Mit SAS Formaten Programme steuern

Daniel Schulte
viadee Unternehmensberatung AG
Anton-Bruchausen-Straße 8
48147 Münster
daniel.schulte@viadee.de

Zusammenfassung

Mit Formaten können aus den zwei nativen SAS Datentypen (Numerisch und Zeichenketten) sehr unterschiedliche Ausgaben erzeugt werden. Diese Formate können aber zum einen selbst erzeugt werden und dann zum anderen auch für die Programmsteuerung genutzt werden.

Schlüsselwörter: Formate, Steuerung, SAS Base

1 Grundlagen zu SAS Formaten

Schaut man sich andere Datenbanksysteme an, ist man dort mit einer Reihe von unterschiedlichsten Datentypen konfrontiert und muss sich dort für die Verwendung den besten herauspicken. Bei SAS (Foundation) ist die Wahl auf Numerisch und Zeichenketten beschränkt. Lediglich die Länge ist bei der Deklaration von Variablen und damit Feldern in einer SAS Tabelle ein notweniger Parameter.
Möchte man jetzt aber zum Beispiel mit Datumswerten arbeiten, wendet SAS folgenden Trick an: Ein Datumswert wird als Anzahl Tage vom 01. Januar 1960 als numerischer Wert gespeichert. Um dann ein Datum auszugeben, wird ein Format genutzt, um die Zahl 21980 als 06.März 2020 auszugeben. Die gleiche Zahl kann aber auch als Wochentag (Freitag) oder ganz anders ausgegeben werden. Hier kommt es lediglich auf das verwendete SAS Format an.
Die Formate können entweder im Programmcode für eine bestimmte Operation genutzt werden oder als Metainformation an eine Variable angeheftet werden.
Die Formate werden aber lediglich für die Ausgabe verwendet. Die Speicherung in der Tabelle bzw. im PDV erfolgt weiterhin über den nativen Datentyp.

Hinweis:
Neben den in diesem Beitrag beschriebenen Ausgabe Formaten gibt es auch noch Eingabe Formate. Diese werden von SAS für Einleseoperationen verwendet. Auf diese Formate wird hier nicht eingegangen!

Auch ist die Ausgabebreite im Alltag ein wichtiger Parameter an den Formaten. Die hier genutzten Codebeispiele sind zur Darstellungszwecken vereinfacht!

1.1 Mitgelieferte SAS Formate

SAS liefert direkt eine ganze Menge an Formaten mit. Diese können in drei Kategorien eingeteilt werden:

- Charater
- Date & Time
- Numeric

1.1.1 Character Formate

Character Formate dienen zur Manipulation von Zeichenketten. Man kann sie durch ein vorangestelltes $ Zeichen identifizieren und mit Zeichenketten anwenden. Ein Beispiel wäre z.B. das $UPCASE Format. Damit lässt sich eine Zeichenkette mit Groß- und Kleinschreibung für die Ausgabe in Großbuchstaben umwandeln.

1.1.2 Date & Time Formate

Datums und Uhrzeit Formate nutzten die SAS Definition eines Datums (Tage zwischen dem 01. Januar 1960 und dem jeweiligen Tag) bzw. einer Uhrzeit (Sekunden seit Mitternacht), um dann über die Formate eine für Menschen lesbare Ausgabe daraus zu erzeugen.
Hier sind auch lokale Unterschiede verfügbar:

- ddmmyy10. erzeugt aus 21980 den 06.03.2020
- mmddyy10. erzeugt aus 21980 den 03.06.2020

Für internationale Ausgaben sollte auf den üblichen Datumsstandard geachtet werden und ggf. mit notiert werden.

1.1.3 Numeric Formate

Über Numeric Formate können Zahlen andersartig ausgegeben werden. Das Standard Numerische Format W.d kann aber auch genutzt werden, um die nativen Inhalte einer Variablen auszugeben. Wenn eine Variable z.B. ein Datumsformat in den Metadaten hinterlegt hat, kann so der reine nummerische Wert ausgegeben werden.

Die Angaben W und D spiegeln bei Formaten die Ausgabebreite (W = width) und Anzahl Dezimalstellen nach dem Komma wider.

Mit W.d ist z.B. 8.2 gemeint 8 Stellen insgesamt und 2 Nachkommastellen.

2 Erstellen eigener Formate

Um eigene Formate zu erzeugen, liefert SAS die Prozedur PROC FORMAT mit. Formate, die so erzeugt werden, legt das SAS System standardmäßig in einem SAS Katalog (formats) in der Bibliothek WORK ab und sie können in der Session dann direkt genutzt

werden, ohne den Suchpfad anzupassen. Wichtig hierbei: wie alles im WORK, wird der Bereich geleert und Formate gehen dann verloren. Wird das Format in der nächsten Session dann ohne erneutes Anlegen angesprochen, kommt es zu Fehlermeldung.

Formate lassen sich aber auch in eine permanente Bibliothek generieren. Dafür muss lediglich der Parameter LIB=<AUSGABE LIB> im PROC FORMAT hinzugefügt werden. So generierte Formate liegen allerdings damit nicht mehr im Suchpfad von SAS und es kommt beim Versuch, das Format zu nutzen zu einer Fehlermeldung. Den Suchpfad kann man mit der Option FMTSEARCH anpassen:

OPTIONS FMTSEARCH=(<AUSGABE LIB>);

Bei Bedarf können hier auch mehrere Bibliotheksnamen hintereinander verkettet werden. Formate können so zentral bereitgestellt werden und User müssen lediglich das Verzeichnis per Libname Statement allokieren und den Suchpfad erweitern.

Bei individuellen Formaten kann man drei Vorgehensweisen nutzen

- Werte(bereich) > Ausgaben
- Ausgabemuster
- Tabellen als Eingaben

2.1 Werte(bereich) > Ausgaben

Hierbei werden einzelne Werte oder Wertebereiche einem Ausgabewert zugeordnet. Ein typisches Beispiel ist eine Klassenbildung, in der es darum geht, Wertebereiche zu definieren und dann mit den Klassen weiter zu arbeiten

```
proc format lib=LIB_FMT;
    value altersklasse
       low - 0    ='invalid'
       1   - 11   ='below 12 years'
       12  - 14   ='between 12 and 14'
       15  - 17   ='over 15 years'
       18  - high ='invalid'
       ;
run;
```

Hierbei sollte beachtet werden, dass außerhalb des üblichen Bereichs liegende Werte ebenfalls erfasst werden. Mit „low“ und „high“ können diese bis in den niedrigst- bzw. höchstmöglichen Wert abgegrenzt werden.

2.2 Ausgabemuster

Bei Ausgabemustern kann man Zahlen in einer gewünschten Darstellungsform ausgeben. Die Ziffer 0 entspricht dabei einer optionalen und die 9 einer forcierten Ausgabe.

```
proc format lib=LIB_FMT;
    picture  Gewicht
      low - <100    ='99.9'
      100 - <2000   ='0999'
      2000- high    ='invalid'
      ;
run;
```

Über dieses Beispiel lassen sich kleine Werte (unter 100) mit einer Nachkommastelle und Werte zwischen 100 und 2000 ohne darstellen.

Das Ausgabemuster kann darüber hinaus auch mit einem Multiplier versehen werden.

```
proc format lib=LIB_FMT;
    picture lbs2kg (round)
       low - high  = '0099.9' (multiplier=4.5359);
run;
```

Dabei ist jedoch zu beachten, dass sich das Muster ggf. in der Positionierung der Stellen verschiebt.

2.3 Tabellen als Eingaben

Für die Erzeugung von Formaten können auch Steuertabellen genutzt werden. Der Mindestumfang einer solchen Tabelle sind die Felder:

- FMTNAME für den Namen des Formats
- START Eingangswert
- LABEL Ausgabewert

Um einen Wertebereich abzubilden, ist zusätzlich das Feld END notwendig.

	FMTNAME	START	END	LABEL
1	Alterskls	LOW	11	unter 12
2	Alterskls	12	14	Zwischen 12 und 14
3	Alterskls	15	HIGH	älter 14

Abbildung 1: Beispiel einer Steuertabelle

Der Vorteil, Formate aus Steuertabelle(n) zu erzeugen, liegt vor allem in der Flexibilität. Diese Tabellen können außerhalb einer Softwarelieferung bereitgestellt werden und ermöglichen darüber einen dynamischen Umgang mit Parametern.

3 Programmsteuerung über Formate

Mit den selbst erzeugten Formaten hat man neben einer Darstellungsoption, auch einer Steuerfunktion. Wird das Label nicht in einen Fließtext, sondern in Schalter (1 oder 0) gewandelt, kann man diese Information für Filter oder andere Steuerungen verwenden.

Bei klassischen Bedingungen, in welchen die Konditionen hart im Quelltext hinterlegt sind, muss man für Änderungen sicherstellen, dass diese

- Konsistent — Alle relevanten Stellen wurden modifiziert
- Getestet — Das Programm ist syntaktisch einwandfrei
- Nachvollziehbar — Andere Entwickler / Tester müssen die Änderung nachvollziehen können

sind.

Wenn die Bedingung in dem folgenden Beispiel angepasst werden:

```
data work.Ausgabe1 (keep= Name age) ;
    set sashelp.class;
    if age = 13;
run;
```

um nicht nur die Einträge mit dem Alter =13 auszugeben, sondern auch mit dem Alter = 11, ist eine Umformulierung des IF Statements in beispielsweise

- `age = 13 or age = 11`
- `age in (13,11)`

notwendig. Wird der Wahrheitswert nun über ein Format abgebildet, kann dieses erweitert werden, und der Programmcode bleibt davon unberührt.

	FMTNAME	START	LABEL
1	SWALTER	13	1
2	SWALTER	OTHER	0

Abbildung 2: Steuertabelle mit Schalter

Wir die o.g. Tabelle als Eingabe für PROC Format genutzt, wird das Format SWALTER erzeugt, welches dann als Steuerkriterium verwendet werden kann.

```
data work.Ausgabe2 (keep= Name age) ;
    set sashelp.class;
    if put(age,SWALTER.)="1";
run;
```

	Name	Age
1	Alice	13
2	Barbara	13
3	Jeffrey	13

Abbildung 3: Ausgabe aus DataStep

Wird die Steuertabelle nun erweitert und wird das Format neu generiert(!),

	FMTNAME	START	LABEL
1	SWALTER	11	1
2	SWALTER	13	1
3	SWALTER	OTHER	0

Abbildung 4: modifizierte Steuertabelle mit Schalter

```
data work.Ausgabe2 (keep= Name age) ;
    set sashelp.class;
    if put(age,SWalter.)="1";
run;
```

erzeugt das gleiche Programm einen anderen Output:

	Name	Age
1	Alice	13
2	Barbara	13
3	Jeffrey	13
4	Joyce	11
5	Thomas	11

Abbildung 5: Ausgabe aus DataStep mit neu generiertem Format

Gleiches kann man auch im SQL anwenden:

```
select Name, Age
  from sashelp.class
    where put(age,SWAlter.)="1"
;
```

Da über diese Syntax komplexere Zusammenhänge einfacher dargestellt werden können, ist der Programmcode für Außenstehende deutlich besser zu verstehen.

Wird konsequent über ein Format gesteuert, sind mit einer Anpassung der Formattabelle und der Generierung des Formats alle Programmstellen mit der neuen Bedingung ausgestattet.

Da es sich bei der Änderung nur um die Anpassung an der Steuertabelle handelt, muss nur diese fachlich und technisch geprüft werden. Der Aufwand für Tests und Qualitätssicherung sinkt.