_Free.
- Set the **Policy_Age flag to 'Y'** if the *Policy Age* is greater than or equal to 3 years.

```
IF INTCK('year',Original_Policy_Date,Policy_Renewal_Date,'continuous') >=3
THEN Policy_Age = 'Y';
ELSE Policy_Age = 'N';
```

- Set the **Claim_Free flag to 'Y'** if the policy is *Claim Free* for greater than or equal to 2 years.

```
IF (Claim_Exists = 'Y' AND
    INTCK('year',Last_Claim_Date,Policy_Renewal_Date,'continuous') >=2)
OR
   (Claim_Exists = 'N' AND
    INTCK('year',Original_Policy_Date,Policy_Renewal_Date,'continuous') >=2)
THEN Claim_Free = 'Y';
ELSE Claim_Free = 'N';
```

- Concatenate the *Bits* as the *Bit Level Data* named **Discount**.

```
Discount = Policy_Age||Claim_Free;
```

- Create a *Bit Level Data Comparison* using a **SELECT Discount** with **four WHEN's**.

```
SELECT Discount;
  WHEN ('NN') Discount_Percent = 0;
  WHEN ('NY') Discount_Percent = 10;
  WHEN ('YN') Discount_Percent = 10;
  WHEN ('YY') Discount_Percent = 10;
END;
```

Option 3

- Using the same criteria as **Option 2**, you can reduce the **SELECT Discount** down to **one WHEN**.

```
SELECT Discount;
  WHEN ('NN') Discount_Percent = 0;
  OTHERWISE Discount_Percent = 10;
END;
```

Tip for Success

Create all of the results before reducing in order to ensure completeness.

Option 4

- Set **Policy_Age** to the *Policy Age* in years with a **leading zero**.

```
Policy_Age =
PUT(INTCK('year',Original_Policy_Date,Policy_Renewal_Date,'continuous'),Z2.);
```

- Set **Claim_Free** to the number of years *Claim Free* with a **leading zero**.

```
IF Claim_Exists = 'Y'
THEN Claim_Free = PUT(INTCK('year',Last_Claim_Date,Policy_Renewal_Date,'continuous'),Z2);
ELSE Claim_Free = PUT(INTCK('year',Original_Policy_Date,Policy_Renewal_Date,'continuous'),Z2);
```

- Concatenate the *Bits* as the *Bit Level Data* named **Discount**.

```
Discount = Policy_Age||Claim_Free;
```

- Create a *Bit Level Data Comparison* using a **SELECT** with **two WHEN's**.

```
SELECT;
  WHEN (Discount >= '0300') Discount_Percent = 10;
  WHEN (Substring(Discount,3,2) >= '02') Discount_Percent = 10;
  OTHERWISE Discount_Percent = 0;
END;
```

Tip for Success

Initially code all four ways in order to determine which option you prefer to use.

At this point, you are asking yourself, "Which option am I supposed to use?"

- The simple answer is the option you ultimately decide to code. Each of these coding options will work as long as you code the *Bit Level Data Comparison* correctly. Next, we will explore the other most commonly used *Truth Table* **~** the *AND Logical Operator*.

Quick Review ~ your power to know began with the following definitions:

- **Bit ~** one piece of information needed to help build business logic.
- **Bit Level Data ~** combination of *Bits* needed to help build business logic.
- **Bit Level Data Comparison ~** comparison of *Bit Level Data* within the business logic.

Business Logic Scenario using the AND Logical Operator:

- A customer receives a 10% Discount if they are ***Paperless ~*** and ***~ Claim Free*** for at least 2 years.

Option 1

Here is the complete business logic before we apply the *Bit Level Data Comparison*:

```
/* Determine if the Account is Paperless – Account_Comm values are M–Regular Mail, E–Email, W-Web */
IF Account_Comm <> 'M'
AND
/* Determine if the policy is Claim Free for greater than or equal to 2 years when claim exists */
((Claim_Exists = 'Y' AND INTCK('year',Last_Claim_Date,Policy_Renewal_Date,'continuous') >=2)
OR
/* Determine if the policy is Claim Free for greater than or equal to 2 years when no claims */
(Claim_Exists = 'N' AND INTCK('year',Original_Policy_Date,Policy_Renewal_Date,'continuous') >=2))
THEN Discount _Percent = 10;
ELSE Discount_Percent = 0;
```

How to determine the *Discount Percent* using *Bit Level Data Comparison*:

Questions to ask yourself ~

What are the *Bits*?

- Paperless
- Claim Free

How are the *Bits* populated?

- Paperless
  - Yes/No flag
    - 'Y' for Paperless.
    - 'N' for not Paperless.
- Claim Free
  - Yes/No flag
    - 'Y' for 2 years or older.
    - 'N' for less than 2 years.
  - Number of years ~ 2-digit integer with leading zero and no decimal.

Tip for Success ~ Reminder

Only focus on one *Bit* at a time in order to simplify intermediate logic.

Option 2

- Name the first *Bit* ~ Paperless.
- Name the second *Bit* ~ Claim_Free.
- Set the **Paperless flag to 'Y'** if the *Policy Communication* is NOT set to **'M'**.

```
IF Policy_Comm = 'M'
THEN Paperless = 'N';
ELSE Paperless = 'Y';
```

- Set the **Claim_Free flag to 'Y'** if the policy is *Claim Free* for greater than or equal to 2 years.

```
IF (Claim_Exists = 'Y' AND
    INTCK('year',Last_Claim_Date,Policy_Renewal_Date,'continuous') >=2)
OR
   (Claim_Exists = 'N' AND
    INTCK('year',Original_Policy_Date,Policy_Renewal_Date,'continuous') >=2)
THEN Claim_Free = 'Y';
ELSE Claim_Free = 'N';
```

- Concatenate the *Bits* as the *Bit Level Data* named **Discount**.

```
Discount = Paperless||Claim_Free;
```

- Create a *Bit Level Data Comparison* using a **SELECT Discount** with **four WHEN's**.

```
SELECT Discount;
  WHEN ('NN') Discount_Percent = 0;
  WHEN ('NY') Discount_Percent = 0;
  WHEN ('YN') Discount_Percent = 0;
  WHEN ('YY') Discount_Percent = 10;
END;
```

Option 3

- Using the same criteria as **Option 2**, you can reduce the **SELECT Discount** down to **one WHEN**.

```
SELECT Discount;
  WHEN ('YY') Discount_Percent = 10;
  OTHERWISE Discount_Percent = 0;
END;
```

Tip for Success ~ Reminder

Create all of the results before reducing in order to ensure completeness.

Option 4

- Set the **Paperless flag to 'Y'** if the *Policy Communication* is NOT set to **'M'**.

```
IF Policy_Comm = 'M'
THEN Paperless = 'N';
ELSE Paperless = 'Y';
```

- Set **Claim_Free** to the number of years *Claim Free* with a **leading zero**.

```
IF Claim_Exists = 'Y'
THEN Claim_Free = PUT(INTCK('year',Last_Claim_Date,Policy_Renewal_Date,'continuous'),Z2);
ELSE Claim_Free = PUT(INTCK('year',Original_Policy_Date,Policy_Renewal_Date,'continuous'),Z2);
```

- Concatenate the *Bits* as the *Bit Level Data* named **Discount**.

```
Discount = Policy_Comm||Claim_Free;
```

- Create a *Bit Level Data Comparison* using a **SELECT** with **two WHEN's**.

```
SELECT;
  WHEN (Discount >= 'Y02') Discount_Percent = 10;
  OTHERWISE Discount_Percent = 0;
END;
```

THE POWER TO BUILD

Now we will use a business logic scenario combining the *OR* and the *AND Logical Operators* to greatly expand your power to build more efficient programs that are easier to code, read, update, and maintain.

Business Logic Scenario combining the OR and the AND Logical Operators:

- *Policy Age* of 10 years is *Level Gold* **~** 5 years is *Level Silver* **~** 2 years is *Level Bronze*.
- *Tier One* is *Claim Free* for 2 years **~** has *Multiple Policies* **~** or **~** has a *Security System*.
- *Tier Two* is not *Claim Free* for 2 years **~** has a *Single Policy* **~** and **~** does not have a *Security System*.
- *Level Gold Tier One* has a 30% Discount **~** *Level Gold Tier Two* has a 20% Discount.
- *Level Silver Tier One* has a 20% Discount **~** *Level Silver Tier Two* has a 10% Discount.
- *Level Bronze Tier One* has a 10% Discount **~** *Level Bronze Tier Two* has a 0% Discount.

Here is a mapping of this business logic presented in a table:

Level	Tier	Policy Age	Claim Free 2 years	Multiple Policies	Security System	Discount
Gold	One	10	N	N	Y	30%
Gold	One	10	N	Y	N	30%
Gold	One	10	N	Y	Y	30%
Gold	One	10	Y	N	N	30%
Gold	One	10	Y	N	Y	30%
Gold	One	10	Y	Y	N	30%
Gold	One	10	Y	Y	Y	30%
Gold	Two	10	N	N	N	20%
Silver	One	5	N	N	Y	20%
Silver	One	5	N	Y	N	20%
Silver	One	5	N	Y	Y	20%
Silver	One	5	Y	N	N	20%
Silver	One	5	Y	N	Y	20%
Silver	One	5	Y	Y	N	20%
Silver	One	5	Y	Y	Y	20%
Silver	Two	5	N	N	N	10%
Bronze	One	2	N	N	Y	10%
Bronze	One	2	N	Y	N	10%
Bronze	One	2	N	Y	Y	10%
Bronze	One	2	Y	N	N	10%
Bronze	One	2	Y	N	Y	10%
Bronze	One	2	Y	Y	N	10%
Bronze	One	2	Y	Y	Y	10%
Bronze	Two	2	N	N	N	0%

As we continue together on our exciting efficiency journey, we will replace more complex multi-layered and time-consuming inefficient code with simpler and efficient *Bit Level Data Comparisons*. Please note we will be starting out slowly at first ~ but take heart, we will be gaining momentum along the way and the efficiency destination we are driving to will be worth the trip! ☺☺

Option 1

Here is the complete business logic for *Level Gold* before we apply the *Bit Level Data Comparison*:

```
/* Determine the Policy Age and if it is greater than or equal to 10 years */
IF INTCK('year',Original_Policy_Date,Policy_Renewal_Date,'continuous') >=10
AND
/* Determine if the policy is Claim Free for greater than or equal to 2 years when claim exists */
(
((Claim_Exists = 'Y' AND INTCK('year',Last_Claim_Date,Policy_Renewal_Date,'continuous') >=2)
OR
/* Determine if the policy is Claim Free for greater than or equal to 2 years when no claims */
(Claim_Exists = 'N' AND INTCK('year',Original_Policy_Date,Policy_Renewal_Date,'continuous') >=2))
OR
Multiple_Policy = 'Y'
OR
Security_System = 'Y'
)
THEN DO;
        Level = 'Gold';
        Tier = 'One';
        Discount _Percent = 30;
    END;
ELSE;
/* Determine the Policy Age and if it is greater than or equal to 10 years */
IF INTCK('year',Original_Policy_Date,Policy_Renewal_Date,'continuous') >=10
AND
/* Determine if the policy is Claim Free for less than 2 years when claim exists */
(
(Claim_Exists = 'Y' AND INTCK('year',Last_Claim_Date,Policy_Renewal_Date,'continuous') < 2)
AND
Multiple_Policy = 'N'
AND
Security_System = 'N'
)
THEN DO;
        Level = 'Gold';
        Tier = 'Two';
        Discount _Percent = 20;
    END;
```

Are we there yet?

Please note that *Level Silver* and *Level Bronze* would be coded in a similar manner.

How to determine the *Discount Percent* using *Bit Level Data Comparison*:

Questions to ask yourself ~

What are the *Bits*?

- Policy Age
- Claim Free
- Multiple Policies
- Security System

How are the *Bits* populated?

- Policy Age
  - '10' for 10 years or older.
  - '05' for 5 years or older.
  - '02' for 2 years or older.
- Claim Free
  - Yes/No flag
    - 'Y' for 2 years or older.
    - 'N' for less than 2 years.
- Multiple Policies
  - Yes/No flag
    - 'Y' for Multiple Policies.
    - 'N' for Single Policy.
- Security System
  - Yes/No flag
    - 'Y' for Security System.
    - 'N' for No Security System.

Tip for Success ~ Reminder

Only focus on one *Bit* at a time in order to simplify intermediate logic.

Option 2

- Set **Policy_Age to '10', '05', or '02'** depending upon the *Policy Age*.

```
P_Age = IF INTCK('year',Original_Policy_Date,Policy_Renewal_Date,'continuous')
SELECT;
  WHEN (P_Age >= 10 THEN Policy_Age = '10';
  WHEN (P_Age >= 5   THEN Policy_Age = '05';
  WHEN (P_Age >= 2   THEN Policy_Age = '02';
  OTHERWISE Policy_Age = '00';
END;
```

- Set the **Claim_Free flag to 'Y'** if the policy is *Claim Free* for greater than or equal to 2 years.

```
IF (Claim_Exists = 'Y' AND
     INTCK('year',Last_Claim_Date,Policy_Renewal_Date,'continuous') >=2)
OR
   (Claim_Exists = 'N' AND
     INTCK('year',Original_Policy_Date,Policy_Renewal_Date,'continuous') >=2)
THEN Claim_Free = 'Y';
ELSE Claim_Free = 'N';
```

- Set the **Multiple_Policies flag to 'Y'** if Policy_Count is greater than 1.

```
IF Policy_Count > 1
THEN Multiple_Policies = 'Y';
ELSE Multiple_Policies = 'N';
```

- Set the **Security_System flag to 'Y'** if Security_Type values are A–Alarm, D–Dog, G–Guard, M–Moat.

```
IF Security_Type = ''
THEN Security_System = 'N';
ELSE Security_System = 'Y';
```

- Concatenate the *Bits* as the *Bit Level Data* named **Discount**.

```
Discount = Policy_Age||Claim_Free||Multiple_Policies||Security_System;
```

- Create a *Bit Level Data Comparison* using a **SELECT Discount** with **twenty-four WHEN's**.

```
SELECT Discount;
  WHEN ('10NNY') DO; Level = 'Gold'; Tier = 'One'; Discount_Percent = 30; END;
  WHEN ('10NYN') DO; Level = 'Gold'; Tier = 'One'; Discount_Percent = 30; END;
  WHEN ('10NYY') DO; Level = 'Gold'; Tier = 'One'; Discount_Percent = 30; END;
  WHEN ('10YNN') DO; Level = 'Gold'; Tier = 'One'; Discount_Percent = 30; END;
  WHEN ('10YNY') DO; Level = 'Gold'; Tier = 'One'; Discount_Percent = 30; END;
  WHEN ('10YYN') DO; Level = 'Gold'; Tier = 'One'; Discount_Percent = 30; END;
  WHEN ('10YYY') DO; Level = 'Gold'; Tier = 'One'; Discount_Percent = 30; END;
  WHEN ('10NNN') DO; Level = 'Gold'; Tier = 'Two'; Discount_Percent = 20; END;
```

```
   WHEN ('05NNY') DO; Level = 'Silver'; Tier = 'One'; Discount_Percent = 20; END;
   WHEN ('05NYN') DO; Level = 'Silver'; Tier = 'One'; Discount_Percent = 20; END;
   WHEN ('05NYY') DO; Level = 'Silver'; Tier = 'One'; Discount_Percent = 20; END;
   WHEN ('05YNN') DO; Level = 'Silver'; Tier = 'One'; Discount_Percent = 20; END;
   WHEN ('05YNY') DO; Level = 'Silver'; Tier = 'One'; Discount_Percent = 20; END;
   WHEN ('05YYN') DO; Level = 'Silver'; Tier = 'One'; Discount_Percent = 20; END;
   WHEN ('05YYY') DO; Level = 'Silver'; Tier = 'One'; Discount_Percent = 20; END;
   WHEN ('05NNN') DO; Level = 'Silver'; Tier = 'Two'; Discount_Percent = 10; END;
   WHEN ('02NNY') DO; Level = 'Bronze'; Tier = 'One'; Discount_Percent = 10; END;
   WHEN ('02NYN') DO; Level = 'Bronze'; Tier = 'One'; Discount_Percent = 10; END;
   WHEN ('02NYY') DO; Level = 'Bronze'; Tier = 'One'; Discount_Percent = 10; END;
   WHEN ('02YNN') DO; Level = 'Bronze'; Tier = 'One'; Discount_Percent = 10; END;
   WHEN ('02YNY') DO; Level = 'Bronze'; Tier = 'One'; Discount_Percent = 10; END;
   WHEN ('02YYN') DO; Level = 'Bronze'; Tier = 'One'; Discount_Percent = 10; END;
   WHEN ('02YYY') DO; Level = 'Bronze; Tier = 'One'; Discount_Percent = 10; END;
   WHEN ('02NNN') DO; Level = 'Bronze'; Tier = 'Two'; Discount_Percent = 0; END;
   OTHERWISE DO; Level = ''; Tier = ''; Discount_Percent = 0; END;
END;
```

Tip for Success ~ Reminder

Create all of the results before reducing in order to ensure completeness.

Option 3

- Using the same criteria as **Option 2**, you can reduce the **SELECT Discount** down to **six WHEN's**.

```
SELECT;
   WHEN (Discount = '10NNN') DO; Level = 'Gold'; Tier = 'Two'; Discount_Percent = 20; END;
   WHEN (Discount > '10NNN') DO; Level = 'Gold'; Tier = 'One'; Discount_Percent = 30; END;
   WHEN (Discount = '05NNN') DO; Level = 'Silver'; Tier = 'Two'; Discount_Percent = 10; END;
   WHEN (Discount > '05NNN') DO; Level = 'Silver'; Tier = 'One'; Discount_Percent = 20; END;
   WHEN (Discount = '02NNN') DO; Level = 'Bronze'; Tier = 'Two'; Discount_Percent = 0; END;
   WHEN (Discount > '02NNN') DO; Level = 'Bronze'; Tier = 'One'; Discount_Percent = 10; END;
   OTHERWISE DO; Level = ''; Tier = ''; Discount_Percent = 0; END;
END;
```

Time to stop for some hot chocolate to celebrate our journey together ☺☺!

CONCLUSION

Your *Power To Know* how to create *Bit Level Data Comparisons* enables the *Power To Transform* complex business rules, requirements, and logic into simplified code which leads to the *Power To Build* more efficient programs that are easier to code, read, update, and maintain.

It's not what the world holds for you,
it's what YOU bring to it!
Anne of Green Gables

It's not what the SAS World holds for you, it's what YOU bring to it! As a SAS professional, you are inquisitive, research oriented, and solution driven. Your optimistic and tenacious desire to design a quality program fuels your thoroughness and attention to detail. When you are in your SAS zone, you are relentless in your pursuit to maximize your analysis, development, and programming.

Don't be a reservoir, be a river. **John C. Maxwell**

SAS programming is *Mind Art* **~** a creative realm where each of you is an artist. Continue to develop and build on your many skills and talents. Keep looking for new ways to share your God-given abilities and ideas. Don't be a reservoir of SAS knowledge **~** be a refreshing river flowing outward to empower those around you. Always remember, your contributions make a positive impact in the world!

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

Plan on coming back to the *MWSUG Regional Conference* next year to shed some light on the exciting things you are learning. All of us are on the SAS journey with you, and we look forward to your teaching sessions in the future. As we conclude, we want to introduce you to our SAS mascot, *Smiley*, who represents the joy that each of us experience as we find better ways to accomplish grand and noble deeds using SAS. The three of us hope we have enriched your SAS knowledge.

Thank you for sharing part of your SAS journey with us ~
☺☺ Happy SAS Trails to you until we meet again!

ACKNOWLEDGMENTS

We want to thank *Misty Johnson*, the Academic Chair; *Dave Foster*, the Operations Chair; and *David Corliss*, the Business Leadership Section Chair; at the MWSUG 2024 Regional Conference for accepting our White Paper Presentation and for their dedicated service. We also want to express our appreciation to the *Executive Board*, *Conference Leaders*, and *SAS Institute* for their comprehensive efforts in organizing this exciting and illuminating learning experience. You inspire us to keep sharing what we are learning, and we hope to be a light of encouragement to you as well ~ *Your friends, Kent* & *Ronda Team Phelps*.

MEET THE AUTHORS

Writing is a permanent legacy.

John C. Maxwell

Kent **&** Ronda **Team** Phelps are *The **SAS**keteers: All for **SAS** and **SAS** for All!* They founded *Illuminator Coaching, Inc.*, as a platform to illuminate the ever-expanding wonders of SAS. Kent is a *Senior Data, Technical, and Business Analyst, Developer, Engineer, Programmer, Trainer, Coach, Consultant*, and *SAS® Certified Professional Programmer* who has happily programmed in SAS since 2007. He enjoys coaching, encouraging, and equipping you to fulfill your life, career, and leadership potential ~ as you build an enduring legacy of inspiration, excellence, and honor. Ronda is a *Writer and Coach* who previously served in the banking and insurance industries. She believes that YOU are a gift the world is waiting to receive. She enjoys coaching, encouraging, and equipping you to pursue your unique destiny ~ as you navigate your life journey with intentionality, fulfilling purpose, and enduring hope.

"Don't be a warehouse of knowledge ~
be a lighthouse guiding people from where they are to where they want to go."

Kent & Ronda Team Phelps

CONTACT INFORMATION

☺ **We invite you to share your questions and comments with us** ☺

Kent **&** Ronda **Team** Phelps ~ *The **SAS**keteers* ~ *All for **SAS** & **SAS** for All!*

E-mail: SASketeers@illuminatorcoaching.com

REFERENCES

(2024) Dictionary.com, Definition of 'Bit'. https://www.dictionary.com/browse/bit

(2024) Merriam-Webster Dictionary, Definition of 'Bit'. https://www.merriam-webster.com/dictionary/bit

(2024) SAS®. https://blogs.sas.com/content/iml/2017/05/15/intck-intnx-intervals-sas.html

(2024) SAS®. https://documentation.sas.com/doc/en/pgmsascdc/v_055/lepg/n1szmwbyyd88azn0ztn08d8d0iq4.htm

(2024) SAS®. https://sasexamplecode.com/add-leading-zeros-to-a-variable-in-sas/

(2024) Techopedia, Definition of 'Bit'. https://www.techopedia.com/definition/23954/bit

(2024) Phelps, Kent & Ronda Team, Training Course: *Automate Manual Excel and PDF Report Creation with Dynamic SAS® Code ~ Your Newest BFF (Best Friend Forever) in SAS®*; 31st Annual MidWest SAS® Users Group (MWSUG) 2024 Regional Conference in Milwaukee, WI; *The SASketeers ~ All for SAS & SAS for All!* ~ Illuminator Coaching, Inc., Des Moines, IA, USA.
https://www.mwsug.org/2024/training.html#sat07

(2024) Phelps, Kent & Ronda Team, Training Course: *The Joinless Join ~ The Impossible Dream Come True; Using SAS® Enterprise Guide®, PROC SQL, and DATA Step*; 31st Annual MidWest SAS® Users Group (MWSUG) 2024 Regional Conference in Milwaukee, WI; *The SASketeers ~ All for SAS & SAS for All!* ~ Illuminator Coaching, Inc., Des Moines, IA, USA. https://www.mwsug.org/2024/training.html#sat04

(2021) Phelps, Kent & Ronda Team, Livestream Presentation: *Joinless Join: The Impossible Dream Come True, Using SAS® Enterprise Guide®, PROC SQL, and DATA Step*; 29th Annual Southeast SAS® Users Group (SESUG) 2021 Regional Conference (*Virtual*); *The SASketeers ~ All for SAS & SAS for All!* ~ Illuminator Coaching, Inc., Des Moines, IA, USA.
https://www.lexjansen.com/sesug/2021/SESUG2021_Paper_96_Final_PDF.pdf

(2021) Phelps, Kent & Ronda Team, Video Presentation: *Base SAS® & SAS® Enterprise Guide® Automate Your SAS® World with Dynamic Code ~ Forwards & Backwards; Your Newest BFF (Best Friend Forever) in SAS®*; SAS® Global Forum (SGF) 2021 (*Virtual*); *The SASketeers ~ All for SAS & SAS for All!* ~ Illuminator Coaching, Inc., Des Moines, IA, USA.
https://communities.sas.com/t5/SAS-Global-Forum-Proceedings/Base-SAS-and-SAS-Enterprise-Guide-Automate-Your-SAS-World-With/ta-p/726370

(2020) Phelps, Kent & Ronda Team, Presentation: *Joinless Join: The Impossible Dream Come True, Using SAS® Enterprise Guide®, PROC SQL, and DATA Step*; SAS® Global Forum (SGF) 2020 in Washington, D.C., (Presentation was published but not given due to SGF changing to a reduced Virtual Forum); *The SASketeers ~ All for SAS & SAS for All!* ~ Illuminator Coaching, Inc., Des Moines, IA, USA.
https://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2020/5169-2020.pdf

(2019) Phelps, Kent & Ronda Team, Hands-On Workshop (HOW): *Base SAS® & SAS® Enterprise Guide® Automate Your SAS® World with Dynamic Code ~ Forwards & Backwards; Your Newest BFF (Best Friend Forever) in SAS®*; 30th Annual MidWest SAS® Users Group (MWSUG) 2019 Regional Conference in Chicago, IL; *The SASketeers ~ All for SAS & SAS for All!* ~ Illuminator Coaching, Inc., Des Moines, IA, USA; ***Awarded Best Hands-On Workshop***. http://www.mwsug.org/proceedings/2019/HW/MWSUG-2019-HW-086.pdf

(2018) Phelps, Kent & Ronda Team, Hands-On Workshop (HOW): *The Joinless Join ~ The Impossible Dream Come True; Expand the Power of Base SAS® and SAS® Enterprise Guide® in a New Way*; 29th Annual MidWest SAS® Users Group (MWSUG) 2018 Regional Conference in Indianapolis, IN; *The SASketeers ~ All for SAS & SAS for All!* ~ Illuminator Coaching, Inc., Des Moines, IA, USA.
http://www.mwsug.org/proceedings/2018/HW/MWSUG-2018-HW-98.pdf

(2018) Phelps, Kent & Ronda Team, Advanced Presentation: *Base SAS® and SAS® Enterprise Guide®: Automate Your SAS® World with Dynamic Code ~ Your Newest BFF (Best Friend Forever) in SAS®*; SAS® Global Forum (SGF) 2018 in Denver, CO; *The SASketeers ~ All for SAS & SAS for All!* ~ Illuminator Coaching, Inc., Des Moines, IA, USA.
http://www.sas.com/content/dam/SAS/support/en/sas-global-forum-proceedings/2018/2857-2018.pdf

(2017) Phelps, Kent & Ronda Team, Hands-On Workshop (HOW): *Base SAS® and SAS® Enterprise Guide® ~ Automate Your SAS® World with Dynamic Code; Your Newest BFF (Best Friend Forever) in SAS®*; 28th Annual MidWest SAS® Users Group (MWSUG) 2017 Regional Conference in St. Louis, MO; *The SASketeers ~ All for SAS & SAS for All!* ~ Illuminator Coaching, Inc., Des Moines, IA, USA.
http://www.mwsug.org/proceedings/2017/HW/MWSUG-2017-HW03.pdf

(2016) Phelps, Kent & Ronda Team, Hands-On Workshop (HOW): *The Joinless Join ~ The Impossible Dream Come True; Expand the Power of Base SAS® and SAS® Enterprise Guide® in a New Way*; 27th Annual MidWest SAS® Users Group (MWSUG) 2016 Regional Conference in Cincinnati, OH; *The SASketeers ~ All for SAS & SAS for All!* ~ Illuminator Coaching, Inc., Des Moines, IA, USA.
http://www.mwsug.org/proceedings/2016/HW/MWSUG-2016-HW03.pdf

(2016) Phelps, Kent & Ronda Team, Presentation: *Base SAS® and SAS® Enterprise Guide® ~ Automate Your SAS® World with Dynamic Code; Your Newest BFF (Best Friend Forever) in SAS®*; 27th Annual MidWest SAS® Users Group (MWSUG) 2016 Regional Conference in Cincinnati, OH; *The SASketeers ~ All for SAS & SAS for All!* ~ Illuminator Coaching, Inc., Des Moines, IA, USA.
http://www.mwsug.org/proceedings/2016/TT/MWSUG-2016-TT11.pdf

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.

DISCLAIMER

We have endeavored to provide accurate and beneficial information in this *SAS® Conference Training Course / Hands-On Workshop / Presentation / White Paper*. This information is provided in *'Good Faith'* and *'As Is'* without any kind of warranty, either expressed or implied. Recipients acknowledge and agree that we and/or our company are/is not, and never will be, liable for any problems and/or damages whatsoever which may arise from the recipient's use of the information in this paper. Please refer to your Operating System (e.g., UNIX, Microsoft Windows, or IBM z/OS) Manual, Installation Configuration, and/or in-house Technical Support for further guidance in how to create the SAS® code presented.

Taking time to reach out to help and encourage one another on the journey of life makes each day more worthwhile and enjoyable for everyone ~

Make Each Day Count!