Lehr- und Handbücher der Statistik

Herausgegeben von
Universitätsprofessor Dr. Rainer Schlittgen

Bisher erschienene Werke:

Caspary/Wichmann, Lineare Modelle
Chatterjee/Price (Übers. Lorenzen), Praxis der Regressionsanalyse, 2. Auflage
Degen/Lorscheid, Statistik-Aufgabensammlung
Harvey (Übers. Untiedt), Ökonometrische Analyse von Zeitreihen, 2. Auflage
Harvey (Übers. Untiedt), Zeitreihenmodelle, 2. Auflage
Heiler/Michels, Deskriptive und Explorative Datenanalyse
Oerthel/Tuschl, Statistische Datenanalyse mit dem Programmpaket SAS
Pokropp, Lineare Regression und Varianzanalyse
Rinne, Wirtschafts- und Bevölkerungsstatistik
Schlittgen, Statistik, 5. Auflage
Schlittgen/Streitberg, Zeitreihenanalyse, 5. Auflage

Statistische Datenanalyse mit dem Programmpaket SAS

Von
Diplom-Kaufmann
Frank Oerthel
Universität Passau
und
Diplom-Kaufmann
Stefan Tuschl
Universität Passau

R. Oldenbourg Verlag München Wien

Adresse der Autoren:

Universität Passau
Lehrstuhl für Statistik
Dipl.-Kfm. Frank Oerthel
Dipl.-Kfm. Stefan Tuschl
Innstraße 27
94032 Passau

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Oerthel, Frank:
Statistische Datenanalyse mit dem Programmpaket SAS / von
Frank Oerthel und Stefan Tuschl. - München ; Wien :
Oldenbourg, 1995
(Lehr- und Handbücher der Statistik)
ISBN 3-486-23349-1
NE: Tuschl, Stefan:

Gesamtherstellung: R. Oldenbourg Graphische Betriebe GmbH, München

ISBN 3-486-23349-1

Inhaltsverzeichnis

Vorwort

In nahezu jeder Wissenschaft stellt sich dem Forschenden die Aufgabe, eine Vielzahl von Daten statistisch auszuwerten. Im Zeitalter der elektronischen Datenverarbeitung ist diese Aufgabe durch den Einsatz von Rechnern in erheblich kürzerer Zeit zu bewältigen als früher. Auch läßt sich dadurch der Umfang des zu analysierenden Datenmaterials erheblich ausweiten. Datenmengen, deren Auswertung "von Hand" schlicht unmöglich ist, stellen für einen Rechner kein Problem dar. Aus diesem Grund sollte sich der Studierende der Statistik bereits frühzeitig mit der Anwendung entsprechender Software vertraut machen - der Praktiker wird ohnehin nicht mehr ohne sie auskommen.

Zur computergestützten Datenauswertung existiert eine Vielzahl von Statistik-Programmen und -Programmpaketen. Davon ist das weit verbreitete Programmpaket SAS (*Statistical Analysis System*) bezüglich der Anzahl der zur Verfügung gestellten Funktionen und Prozeduren das umfangreichste und nach unserer Ansicht für fast alle Anforderungen des Anwenders auch das beste.

Das Arbeiten mit SAS zu erleichtern, ist ein Ziel dieses Textes. Ein zweites, mindestens ebenso wichtiges, ist die Vermittlung der hinter den einzelnen Prozeduren stehenden statistischen Theorie. Damit soll einerseits dem Leser die Möglichkeit gegeben werden, die Berechnungen des Programms nachzuvollziehen, und andererseits die durch die einfache Handhabung computergestützter Datenanalyseverfahren "provozierte" Anwendung inadäquater statistischer Verfahren vermieden werden.

Das Buch richtet sich an jeden SAS-Anwender, an den Studierenden der Statistik genauso wie an den Praktiker, der im Rahmen seiner beruflichen Tätigkeit mit dem Programm arbeitet. Zum Verständnis des Textes sind Grundkenntnisse in Statistik erforderlich, wie sie heute in jedem Studienfach, dessen Bestandteil eine Statistik-Ausbildung im Grundstudium ist, vermittelt werden.

Es versteht sich von selbst, daß ein Programmpaket, dessen Handbücher mehrere Tausend Seiten umfassen, in diesem Text nicht vollständig beschrieben werden kann. Wir waren deshalb gezwungen, eine Auswahl der zu behandelnden Prozeduren zu treffen, die nach dem Kriterium ihrer Anwendbarkeit in den Wirtschafts-, Sozial- und Naturwissenschaften erfolgte. Leider mußten trotz ihrer diesbezüglichen Relevanz einige interessante, jedoch sehr umfangreiche bzw. komplexe statistische Verfahren wegfallen, so. z.B. die Schätzung von allgemeinen linearen Modellen mit der

Prozedur GLM, das komplette Gebiet der Zeitreihenanalyse, für das in SAS ein eigenes Programmodul (SAS/ETS) existiert, die Analyse von Überlebenszeiten oder Verfahren der nichtlinearen Diskriminanz- und Regressionsanalyse.
Schließlich konnten auch die behandelten Prozeduren und deren Syntax nicht bis ins letzte Detail erläutert werden, sondern es war eine Beschränkung auf das unserer Ansicht nach Wesentliche vonnöten. Jede Prozedur in SAS bietet eine Vielzahl von Optionen an, von denen einige wohl aber nur selten gebraucht werden. Diesbezügliche Wünsche für künftige Auflagen nehmen wir jedoch gerne entgegen.

Die Ausführungen beziehen sich auf den Stand der Programm-Version 6. Damit wurden auch sämtliche Beispiele gerechnet. Alle Änderungen, die bis einschließlich der Unterversion 6.10 auftreten, sind bereits berücksichtigt. Zu neuen Programm-Versionen von SAS ist anzumerken, daß sich die Syntax der Statistik-Prozeduren im allgemeinen nicht wesentlich ändert. Vorhandene Optionen fallen normalerweise nicht weg, so daß Änderungen überwiegend im Hinzukommen neuer Optionen und Prozeduren zu sehen sind. Das Buch wird also über einen längeren Zeitraum hinweg nichts an Aktualität verlieren.

Für die wertvolle Unterstützung bei der Veröffentlichung danken wir Herrn Prof. Dr. Walter Schweitzer, Passau. Zu Dank verpflichtet sind wir auch Frau Dr. Cornelia Baumgartner, Passau, für ihre Anregungen. Bei Herrn Martin Weigert möchten wir uns für die freundliche Aufnahme des Buches in den Oldenbourg Verlag bedanken.

Wir hoffen, mit dem vorliegenden Buch eine nützliche Arbeitshilfe zur Verfügung zu stellen und sind jederzeit für Anregungen, Verbesserungsvorschläge und Hinweise auf Fehler dankbar.

Frank Oerthel und Stefan Tuschl

0. Grundstrukturen des SAS-Systems

In diesem Abschnitt sollen dem Leser, der noch nie mit dem SAS-System gearbeitet hat, die Grundelemente und Grundstrukturen dieses Datenanalysesystems nähergebracht werden. Alle folgenden Kapitel setzen die Kenntnis dieser Grundstrukturen voraus. Für die überwiegende Mehrzahl der Anwender sollten die vermittelteten Kenntnisse ausreichen, um die Statistik-Prozeduren ohne Probleme ausführen zu können. Wünscht ein Anwender tiefergehende bzw. weitere Kenntnisse in der Bedienung von SAS, so sei ihm die einschlägige Literatur empfohlen.

Angemerkt sei, daß der Umgang mit SAS nicht ohne weiteres allgemein dargestellt werden kann, da die Durchführung vieler Operationen, je nachdem, unter welchem Betriebssystem SAS läuft, auf unterschiedliche Weise zu bewerkstelligen ist. Die Autoren selbst verwenden "SAS für *WINDOWS*", das wohl die bequemste Implementierung von SAS darstellt. Viele Operationen sind hier dem Standard der graphischen Benutzeroberfläche *WINDOWS* angepaßt, so daß Kenntnisse aus anderen *WINDOWS*-Programmen teilweise direkt auf SAS übertragen werden können.

Weiterhin gilt es zu beachten, daß das Programmpaket SAS aus verschiedenen **Modulen** besteht, von denen nicht alle installiert sein müssen. Das **Grundmodul** SAS/BASE ist jedoch in jedem Fall erforderlich. Für die statistische Datenauswertung benötigt man weiterhin das Modul SAS/STAT. Mit Ausnahme der im Abschnitt 0.3.4. beschriebenen Prozedur FSEDIT erfordern sämtliche in diesem Buch besprochenen statistischen Analyseverfahren lediglich diese beiden Module. In einigen Abschnitten wird ergänzend das Erstellen **hochauflösender Graphiken** besprochen. Zur Durchführung dieser Programme muß das Modul SAS/GRAPH installiert sein.

Die nachfolgende Darstellung der Grundstrukturen des SAS-Systems erfolgt **betriebssystemunabhängig**, d.h. die vorgestellten Anweisungen und Befehle können in jeder SAS-Implementierung verwendet werden.

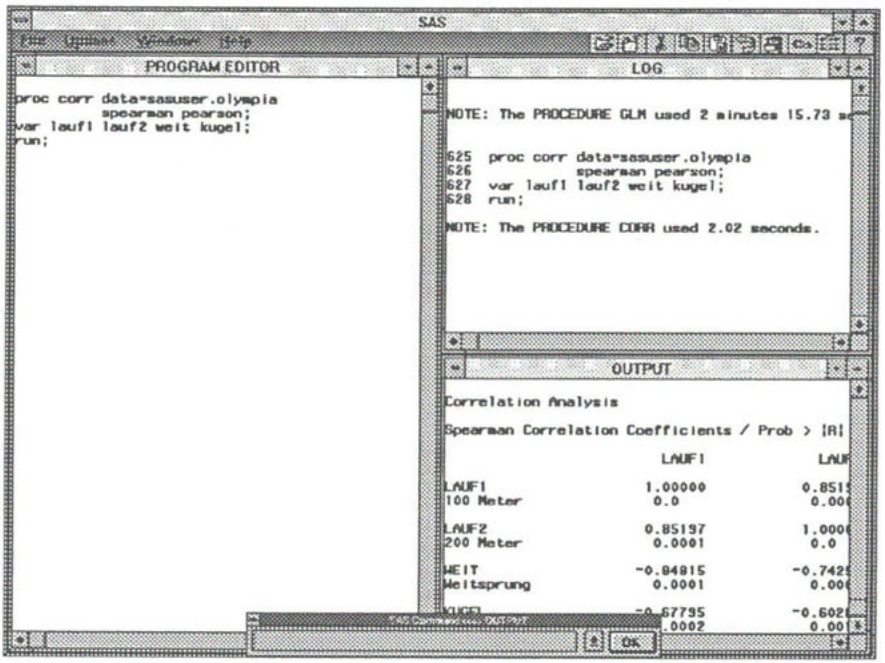
0.1. Der SAS-Display-Manager

SAS kann auf zwei Arten gestartet werden: Im **nichtinteraktiven (batch-)** und im **interaktiven** Modus. Im ersten Fall schreibt man sämtliche Anweisungen an das SAS-System in eine **Datei**, die dann dem System vollständig zur Abarbeitung übergeben wird. Das System liefert dann eine Datei mit den Ergebnissen und eine mit Fehlermeldungen und anderen Nachrichten zurück. Im interaktiven Modus erzeugt das SAS-System eine **Benutzer-**

oberfläche auf dem Bildschirm, in die man die Anweisungen eingibt und sogleich zur Abarbeitung übergeben kann. Die Ergebnisse sowie die Nachrichten erhält man hier unmittelbar auf den Bildschirm. Bei Fehlern kann das Programm ohne Aufwand sofort korrigiert werden. Auch bereits laufende Programme können oft noch modifiziert werden. Dieser Modus ist deshalb der empfehlenswertere. Innerhalb des interaktiven Modus existieren auch **menügeführte Oberflächen** (z.B. SAS/ASSIST), bei deren Verwendung der Anwender überhaupt keine Kenntnisse der SAS-Syntax mehr benötigt. Er wird vielmehr durch die in Interviewtechnik gestalteten Menüs zum Ziel geführt. Dieser Vorteil wird jedoch mit einem übermäßig hohem Zeitaufwand erkauft, den man benötigt, um die ganze Menühierarchie zu durchwandern. Auf diese Oberflächen wird deshalb nachfolgend auch nicht weiter eingegangen.

Der **interaktive** Modus wird gestartet, indem nach der Eingabeaufforderung des Betriebssystems der Befehl SAS eingegeben bzw. ein Symbol für SAS mit der Maus "angeklickt" wird (falls eine graphische Benutzeroberfläche wie z.B. *WINDOWS* verwendet wird). Wird SAS im interaktiven Modus betrieben, so setzt sich der SAS-Bildschirm (**Display**) stets aus drei Teilen zusammen. Das bedeutet nicht, daß diese drei Teilbildschirme (auch als **Screens**, **Fenster** oder **Windows** bezeichnet) alle gleichzeitig sichtbar sind. Sie existieren jedoch gleichzeitig und können bei Bedarf in den Vordergrund geholt werden. Jedes dieser Fenster enthält auch eine **Kommandozeile**, die sog. *Command Line*, in die Kommandos (nicht Programme!) an das System zu schreiben sind. Diese Kommandos bezeichnet man als **DMS-Kommandos** (**D**isplay-**M**anager-**S**ystem-Kommandos). In manchen Implementierungen gibt es auch nur eine Command Line für alle drei Fenster. Hier eingegebene Kommandos gelten dann für das gerade **aktive Fenster** (das Fenster, in dem sich der Cursor gerade befindet). Es können auch gleich mehrere Kommandos in die *Command Line* eingegeben werden, sie sind dann jeweils durch ein Semikolon zu trennen.
Menügeführte **Hilfe** zu SAS erhält man durch Eingabe von HELP in die *Command Line*. **Beendet** wird SAS durch Eingabe des Kommandos BYE in die *Command Line*.

In "SAS für *WINDOWS*" hat der Bildschirm z.B. folgendes Aussehen:



Hier sind die drei erwähnten Fenster sowie die *Command Line* (unten) gleichzeitig sichtbar. Die drei Fenster sind:

- *Der Program Editor.* In diesen Bildschirm werden die **Programme** eingegeben, die SAS ausführen soll. Die vom Editor angebotenen **Editierhilfen** sind je nach Betriebssystem unterschiedlich handzuhaben. Mit dem DMS-Befehl SUBMIT schickt man ein hier erstelltes Programm zur Abarbeitung an das SAS-System. Danach ist das Programm aus dem Program Editor verschwunden, jedoch in einem **Puffer** gespeichert, von wo es mit dem Kommando RECALL wieder in den Program Editor geholt werden kann.

- *Der Log-Screen.* Auf diesem Bildschirm erscheinen alle **Mitteilungen** des SAS-Systems an den Benutzer, z.B. Fehlermeldungen, Informationen über die verbrauchte Rechenzeit, Warnhinweise etc.

- *Der Output-Screen.* Auf diesem Bildschirm erscheint der von SAS-Programmen erzeugte **Output**.

Aktiviert wird ein Fenster von einem anderen aus, indem der Kurzname des Fensters (**PGM, LOG, OUT**) in die *Command Line* eingegeben wird. In einigen Implementierungen gibt es dafür (wie übrigens für die meisten SAS-

Kommandos) aber auch einfachere Möglichkeiten, z.B. die Verwendung von Funktionstasten, das Setzen des Cursors in das entsprechende Fenster oder das "Mausklicken" unter *WINDOWS*. Nimmt ein Fenster nicht den ganzen Bildschirm ein, so kann es durch das Kommando ZOOM **vergrößert** und mit ZOOM OFF wieder auf die ursprüngliche Größe **verkleinert** werden. Ist der Inhalt eines Fensters zu umfangreich, um auf dem Bildschirm dargestellt zu werden, so kann mit dem Befehl FORWARD **eine Bildschirmseite nach vorn geblättert** werden, mit dem Befehl BACKWARD **eine Seite zurück**. Zur letzten Seite gelangt man mit BOTTOM, zur ersten mit TOP. Der Inhalt eines jeden Fensters kann separat in eine Datei **gespeichert** werden, wozu der Befehl FILE *<Dateiname>* dient. Wird mit diesem Befehl ein Programm abgespeichert, so kann es in einer späteren SAS-Sitzung mit dem Befehl INCLUDE *<Dateiname>* wieder **geladen** werden. **Gelöscht** wird der Inhalt eines Fensters mit dem Befehl CLEAR. Den wichtigsten und am häufigsten benötigten Kommandos sind in den meisten Implementierungen **Funktionstasten** zugeordnet. Eine Übersicht über die Belegung der Funktionstasten erhält man durch Eingabe des Kommandos KEYS. Damit kann die Funktionstastenbelegung auch verändert und somit den persönlichen Bedürfnissen angepaßt werden. Man beachte jedoch, daß die Funktionstasten in verschiedenen Fenstern unter Umständen verschiedene Belegungen haben können.

0.2. Allgemeiner Aufbau eines SAS-Programms

Ein SAS-Programm besteht allgemein aus folgenden Typen von Anweisungen:

- **Globale Anweisungen (global statements)**
- **Datenmanipulationsschritte (DATA steps)**
- **Datenverarbeitungsschritte (PROC steps, Prozeduren)**

In einem SAS-Programm müssen nicht alle drei Typen von Anweisungen enthalten sein, sondern lediglich einer. Jede Anweisung wird im SAS-Programm mit einem Semikolon abgeschlossen.
Mit **globalen Anweisungen** werden z.B. das Seitenlayout, Titel, Fußzeilen etc. des Outputs festgelegt. Sie beziehen sich auf alle folgenden Anweisungen und sind während einer SAS-Sitzung so lange gültig, bis neue globale Anweisungen gesetzt werden. Werden globale Anweisungen nicht gesetzt, verwendet SAS entsprechende **Voreinstellungen**, die in fast allen Fällen auch übernommen werden können.
Datenmanipulationsschritte werden innerhalb des Programms durch das Schlüsselwort DATA eingeleitet. Damit SAS Daten analysieren kann,

müssen sie vom System zunächst einmal eingelesen und in ein für SAS verarbeitbares Format gebracht werden. Dazu dienen die **DATA-Steps**. Auch die Manipulation von Daten (Verändern bereits bestehender Daten, Erzeugen von neuen Daten aus bereits früher eingelesenen, Hinzufügen neuer Daten zu bereits bestehenden etc.) geschieht stets innerhalb eines DATA-Steps.
Datenverarbeitungsschritte werden innerhalb des Programms durch das Schlüsselwort PROC eingeleitet. Hier erfolgt die eigentliche Datenanalyse.

Die letzte Anweisung in einem SAS-Programm ist stets die Anweisung RUN. Fehlt sie, so wird ein Programm zwar abgearbeitet und die Syntax auf Fehler überprüft, jedoch solange nicht tatsächlich ausgeführt, bis die Anweisung RUN nachträglich eingegeben wird. Manche Prozeduren (z.B. die in Kapitel 6 besprochene Prozedur REG) laufen auch nach der Abarbeitung des sie aufrufenden Programms weiter und warten auf weitere Eingaben. In der Prozedur REG können z.B. die Residuenplots auf diese Weise nachträglich angefordert werden. Solche Prozeduren beendet man durch nachträgliche Eingabe der Anweisung QUIT in den Program Editor. Soll dieses Weiterlaufen von Anfang an unterbunden werden, so ist die Anweisung QUIT gleich am Ende des Programms (nach RUN) anzugeben.

0.3. Datenerfassung in SAS

SAS kennt mehrere Möglichkeiten der Datenerfassung, z.B. das Einlesen aus einer **externen Datei** im ASCII-Format, die **direkte Eingabe** im SAS-Programm (nur bei geringer Datenmenge zu empfehlen) oder die **maskenorientierte Dateneingabe** mittels der Prozedur **FSEDIT**. Letztere ist vor allem bei großen Datenmengen zu empfehlen. Man beachte jedoch, daß diese Prozedur nur dann zur Verfügung steht, wenn das Modul SAS/FSP installiert ist.
Das Ergebnis ist in allen drei Fällen eine **SAS-Datendatei (SAS Dataset)**, also eine Datei, die die zu analysierenden Daten in einer für SAS verständlichen Form beinhaltet. Der eigentlich redundante Begriff "Datendatei" dient zur Abgrenzung von SAS-Dateien mit anderem Inhalt, z.B. Programm-Dateien oder Output-Dateien.

0.3.1. Verwaltung von SAS-Datendateien

Der **Name** einer SAS-Datendatei setzt sich stets aus **zwei Komponenten** zusammen, der sogenannten *libref* **(library reference)** und dem eigentlichen *dateinamen*. Diese beiden Komponenten werden durch einen Punkt

voneinander getrennt. Allgemein lautet die Bezeichnung einer SAS-Datendatei also

libref.dateiname

Die *libref* bezeichnet einen **Ordner (library)**, in dem die Datei abgelegt werden soll und ist vergleichbar mit einem **Verzeichnis** (directory) auf der Betriebssystemebene. Für Datendateien ist normalerweise bereits eine *libref* voreingestellt. Sie kann aber auch vom Benutzer mit einem SAS-Programm erstellt werden. Dazu dient die Anweisung LIBNAME , die allgemein die Form

LIBNAME *libref 'Verzeichnisname im Betriebssystem'* ;

hat.
Soll beispielsweise eine *libref* mit der Bezeichnung "sasdat" erstellt werden, deren Inhalte auf Betriebssystemebene im Verzeichnis "C:\sas\daten" gespeichert werden sollen, so lautet das notwendige Programm:

```
libname  sasdat  'c:\sas\daten';
run;
```

War das Erstellen der *libref* mit diesem Programm erfolgreich, so erscheint eine bestätigende Meldung im Log-Screen:

```
02    libname sasdat 'c:\sas\daten';
NOTE: Libref SASDAT was successfully assigned as follows:
      Engine:        V608
      Physical Name: C:\SAS\DATEN
03   run;
```

Wird nun eine neue SAS-Datendatei mit der Bezeichnung `sasdat.umfrage` angelegt, bedeutet dies, daß die Datei `umfrage` im Ordner (library) `sasdat` abgelegt und vom Betriebssystem in das Verzeichnis `C:\sas\daten` gespeichert wird. In jedem SAS-Programm kann diese Datei dann unter der Bezeichnung `sasdat.umfrage` verwendet werden.
Man beachte, daß eine derart angelegte *libref* nur in der SAS-Sitzung existiert, in der sie angelegt wurde. Soll auf die Datei `sasdat.umfrage` in einer späteren SAS-Sitzung wieder zugegriffen werden, so ist zu Beginn dieser Sitzung die *libref* `sasdat` zunächst mit Hilfe des obigen Programms erneut zu erstellen. Man sollte deshalb nach Möglichkeit auf eine von SAS

standardmäßig erzeugte *libref* zum Ablegen der Dateien zurückgreifen. Eine Übersicht über alle vorhandenen *librefs* (sowohl die standardmäßig von SAS als auch die vom Benutzer erzeugten) erhält man mit dem DMS-Kommando LIBNAME.

Wenn beim Erstellen einer neuen SAS-Datendatei keine *libref* angegeben wird, sondern nur der eigentliche *dateiname*, legt SAS die Datei automatisch im Ordner `work` ab. Das ist eine *libref*, die in SAS stets vorhanden ist und zum Ablegen **temporärer** Dateien dient.
Eine Datei, die **ohne** *libref* erstellt wird, ist demnach eine **temporäre Datendatei**. Sie steht **nur** in der momentan laufenden SAS-Sitzung zur Verfügung. Wird SAS beendet und neu gestartet, ist diese Datei verschwunden und muß gegebenenfalls erneut erstellt werden.
Wird beim Erstellen der Datei hingegen eine *libref* in deren Namen aufgenommen, erstellt SAS eine **permanente Datendatei**. Einmal erstellt, steht sie in allen künftigen SAS-Sitzungen zur Verfügung (auch nach einem eventuellen "Absturz" des Rechners).

0.3.2. Das Einlesen von Daten aus einer ASCII-Datei

Sind die zu verarbeitenden Daten bereits im sog. ASCII-Format auf einem Speichermedium verfügbar, können diese von SAS spaltenweise eingelesen werden. Dazu dient der **DATA-Step**. Ein Programm zum Erfassen externer Daten hat allgemein folgende Syntax:

DATA *SAS-Dateiname*;
INFILE '*Name der externen Datei*';
INPUT *Variablenliste*;

Wie ein konkretes Programm aussieht, soll anhand eines Beispiels gezeigt werden. Die einzulesende ASCII-Datei habe folgendes Aussehen:

```
a 4000 3500
b 6800 7200
c 2100 2000
d 4500 2800
.    .    .
.    .    .
.    .    .
```

Diese Daten werden mit dem folgenden Programm eingelesen:

```
data sasuser.daten12;
infile 'c:\daten\umsatz.dat';
input  filiale $  umsatz  kosten;
run;
```

Dabei ist nach DATA der Name angegeben, den die Datendatei im SAS-Format tragen soll (`sasuser.daten12`). Wird hier kein *SAS-Dateiname* angegeben, vergibt SAS den Namen `data1` in der library `work`. Eine zu einem späteren Zeitpunkt innerhalb derselben SAS-Sitzung ohne Namensangabe erzeugte Datei erhielte dann den Namen `data2` usw. Diese Dateien existieren nur innerhalb der Sitzung, in der sie erzeugt wurden.
Nach INFILE ist der Name angegeben, den die ASCII-Datei, deren Inhalt SAS einlesen soll, im Betriebssystem trägt (`c:\daten\umsatz.dat`). Hierbei ist zu beachten, daß dieser Dateiname in **Hochkommata** zu schreiben ist. Nach INPUT sind die Namen (maximal acht Zeichen) der einzulesenden Variablen angegeben (`filiale`, `umsatz` und `kosten`). **Alphanumerische Variablen** (Zeichenketten) sind durch ein nachgestelltes $-Zeichen zu kennzeichnen (im Beispiel die Variable `filiale`). Bei Variablen, deren Werte **Dezimalzahlen** sind, ist zu beachten, daß SAS nur den **Dezimalpunkt** kennt, nicht aber das Dezimalkomma!

Nicht getrennte Werte in einer Zeile der ASCII-Datei

Sind die Werte eines Datensatzes in der ASCII-Datei nicht durch ein **Leerzeichen** getrennt, so ist nach jedem Variablennamen die **Angabe der Spalten** erforderlich, die die Werte der Variablen enthalten. Die INPUT-Anweisung müßte dann

```
...
input   filiale $ 1  umsatz 2-5  kosten 6-9;
...
```

lauten, wenn die Beispieldatei folgendes Aussehen hätte:

```
a40003500
b68007200
c21002000
d45002800
    .
    .
    .
```

Mehrere Objektdatensätze in einer Zeile

Befinden sich in **einer Zeile** der einzulesenden ASCII-Datei die Merkmalswerte für **mehrere Objekte**, so ist dies durch Hintanstellen von @@ an das Ende der INPUT-Anweisung dem SAS-System mitzuteilen. Die INPUT-Anweisung müßte also

```
...
input   filiale $  umsatz  kosten  @@;
...
```

lauten, wenn die ASCII-Datei im Beispiel folgendes Aussehen hätte:

```
a 4000 3500 b 6800 7200 c 2100 2000 d 4500 2800
.    .    .  .    .    .  .    .    .  .    .
.    .    .  .    .    .  .    .    .  .    .
.    .    .  .    .    .  .    .    .  .    .
```

Einlesen eines Objekdatensatzes aus mehreren Zeilen

Wenn sehr viele Merkmale erhoben worden sind, kann es vorkommen, daß in der ASCII-Datei die Beobachtungswerte für **ein Objekt mehr als nur eine Zeile** beanspruchen. Dies muß SAS in der INPUT-Anweisung mitgeteilt werden. Befinden sich zum Beispiel die Werte der Variablen `filiale` in der **jeweils ersten Zeile** eines Objektdatensatzes und die Werte der Variablen `umsatz` und `kosten` in der **jeweils zweiten Zeile**, müßte die INPUT-Anweisung

```
...
input   #1 filiale $
        #2 umsatz  kosten;
...
```

lauten. Die ASCII-Datei hätte hier also folgendes Aussehen:

```
a
4000 3500
b
6800 7200
c
2100 2000
 .    .
 .    .
```

Fehlende Beobachtungswerte

Fehlen in der ASCII-Datei Beobachtungswerte, weist SAS dem entsprechenden Objekt in der betreffenden Variablen den **missing value** (fehlenden Beobachtungswert) zu. SAS symbolisiert den missing value in der Datendatei durch einen Punkt.

Überprüfen der Datendatei

Im Anschluß an das Einlesen der Daten sollte man die Datendatei auf ihre Korrektheit hin überprüfen, was dadurch geschieht, daß man den Inhalt mit **PROC PRINT** ausgeben läßt. Das dazu erforderliche Programm würde im Beispiel lauten:

```
proc print data=sasuser.daten12;
run;
```

Es liefert den folgenden (selbsterklärenden) Output:

OBS	FILIALE	UMSATZ	KOSTEN
1	a	4000	3500
2	b	6800	7200
3	c	2100	2000
4	d	4500	2800
.	.	.	.
.	.	.	.
.	.	.	.

Allgemeine Informationen über eine SAS-Datendatei (Datum der Erstellung, Anzahl und Bezeichnung der Variablen etc.) erhält man mit der Prozedur **CONTENTS**. Für die Beispieldatei liefert das Programm

```
proc contents data=sasuser.daten12;
run;
```

folgende Ausgabe (auf die jedoch nicht weiter eingegangen werden soll):

```
CONTENTS PROCEDURE

Data Set Name: SASUSER.DATEN12
Member Type:   DATA
Engine:        V608
Created:       10:52 Friday, February 10, 1995
Last Modified: 10:52 Friday, February 10, 1995
Protection:
Data Set Type:
Label:

Observations:          17
Variables:             3
Indexes:               0
Observation Length:    24
Deleted Observations:  0
Compressed:            NO
Sorted:                NO


-----Engine/Host Dependent Information-----

Number of Data Set Pages: 1
Data Set Page Size:       4096
File Format:              607
First Data Page:          1
Max Obs per Page:         169
Obs in First Data Page:   17


CONTENTS PROCEDURE

----Alphabetic List of Variables and Attributes----

#    Variable    Type    Len    Pos
-----------------------------------
1    FILIALE     Char      8      0
3    KOSTEN      Num       8     16
2    UMSATZ      Num       8      8
```

0.3.3. Direkte Dateneingabe im SAS-Programm

Stehen die zu analysierenden Daten nicht bereits auf einem Speichermedium zur Verfügung, so können sie auch **direkt im SAS-Programm** (DATA-Step) eingegeben werden. Diese Vorgehensweise empfiehlt sich jedoch nur bei vergleichsweise **geringen Datenmengen**. Ab einer gewissen Variablen- oder Fallzahl wird die direkte Eingabe nämlich schnell unübersichtlich und

unnötig kompliziert. In diesem Fall sollten eigene Programme zur Datenerfassung (z.B. Tabellenkalkulationsprogramme) mit anschließendem Einlesen aus einer ASCII-Datei oder die SAS-Prozedur FSEDIT (vgl. Abschnitt 0.3.4.) verwendet werden. Die Syntax eines DATA-Steps mit direkter Dateneingabe lautet:

DATA *SAS-Dateiname*;
INPUT *Variablenliste*;
CARDS;
<Daten>
;

Die DATA- und die INPUT-Anweisung entsprechen exakt den im vorherigen Abschnitt (Einlesen einer ASCII-Datei) erläuterten gleichlautenden Anweisungen. Die eigentliche Dateneingabe wird mit der Anweisung CARDS eingeleitet. In der nächsten Zeile kann die Dateneingabe beginnen. Sie wird mit einem Semikolon, das in eine eigene Zeile geschrieben werden muß, abgeschlossen. Auch hier sei nochmals darauf hingewiesen, daß **Dezimalzahlen mit Dezimalpunkt** (nicht mit Komma) einzugeben sind.
Das Programm zum Einlesen der Beispieldatendatei `sasuser.daten12` müßte bei direkter Dateneingabe also

```
data sasuser.daten12;
input  filiale $  umsatz  kosten;
cards;
a 4000 3500
b 6800 7200
c 2100 2000
d 4500 2800

weitere Datenzeilen

;
run;
```

lauten.

Fehlende Werte

Fehlt der Beobachtungswert einer Variablen bei einem Objekt, so ist statt des Wertes ein Punkt einzugeben. SAS markiert diesen Wert dann automatisch als fehlend.

0.3.4. Datenerfassung mit der Prozedur FSEDIT

Die Dateneingabe mittels FSEDIT gleicht dem Ausfüllen eines Fragebogens. FSEDIT wird nicht innerhalb eines DATA-Steps ausgeführt, sondern als Prozedur in einem PROC-Step. Mit FSEDIT können nicht nur neue Dateien erstellt, sondern auch bereits bestehende modifiziert werden. Es sei nochmals darauf hingewiesen, daß diese Prozedur nur zur Verfügung steht, wenn das Programm-Modul SAS/FSP installiert ist.

Die Syntax dieser Prozedur lautet allgemein:

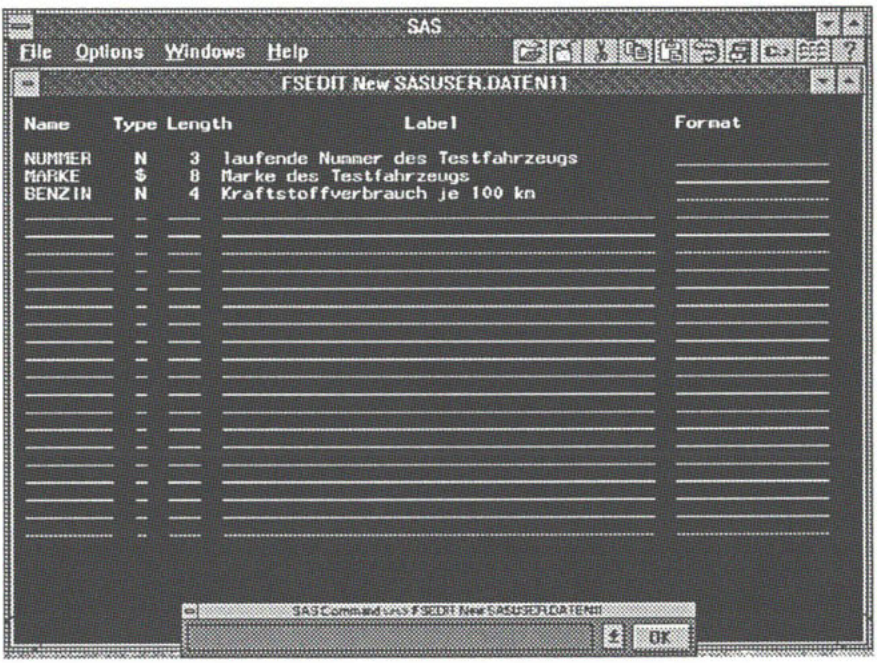
PROC FSEDIT NEW= | DATA= *SAS-Datendatei* ;

Soll eine neue Datendatei erstellt werden, ist die Option NEW= zu verwenden, die Option DATA= dient der Modifikation einer bereits bestehenden Datendatei.

Zum Erstellen einer neuen Datei (z.B. mit der Bezeichnung *sasuser.daten11*) wäre somit folgendes Programm erforderlich:

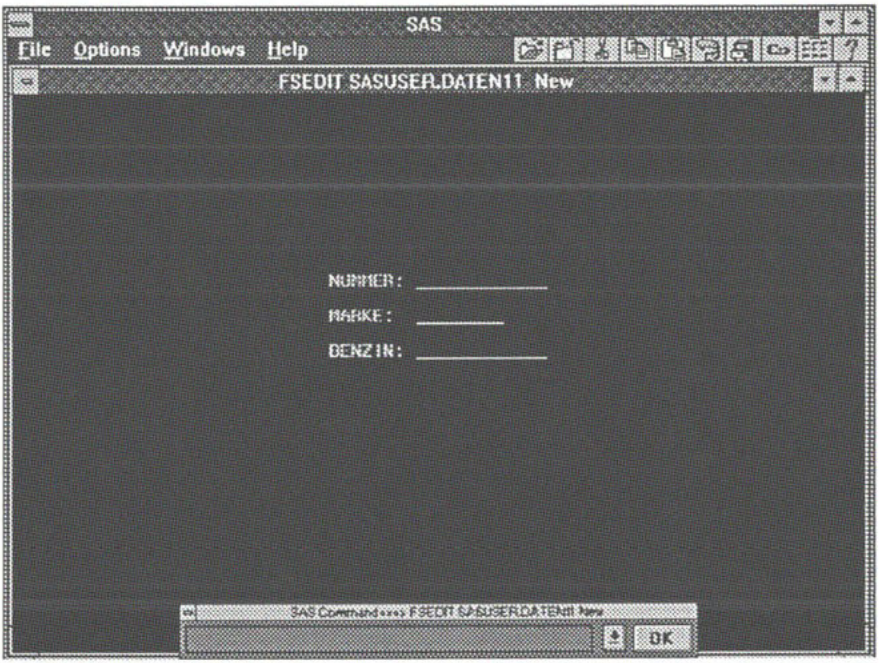
```
proc fsedit  new=sasuser.daten11;
run;
```

Nach dem Abschicken des Programms erscheint das folgende (hier bereits teilweise ausgefüllte, zunächst aber leere) Formular auf dem Bildschirm:



Jede Zeile ist dabei für eine zu erzeugende Variable reserviert. In das Feld `Name` ist der **Name der Variablen** einzutragen, in das Feld `Type` für **numerische** Variablen ein N und für **alphanumerische** Variablen (Zeichenketten) ein $. Wird hier nichts eingetragen, nimmt SAS eine numerische Variable an. In das Feld `Length` schreibt man die **Anzahl der Bytes**, die zur Speicherung eines Wertes dieser Variablen verwendet werden soll. Bei alphanumerischen Variablen entspricht diese der Anzahl der Zeichen, die ein Wert maximal haben darf. Auch dieses Feld muß jedoch nicht unbedingt ausgefüllt werden, SAS verwendet dann als Voreinstellung den Wert 8. Unter `Label` kann eine **erläuternde Beschreibung** der Variablen eingetragen werden, die dann in jedem Output erscheint. Unter `Format` können **Ausgabeformate** (v.a. für numerische Variablen) festgelegt werden. Hier ist ein Wert vom Typ x.y einzutragen, wobei x die **Gesamtzahl der Stellen** einer Zahl ist (**einschließlich des Dezimalpunkts!**) und y die Anzahl der Nachkommastellen. Ein Ausgabeformat 6.2 bedeutet also, daß die Werte der Variablen im Output (nicht intern!) zwei Dezimalstellen aufweisen, und vor dem Dezimalpunkt maximal 3 Stellen haben.
Dieses Fenster wird durch Eingabe von END in die *Command Line* verlassen, woraufhin man von SAS aufgefordert wird, den Befehl ADD einzugeben, um mit der **Dateneingabe** zu beginnen, oder nochmals END, um die Datei zunächst leer zu belassen und FSEDIT zu beenden. Gibt man ADD

ein, erscheint ein weiteres Formular (eine sog. Eingabemaske), in das man nun die Variablenwerte des ersten Objektes einträgt:



Diese Eingabe ist mit ADD abzuschließen (wofür aber zur einfacheren Handhabung eine Funktionstaste mit ADD belegt ist), dadurch erscheint wieder eine leere Maske für das nächste Objekt. Nach Eingabe der Werte des letzten Objekts ist statt ADD der Befehl END einzugeben. Damit wird die Prozedur FSEDIT verlassen. Wurden mehr Variablen erzeugt, als gleichzeitig auf dem Bildschirm darstellbar sind, so zerlegt SAS die Eingabemaske in **Teilmasken**. Die nächste Teilmaske erreicht man durch Eingabe des Befehls RIGHT, eine Teilmaske zurück gelangt man mit LEFT.
Weitere Befehle sind am Ende des nächsten Abschnitts beschrieben.

Fehlende Beobachtungswerte

Liegt der Beobachtungswert einer Variablen bei einem Objekt nicht vor, so ist das entsprechende Feld der Eingabemaske leer zu belassen. SAS markiert dann in der Datendatei diesen Wert automatisch als fehlend und vergibt als Symbol für den missing value einen Punkt.

0.3.5 Modifikation einer vorhandenen Datendatei mit FSEDIT

Natürlich können einer bereits bestehenden Datei **nachträglich weitere Datensätze angehängt** und auch **bestehende Datensätze modifiziert** werden. Dazu ruft man die Prozedur FSEDIT mit der Option *DATA=* anstelle von *NEW=* auf. Im Beispiel wäre also das Programm

```
proc fsedit  data=sasuser.daten11;
run;
```

einzugeben. Es erscheint ein Bildschirm mit den Daten des ersten Objektes. Man kann dann mit einer Reihe von Befehlen (für die meisten gibt es aber auch hier Funktionstasten) die **bisherigen Datensätze durchblättern** und **korrigieren** bzw. wieder mit ADD **neue Datensätze anhängen**. Beendet wird die Prozedur auch in diesem Fall durch Eingabe von END.
Die für die *Command Line* der Prozedur FSEDIT erlaubten Kommandos sind im einzelnen:

n	Anzeigen des n-ten Objektes (n steht für eine natürliche Zahl)
ADD	Ausgeben eines leeren Bildschirms zum Erfassen eines neuen Datensatzes
BACKWARD	Ausgeben des vorigen Datensatzes
CANCEL	Durchgeführte Korrekturen am aktuellen Objekt rückgängig machen
DUPLICATE	Duplizieren des angezeigten Datensatzes an das Ende der Datei
END	Beenden der Prozedur
FIND *<Bedingung>*	Durchsuchen der Datendatei nach Objekten mit erfüllter *<Bedingung>*. *<Bedingung>* ist vom Typ *<Variable>* = | < | > | <= | >= | NE *Wert* NE steht dabei für not equal (ungleich). Im Beispiel würde der Befehl `find benzin >= 9.5` das nächste Objekt (Auto) der Datendatei suchen, dessen Benzinverbrauch mindestens 9,5 Liter beträgt.
FORWARD	Ausgeben des nächsten Datensatzes

HELP	Anfordern von Informationen über das Feld, in dem gerade der Cursor steht
KEYS	listet die aktuelle Funktionstastenbelegung auf.
LEFT	Paßt die Eingabemaske für ein Objekt nicht auf eine Bildschirmseite, so wird sie von SAS in Teilmasken zerlegt. Mit diesem Befehl gelangt man dann in die vorherige Teilmaske des aktuellen Objekts.
RIGHT	Vgl. LEFT. Man kommt in die nächste Teilmaske des aktuellen Objekts.
RFIND	Wiederholt die letzte Suchanweisung (vgl. Befehl FIND)
WHERE *<Bedingung>*	Schränkt die Datenfälle zum Bearbeiten temporär gemäß *<Bedingung>* ein. Zur Syntax zulässiger *<Bedingungen>* vgl. man den Befehl FIND.

Diese Befehle gelten auch, wenn eine neue Datendatei mit FSEDIT und der Option NEW= erstellt wird.

0.4. Zusammenfügen bestehender SAS-Datendateien

Wurden die zu analysierenden Daten in verschiedenen SAS-Datendateien gespeichert, so lassen sie sich dennoch in einer einzigen Datendatei **zusammenfassen**. Im folgenden wird die Vorgehensweise für zwei zu verschmelzende Dateien vorgestellt. Die Syntax ändert sich nicht, falls **mehr als zwei Dateien** zusammengefügt werden sollen, es sind dann lediglich entsprechend mehr Dateinamen anzugeben.
Grundsätzlich kann man beim Zusammenfügen von SAS-Datendateien zwei Fälle unterscheiden:

- Die Dateien werden **hintereinander verkettet**.
- Die Dateien werden **verschränkt (interleaving)**.

Auf beide Fälle wird nachfolgend näher eingegangen.

0.4.1. Hintereinander Verketten von SAS-Datendateien

Diese Vorgehensweise ist angebracht, falls in den zu verkettenden Dateien die **gleichen Variablen** enthalten sind. Das bedeutet, daß die gleichen Merkmale an mehreren Gruppen von Objekten gemessen wurden, von denen jede eine eigene Datendatei beansprucht. Die Dateien werden dabei quasi **untereinander** gesetzt. In der resultierenden Datei befinden sich dann die Beobachtungswerte **sämtlicher** Objekte.
Die **Syntax zum Verketten** von Datendateien besteht aus lediglich einer Anweisung, die **innerhalb eines DATA-Steps** stehen muß:

SET *<datei1>* *<datei2>* ;

Stünden im Beispieldatensatz der Umsatz und die Kosten der Filialen a bis g in der Datendatei `sasuser.daten13`, die entsprechenden Größen der Filialen h bis q in der Datendatei `sasuser.daten14`, und sollen diese beiden Dateien zur neuen Datendatei `sasuser.daten15` verkettet werden, wäre folgendes Programm erforderlich:

```
data  sasuser.daten15;
set  sasuser.daten13  sasuser.daten14;
run;
```

Die resultierende Datei `sasuser.daten15` enthielte dann den Umsatz und die Kosten sowie die Bezeichnung (a bis q) **aller** Filialen, wie ein Blick auf den Output des folgenden Programms zeigt:

```
proc print  data=sasuser.daten13;
proc print  data=sasuser.daten14;
proc print  data=sasuser.daten15;
run;
```

Das Programm erzeugt folgenden (selbsterklärenden) Output:

```
/*  Inhalt der Datei sasuser.daten13  */

OBS    FILIALE    UMSATZ    KOSTEN

 1        a        4000      3500
 2        b        6800      7200
 3        c        2100      2000
 4        d        4500      2800
 5        e        3200      1800
 6        f        4900      3000
 7        g        7800      4500

/*  Inhalt der Datei sasuser.daten14  */

OBS    FILIALE    UMSATZ    KOSTEN

  1       h        3200      2900
  2       i        4700      4600
  3       j        3700      2300
  4       k        5000      3000
  5       l        3800      4200
  6       m        5200      3200
  7       n        7900      6800
  8       o        4700      2300
  9       p        5600      3400
 10       q        4500      2500


/*  Inhalt der verketteten Datei sasuser.daten15  */

OBS    FILIALE    UMSATZ    KOSTEN

  1       a        4000      3500
  2       b        6800      7200
  3       c        2100      2000
  4       d        4500      2800
  5       e        3200      1800
  6       f        4900      3000
  7       g        7800      4500
  8       h        3200      2900
  9       i        4700      4600
 10       j        3700      2300
 11       k        5000      3000
 12       l        3800      4200
 13       m        5200      3200
 14       n        7900      6800
 15       o        4700      2300
 16       p        5600      3400
 17       q        4500      2500
```

0.4.2. Verschränken (interleaving) von Datendateien

Befinden sich für **ein** Objekt die Beobachtungswerte einer Reihe von Variablen in einer ersten SAS-Datendatei und in einer zweiten die Beobachtungswerte weiterer Variablen (desselben Objekts!), so ist beim Zusammenfügen der beiden Dateien das sogenannte **interleaving** angebracht. Die Dateien werden hierbei quasi **nebeneinander** gesetzt. In der resultierenden Datei stehen in einer Zeile die Beobachtungswerte **sämtlicher** Variablen für **ein** Objekt.
Die Variablen sollten in den zu verschränkenden Dateien unterschiedliche Bezeichnungen haben, um Mehrdeutigkeiten zu vermeiden.

Die **Syntax** zum Verschränken von SAS-Datendateien besteht ebenfalls aus nur einer Anweisung. Auch diese muß innerhalb eines DATA-Steps stehen. Sie lautet:

MERGE <*datei1*> <*datei2*> ;

Hierbei erfolgt eine 1:1-Kopplung der beiden Dateien. Dies kann unter Umständen zu Problemen führen, etwa dann, wenn die Anzahl der Objekte in den beiden Dateien nicht übereinstimmt, weil z.B. die Beobachtungswerte der in der <*datei2*> enthaltenen Variablen für ein oder mehrere Objekte nicht vorliegen. In diesem Fall fügt SAS eigentlich verschiedene Objekte zusammen, was normalerweise keinen Sinn macht.

Dies kann vermieden werden, wenn wenigstens eine Schlüsselvariable in allen zu verschränkenden Dateien vorkommt, anhand derer sich jedes Objekt eindeutig identifizieren läßt. Werden die zu verschränkenden Dateien nach den Werten dieser Schlüsselvariablen sortiert (was mit PROC SORT geschieht, siehe Beispiel), dann bewirkt die Anweisung BY nach der MERGE-Anweisung, daß die Objekte nur zusammengefügt werden, wenn sie im Wert der Schlüsselvariablen übereinstimmen.

Die **Syntax** lautet dann:

MERGE <*datei1*> <*datei2*> ;
BY <*schlüsselvariable*> ;

Würden im Beispieldatensatz für jede Filiale die Bezeichnung der Filiale sowie die Werte der Variablen Umsatz und Kosten in der Datei `sasuser.daten15` stehen und in einer weiteren Datendatei `sasuser.daten16` wiederum die Bezeichnung jeder Filiale sowie die Größe ihrer Verkaufsfläche (Variable `flaeche`), sähe das Programm zur

"gesicherten" Verschränkung (neue Datei `sasuser.daten17`) folgendermaßen aus:

```
proc sort  data=sasuser.daten15;
by filiale;

proc sort  data=sasuser.daten16;
by filiale;

data  sasuser.daten17;
merge  sasuser.daten15  sasuser.daten16;
by filiale;
run;
```

In der resultierenden Datei `sasuser.daten17` befinden sich dann für jede Filiale die Variablen `filiale`, `umsatz`, `kosten` und `flaeche`, wie das folgende Programm zeigt:

```
proc print  data=sasuser.daten15;
proc print  data=sasuser.daten16;
proc print  data=sasuser.daten17;
run;
```

Es liefert den folgenden (selbsterklärenden) Output:

```
/*  Inhalt der Datei sasuser.daten15  */

OBS    FILIALE    UMSATZ    KOSTEN

  1       a        4000      3500
  2       b        6800      7200
  3       c        2100      2000
  4       d        4500      2800
  5       e        3200      1800
  6       f        4900      3000
  7       g        7800      4500
  8       h        3200      2900
  9       i        4700      4600
 10       j        3700      2300
 11       k        5000      3000
 12       l        3800      4200
 13       m        5200      3200
 14       n        7900      6800
 15       o        4700      2300
 16       p        5600      3400
 17       q        4500      2500
```

```
/*  Inhalt der Datei sasuser.daten16  */

OBS     FILIALE     FLAECHE

  1        a          150
  2        b          175
  3        c          210
  4        d          300
  5        e          180
  6        f          490
  7        g          200
  8        h          220
  9        i          170
 10        j          250
 11        k          190
 12        l          290
 13        m          350
 14        n          300
 15        o          390
 16        p          400
 17        q          250

/*  Inhalt der verschränkten Datei sasuser.daten17  */

OBS     FILIALE     UMSATZ     KOSTEN     FLAECHE

  1        a         4000       3500        150
  2        b         6800       7200        175
  3        c         2100       2000        210
  4        d         4500       2800        300
  5        e         3200       1800        180
  6        f         4900       3000        490
  7        g         7800       4500        200
  8        h         3200       2900        220
  9        i         4700       4600        170
 10        j         3700       2300        250
 11        k         5000       3000        190
 12        l         3800       4200        290
 13        m         5200       3200        350
 14        n         7900       6800        300
 15        o         4700       2300        390
 16        p         5600       3400        400
 17        q         4500       2500        250
```

0.5. Datenmanipulationen

Mit Hilfe von SAS-Befehlen können bereits eingegebene Variablen **verändert**, bzw. neue Variablen **definiert** werden. Diese Datentransformationen sind immer **innerhalb** eines **DATA-Schrittes** (entweder direkt beim Einlesen der Daten oder beim Hinzufügen von neuen Variablen mit Hilfe des **SET**-Befehls in eine bestehende Datei) anzugeben.

0.5.1. Wertzuweisungen in SAS

Einer (neuen) Variablen wird folgendermaßen ein arithmetischer Ausdruck **zugewiesen**:

Variable = arithmetischer Ausdruck

Dieser arithmetische Ausdruck kann Variablen(namen) und/oder Zahlen, beinhalten, die durch die folgenden arithmetischen Operatoren miteinander verknüpft werden können:

+	Addition
-	Subtraktion
*	Multiplikation
/	Division
**	Potenzieren

Für das Programmieren von komplizierteren Formeln empfiehlt sich außerdem das Setzen von Klammern.

Wurde z.B. die ASCII-Datei `umsatz.dat` als SAS-Datendatei `sasuser.daten12`, die die Variablen `filiale` (alphanumerisch) `umsatz` und `kosten` enthält, eingelesen so, kann der in den Filialen erzielte Gewinn mit Hilfe von SAS berechnet werden als

```
gewinn = umsatz - kosten
```

bzw. in einem DATA-Schritt als Variable `gewinn` der Datei `sasuser.daten12` hinzugefügt werden als

```
data sasuser.daten12;
set sasuser.daten12;
gewinn=umsatz-kosten;
proc print data=sasuser.daten12;
run;
```

Diese Datenmanipulation liefert dann den Output:

OBS	FILIALE	UMSATZ	KOSTEN	GEWINN
1	a	4000	3500	500
2	b	6800	7200	-400
3	c	2100	2000	100
4	d	4500	2800	1700
5	e	3200	1800	1400
6	f	4900	3000	1900
7	g	7800	4500	3300
8	h	3200	2900	300
9	i	4700	4600	100
10	j	3700	2300	1400
11	k	5000	3000	2000
12	l	3800	4200	-400
13	m	5200	3200	2000
14	n	7900	6800	1100
15	o	4700	2300	2400
16	p	5600	3400	2200
17	q	4500	2500	2000

Will man zusätzlich pro Filiale den ausgewiesenen Gewinn in Dollar umrechnen (der Kurs sei mit 1 $ = 1,50 DM festgelegt) und als Variable `gewinnd` der Datendatei hinzufügen, so ist folgendes Programm nötig:

```
data sasuser.daten12;
set sasuser.daten12;
gewinnd=gewinn/1.5;
proc print data=sasuser.daten12;
var gewinn gewinnd;
run;
```

Es liefert den folgenden Output:

OBS	GEWINN	GEWINND
1	500	333.33
2	-400	-266.67
3	100	66.67
4	1700	1133.33
5	1400	933.33

6	1900	1266.67
7	3300	2200.00
8	300	200.00
9	100	66.67
10	1400	933.33
11	2000	1333.33
12	-400	-266.67
13	2000	1333.33
14	1100	733.33
15	2400	1600.00
16	2200	1466.67
17	2000	1333.33

0.5.2. Arithmetische und statistische Funktionen

SAS bietet auch zahlreiche **arithmetische Funktionen** an, mit deren Hilfe Datentransformationen vorgenommen werden können.

Die Syntax für eine Datentransformation mit Hilfe einer arithmetischen Funktion lautet:

Variable = Funktionsname(argument[1,argument2,....])

Argument kann dabei entweder eine einzige Zahl oder ein Variablenname sein. Bei der Angabe eines Variablennamens führt SAS die Funktionsvorschrift für alle vorhandenen Werte der Variablen aus.

SAS bietet u.a. die folgenden *Funktionen* an (im folgenden wird Argument mit '*ARG*' abgekürzt):

ABS(ARG)	berechnet den Absolutbetrag des Arguments.
MOD(ARG1,ARG2)	Modulo-Algebra. Berechnet den Rest der Division von *ARG1/ARG2*.
SIGN(ARG)	ermittelt das Vorzeichen des Arguments.
SQRT(ARG)	ermittelt die Quadratwurzel des Arguments.
ROUND(ARG)	rundet auf den nächsten ganzzahligen Wert des Arguments.
EXP(ARG)	berechnet den Wert der Exponentialfunktion des Arguments zur Basis e (Eulersche Zahl).

LOG(ARG)	berechnet den Logarithmus des Arguments zur Basis e.
LOG2(ARG)	berechnet den Logarithmus des Arguments zur Basis 2.
LOG10(ARG)	berechnet den Logarithmus des Arguments zur Basis 10.
SIN(ARG)	berechnet den Sinus des Arguments.
COS(ARG)	berechnet den Cosinus des Arguments.
LAG[K](ARG)	verschiebt alle Werte des angegebenen Arguments in ihrer Reihenfolge um k Positionen ($k \leq 100$) nach hinten. Die freiwerdenden Positionen 1, ..., (k-1) werden mit fehlenden Werten (Symbol '.') besetzt. Für k = 1 genügt die Angabe *LAG(ARG)* und alle Werte werden um eine Position nach hinten verschoben.

Wird eine Funktionsanweisung innerhalb eines DATA-Schrittes angegeben, in dem Variablen eingelesen werden, so weist die erzeugte Variable - falls die Funktion für einen **konkreten** Wert und nicht für alle Variablenwerte berechnet wird - den resultierenden Funktionswert nicht nur einmal auf, sondern entsprechend der **Anzahl** der Werte der übrigen Variablen.

Soll der in der Datendatei `sasuser.daten12` in Dollar umgerechnete Gewinn der Filialen (Variable `gewinnd`) noch auf ganzzahlige Werte gerundet und der Datendatei hinzugefügt werden, so kann dies auf die folgende Weise geschehen:

```
data sasuser.daten12;
set sasuser.daten12;
gewinnr=round(gewinnd);
proc print data=sasuser.daten12;
var gewinn gewinnd gewinnr;
run;
```

Diese Anweisungen ergeben dann den folgenden Output:

OBS	GEWINN	GEWINND	GEWINNR
1	500	333.33	333
2	-400	-266.67	-267
3	100	66.67	67
4	1700	1133.33	1133
5	1400	933.33	933
6	1900	1266.67	1267
7	3300	2200.00	2200
8	300	200.00	200
9	100	66.67	67
10	1400	933.33	933
11	2000	1333.33	1333
12	-400	-266.67	-267
13	2000	1333.33	1333
14	1100	733.33	733
15	2400	1600.00	1600
16	2200	1466.67	1467
17	2000	1333.33	1333

SAS bietet auch zahlreiche **statistische** Funktionen:

Verteilungsfunktionen

Hier muß beachtet werden, daß SAS nur den Wert der Verteilungsfunktion für einen gegebenen Wert x (also für eine gegebene Zahl) berechnet.

POISSON(λ,x) berechnet für den Wert x den Wert der Verteilungsfunktion einer POISSON-Verteilung mit dem Parameter λ.

PROBBETA(x,a,b) berechnet für den Wert x den Wert der Verteilungsfunktion einer Beta-Verteilung mit den Parametern a und b.

PROBBNML(θ,n,x) berechnet für den Wert x den Wert der Verteilungsfunktion einer Binomial-Verteilung mit den Parametern θ und n.

PROBCHI(x,k[,δ]) berechnet für den Wert x den Wert der Verteilungsfunktion einer χ^2-Verteilung mit k Freiheitsgraden. Wird für δ ein Wert angegeben, so wird für x der Wert der Verteilungsfunktion