

Neu in CNTA und CNTW

2. 4. 2018

Ränder MARGINS für die Ausgabe mit PAGE

Das Statement PAGE erhält den zusätzlichen Parameter:

MARGINS = < L < : n > > < , R < : n > > < , T < : n > > < , B < : n > >

Diese Angabe legt die Randbreiten und Randlinien der Ausgabeseiten fest. Ohne MARGINS wird ohne Randbreiten. Zum Beispiel erzeugt

```
MARGINS=T:3, L:10
```

einen oberen Rand von 3 Zeilen Höhe und einen linken Rand von 10 Zeichenpositionen Breite.

- L** Angaben zum linken Rand des Blattes.
- R** Angaben zum rechten Rand des Blattes.
- T** Angaben zum oberen Rand des Blattes.
- B** Angaben zum unteren Rand des Blattes.
- n Hinter T und B ist dies die Anzahl Zeilen über und unter den Ausgabedaten einer Seite. Hinter L und R ist es die Anzahl Zeichenpositionen rechts und links der Ausgabedaten. Die Texte HEADL, HEADC, HEADR, FOOTL, FOOTC, FOOTR werden *innerhalb* des oberen und unteren Randes gedruckt. Reicht der hier angegebene Rand nicht für diese Texte, so vergrößert das Programm die Ränder automatisch.

15.34. 2018

Größere Tabellen für die EXCEL-Ausgabe

Bisher waren Kreuztabellen mit dem Statement PAGEP nur für Seitengrößen im DIN A4 Format möglich. Größere Tabellen ließen sich nur mit PAGE erzeugen. Damit konnten Tabellengestaltungen mit verschiedenen Schriften, Logos, Linien und Farben nicht nach EXCEL übertragen werden.

Im Statement PAGEP ermöglicht der neue Parameter FORM=EXCEL jetzt Tabellen von 200 cm Breite und 50 cm Höhe. Bei normaler Schriftgröße sind damit EXCEL-Tabellen mit über 100 Zeilen und Spalten möglich, bevor ein Umbruch der Tabellen nötig wird.

Eine Ausgabe als PDF-Datei ist damit ebenfalls möglich, auf Drucker allerdings nur bei sehr großen Papierformaten sinnvoll.

16. 2. 2018

INSERT=NO im XTAB-Statements

INSERT=NO setzt die Angabe INSERTS aus früheren XTAB-Statements außer Kraft.

2. 1. 2018

Überarbeitung des WEIGHT-Statements

Die mögliche Anzahl von Randbedingungen wurde für große Wichtungsaufgaben von 30 auf 100 erhöht. Gleichzeitig wurden Laufzeiten und Speicherbedarf solcher Arbeiten reduziert.

Dabei wurde die Syntax für den Parameter FUNCTION verändert:

FUNCTION=E	optimiert die Effektivitätswerte
FUNCTION=S	optimiert den Informationsgehalt nach Shannon
FUNCTION=M	vermeidet kleine Gewichte nahe 0 bei Schonung der Effektivitätswerte

Die bisherigen Parameter werden weiter akzeptiert und folgendermaßen zugeordnet:

FUNCTION=1 und FUNCTION=5	auf	FUNCTION=S
FUNCTION=2 und FUNCTION=3	auf	FUNCTION=E
FUNCTION=4	auf	FUNCTION=M

Schließlich wurde FUNCTION=I neu eingeführt:

Diese Funktion verzichtet auf die Optimierungen der bisherigen Funktionen, erlaubt aber größere Aufgabenstellungen mit sehr vielen und umfangreichen Randbedingungen, ohne auf Laufzeit- und Speichergrenzen zu stoßen. Außerdem liefert sie bei nicht vollständig lösbarer Wichtungsaufgaben geringere Soll-Ist-Abweichungen, jedoch mit Abstrichen bei Effektivität und Informationsgehalt.

30.3.2017

Erweiterung des END-Statements

```
-END < | CASE | >
      | ALL |
```

Die hinter END stehenden Folgestatements werden nicht mehr ausgeführt.

Ohne weitere Angaben wird die Verarbeitung des laufenden Falles direkt hinter den Folgestatements fortgesetzt. CASE beendet darüber hinaus die Verarbeitung des laufenden Falles durch alle noch folgenden Statements.

ALL beendet die gesamte Verarbeitung vorzeitig.

28. 1. 2017

Erweiterung von SORT in XTAB

Es wurde die Möglichkeit geschaffen, Gruppen von Zeilen und Spalten in den Tabellen zu sortieren ohne die Zeilen und Spalten innerhalb der Gruppen zu vertauschen. Die Auswahl der Gruppen erfolgt über die Klassenbuchstaben A ... Z in eckigen Klammern:

Die Angaben

```
ROW=TOTAL;B{C1.1;...5};B{C2.1;...3}
SORT=ROWS1 [B]
```

vertauschen die beiden Klassen B abhängig von den Werten ihrer ersten Zeilen. Die Zeilen innerhalb der Klassen B bleiben unsortiert.

Im Beispiel:

```
ROW=TOTAL;B{C1.1;...5};B{C2.1;...3}
SORT=ROWS1 [B] (B)
```

tauscht [B] zunächst die Klassen B gegeneinander aus, abhängig von den Werten ihrer ersten Zeilen. Danach sortiert (B) die Zeilen innerhalb der beiden Klassen B{ }.

Mehrstufige Klassenangaben begrenzen die Vertauschungen:

```
ROW=A{C10.1;B{C1.1;D{C1.2;...4}}};
      B{C2.1;D{C2.2;...5}}};
      A{C10.1;B{C3.1;D{C3.2;...6}}};
SORT=ROWS1 [A] [B] (D)
```

Zunächst vertauscht [A] die beiden Klassen A miteinander, da diese nicht in übergeordneten Klassen liegen.

Danach vertauscht [B] die beiden Klassen B{C1.1 ... } und B{C2.1 ... } gegeneinander, da sie in der gleichen übergeordneten Klasse A liegen. Die Klasse B{C3.1 ... } wird nicht verschoben, da sie in einer getrennten Klasse A liegt. Abschließend sortiert (D) die Zeilen jeweils innerhalb der drei Klassen D{ }.

Um dies zu ermöglichen, waren die bisherigen Verarbeitungsregeln für Klassenbuchstaben A{ } ... Z{ } zu ändern. Die Statements

```
ROW=TOTAL;A{C1.1;...5};A{C2.1;...3}
SORT=ROWS1 (A)
```

sortierten bisher die Zeilen C1.1; ... 5 und C2.1;...3 gemeinsam, was wenig sinnvoll war.

Jetzt werden die Zeilen C1.1; ... 5 und C2.1;...3 getrennt voneinander sortiert. Dafür war bisher zu schreiben:

```
ROW=TOTAL;A{C1.1;...5};B{C2.1;...3}
SORT=ROWS1 (A) (B)
```

10. 1. 2017

INDEX-Statement mit HEAD=NO

Die Angabe HEAD=NO im Statement INDEX unterdrückt die Ausgabe der Überschrift *Inhaltverzeichnis*. Die Indexverweise beginnen ganz oben auf dem Blatt.

11.11 .2016

N-Variable als Datum in SPSS-Dateien ausgeben

Die Folgestatements von OUTPUT-EXT mit SPSS wurden um die Angabe DATE erweitert. Bei der Ausgabe von N-Variablen in SPSS-Dateien sorgt dies dafür, dass die daraus erstellte SPSS-Variable ihre Werte als Datum anzeigt. Dazu muss die N-Variable das Datum in der Form JJJJ MM TT enthalten. Dieses darf nicht vor dem 1. 1. 1970 liegen. Zum Beispiel:

```
OUTPUT-EXT SPSS ...  
-N22 DATE
```

2.10 .2016

PRINT=EMPTY in XTAB

PRINT... = EMPTY gibt wie PRINT... = NO an der entsprechenden Tabellenposition kein Zählergebnis aus, stellt aber zusätzlich in den CSV- und XLSX-Dateien eine leere Spalte an die PRINT...-Position.

15. 9 .2016

Effektstärken für t-Tests

Für die t-Tests mit MTT und MTTU ist es jetzt möglich, Schwellenwerte 0.001 ... 99.999 der Effektstärken nach J. Cohen vorzugeben. Zum Beispiel

```
PRINT=SIG, 5, E0.2, E0.5, E0.8
```

Danach werden nur solche Wertepaare mit den Buchstaben a ... z markiert, die sich im Sinne des t-Tests auf dem Signifikanzniveau von 5% unterscheiden und zusätzlich die Effektstärke 0,2 überschreiten. Effektstärken zwischen 0,2 und 0,5 werden durch Kleinbuchstaben gekennzeichnet, solche zwischen 0,5 und 0,8 durch Großbuchstaben und Stärken über 0,8 durch Großbuchstaben mit Ausrufungszeichen dahinter.

17. 8 .2016

Erweiterungen zur Markierung signifikanter Unterschiede bei t-Tests

Die maximale Anzahl von Markierungszeichen hinter TMARK wurde von 50 auf 200 erhöht. Die zusätzliche Angabe von M hinter PRINT=SIG stellt bei signifikanten Unterschieden ein Minuszeichen - vor das Markierungszeichen a ... z des kleineren Wertes. Zum Beispiel:

```
PRINT=SIG, 5, M
```

24.6.2016

Neue Datumsfunktionen

Folgende Datumsfunktionen lassen sich in logischen und numerischen Ausdrücken verwenden.

- WEEK** Wochennummer eines Datums nach der ISO-Norm 8601.
 Enthält die Variable N2 das Datum 30.6.2016 in der Form 20160630, so stellt das Statement

$$-N1=WEEK(N2)$$
 die Wochennummer 201626 in die Variable N1. Die Jahresangabe in dem Ergebnis ist erforderlich, da die ersten Tage eines Jahres in die letzte Woche des Vorjahres fallen können und die letzten Tage eines Jahres in die erste Woche des Folgejahres.
 Das folgende Statement löst die Wochennummer 26 des obigen Beispiel aus der Variablen N1 heraus und stellt sie in die Variable N3:

$$-N3=N1-FL(N1/100)*100$$
 Ein ungültiges Datum beantwortet die Funktion mit dem Ergebnis Z0 (keine Angabe) und einer Fehlermeldung im Zählprotokoll. Der Eingabewert Z0 liefert ebenfalls das Ergebnis Z0, jedoch ohne Fehlermeldung.
- DAYW** Wochentag 1 ... 7 eines Datums für Montag ... Sonntag.
 Enthält die Variable N2 das Datum 30.6.2016 in der Form 20160630, so stellt das Statement
 Das Statement

$$-C1=DAYW(N2)$$
 setzt die Merkmale 1 ... 7 der Variablen C1 abhängig vom Datum in der Variablen N2. Enthält N2 das Datum 30.6.2016 in der Form 20160630, so wird das Merkmal 4 der Variablen C1 für *Donnerstag* gesetzt.
 Ein ungültiges Datum beantwortet die Funktion mit dem Ergebnis Z0 (keine Angabe) und einer Fehlermeldung im Zählprotokoll. Der Eingabewert Z0 liefert ebenfalls das Ergebnis Z0, jedoch ohne Fehlermeldung.
- DAYY** Tag 1 ... 366 eines Datums im Jahr.
 Enthält die Variable N2 das Datum 30.6.2016 in der Form 20160630, so stellt das Statement

$$-N1=DAYY(N2)$$
 den Tag 182 des Jahres in die Variable N1.
 Ein ungültiges Datum beantwortet die Funktion mit dem Ergebnis Z0 (keine Angabe) und einer Fehlermeldung im Zählprotokoll. Der Eingabewert Z0 liefert ebenfalls das Ergebnis Z0, jedoch ohne Fehlermeldung.

3.5.2016

Lange Signifikanzmarkierungen bei t-Tests

t-Tests mit vielen signifikanten Unterschieden können sehr lange Ketten von Buchstaben in den Kreuztabellen erzeugen. Bei Platzmangel in den Spalten werden diese in mehrere Zeilen zerlegt.

Bisher mussten dafür hinter ROWS passende Leerzeilen eingefügt werden, um das Überschreiben von Zählergebnissen zu verhindern. Das Programm sorgt jetzt automatisch für den erforderlichen Platz.

16.3.2016

Kopf- und Fußzeilen in PAGE und PAGEP

Die bisherigen Angaben HEAD, FOOT, NR, DATE und TIME werden durch die neuen Angaben HEADL, HEADC, HEADR sowie FOOTL, FOOTC und FOOTR ersetzt, die mehr Gestaltungsmöglichkeiten bieten. Die alten Parameter sind weiter möglich, sollten aber in neuen Statementdateien vermieden werden, da die Vermischung von alten und neuen Parametern zu unerwarteten Ergebnissen führen kann.

HEADL	=	NO
HEADC		<,'t'> <, FTi > <, K >
HEADR		

FOOTL	=	NO
FOOTC		<,'t'> <, FTi > <, K >
FOOTR		

Diese Angaben erzeugen Kopf und Fußtexte auf jedem Blatt.

Ohne sie erscheinen Name und laufende Versionsnummer des Programms mit dem Namen des Anwenders als Blattüberschrift links oben und die Seitennummerierung: *Seite 1*...rechts unten.

HEADL Kopftext links oben auf dem Blatt.
HEADC Kopftext oben mittig auf dem Blatt.
HEADR Kopftext rechts oben auf dem Blatt.

FOOTL Fußtext links unten auf dem Blatt.
FOOTC Fußtext unten mittig auf dem Blatt.
FOOTR Fußtext rechts unten auf dem Blatt.

- 't' Ein- oder mehrzeiliger Text, wie im Abschnitt *Textzeilen* beschrieben.
Die Angaben ~DATE~ und ~TIME~ stellen Datum und Uhrzeit des Auswertungslaufs an beliebiger Stelle in die Textzeilen. ~NR~ setzt die Seitennummer des laufenden Blattes ein.
- FTi** Schrift FONTi aus dem Statement PAGEP für den Text.
Fehlt diese Angabe, so wird für die Kopftexte die Schrift FONT4 und für die Fußtexte FONT2 verwendet.
- K** Werden mit der Angabe LOGOS rechts oder links Logos auf das Blatt gestellt, so verschieben diese Logos die Kopf- und Fußtexte. Mit der Angabe K überschreiben die Texte die Logos..
- NO** Nach dieser Angabe wird der jeweilige Text nicht mehr ausgegeben.

Zum Beispiel:

```
PAGEP HEADL='Studie 325'/'Welle 2: ~DATE~' HEADR='Seite ~NR~'
```

Änderungen im Statement INDEX

Hier werden entsprechend die Parameter PAGEHEADL, PAGEHEADC, PAGEHEADR sowie PAGEFOOTL, PAGEFOOTC, PAGEFOOTR neu eingeführt. Die alten Angaben PAGEHEAD und PAGEFOOT bleiben weiterhin wirksam, sollten aber in neuen Statementdateien nicht mehr verwendet werden.

3.2.2016

BZ in IF-Statements

In den IF-Statements ist hinter dem logischen Ausdruck die Angabe **BZ** für *blank to zero* möglich. Diese Angabe ersetzt fehlende Werte von N-Variablen im Ausdruck durch Null. In dem Statement

```
-IF (N10+N11)>10
```

ist der logische Ausdruck nicht erfüllt, wenn N10 den Wert Z0 und N11 den Wert Z0 besitzt.

Durch BZ hinter dem Ausdruck wird der Wert Z0 in N10 als 0 verarbeitet, so dass der logische Ausdruck wahr wird.

19.1.2016

Anzahl Angaben einer Variablen in XTAB

Hinter ROW, COL und FILTER ist für jede C-Variable Cn die Angabe Cn.F möglich.

Diese Angabe erzeugt zu jeder möglichen Anzahl von Angaben pro Fall einen Operanden.

Wird die Variable C10 zum Beispiel drei Merkmale definiert, so erzeugt C10.F einen Operanden für 0 Angaben, und je einen für 1 Angabe, für 2 Angaben und für 3 Angaben.

Für jeden statistischen Fall erhält nur der Operand den Wert 1, dessen Anzahlbedingung erfüllt ist. Die übrigen erhalten den Wert 0. In den Tabellen erscheinen nur die Zeilen oder Spalten, zu denen auch Fälle vorliegen.

Die Statements:

```
XTAB ROW=TOTAL;C0.F
      COL=TOTAL;C1
```

könnten die folgende Tabelle erzeugen:

	TOTAL	Variable C1		
		1	2	3
TOTAL	22	10	7	5
Variable C0				
3 Angaben	2	1	1	-
2 Angaben	6	3	1	2
1 Angabe	12	4	5	3
0 Angaben	2	2	-	-

Die festen Texte *Angabe* und *Angaben* lassen sich mit den Folgestatements `-FS` und `-FP` von DEFS den eigenen Wünschen anpassen.

7.1.2016

Folgestatements zu XTAB

Die Wertezuweisungen können jetzt Zeilen- und Spaltensummen erzeugen:

```
XTAB ...
-R (-1) = R (1 . . . -2)
```

Hier bildet $R(1...-2)$ die Summen aus der ersten bis zu vorletzten Zeile der Tabelle und stellt das Ergebnis in die letzte Zeile. Entsprechend erzeugt

```
XTAB ...
-C (1) = R (A)
```

die Summe aus den Spalten mit dem Klassenbuchstaben A und stellt das Ergebnis in die Spalte 1.

Label-Texte für dichotome Variable in der SPSS-Ausgabe

Werden C-Variable über OUTPUT-EXT in eine SPSS-Datei als dichotome Variable im U-Format ausgegeben, so wurde bisher als Variablen-Label der Merkmalstext und als Werte-Label der Variablentext verwendet.

Im Statement OUTPUT-EXT lassen sich jetzt mit dem Parameter CSTD davon abweichende Texte festlegen:

```
CSTD = | U <, label > |
      | UM <, label > |
```

Für *label* sind folgende Angaben möglich:

label	Variable Label	Value Label
I	Merkmalstext	1 = Variablentext
IV	Merkmalstext Variablentext	1 = gewählt 0=nicht gewählt
VI	Variablentext Merkmalstext	1 = gewählt 0=nicht gewählt
V	Variablentext	1 = Merkmalstext

Ohne diese Angabe wird der Labeltyp I verwendet. Die Value-Label der Labeltypen IV und VI lassen sich durch beliebige andere Texte ersetzen. Dazu dienen hinter dem Statement DEFS die Folgestatements QUOTED für *gewählt* und NOTQUOTED für *nicht gewählt*. Zum Beispiel:

```
DEFS
-QUOTED 'genannt'
-NOTQUOTED 'nicht genannt'
```

In den Folgestatements von OUTPUT-EXT lassen sich die Textregeln auch für einzelne Variable festlegen.

Zum Beispiel:

```
-C10 U,IV 1='genannt' 0='nicht genannt'
```

15.7.2015

Transponierte Daten mit OUTPUT-INT

Die Angabe TRANS speichert alle Variablenwerte neben der normalen Form ein zweites Mal transponiert in der Datei. Der neue Parameter TRANSONLY speichert die Variablenwerte nur einmal in der transponierten Form. Das reduziert die Größe der internen Datei, ist aber nur zu empfehlen, wenn niemals ein größerer Teil der Variablen gleichzeitig auszuwerten ist.

Außerdem wurde die Laufzeit zur Erstellung der transponierten Daten reduziert: Es ist dazu nur noch ein Datendurchlauf erforderlich.

9.6.2015

Erweiterung in XTAB für PRINT ... IF

Sollte die Aufbereitung eines Zählergebnisses in XTAB von den Werten der ersten Zeile oder Spalte abhängig gemacht werden, war eine Angabe der folgenden Art erforderlich:

```
XTAB PRINT=ABS IF MAXZ (R(1)) < 80, '&*', F1
```

Dafür ist jetzt die etwas näher liegende Form möglich:

```
XTAB PRINT=ABS IF ROW1 < 80, '&*', F1
```

18.4.2015

Erweiterung der Variablennamen

Variablennamen dürfen jetzt zwischen zwei und zehn Zeichen lang sein und aus den Buchstaben A ... Z, den Ziffern 0 ... 9 und dem Sonderzeichen _ bestehen. Umlaute Ä, Ö, Ü sowie ß sind nicht möglich. Wie bisher wird der Variablentyp über das erste Zeichen C, N oder T festgelegt.

Endet ein Variablenname mit einer Zahl, so ergeben führende Nullen keine neue Variable.

So bezeichnen NX001, NX01 und NX1 dieselbe Variable. Dagegen sind NX und NX0 voneinander verschieden.

Bei der Auswahl von Variablen im G-Statement hinter DEFS und in den Folgestatements von OUTPUT-INT, OUTPUT-EXT und CODEBOOK sind jetzt auch die Wildcard-Zeichen * und ? möglich.

Zum Beispiel:

-CA??	wählt alle Variablen, die aus vier Zeichen bestehen und mit CA beginnen,
-C*	wählt alle Variablen, die mit dem Zeichen C beginnen,
-N*1...9	wählt alle N-Variablen, die mit den Nummern 0 ... 9 enden, etwa NXY5.

15.3.2015

Einfärben von Tabellenspalten mit INSERT

Im Statement XTAB lassen sich ausgewählte Tabellenspalten über die Angabe INSERT mit Grautönen G1 ... G16 oder G1 ... 16 oder CO1 ... CO16 unterlegen.

Bisher unterlegte das Statement

```
XTAB COL=C15.1:G8
```

die Tabellenspalte mit dem Grauton G8. Dabei wurde neben den Zählerwerten auch die unterste Stufe der Kopftexte eingefärbt, nicht jedoch der Variablentext von C15 und auch nicht darüber stehende Texte aus Staffellungen der Art

```
XTAB COL={C1}{C15.1:G8}
```

Die neue Angabe

```
XTAB COL={C1}{C15.1} INSERT=C(1):G8
```

unterlegt neben den Zählerwerten der Spalte auch all ihre Kopftexte mit dem Grauton G8.

11. 8. 2014

Statement CLEAN zur Datenbereinigung

Das neuen CLEAN-Statements sind Folgestatements zu ADD-PROC, MOD-PROC, UPD-PROC und OUTPUT-EXT. Sie dienen dazu, fehlerhafte Variablenwerte zu bereinigen und darüber eine Statistik in das Zählprotokoll auszugeben.

In CLEAN-Statement lassen sich beliebig komplizierte Prüfbedingungen formulieren. Wie bei IF-Statements werden die dahinter liegenden Statements nur ausgeführt, wenn die Prüfbedingung erfüllt ist. Andernfalls wird die Verarbeitung beim nächsten CLEAN- oder ECLEAN-Statement fortgesetzt. Das ECLEAN-Statement beendet die CLEAN-Funktion und führt zur normalen Verarbeitung ohne Protokollierung zurück.

Auf ein CLEAN-Statement folgen Wertezuweisungen zur Korrektur der fehlerhaften Variablenwerte. Weichen dabei die alten von den neuen Variablenwerten ab, so wird dies als Korrektur in der CLEAN-Statistik ausgewiesen. Diese Statistik zeigt die Anzahl der Korrekturen zu den CLEAN-Statements und den dazugehörigen Wertezuweisungen.

Im Bereich eines CLEAN-Statement sind alle Folgestatements zulässig, ausgenommen Wertezuweisungen mit externen Datenfeldern links vom Gleichheitszeichen. Insbesondere sind DO- IF- PRT- und TST-Statements möglich.

Die Statements:

```
add-proc
-clean c1.1 & c0.1
-n1=5
-c2.1=1
-clean ^c1.1 & c0.1
-n1=10
-c2=1
-eclean
```

könnten folgende Statistik im Zählprotokoll erzeugen:

		=====		=====	
		CLEAN-Protokoll		ADD-PROC	
		=====		=====	
Statements		Bedingung erfüllt		korrekturen	
18	CLEAN c1.1 & c0.1.....	49	96.1%	2	3.9%
19	n1=5.....			1	2.0%
20	c2.1=1.....			1	2.0%
21	CLEAN ^c1.1 & c0.1.....	1	2.0%	1	2.0%
22	n1=10.....			1	2.0%
23	c2=1.....			1	2.0%
-----		-----		-----	
Gesamt		51	100.0%	3	5.9%
Anzahl Fälle mit korrektoren		3	5.9%	4	7.8%
		=====		=====	
		CLEAN-Protokoll pro Fall			
		=====		=====	
Fall-Identifikation		Korrekturen		Statement	
433		2	2	21	
422		1	1	18	
411		1			

5.7.2014

Externe Dateien im XLSX-Format

Die Statements INPUT-EXT und FETCH-FILE erlauben jetzt die Angabe XLSX. Damit lassen sich Dateien im XLSX-Format von Excel einlesen. Die Verarbeitung der Eingabedaten erfolgt analog zu CSV-Dateien mit der Angabe SEP in INPUT-EXT.

Das Statement OUTPUT-EXT kann über die Angaben XLSX und AUTOXLSX derartige Dateien für Excel erstellen.

31. 5. 2014

Operand KOMP in ROW, COL und FILTER von XTAB

Dieser Operand liefert als Komplement für C-Variable die Zahl der Fälle, die in den davorliegenden Zeilen oder Spalten nicht gezählt wurden.

Zu Beginn von ROW, COL und FILTER besitzt KOMP für jeden statistischen Fall den Wert 1 oder *wahr*. Sobald in ROW, COL oder FILTER ein Operand vor KOMP der Form *Cn.m* oder *Cn.Zi* oder ein logischer Ausdruck erfüllt ist, erhält KOMP den Wert 0 oder *falsch*. Zum Beispiel:

```
ROW=TOTAL; C1.1; C1.Z0; C1.2 | .3; KOMP
```

Hier liefert KOMP die Anzahl der Fälle, für die weder das Merkmal *C1.1* erfüllt ist, noch die Bedingung *C1.Z0* vorliegt und auch der logische Ausdruck *C1.2/.3* nicht wahr ist. Operanden wie TOTAL, N1 oder MEAN(N1) haben keinen Einfluss auf KOMP.

Innerhalb einer Staffel {...} ist KOMP nur abhängig von den Zeilen oder Spalten der gleichen Staffel. Zum Beispiel:

```
ROW=TOTAL; C10.1; {C1.1; C1.2; KOMP} {C2.1; C2.2; KOMP}
```

Hier steht KOMP in der ersten Staffel für die Fälle, für die *C1.1* und *C1.2* nicht erfüllt sind und KOMP in der zweiten Staffel für die Fälle, für die *C2.1* und *C2.2* nicht erfüllt sind.

Hinter einer Stafflung {...} ist KOMP nur abhängig von den Zeilen oder Spalten, die zwischen dem Ende der Staffel und KOMP liegen. Zum Beispiel:

```
ROW=TOTAL; {C10.1} {C1.1; C1.2}; C2.1; C2.2; KOMP
```

Hier liefert KOMP die Anzahl der Fälle, für die *C2.1* und *C2.2* nicht erfüllt sind; dies unabhängig von den davorstehenden Angaben.

25. 5. 2014

Logische Ausdrücke in ROW, COL und FILTER von XTAB

In XTAB sind jetzt alle logischen Ausdrücke möglich, auch mit dem neuen Operanden KOMP. Sie werden durch die Operatoren `&` `|` `¬` `<` `>` `=` von den numerischen Ausdrücken unterschieden.

Zum Beispiel:

```
ROW=(C17.1|.4)&C20.Z1
```

Logische Ausdrücke in XTAB werden – anders als die numerischen – fallweise ausgewertet. So werden in dem Beispiel

```
ROW=( (N1*N2) > 17) & C20.1
```

für jeden statistischen Fall die Werte von N1 und N2 miteinander multipliziert. Ist das Ergebnis größer 17 und gleichzeitig die Bedingung C20.1 erfüllt, so wird der vorliegende Fall in der entsprechenden Tabellenzeile gezählt.

Der Ausdruck

```
ROW=N1*N2
```

enthält dagegen keinen der Operatoren `&` `|` `¬` `<` `>` `=`. Es liegt also ein numerischer Ausdruck vor, für den zunächst die Summen von N1 und N2 für alle statistischen Fälle getrennt gebildet werden. Erst bei der Tabellenausgabe werden diese Summen miteinander multipliziert.

22. 4. 2014

Erstellung von SPSS-SAV-Dateien durch OUTPUT-EXT

Das Statement OUTPUT-EXT erstellt jetzt direkt SPSS-SAV-Dateien, ohne Umweg über Syntaxdateien im SPS-Format. Dazu ist im Ausgabeformat SPSS lediglich die Angabe VNAME wegzulassen und hinter FNAME die Dateierweiterung SAV zu verwenden.

27. 3. 2014

Überarbeitung des WEIGHT-Statements

Mit FUNCTION=4 wird ein weiteres Optimierungsverfahren eingeführt, das den Informationsverlust im Sinne der Shannon-Entropie durch die Gewichte minimiert. Entsprechend erscheint jetzt im Wichtungsprotokoll zusätzlich zum Effektivitätswert in Prozent auch der Informationsverlust in Promille.

Mit FUNCTION=5 wird eine Weiterentwicklung von FUNCTION=1 eingeführt. Bei guter Vermeidung sehr kleiner Gewichte nahe bei 0 werden bessere Effektivitätswerte erreicht. Dazu wurde die Zielfunktion

$$\sum_i \left(x_i^2 + \frac{1}{x_i^2} \right)$$

ersetzt durch

$$\sum_i \left(x_i^2 + \frac{1}{x_i} \right)$$

Das Verfahren FUNCTION=1 bleibt für die Kompatibilität alter Auswertungen erhalten, wird aber für neue Aufgaben nicht mehr empfohlen.

Bei fehlender FUNCTION-Angabe wird jetzt FUNCTION=4 gewählt.

Schließlich werden im Wichtungsprotokoll alle Merkmalskombinationen mit besonders großen oder kleinen Gewichten angezeigt.

24. 2. 2014

Übergabe von Variablentexten in das Editierfenster für Statements

Die Texte der Variablenanzeige in CNTW lassen sich durch Mausklick in das Editierfenster für Statements kopieren.

Durch Klicken auf Variablentexte und Textelemente wird im Statementfenster eine vollständige Variablendefinition für DEFS erzeugt. Das ermöglicht eine bequeme Korrektur von Texten aus internen Dateien.

Klicks auf Merkmals- und Gruppentexte übertragen lediglich die Texte selbst.

14. 2. 2014

Ausgabe von Zählprotokollen als txt-Dateien

Aus der Anzeige der Zählprotokolle in CNTW lassen sich txt-Dateien erstellen.

31. 1. 2014

Eingabe von Triple-S-Dateien

CNTW kann jetzt Triple-S-Dateien einlesen und in CNT-interne Dateien umwandeln.
Dazu dient die Prototyp-Datei *Daten von Triple-S an CNT übergeben* (triplesCNT.ptt).

18. 1. 2014

Erweiterung der Startseite von CNTlight

Die Projektschaltflächen dieser Seite lassen sich mit gedrückter linker Maustaste neu positionieren.
Außerdem ist es möglich, sie auf mehrere Spalten zu verteilen. Dazu ist zunächst die Spalte der Projektschaltflächen an ihrem rechten Rand mit gedrückter linker Maustaste zu verbreitern.

10. 11. 2013

Externe Dateien im UTF8-Code

Externe Dateien INPUT-EXT, OUTPUT-EXT und FETCH-FILE lassen sich im UTF8-Code verarbeiten.
Dies gilt sowohl für alphanumerische ID- und KEY-Felder als auch für Textwerte in A-, AK- und AS-Feldern.

25. 7. 2013

Erweiterungen für Variable

Die Kapazität der numerischen Variablen wurde auf 18 Dezimalziffern erhöht, wobei auch 18 Nachkommastellen möglich sind.

Textvariable vertragen jetzt auch das UTF8-Format. Es ist daher möglich Texte aus externen Dateien im UTF8-Format direkt in Textvariable einzulesen. Alle Wertezuweisungen, Ausdruck von Variablenwerten sowie Codetransformationen zum Beispiel in OUTPUT-EXT werden korrekt im UTF8-Format ausgeführt.

25. 7. 2013

Erweiterungen für interne Dateien

Die bisherigen Größenbegrenzungen der internen Dateien sind aufgehoben: Alle vom Betriebssystem unterstützten Dateigrößen sind möglich.

Die Grenzen für transponierte Dateien (hohe Fallzahlen oder lange Variablenwerte) sind beseitigt. Die Laufzeiten zur Erstellung großer transponierter Dateien wurden spürbar reduziert.

Die maximale Länge der Datensätze ist auf 10 MB erhöht.

Numerische Fall-Identifikationen dürfen bis zu 18 Dezimalziffern besitzen und alphanumerische Fall-Identifikationen können bis zu 128 Zeichen lang werden.

Die Gesamtlänge der Fall-Identifikationen bei mehreren ID-Stufen ist nicht mehr begrenzt (bisher maximal 50 Zeichen).

26. 6. 2013

Erweiterungen für die t-Tests

Bisher waren die Buchstaben a, b, c ... z zur Markierung signifikanter Unterschiede fest vorgegeben. Mit dem neuen Parameter TMARK im Statement XTAB lässt sich eine abweichende Zeichenfolge für diesen Zweck wählen. Zum Beispiel:

```
XTAB TMARK='zyxwvuääöüßαβγδ'
```

Die Zeichen hinter TMARK werden in der hier angegebenen Reihenfolge zum Markieren der Zeilen oder Spalten verwendet. Es sind beliebige druckbare Zeichen möglich. Es ist jedoch zu beachten, dass für PRINT=SIG mit mehreren Signifikanzschwellen die schärferen Bedingungen durch Großbuchstaben gekennzeichnet werden. Dafür sollten hinter TMARK Zeichen gewählt werden, die Gegenstücke in Großschrift besitzen.

Weiter ist es jetzt möglich, gleichzeitig mit drei Signifikanz-Niveaus zu arbeiten. Zum Beispiel:

```
XTAB PRINT=sig,2,5,10
```

Wertepaare, die sich nur gemäß der schwächsten Bedingung (hier 10%) unterscheiden, werden wie bisher durch Kleinbuchstaben gekennzeichnet. Die die nächste Stufe wird durch Großbuchstaben angezeigt und die dritte durch Großbuchstaben mit Ausrufungszeichen dahinter.

Es ist jetzt möglich, signifikante Unterschiede in den Tabellen durch Farbe oder Grauton deutlich zu machen. Zum Beispiel

```
XTAB PRINT=sig,2,CO5
```

Findet der t-Test signifikante Unterschiede zwischen zwei Tabellenzellen, so werden die entsprechenden Markierungen mit der Farbe CO5 aus dem PRINT-Statement unterlegt. Zellen ohne t-Test-Markierungen werden nicht gefärbt. Die Angabe

```
XTAB PRINT=sig,2,CO5(L12)
```

färbt die ganze Tabellenzelle und nicht nur die Markierungszeichen.

Die Möglichkeiten, Zeilen und Spalten miteinander zu vergleichen, wurden erweitert. Zum Beispiel

```
XTAB PRINT=C(1 3)SIG(1...3)5 C(2)SIG(1...3 10)5
```

Hiermit werden die Spalten 1 bis 3 untereinander verglichen und die Spalte 2 zusätzlich mit der Spalte 10.

14. 6. 2013

Einfärben von Zeilen und Spalten in XTAB

Bisher hinterlegte das Statement

```
XTAB ROW=c0.1;.2:G5;.3 COL=c0.1;.2:G5;.3
```

die Zeilen und Spalten auf folgende Weise mit dem Grauton G5:

	Variablen- text C0	Variablen- text C0	Variablen- text C0
	Merkmal 1	Merkmal 2	Merkmal 3
Variablentext C0 Merkmal 1	255	255	255
Variablentext C0 Merkmal 2	255	255	255
Variablentext C0 Merkmal 3	255	255	255

Jetzt erzeugt das gleiche Statement diese Tabelle:

	Variablentext C0		
	Merkmal 1	Merkmal 2	Merkmal 3
Variablentext C0 Merkmal 1	255	255	255
Merkmal 2	255	255	255
Merkmal 3	255	255	255

12. 6. 2013

Vergleich unterschiedlich langer Textwerte

Der Vergleich

```
-T1='Modell11'  
-IF T1='Modell112'  
...
```

unter ADD-PROC, UPD-PROC und MOD-PROC liefert jetzt das Ergebnis *falsch* und nicht mehr *wahr*.

5. 5. 2013

externe Dateien im UTF8-Format

Die Statements INPUT-EXT, OUTPUT-EXT, FETCH-FILE und REPORT-FILE erlauben jetzt auch das Dateiformat UTF8.

28. 2. 2013

Selektion von Fällen im Statement CODEBOOK

Die Auswertung durch CODEBOOK lässt sich jetzt auf Teilmengen der statistischen Fälle beschränken: Dazu dient die Angabe:

CODEBOOK SELECT = logischer Ausdruck

Nur Fälle, die den logischen Ausdruck erfüllen, nehmen an der Wichtung teil.

15. 1. 2013

Ergänzen bestehender REPORT-Dateien

Das Statement REPORT-FILE hat den neuen Parameter APPEND erhalten:

Diese Angabe erhält bestehende REPORT-Dateien. Die mit -REPORT ausgegebenen Daten werden an das Ende der Datei gestellt. Existiert die Datei noch nicht, so wird sie neu angelegt. Ohne APPEND löscht das Programm eventuell vorhandene Dateien.

Zum Beispiel:

```
REPORT-FILE FNAME=Fehler.txt APPEND
```

12. 10. 2012

Umgebungsvariable in Dateinamen

In allen Dateinamen lassen sich Umgebungsvariable verwenden, sofern sie im Betriebssystem definiert sind. Zum Beispiel:

```
INPUT-INT FNAME=%STUDIEN%\NR%KUNDE%\STUDIE27.INT
```

mit den beiden Umgebungsvariablen STUDIEN und KUNDE.

30. 8. 2012

HOLECOUNT-Auswertung von CSV-Dateien

Es ist jetzt möglich, auch CSV-Dateien sinnvoll mit HOLECOUNT auszuwerten. Dazu wurde ein neues Auswertungsschema entwickelt. So könnten die Statements

```
INPUT-EXT FNAME='test6.csv'
          SEP=';'
HOLECOUNT
-K1 (2...4)
```

folgende Auswertung erzeugen:

Satz 1		Anzahl Fälle: 1204	
Satz/Spalte			
1/2			
keine Angabe		235	19,5%
eine Angabe		523	43,4%
zwei Angaben		327	27,2%
drei oder mehr Angaben		119	9,9%
1		368	30,6%
2		216	17,9%
3		211	17,5%
4		126	10,5%
5		99	8,2%
6		85	7,1%
7		85	7,1%
8		80	6,6%
9		80	6,6%
10		71	5,9%
11		66	5,5%
12		65	5,4%
1/3			
keine Angabe		1160	96,3%
eine Angabe		44	3,7%
-2,17		1	0,1%
0		1	0,1%
10		5	0,4%
20		14	1,2%
25		2	0,2%
30		14	1,2%
35		7	0,6%
1/4			
eine Angabe		1204	100,0%
Berlin		128	10,6%
Bremen		127	10,5%
Dresden		119	9,9%
Frankfurt		103	8,6%
Hamburg		139	11,5%
Hannover		131	10,9%
Köln		108	9,0%
Leipzig		106	8,8%
München		123	10,2%
Stuttgart		120	10,0%

Die Mehrfachnennungen in Spalte 3 stammen aus Datenwerten der Form:

2 5-7 12

die als Angaben 2, 5, 6, 7 und 12 verstanden werden.

15. 7. 2012

Häufigkeit von Werten der N- und T-Variablen in CODEBOOK

Die Angabe NVAL=FREQ im Statement CODEBOOK weist zu den N-Variablen alle vorkommenden Werte mit ihrer Häufigkeit aus.

Entsprechend zeigt die Angabe TVAL=FREQ zu den T-Variablen alle vorkommenden Texte mit ihrer Häufigkeit vor. Zum Beispiel:

CODEBOOK			
Anzahl Fälle.....		2888	
T1:255 andere Urlaubsländer			
Fälle mit Angaben.....	34	1,2%	
Fälle ohne Angaben.....	2854	98,8%	
Häufigkeit:			
Danemark.....	2	0,1%	0,1%
Dänemark.....	5	0,2%	0,2%
England.....	14	0,5%	0,7%
Portugal.....	4	0,1%	0,9%
dänemark.....	5	0,2%	1,0%
england.....	4	0,1%	1,2%
N1:1 A1. Freizeitaktivitäten: Aktiv Sport treiben			
Fälle mit Angaben.....	3568	100,0%	
Häufigkeit:			
1.....	721	20,2%	20,2%
2.....	1274	35,7%	55,9%
3.....	931	26,1%	82,0%
4.....	465	13,0%	95,0%
5.....	177	5,0%	100,0%

27. 6. 2012

Behandlung der Z0-Werte in OUTPUT-EXT mit SPSS und SPSS-UTF8

Neu eingeführt wurden in den Folgestatements die Parameter MV= zur Deklaration von Missing-Values und Z0= zur Umsetzung der Z0-Werte aus CNT in andere Werte. Zum Beispiel:

-N12 MV=5...7,12 Bei der Variablendefinition in SPSS werden die Werte 5 bis 7 und 12 als Missing-Values deklariert.

-C125 Z0=-999 In die SPSS-Datensätze wird für Z0 statt Blank der Wert -999 ausgegeben.

21. 6. 2012

Erweiterungen in OUTPUT-INT für die Datenerfassung mit CNTD

Es ist jetzt möglich, einzelne Variable während der Datenerfassung mit konstanten Werten zu füllen. Außerdem lassen sich zu jeder Variablen bei der Erfassung zulässige Werte vorgeben. Zum Beispiel:

-C10=1 5 7...9 24	Die Variable C1 erhält bei der Erfassung die Merkmalswerte 3, 5, 6, 7 und 17.
-C12 (3 5...7 17)	Es lassen sich nur die Merkmalswerte 3, 5, 6, 7 und 17 eingeben.
-N10[2] = -1.5	Die Variable N10[2] erhält bei der Erfassung den Wert -1,5.
-N10 (3 5...7 -17)	Es lassen sich nur die Werte 3, 5, 6, 7 und -17 eingeben.
-T10='Studie132'	Die Variable T10 wird bei der Erfassung mit dem Text <i>Studie132</i> gefüllt.
-T11=USER	Die Variable T11 wird mit dem User-Namen gefüllt, unter dem der Erfasser in Windows angemeldet ist.

5. 5. 2012

Erweiterungen zu INPUT-SEC

Die beiden folgenden Operanden sind in den Folgestatements von ADD-PROC, MOD-PROC, UPD-PROC und OUTPUT-EXT zulässig als auch hinter FILTER, ROW und COL in XTAB.

UPDSEC Dieser Operand besitzt den Wert 1 = *wahr* für statistische Fälle aus INPUT-EXT oder INPUT-INT, zu denen ein Fall mit passender Identifikation aus einer INPUT-SEC-Datei eingelesen wurde. In allen anderen Fällen besitzt UPDSEC den Wert 0 = *falsch*.

ADDSEC Dieser Operand besitzt den Wert 1 = *wahr* für statistische Fälle aus INPUT-SEC, zu denen kein Fall aus INPUT-EXT oder INPUT-INT mit passender Identifikation vorhanden ist. Die Bedingung ADDSEC ist auch für alle statistischen Fälle aus INPUT-SEC mit der Angabe ADD oder ADDP erfüllt. In allen anderen Fällen besitzt ADDSEC den Wert 0 = *falsch*.

Zum Beispiel hinter MOD-PROC:

```
-C1.1=ADDSEC | UPDSEC
```

Hier wird das Merkmal C1.1 gesetzt, wenn zu dem laufenden Fall Daten aus INPUT-SEC vorliegen, egal ob mit oder ohne Daten aus INPUT-EXT und INPUT-INT.

2. 5. 2012

Erweiterungen zum Statement FUSION

Die Angaben MIN und MAX zur Steuerung der der Zuweisung von den Spendern zu den Empfängern haben sich als unhandlich erwiesen und wurden durch folgende Angaben ersetzt:

$$\text{BESTFIT} = \left| \begin{array}{c} \text{ON} \\ n \end{array} \right|$$

Ohne diese Angabe werden die Spender möglichst gleich häufig verwendet. Dadurch bleiben die Häufigkeitsverteilungen des Spenderbestandes so weit wie möglich erhalten.

ON Jeder Empfänger erhält einen Spender mit größtmöglicher Ähnlichkeit der Verbindungsvariablen. Jeder Spender kann beliebig oft verwendet werden.

n Jeder Empfänger erhält ebenfalls einen Spender größtmöglicher Ähnlichkeit. Jeder Spender wird höchstens n mal verwendet.

LIMIT = n

Mindestwert $n=0\dots 100$ für die Ähnlichkeit zwischen Spender und Empfänger. Ohne diese Angabe werden bei schwierigen Zuordnungsproblemen auch Spender und Empfänger mit geringer Ähnlichkeit verbunden. Dabei steht 100 für maximale und 0 für minimale Ähnlichkeit.

LIMIT verhindert Verbindungen mit kleineren Ähnlichkeitswerten als n . Als Folge können Empfänger ohne Spender verbleiben.

Pro Verarbeitungslauf sind jetzt unbegrenzt viele FUSION-Statements ,möglich.

Außerdem lassen sich die Fusionen auf Teilmengen der Spender- und Empfängerdatei beschränken:

Dazu dienen die Angaben:

SELECT = logischer Ausdruck

Nur Fälle der Empfängerdatei, die den logischen Ausdruck erfüllen, nehmen an der Fusion teil. Allen anderen Fällen werden keine neuen Werte zugewiesen. Der logische Ausdruck darf nur Variable aus INPUT-INT oder DEFS enthalten.

Zum Beispiel vergibt

```
SELECT=C10.Z0
```

nur diejenigen Empfänger neue Werte aus der Spenderdatei FNAME, deren Variable C10 keine Angaben besitzt.

SELECTD = logischer Ausdruck

Nur Fälle der Spenderdatei FNAME, die den logischen Ausdruck erfüllen, nehmen an der Fusion teil. Alle anderen Fällen werden nicht zur Vergabe neuer Werte verwendet. Der logische Ausdruck darf nur Variable der Spenderdatei enthalten.

Zum Beispiel wählt

```
SELECT=C20.Z1
```

nur Spender zur Fusion aus, deren Variable C20 genau eine Angabe besitzt.

Damit ist es unter anderem möglich, in einem Datendurchlauf für mehrere Variable über die FUSION-Statements fehlende Werte zu ergänzen.

2. 5. 2012

Selektion von Fällen im Statement WEIGHT

Die Wichtung lässt sich jetzt auf Teilmengen der statistischen Fälle beschränken: Dazu dient die Angabe:

SELECT = *logischer Ausdruck*

Nur Fälle, die den logischen Ausdruck erfüllen, nehmen an der Wichtung teil. In allen anderen Fällen wird der Wert der Gewichtsvariablen aus NEW oder NEWFACT nicht verändert.

Zum Beispiel vergibt

```
SELECT=C10.1
```

nur denjenigen Fällen Gewichte, die die Bedingung C10.1 erfüllen.

Damit wird es möglich, verschiedenen Teilmengen einer Datei mit unterschiedlichen WEIGHT-Statements Gewichte zuzuweisen und diese in einem Datendurchlauf in eine einzige Gewichtsvariable zu stellen.

30. 3. 2012

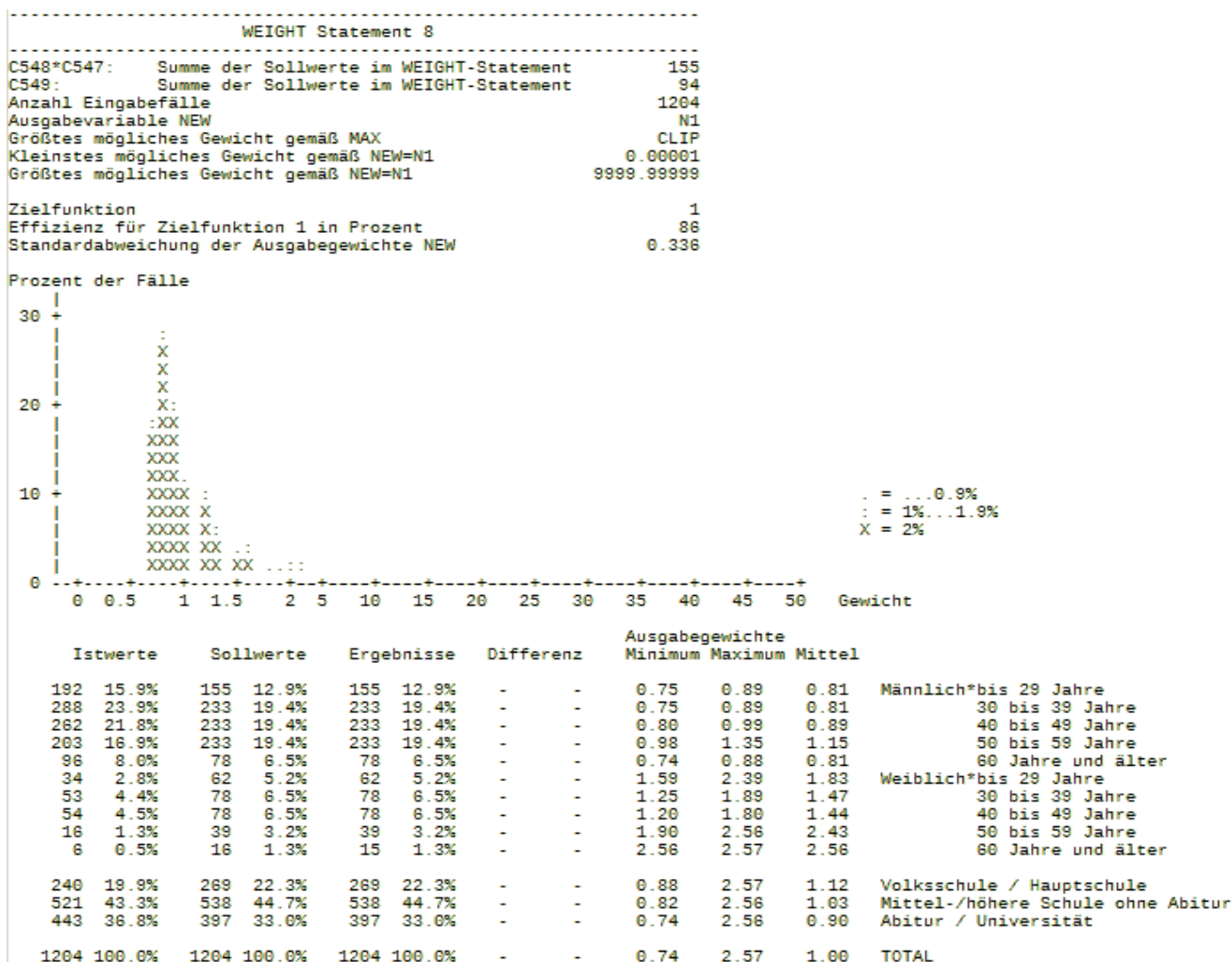
Erweiterungen für OUTPUT-EXT mit SPSS... und AUTO...

In den Folgestatements von OUTPUT-EXT mit AUTO... und SPSS... sind jetzt auch die Statements -IF, -IFN, ELSE und EIF möglich.

9. 1. 2012

Histogramm der Gewichte im WEIGHT-Protokoll

Das Protokoll des WEIGHT-Statements zeigt jetzt in einem Histogramm, wie häufig die gefundenen Gewichte in den statistischen Fällen verwendet werden:



2. 12. 2011

UTF8-Code und internationale Zeichensätze

Das Programm kann jetzt beliebige internationale Zeichensätze verarbeiten.
 Dazu sind die Statementdateien mit einem geeigneten Editor im UTF8-Code zu erstellen.
 Der einfache Statementeditor aus CNTW ist dazu in der Lage.
 Für die Aufbereitung der Ergebnisse ist eine Schrift auszuwählen, die über die gewünschten Zeichen verfügt.
 Ziemlich umfangreich ist hier zum Beispiel *Arial Unicode MS*.

CNTA und CNTW erkennen Statementdateien im UTF8-Code automatisch.

Die mit PAGE FNAME= erzeugten Ausgabedateien werden im Format CHARSET= aus den Dateien CNTA.INI und CNTW.INI erstellt. Gleiches gilt für die Protokolldatei *cntlst* aus CNTA.

Hier ein Beispiel:

DEFS

```
-C1:2 'Geschlecht' 1='Männer' 2='Frauen'
-C2:2 'секс' 1='мужчины' 2='женщины'
-C3:2 '性别' 1='男子' 2='妇女'
```

PAGEP FONT2='Arial Unicode MS', N, 4mm

XTAB ROW = C1; C2; C3

COL=TOTAL: 'Gesamt'; TOTAL: 'общий'; TOTAL: '总计'

	Gesamt	общий	总计
Geschlecht			
Männer	992	992	992
Frauen	3602	3602	3602
секс			
мужчины	992	992	992
женщины	3602	3602	3602
性别			
男子	992	992	992
妇女	3602	3602	3602

22. 11. 2011

Excel-Ausgabe im XLS- und XLSX-Format

Die Statements PAGE und PAGEP können jetzt direkt Excel-Dateien im XLS- und XLSX-Format erzeugen. Als Voraussetzung dafür müssen auf dem jeweiligen Computer die Excel-Programme installiert sein.
 Die Parameter dazu lauten analog zu der XML-Angabe:

```
XLS = | dateiname <,KEEP > |
      | NO |
```


1. 11. 2011

Freie Auswahl von Schriften

In den Auswertungen lassen sich beliebige True-Type- und Open-Type-Schriften verwenden. Dazu wurde die Schriftdefinition im Statement PAGEP um die Angabe *name* erweitert:

FONTi = $\left| \begin{array}{c} \mathbf{H} \\ \mathbf{T} \\ \mathbf{L} \\ \mathbf{N} \\ \textit{name} \end{array} \right| , \left| \begin{array}{c} \mathbf{N} \\ \mathbf{I} \\ \mathbf{B} \\ \mathbf{BI} \end{array} \right| , g \langle ,z \rangle \langle , \left| \begin{array}{c} \mathbf{COi} \\ \mathbf{Gi} \end{array} \right| \rangle$

H	Schriftart <i>Helvetica</i> ähnlich <i>Arial</i> .
T	Schriftart <i>Times</i> ähnlich <i>Times New Roman</i> .
L	Schriftart <i>Line Printer</i> ähnlich <i>Courier New</i> .
N	Schriftart <i>Helvetica Narrow</i> ähnlich <i>Arial Narrow</i> .
<i>name</i>	Name einer Schriftfamilie, wie etwa von Microsoft Word zur Schriftauswahl angezeigt. z.B. <i>Arial</i> , <i>Times New Roman</i> , <i>Arial Unicode MS</i> . Enthält der Name Leerzeichen, so ist er in Hochkommas ' einzuschließen. Zulässig sind True-Type- und Open-Type-Schriften. Die Schrift sollte in Windows installiert sein. Es genügt aber auch, die zugehörige Schriftendatei in das Programmverzeichnis von CNTW oder CNTA zu stellen.

N	Normalschrift: Schriftstil weder fett noch kursiv.
I	Kursive Schrift (italic).
B	Fette Schrift (bold).
BI	Fette und kursive Schrift (bold italic).

g	Schriftgröße in einer der oben beschriebenen Maßeinheiten
z	Zeilenabstand bei mehrzeiligen Texten in einer der oben beschriebenen Maßeinheiten. Fehlt diese Angabe, so wird als Zeilenhöhe die Schriftgröße verwendet.
Gi COi	Mit diesen Angaben erscheint die Schrift nicht schwarz sondern im Grauton G1 ... G16 oder der Farbe CO1 ... CO16. Im Statement PAGE sind diese Angaben wirkungslos.

So könnte zum Beispiel das Statement

PAGEP FONT2='Apple Chancery',N,4mm

folgende Tabelle erzeugen:

	TOTAL	Alter			
		Von 18 bis 20 Jahre	Von 21 bis 25 Jahre	Von 26 bis 30 Jahre	Von 31 bis 35 Jahre
TOTAL	4594	3	238	1101	1203
<i>Schulabschluss</i>					
<i>Volksschule, Hauptschule</i>	110	1	6	13	19
<i>Weiterführende Schule ohne Abitur</i>	621	1	55	117	127
<i>Abitur</i>	579	1	55	152	147
<i>Studium ohne Abschluss</i>	240	-	14	43	54
<i>Studium mit Abschluss</i>	2998	-	101	765	847
<i>kein Abschluss</i>	46	-	7	11	9

12. 10. 2011

Farbig unterlegte Zählergebnisse

Die Zählergebnisse lassen sich, abhängig von ihrem Wert, auf verschiedene Weise mit Farben unterlegen. Zum Beispiel:

```
XTAB ...
PRINT = C(1) ABS, CO16
C(2) ABS, CO5(10...1)
C(3) ABS, CO5(1...10)
C(4) ABS, G8(10...1)
C(5) ABS, G8(1...10)
C(6) ABS, CO2(6...10) IF<=5, CO10(5...0)
C(7) ABS, HIST, CO15(0...10)
C(8) ABS, HISTO, CO14(10...1)
```

Spalte 1	Spalte 2	Spalte 3	Spalte 4	Spalte 5	Spalte 6	Spalte 7	Spalte 8
1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9
10	10	10	10	10	10	10	10

C(1) ABS, CO16

unterlegt die Zählergebnisse der Spalte 1 mit der Farbe CO16 unabhängig von ihren Werten.

C(2) ABS, CO5(10...1)

versieht die Werte der Spalte 2 mit der Farbe CO5, wobei die Intensität der Farbe von den Zählergebnisse abhängt: Die Angabe (10...1) gibt dem Wert 1 die volle Farbe CO5, während größere Werte zunehmend schwächer eingefärbt werden.

C(3) ABS, CO5(1...10)

Diese Angabe liefert in Spalte 3 den entgegengesetzten Verlauf der Farben: Der Wert 10 erhält die volle Farbe CO5, während kleinere Werte zunehmend schwächer eingefärbt werden.

C(4) ABS, G8(10...1)

C(5) ABS, G8(1...10)

Diese Angaben erzeugen in Spalte 4 und 5 ähnliche Ergebnisse, jedoch mit dem Grauton G8 anstelle einer Farbe.

C(6) ABS, CO2(6...10) IF<=5, CO10(5...0)

In Spalte 6 werden Werte unter 5 grün und Werte über 5 rot unterlegt. Die Intensität der Farben steigt mit dem Abstand der Werte von 5.

C(7) ABS, HIST, CO15(0...10)

C(8) ABS, HISTO, CO14(10...1)

In den Spalten 7 und 8 machen die Angaben HIST und HISTO die Länge der Farbbalken von den Zählergebnissen abhängig. HISTO unterdrückt zusätzlich die Ausgabe der Zählerwerte.

21.05.2011

Erweiterung der Schriften

Die Angaben FONT1 ... im Statement PAGEP wurden von 9 auf 16 Schriften erweitert, auf die in den Statements mit FT1 ... FT16 zugegriffen wird.

Alle Schriften lassen sich jetzt auch farbig oder in Grautönen ausgeben.

Zu den bisherigen drei Schriftarten H, T und L ist eine vierte eine Schriftart N hinzugekommen, eine schmale Helvetica, die z.B. in Tabellen längere Texte in schmalen Spalten erlaubt. Hier ein Vergleich der Schriften H und N in verschiedenen Farben:

Qualität in der Verarbeitung des Innenraums	Qualität in der Verarbeitung des Innenraums	Qualität in der Verarbeitung des Innenraums	Qualität in der Verarbeitung des Innenraums
---	---	---	---

Darüber hinaus ist es jetzt möglich, beliebige True-Type-Schriften in CNT zu verwenden.

Zur Definition von Schriften existieren folgende Möglichkeiten:

$$\text{FONTi} = \left| \begin{array}{c} \text{H} \\ \text{T} \\ \text{L} \\ \text{N} \end{array} \right| , \left| \begin{array}{c} \text{N} \\ \text{I} \\ \text{B} \\ \text{BI} \end{array} \right| , g < , z > < , \left| \begin{array}{c} \text{COi} \\ \text{Gi} \end{array} \right| >$$

H	Schriftart <i>Helvetica</i> ähnlich <i>Arial</i> .
T	Schriftart <i>Times</i> ähnlich <i>Times New Roman</i> .
L	Schriftart <i>Line Printer</i> ähnlich <i>Courier New</i> .
N	Schriftart <i>Helvetica narrow</i> ähnlich <i>Arial Narrow</i> .
N	Normalschrift: Die einfacher Schriftstil, weder fett noch kursiv.
I	Kursive Schrift (italic).
B	Fette Schrift (bold).
BI	Fette und kursive Schrift (bold italic).
g	Schriftgröße in einer der oben beschriebenen Maßeinheiten
z	Zeilenabstand bei mehrzeiligen Texten in einer der oben beschriebenen Maßeinheiten. Fehlt diese Angabe, so wird als Zeilenhöhe die Schriftgröße verwendet.
Gi COi	Diese Angaben geben die Schrift nicht schwarz sondern im Grauton G1 ... G16 oder der Farbe CO1 ... CO16 aus. Im Statement PAGE sind diese Angaben wirkungslos.

Liegen keine Angaben zu den Schriften FONTi vor, so gelten folgende Voreinstellungen:

FONT1 = H, N, 6d	Helvetica normal, 6 Punkt
FONT2 = H, N, 8d	Helvetica normal, 8 Punkt
FONT3 = H, B, 8d	Helvetica halbfett, 8 Punkt
FONT4 = H, N, 10d	Helvetica normal, 10 Punkt
FONT5 = H, B, 10d	Helvetica halbfett, 10 Punkt
FONT6 = H, B, 12d	Helvetica halbfett, 12 Punkt
FONT7 = T, N, 8d	Times normal, 8 Punkt
FONT8 = T, B, 8d	Times halbfett, 8 Punkt
FONT9 = T, N, 10d	Times normal, 10 Punkt
FONT10 = T, N, 10d	Times halbfett, 10 Punkt
FONT11 = N, N, 8d	Helvetica schmal normal, 8 Punkt
FONT12 = N, B, 8d	Helvetica schmal halbfett, 8 Punkt
FONT13 = N, B, 10d	Helvetica schmal halbfett, 10 Punkt
FONT14 = H, N, 8d, G1	Helvetica normal, 8 Punkt, weiß für dunklen Hintergrund
FONT15 = H, N, 8d, G1	Helvetica halbfett, 8 Punkt, weiß für dunklen Hintergrund
FONT16 = H, B, 10d, G1	Helvetica halbfett, 10 Punkt, weiß für dunklen Hintergrund

Enthalten die Auswertungsstatements keine Angaben zu den Schriften, so gelten folgende Voreinstellungen:

Schrift	XTAB	CODEBOOK	INDEX	TEXT	HOLECOUNT	OUTPUT-EXT
FONT2	Grundschrift	Grundschrift	Grundschrift	Grundschrift	Grundschrift	Grundschrift
FONT3		Variablendefinitionen			Kopftexte	
FONT4	TITLE	Auswertung im Kopf				
FONT5		Überschrift	Überschrift			Überschrift

Darüber hinaus lässt sich die Schriftfarbe auch innerhalb eines Textes durch die Angaben ~Gi~ für die Grautöne G1 ... G16 und ~COi~ für die Farben CO1 ... CO16 verändern. Mit ~G0~..oder ~CO0~.. wird auf schwarze Schrift zurückgestellt.

Zum Beispiel:

```
TEXT '~CO2~Bitte unbedingt beachten:'/
      '~CO0~Die Frage 7 wurde nur von wenigen Befragten beantwortet'
```

Hier wird die erste Zeile rot ausgegeben und die zweite schwarz.

30.04.2011

Texte zu den Arbeitsblättern der Excel-Ausgabe

In den Statements PAGE und PAGEP lassen sich jetzt Texte für die Arbeitsblätter der Excel-Dateien vorgeben:

```
WORKSHEET = | 'name' |
              | TITLE   |
              | BOTTOM  |
              | STUBHEAD|
              | FILTER  |
              | NO      |
```

Ohne sie verwendet das Programm dafür Standardtexte.

name Der hier eingegebene Text wird bei der Excel-Ausgabe als Text für die Arbeitsblätter verwendet.

Sobald dieser Text wechselt, beginnt ein neues Arbeitsblatt.

TITLE Die TITLE-Texte aus XTAB dienen bei der Excel-Ausgabe als Text für die Arbeitsblätter.

BOTTOM Die BOTTOM-Texte aus XTAB dienen bei der Excel-Ausgabe als Text für die Arbeitsblätter.

STUBHEAD Die STUBHEAD-Texte aus XTAB dienen bei der Excel-Ausgabe als Text für die Arbeitsblätter .

FILTER Die FILTER-Texte aus XTAB dienen bei der Excel-Ausgabe als Text für die Arbeitsblätter.

NO Die WORKSHEET-Angaben aus früheren PAGE- oder PAGEP-Statements werden rückgängig gemacht. Das Programm versieht die Arbeitsblätter wieder mit den Standardtexten.

28.02.2011

Textbreiten zum INDEX-Statement

Bisher wurde die Textbreite zum INDEX-Statement vom Programm anhand der Schriftgrößen bestimmt. Der neue Parameter WIDTH im Index-Statement legt diese Breite nach Wunsch fest.

5. 01. 2011

Texte in Groß- oder Kleinschrift übersetzen

Texte lassen sich bei der Übertragung durch Wertezuweisungen in Groß- oder Kleinschrift übersetzen.
Zum Beispiel:

-T2=AS1 (3) UP

Der Text aus dem Datenfeld AS1(3) wird in die Textvariable T2 übertragen und dort in Großschrift übersetzt. Die Zeichen ä, ö, ü werden zu Ä, Ö und Ü, das ß bleibt unverändert.

-T2=AS1 (3) LOW

Der Text wird entsprechend in Kleinschrift übersetzt. Die Zeichen Ä, Ö, Ü werden zu ä, ö und ü, das ß bleibt unverändert.

25. 11. 2010

Tausenderzeichen und Dezimalzeichen in D-Feldern

Bei der Verarbeitung von CSV-Dateien über INPUT-EXT oder FETCH-FILE sind gelegentlich Zahlen wie 1.234,5
1 234,5 oder 1,234.5 einzulesen. Hierfür wurden in den Statements INPUT-EXT und FETCH-FILE die folgenden Parameter eingeführt:

DCHAR = ' x '

x ist ein beliebiges Zeichen, das in den Wertezuweisungen aus D-Feldern als Dezimalzeichen gelten soll. Fehlt diese Angabe, so werden Punkt und Komma als Dezimalzeichen verstanden.

TCHAR = ' x '

x ist ein beliebiges Zeichen, das in den Wertezuweisungen aus D-Feldern als Tausenderzeichen gelten soll. Fehlt diese Angabe, so wird kein Tausenderzeichen erkannt.

Zum Beispiel sorgt die Angabe

```
INPUT-EXT DCHAR='.' TCHAR=', ' ...
```

dafür, dass in der Wertezuweisung

```
-N1=DS1 (5)
```

der Eingabewert

```
1,500.12
```

als 1500,12 in die Variable N1 gestellt wird.

15. 11. 2010

TEXT-Statement mit Tonflächen und Rändern.

Im Statement TEXT kann die Ausgabe jetzt mit Tonflächen hinterlegt und mit Rändern versehen werden. Zum Beispiel erzeugen die Statements

```
TEXT 'Wie hoch ist das Nettoeinkommen aller ständig '-  
      'in Ihrem Haushalt lebenden Personen zusammengenommen?'  
      STYLE=C015  
      WIDTH=130  
      MARGINS=T:1:L2,B:1:L2,L:2:L2,R:2:L2
```

folgende Ausgabe:

```
Wie hoch ist das Nettoeinkommen aller ständig in Ihrem Haushalt lebenden  
Personen zusammengenommen?
```

10. 11. 2010

Erweiterung der Folgestatements von XTAB

Die folgende Funktion wurde neu eingeführt:

ABS(...) Diese Funktion wandelt die Werte einer Zeile, einer Spalte oder einer Variablen Vi in ihre absoluten

Beträge um. Enthält zum Beispiel die Variable V3 die Werte

(5 -2 4 -9)

so füllt das Statement

-V4=ABS(V3)

die Variable V4 mit den Werten

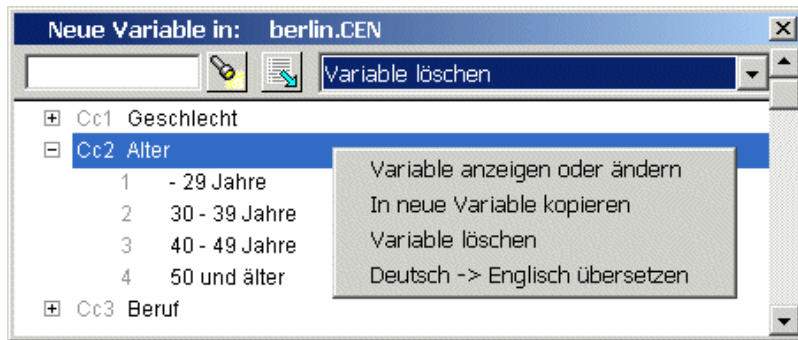
(5 2 4 9)

2. 11. 2010

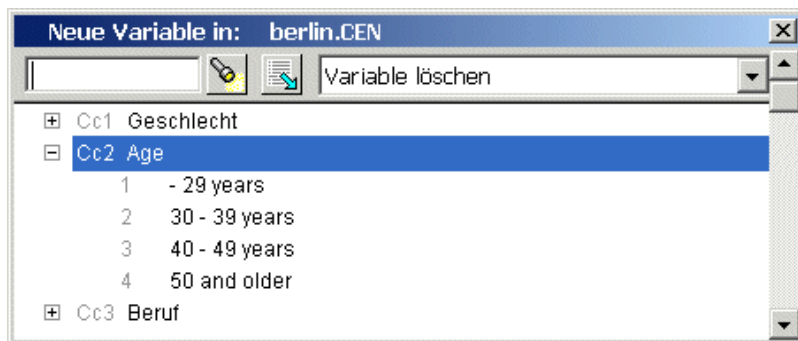
Übersetzung von Variablentexten in andere Sprachen

Die Texte der Variablen aus dem Fenster **Neue Variable** lassen sich von CNTW automatisch in andere Sprachen übersetzen. Voraussetzung ist eine geeignete Eintragung **TRANSLATE** in der Datei **CNTW.INI**.

Der zu übersetzende Text ist im Fenster **Neue Variable** mit der rechten Maustaste auszuwählen. Danach erscheint ein Menü:



Der Menüpunkt *Deutsch -> Englisch übersetzen* übersetzt die ausgewählten Texte:



Das Programm sucht zunächst in der **TRANSLATE**-Datei aus **CNTW.INI** geeignete Übersetzungen. Sind sie dort nicht zu finden und steht ein Internet-Anschluß zur Verfügung, so wird die Übersetzung über den Internet-Dienst *Google Translate* beschafft.

Die so gefundene Übersetzung lässt sich im Fenster **Neue Variable** über den Menüpunkt *Variable anzeigen oder ändern* nachbessern. Außerdem kann der Benutzer die **TRANSLATE**-Datei aus **CNTW.INI** ändern und erweitern.

24. 09. 2010

Aufruf mehrerer Programmläufe aus einer CNTW-Prototypdatei

Die Section-Statements in Prototyp-Dateien wurden folgendermaßen erweitert:

```
[ Sname ] ... IF = ( logischer Ausdruck )   RUN = | CNT
                                                | programmname parameter |
```

Die Angabe RUN erstellt aus den davorliegenden Statements eine Statementdatei *.~S~.
 RUN=CNT wertet diese Statements durch CNT aus.

Mit RUN=*programmname* ... kann auch ein beliebiges anderes Programm – auch mit Parametern – aufgerufen werden.

Die Angabe IF= ... macht die Ausführung der Programme abhängig von logischen Ausdrücken mit @-Variablen.

24. 08. 2010

neues Folgestatement für ADD-PROC, MOD-PROC, UPD-PROC und OUTPUT-EXT

Das Folgestatement –END beendet die Verarbeitung des laufenden Falles in ADD-PROC, MOD-PROC, UPD-PROC oder OUTPUT-EXT. Die dahinter stehenden Folgestatements werden nicht mehr ausgeführt.

Zum Beispiel:

```
ADD-PROC
-IF C17.1
- END
-EIF
...
MOD-PROC
```

Ist hier die Bedingung C17.1 erfüllt, so wird ADD-PROC für den laufenden Fall beendet. MOD-PROC wird dagegen weiter ausgeführt.

28. 07. 2010

Fusionieren interner Dateien

Das neue Statement FUSION ordnet jedem statistischen Fall der laufenden Verarbeitung (Empfänger; Rezipient) einen Datensatz aus einer zusätzlichen internen Datei (Spender, Donor) zu.

Dies geschieht durch Vergleich von Variablenwerten der Empfänger mit entsprechenden Variablenwerten der Spender, wobei möglichst ähnliche Fälle zusammengeführt werden.

Dabei sind verschiedene Strategien möglich: Gleichmäßige Verwendung der Spenderdaten oder größtmögliche Ähnlichkeit der Spender.

Die Spenderdaten lassen sich in bestehende oder neu definierte Variable übernehmen. Die Regeln dazu liefern die Folgestatements des FUSION-Statements.

Das Statement FUSION führt die Spender und Empfänger folgendermaßen zusammen:

Für jedes Empfänger-Spender-Paar werden aus den vorgegebenen Verbindungsvariablen Ähnlichkeitswerte ermittelt. Ein mathematisches Optimierungsverfahren verbindet danach die Spender und Empfänger so, dass die Summe der Ähnlichkeitswerte für die verbundenen Fälle maximal wird. Das Optimierungsverfahren (erweiterter Kuhn-Munkres-Algorithmus) liefert ohne Iterationen ein optimales Ergebnis: Es ist keine Kombination von Spendern und Empfängern möglich, die alle Randbedingungen erfüllt und eine größere Summe von Ähnlichkeitswerten besitzt.

Die Ähnlichkeit zweier Variablenwerte lässt sich mit verschiedenen Verrechnungsformen ermitteln, die stets Werte zwischen 0 und 100 liefern. Dabei steht 0 für sehr geringe und 100 für sehr große Ähnlichkeit. Sind mehrere Paare von Verbindungsvariablen vorgegeben, so wird aus den Ähnlichkeitswerten der einzelnen Variablen der Mittelwert gebildet, als resultierender Ähnlichkeitswert zwischen Spender und Empfänger. Es ist aber auch möglich, die Verbindungsvariablen unterschiedlich zu gewichten, um ihren Einfluss auf den resultierenden Wert zu variieren.

Das Statement FUSION sorgt für die Zusammenführung von Spender- und Empfängerfällen.

Folgestatements hinter FUSION übertragen Variablen aus der Spenderdatei in die Empfängerdaten.

Diese Folgestatements folgen den gleichen Regeln wie die des Statements INPUT-SEC

Alle Details finden sich im Handbuch oder in der Online-Hilfe beim Statement FUSION.

28. 07. 2010

Erweiterung der Folgestatements von XTAB

Die Folgestatements von XTAB zum Verrechnen und Füllen von Zeilen und Spalten kennen jetzt die Funktion ABS zum Erzeugen von Absolutwerten. Zum Beispiel überträgt

$$-C(5) = ABS(C(4))$$

die Werte der Spalte 4 in die Spalte 5 und überschreibt dabei den alten Inhalt der Spalte 5.

Negative Werte der Spalte 4 werden dabei in positive Absolutwerte umgesetzt.

Das Statement

$$-C(4) /= T(1, 4)$$

dividiert alle Werte der Spalte 4 durch den Wert aus Zeile 1 in der Spalte 4.

24. 07. 2010

Erweiterungen für die SPSS-Ausgabe

OUTPUT-EXT erhielt zusätzlich zur Angabe SPSS die Angabe SPSS-UTF8:

SPSS-UTF8 Diese Angabe unterscheidet sich von SPSS nur dadurch, dass die Datei VNAME mit den

Label-Texten im UTF8-Code erstellt wird. Die Datendatei FNAME wird weiterhin im ASCII-Code erstellt.

Enthält eine externe Datendatei Werte im UTF8-Code, die an SPSS weitergereicht werden soll, so empfiehlt es sich diese mit INPUT-EXT ASCII einzulesen. Auf diese Weise lassen sich UTF8-Texte über T-Variable ohne Code-Umwandlung in die SPSS-Datei ausgeben.

Weiter ist es jetzt möglich, bei der Ausgabe von numerischen Variablen in den Folgestatements von OUTPUT-EXT einzelnen Variablenwerten Value Label für SPSS mitzugeben. Zum Beispiel:

```
-N10   -1.5='klein'
        0='mittel'
        +1.5='groß'
```

25. 06. 2010

Einfache Variablenübernahme aus INPUT-SEC

INPUT-SEC ohne Folgestatements überträgt bisher automatisch die Variablenwerte aus INPUT-SEC, sofern eine gleichnamige Variable aus INPUT-INT oder DEFS vorhanden ist.

Zur einfacheren Übernahme der Variablen ist jetzt folgender Parameter in INPUT-SEC verfügbar.

NEWVARS= | ALL
 | ALL(n)
 | MISSING

Diese Angaben sind nur wirksam, wenn keine Folgestatements vorliegen. Sie erzeugen zu den Variablen aus FUSION neue Variable mit den Variablen- und Merkmalstexten sowie den Variablenwerten aus FUSION.

NEWVARS=ALL übernimmt alle Variable aus FUSION in neu definierte Variable. Dazu werden die kleinsten noch freien Variablennummern verwendet.

NEWVARS=ALL(n) vergibt stattdessen die kleinsten noch freien Variablennummern größer oder gleich der Nummer *n*.

NEWVARS=MISSING überträgt zunächst wie bei der normalen Zuordnung die Variablenwerte aus INPUT-SEC in gleichnamige Zielvariable. Zu Variablen aus INPUT-SEC ohne gleichnamiges Gegenstück wird dagegen eine neue Variable mit dem alten Variablennamen angelegt.

Der folgende neue Parameter in INPUT-SEC dient dazu, die Variablentexte der übernommenen Variablen zu kennzeichnen:

VTEXT = 'text'

Diese Angabe stellt die Zeichenfolge *text* vor den Variablentext aller aus der INPUT-SEC-Datei stammenden Variablen, die neu zu definieren sind.

Zum Beispiel stellt

```
VTEXT='Neue Variable'
```

vor alle aus INPUT-SEC übernommenen Variablentexte die Angabe *Neue Variable:* um die Herkunft der Variablen deutlich zu machen.

10. 06. 2010

OUTPUT-EXT mit variabler Satzlänge: Nur mit LF-Zeichen am Satzende

Bisher wurden die Datensätze von OUTPUT-EXT mit variabler Satzlänge durch die beiden Zeichen CR=10 und LF=13 beendet. Die neue Angabe

```
OUTPUT- EXT SI ZE=n, EOLF
```

beendet die Datensätze lediglich mit dem Zeichen LF=13.

18. 05. 2010

Steuerung der Worksheet Einteilung mit PAGEP XML= ...

Die Angabe XML=Dateiname im Statement PAGEP erstellt Excel-Dateien, wobei jeweils 100 Ausgabeseiten ein Worksheet bilden. Durch Einfügen des Statements

PAGEP XML = gleicher Dateiname

wird jetzt stets ein neues Worksheet begonnen. Zum Beispiel erzeugt

```
PAGEP XML = Excel ausgabe. xml
XTAB ROW=TOTAL; C2 COL=TOTAL; C1
XTAB ROW=TOTAL; C3 COL=TOTAL; C1
PAGEP XML = Excel ausgabe. xml
XTAB ROW=TOTAL; C3 COL=TOTAL; C1
INDEX
```

folgende Worksheets in der Datei Excelausgabe.xml:



16. 05. 2010

Erweiterung der Prototyp-Statements

@DEFNAME Diese vordefinierte Variable gibt den Dateinamen für die REP-Dateien der jeweiligen Prototyp-Datei vor. Sie enthält zunächst den Text *Auswertung*. Dieser Text lässt sich an beliebiger Stelle der Prototyp-Datei durch einen anderen Text ersetzen. So bietet das Statement

```
@DEFNAME Tabellen
```

bei der Speicherung einer neuen REP-Datei den Namen

```
Tabelleni.REP
```

an, wobei *i* die kleinste Zahl 1 ... ist, für die im jeweiligen Verzeichnis noch keine Datei mit diesem Namen vorhanden ist. Der Benutzer kann diesen Dateiname durch einen beliebigen anderen ersetzen.

@RUNDATE Diese vordefinierte Variable liefert das Datum des Auswertungslaufs in der Form TT.MM.JJJJ, falls im Menüpunkt *Optionen* als Programmsprache *Deutsch* eingestellt ist und als JJJJ.MM.TT für andere Programmsprachen.

@RUNTIME Diese vordefinierte Variable liefert die Uhrzeit des Auswertungslaufs in der Form SS:MM:SS.

DEFA = Dateiname

Diese Angabe ist in FNAME-Statements anstelle von DEF möglich. Sie öffnet automatisch die hinter DEFA stehende Datei, der Benutzer kann diese nicht auswählen oder verändern. *Dateiname* muß dabei ein vollständiger Name mit Laufwerk und Verzeichnis sein, ohne Platzhalter * und ? oder FIRST, CLOSE und NONAME. Es ist auch nur ein Dateiname möglich.

Hinter DEFA sind jedoch @-Variable zulässig. Zum Beispiel:

```
@INPUTS FNAME 'Steuerdatei' E DEF=*.st
@INPUTV FNAME 'Datei mit Variablen' V
DEFA=@INPUTS[PATH].@INPUTS[NAME].int
```

Über das erste FNAME-Statement wählt der Benutzer eine beliebige Datei mit der Erweiterung *st* aus. Das zweite Statement öffnet danach automatisch eine Datei im gleichen Verzeichnis, mit dem gleichen Dateinamen aber der Erweiterung *int*.

27. 04. 2010

Neue FORM-Angaben in PAGEP

Im Statement PAGEP sind anstelle von FORM=1, 2, 3 auch folgende Angaben möglich:

```
FORM = | LANDSCAPE |
        | PORTRAIT  |
        | MIXED     |
```

LANDSCAPE DIN A4, Querformat

PORTRAIT DIN A4, Hochformat.

MIXED DIN A4, Blattüberschrift und Seitennummern im Hochformat, Daten und Tabellen im Querformat

18. 03. 2010

Ausgabe von Tonflächen in PRINT-Anweisungen

Die farbige Unterlegung von Zählern aus IF-Angaben in PRINT, PRINT1...3 beziehen sich bisher nur auf den zur Print-Anweisung gehörenden Zähler. Sollen aber ganze Ausgabezeilen, in voller Breite unterlegt werden ist dies durch folgende Zusatzangabe zur Tonflächenfarbe möglich:

CO4 (L1) oder G1 (L12)

Durch die Angabe L1 wird die ganze erste Zeile, also der Bereich von PRINT und PRINT1 farbige unterlegt. Entsprechend wird durch L2 die ganze zweite Zeile und damit der Bereich von PRINT2 und PRINT3 unterlegt. Durch die Angabe L12 wird sogar der ganze Ausgabebereich von PRINT bis PRINT3 farbige ausgegeben.

Zum Beispiel liefert:

```
XTAB ROW=TOTAL;C547 COL=TOTAL;C548
PRINT =R(1)ABS R(2...-1) C(1)ABS C(2...-1)ABS IF >= 250,CO16(L12) IF <= 20,CO4(L12)
PRINT1=PROW1,1,%
PRINT3=PCOL1,1,%
STYLE =G2
```

folgende Tabelle

TOTAL	1.204			
	TOTAL	Geschlecht		
		Männlich	Weiblich	
TOTAL	1204 100,0% 100,0%	1041 100,0% 86,5%	163 100,0% 13,5%	
Alter				
bis 29 Jahre	226 18,8% 100,0%	192 18,4% 85,0%	34 20,9% 15,0%	
30 bis 39 Jahre	341 28,3% 100,0%	288 27,7% 84,5%	53 32,5% 15,5%	
40 bis 49 Jahre	316 26,2% 100,0%	262 25,2% 82,9%	54 33,1% 17,1%	
50 bis 59 Jahre	219 18,2% 100,0%	203 19,5% 92,7%	16 9,8% 7,3%	
60 Jahre und älter	102 8,5% 100,0%	96 9,2% 94,1%	6 3,7% 5,9%	

26. 11. 2009

Erweiterungen für INPUT-SEC

Bisher wurden aus INPUT-SEC-Dateien nur Variablenwerte in bereits in INPUT-INT oder DEFS definierte Variable übertragen. Nun ist es möglich, auch Variablendefinitionen mit Variablen- und Merkmalstexten aus INPUT-SEC zu übernehmen. Hierzu gibt es mehrere Wege:

Wenn

`-C10=C15`

ein Folgestatement von INPUT-SEC ist und die Variable C10 weder in INPUT-INT noch unter DEFS vorkommt, so wird die Variable C10 durch dieses Statement mit den Angaben der Variablen C15 aus INPUT-SEC neu definiert und anschließend mit den Werten von C15 versorgt.

Das Folgestatement

`-NEW=C15`

legt ebenfalls eine neue Variable mit den Angaben aus C15 an. Das Programm verwendet dafür die kleinste noch freie Variablennummer.

Das Statement

`-NEW(100)=C15`

führt zum gleichen Ergebnis, verwendet aber als Variablennummer die kleinste noch freie Nummer größer oder gleich 100.

Für INPUT-SEC ohne Folgestatements ist der neue Parameter NEWVARS verfügbar. Er erzeugt zu allen Variablen aus INPUT-SEC, zu denen kein gleichnamiges Gegenstück aus INPUT-INT oder DEFS vorliegt, eine neue Variable mit den Angaben und Texten aus INPUT-SEC.

10. 11. 2009

Verbesserung der XML-Ausgabe

Die Aufbereitung der XML-Dateien wurde an Besonderheiten neuerer Excel-Versionen angepasst. Bei der Tabellenausgabe werden mehrzeilige Texte vollständig in Excel angezeigt und die Spaltenbreiten verbessert.

28. 9. 2009

Codeplan-Ausgabe für OUTPUT-EXT

Bei der Ausgabe externer Dateien mit OUTPUT-EXT und AUTOASCII ... SPSS wurde bisher schon automatisch ein Codeplan erstellt. Dies geschieht nun auch für OUTPUT-EXT mit Wertezuweisungen ohne AUTOASCII ... SPSS.

24. 9. 2009

Korrelations-Koeffizienten in XTAB

Die Angabe CORR hinter PRINT...PRINT3 erstellt Korrelations-Koeffizienten aus den Werten der Zeilen- und Spaltenvariablen. Dazu sind in ROW und COL N-Variable, C-Variable mit Merkmalen und Mittelwerte MEAN von C- und N-Variablen möglich. Zum Beispiel:

```
XTAB ROW=TOTAL;C10
      COL=TOTAL;C20
      PRINT=ABS
      PRINT1=R(2...-1)C(2...-1)CORR,2
```

10. 9. 2009

Externe Dateien über vier Gigabyte

Es ist jetzt möglich, sehr große Dateien über vier Gigabyte mit INPUT-EXT in allen Formaten einzulesen und dabei auch mit SORT zu sortieren. Analog lassen sich derartige externe Dateien über FETCH-FILE einlesen und über OUTPUT-EXT auch ausgeben.

30. 8. 2009

Erweiterungen für t-Tests

Die bisherigen Funktionen MTT und MTTU zur Markierung signifikanter Unterschiede von Tabellenwerten wurden um die Funktion MTTD erweitert. Diese untersucht die Differenzen von Variablenwerten auf der Fall-Ebene (abhängiger oder verbundener Test) und prüft, ob sich der Mittelwert dieser Differenzen signifikant von 0 unterscheidet.

Die Funktionen MTT und MTTU untersuchen die Mittelwerte der dahinter stehenden Variablen. Zum Beispiel:

```
XTAB ROW=MTT{MEAN (N10) }
      COL=C10
```

Hier werden in jeder Spalte die Mittelwerte der Variablen N10 ermittelt und durch den t-Test MTT auf signifikante Unterschiede geprüft.

Neu hinzugekommen sind die Funktionen MTTX und MTTUX, die Mittelwerte der gegenüberliegenden ROW-COL-Seite auf signifikante Unterschiede untersuchen. Zum Beispiel:

```
XTAB ROW=MTTX{C10.1}
      COL=MEAN (N10) ; MEAN (N11)
```

Hier werden innerhalb der Zeile c10.1 die Mittelwerte der Variablen N10 und N11 gebildet und mit dem t-Test MTTX auf signifikante Unterschiede geprüft.

Die Funktionen MTT... können jetzt 260 Zeilen und Spalten miteinander vergleichen statt bisher 26. Dafür werden die Zeilen oder Spalten mit den Markierungen (a) ... (z) (a1) ... (z1) bis (a9) ... (z9) versehen.

Die Markierung (a) ... (z9) der Zeilen und Spalten erfolgt jetzt nach Sortierung durch SORT und Unterdrückung von Zeilen oder Spalten durch SUPPRESS.

Die Funktionen MTT... konnten bisher N-Variable nur zusammen mit MEAN auswerten. Zum Beispiel:

```
XTAB COL=TOTAL;C32
      ROW=TOTAL;MTTU{MEAN (N20) ; MEAN (N30) }
```

Nun ist es möglich, den t-Test auf numerische Variable auch ohne MEAN anzuwenden. Zum Beispiel:

```
XTAB COL=TOTAL;C32
      ROW=TOTAL;MTTU{N20;N30}
```

Wenn innerhalb der Tabelle der Platz nicht für alle Signifikanzangaben ausreicht, so werden diese in mehrere Zeilen umbrochen. Sie können dabei allerdings in die Ergebnisse der folgenden Zeilen hineinreichen! Dazu erscheinen Warnungen im Zählprotokoll. Dies Problem ist nur durch breitere Spalten oder Einfügen von Leerzeilen zu beheben.

2. 4. 2009

Texte auf Ähnlichkeit untersuchen

Bisher konnten unter ADD-PROC, MOD-PROC und UPD-PROC Texte aus T-Variablen mit konstanten Texten oder mit Texten aus anderen T-Variablen verglichen werden. Das Ergebnis war entweder *gleich* oder *ungleich*.

Die neuen Funktionen COMP und COMPI ermitteln dagegen Ähnlichkeitswerte 0 ... 100 zwischen zwei Texten. Dies erfolgt anhand der kleinstmöglichen Anzahl von Editierschritten, mit denen sich die Worte der beiden Texte ineinander überführen lassen. Der Ähnlichkeitswert 100 für sehr große und 0 für sehr geringe Ähnlichkeit.

```
| COMP | ( | 'sss' | | 'ttt' | )
| COMPI | | Tn | | Tn |
| | | | IDi | | IDi |
```

Zum Beispiel vergleicht das Statement

```
-N1=COMP('Hannover' T12)
```

in den Wertezuweisungen den Text Hannover mit dem Wert der Variablen T12 und stellt das Ergebnis 0 ... 100 in die Variable N1.

Die Funktionen lassen sich auch in logischen Ausdrücken verwenden:

```
-C1.1=COMP('Hannover' T12)>80 | COMP('Hildesheim', T12)>80
```

Vor dem Vergleich werden die Leerzeichen vor und hinter den Texten entfernt sowie mehrfache Leerzeichen zwischen den Worten auf ein Leerzeichen verkürzt. Tabulatorzeichen gelten dabei als Leerzeichen.

Die Funktionen liefern folgende Ähnlichkeitswerte:

100	Die Texte sind bis auf Leerzeichen gleich.
99	Nach zusätzlicher Umsetzung der Texte in Kleinbuchstaben sind die Texte gleich.
98	Nach zusätzlicher Umsetzung der Umlaute ä ... in a ... und von ß in s sind die Texte gleich
97	Nach zusätzlicher Umsetzung aller Sonderzeichen in Leerzeichen sind die Texte gleich.
96	Nach zusätzlicher Umsetzung aller doppelten Zeichen in ein Zeichen sind die Texte gleich.
0 ... 95	In den verbleibenden Fällen prüft das Programm, ob die Worte des ersten Textes denen im zweiten Text ähnlich sind. Dazu wird ein Ähnlichkeitswert 0 ... 95 ermittelt, abhängig der kleinstmöglichen Anzahl von Editierschritten (Zeichen einfügen, löschen oder vertauschen) mit denen sich zwei Worte ineinander überführen lassen.

COMP

Diese Funktion prüft, wie stark die Worte des ersten Textes denen des zweiten Textes ähneln. Die Reihenfolge der Worte spielt dabei keine Rolle; auch darf der zweite Text zusätzliche Worte enthalten. So liefern die Aufrufe

```
COMP('Meier' 'Meier') und COMP('Meier' 'Anna Meier')
```

beide das Ergebnis 100.

Mit zwei Funktionsaufrufen lassen sich die Texte auch auf die gleiche Anzahl von Worten überprüfen:

```
-IF COMP(T1 T2)=100 & COMP(T2 T1)=100
```

COMPI

Die Funktion COMP vergleicht nur ganze Worte, die durch Leer- oder Sonderzeichen voneinander getrennt sind. COMPI prüft dagegen, ob die Worte des ersten Textes in den Worten des zweiten Textes enthalten sind. So liefert

COMP ('Last' 'Lastwagen')

nur den Ähnlichkeitswert 34,

COMPI ('Last' 'Lastwagen')

dagegen den Wert 95.

Beispiele

T12	COMP ('Hannover' T12)
Hannover	100
hannover	99
hannover-langenhagen	95
in der nähe von hanover	95
hanofer	73
Hannöversand	54
hohenwerder	12
hameln	9

6. 3. 2009

Tabulatorzeichen in REPORT-Dateien

Mit der Angabe `TAB` lassen sich Tabulatorzeichen in REPORT-Dateien ausgeben.

Zum Beispiel stellt das Statement

```
-REPORT TAB N1 3TAB
```

ein Tabulatorzeichen vor den Wert der Variablen `N1`, gefolgt von drei weiteren Tabulatorzeichen.

4. 3. 2009

Breitenvorgaben für die TEXT-Statements

Das Statement `TEXT` besitzt den neuen Parameter

WIDTH = n

Ohne diese Angabe wird der Text genau in dem vorgegebenen Zeilenlauf ausgegeben.

Zu breite Texte reichen über die verfügbare Blattbreite hinaus; dies jedoch mit einer Fehlermeldung im Zählprotokoll.

Mit der Angabe `WIDTH` wird der Text ein Rechteck von der Breite `n` Millimeter eingepasst.

Auch die übrigen Maßeinheiten aus dem Statement `PAGEP` sind möglich.

25. 2. 2009

Textjustage in den Tabellenköpfen und Zeilen

Durch den Parameter `STYLE` im Statement `XTAB` lassen sich die Texte aus `COL` und `STUBHEAD` in der frei verfügbaren Höhe ausrichten:

T = oben, B = unten, M = mittig.

Zum Beispiel stellt

```
STYLE=COL:M, COL(-1):B
```

die Texte aus `COL` mittig in die frei verfügbare Höhe der Köpfe, die letzten Texte jeder Spalte werden jedoch nach unten ausgerichtet.

Weiter lassen sich die Zeilentexte mit der Angabe `STYLE=ROW` jetzt auch horizontal ausrichten:

L = links, R = rechts, C = zentriert.

27. 1. 2009

Datum und Uhrzeit in den Textzeilen

Mit den Angaben ~DATE~ und ~TIME~ lassen sich Datum und Uhrzeit des Verarbeitungslaufs an beliebiger Stelle in Textzeilen einfügen. Zum Beispiel:

```
PAGEP FOOT='Auswertung vom ~DATE~ um ~TIME~ Uhr'
```

30. 11. 2008

Zusätzliche Reichweitenfunktionen RCHM und RCHX

Neben der Funktion RCH zur Ermittlung der Nettoreichweite von Medienstreuplänen wurden folgende Funktionen neu eingeführt:

RCHM (Nn₁ Nn₂ ...)

Diese Funktion ermittelt die Mehrfachleser der angegebenen Titel, also diejenigen, die alle Titel gleichzeitig lesen. Die Berechnung erfolgt durch:

$$r = p_1 \cdot p_2 \cdot \dots$$

p_i = Kontaktwahrscheinlichkeit aus den Variablen Nn_i.

RCHX (Nn₁ Nn₂ ...)

Diese Funktion ermittelt die Exklusivleser des ersten Titels, also die Leser des ersten Titels, die die übrigen Titel nicht lesen. Die Berechnung erfolgt zum Beispiel bei drei Titeln durch:

$$r(p_1 p_2 p_3) = p_1(1 - p_2 - p_3 + p_2 + p_3)$$

p_i = Kontaktwahrscheinlichkeit aus den Variablen Nn_i.

14. 11. 2008

Vergrößerung der maximalen Merkmalsnummer in MS- und MK-Feldern

Die maximale Merkmalsnummer in diesen Feldern wurde von 9 999 auf 99 999 angehoben.

4. 11. 2008

Erweiterung der Logo-Ausgabe

Die Ausgabemöglichkeiten für Logos wurden verbessert. Anstelle der umständlichen Regel mit Dateinamen LOGO20 ... für die Logonummern 20 ... lassen sich jetzt beliebige Dateinamen verwenden.

Die Logo-Verarbeitung wurde auf Bilddateien im JPEG-Format mit der Dateierweiterung JPG umgestellt. Darüber hinaus lässt sich die Größe der Logos einfach verändern. Um Probleme mit der Darstellungsqualität und Lesbarkeit zu vermeiden, werden PCX- und TIF-Dateien nicht mehr akzeptiert.

Die alten Logo-Angaben sind weiterhin zulässig, wurden jedoch aus Handbuch und Hilfe-Datei entfernt. Sie sollten möglichst nicht mehr verwendet werden.

Die Definition der neuen Logos erfolgt jetzt im Statement PAGEP mit der Angabe:

LOGO *i* = dateiname < , zoomfaktor >

Mit dieser Angabe wird der Zahl *i* eine beliebige Bilddatei im JPEG-Format zugewiesen. Unter der Nummer *i* lässt diese Bilddatei in jede Textzeile als Logo einfügen oder im Statement PAGEP mit der Angabe LOGOS frei auf den Seiten der Auswertungen plazieren.

Bei der Arbeit mit PAGE lassen sich Logos angeben ohne in den Auswertungen zu erscheinen.

*i*Nummer 1 ... 999 unter der das Logo in den Statements aufgerufen werden soll.

dateiname Name einer Bilddatei im JPEG-Format mit der Dateierweiterung JPG.

Fehlt die Dateierweiterung, so wird JPG angenommen.

Der Dateiname kann auch ohne Pfad angegeben werden. Dann sucht das Programm die Logo-Datei in den folgenden Verzeichnissen:

1. Im Verzeichnis der REP-Datei. Neu angelegte Auswertungen mit dem Namen ~N~.REP befinden sich stets im Verzeichnis CNTWORK oder CNTAWORK.
2. Im Verzeichnis der Statementdatei.
3. In dem durch die Angabe LOGOS= in der Datei CNTA.INI oder CNTW.INI festgelegten Verzeichnis.
4. Im Programmverzeichnis CNTW oder CNTA.

zoomfaktor Vergrößerungsfaktor 1 ... 999 als Prozentwert. Fehlt diese Angabe, so wird 100 angenommen und das Logo in Originalgröße ausgegeben.

Beispiel:

```
DEFS
-C1:8 1='~11~ aaa' 2='~12~ bbb' 3='~13~ ccc' 4='~14~ ddd'
      5='~15~ eee' 6='~16~ fff' 7='~17~ ggg' 8='~18~ hhh'
PAGEP LOGO10=c:\cntw\cnt.jpg
      LOGO11=shampooa
      LOGO12=shampoob
      LOGO13=shampooc
      LOGO14=shampood
      LOGO15=shampooe
      LOGO16=shampoof
      LOGO17=shampoog
      LOGO18=shampooh
      L3=0.5,1,C016
      L4=1,2,C016
      MARGINS=T:22:L4
      LOGOS=TL:10
```

```

HEAD='~B~Produktvergleich'
XTAB ROW=TOTAL: 'Gesamt'; ;MEAN (C2);MEAN (C3);MEAN (C4)
COL=C1
LINES=L3
    
```

Diese Statements könnten folgendes Ergebnis liefern:

 **Produktvergleich**

								
	aaa	bbb	ccc	ddd	eee	fff	ggg	hhh
Gesamt	551	538	549	545	532	557	531	515
Preis	5,81	5,82	5,83	5,74	5,84	5,88	5,95	6,04
Qualität	1,39	1,43	1,41	1,46	1,36	1,39	1,40	1,43
Sympathie	3,46	3,44	3,46	3,46	3,48	3,59	3,62	3,45

Im Statement PAGEP werden zunächst die Logos 10 ... 18 definiert: Der JPEG-Datei *c:\cntw\cnt.jpg* mit dem CNT-Logo wird die Nummer 10 zugewiesen und der Datei *shampooh.jpg* schließlich die Nummer 18.

Die Angabe *LOGOS=TL:10* im Statement PAGEP stellt das Logo 10 links oben auf jede Seite der Auswertungen. Die übrigen Logos werden durch die Angaben *~11~* bis *~18~* in die Merkmalstexte der Variablen C1 übernommen. Durch *COL=C1* im Statement XTAB gelangen sie in die Köpfe der Tabelle.

Die Angabe *L3=0.5,1,CO16* in PAGEP erzeugt eine blaue Linie, die über die Angabe *LINES=L3* in XTAB für die Linien in der Tabelle verwendet wird. Mit *L4=1,2,CO16* wird die breite blaue Linie erzeugt, die durch die Angabe *MARGINS=T:22:L4* den oberen Rand von 22mm Höhe abschließt.

30. 10. 2008

SPSS-Ausgabe mit Mehrfachnennungen

Für OUTPUT-EXT mit SPSS wurde die Angabe CSTD erweitert.
 Mit CSTD=UM wird die Ausgabe einer C-Variablen abhängig von der Anzahl ihrer Nennungen.
 Mit Mehrfachnennungen entstehen dichotome und ohne Mehrfachnennungen kategoriale SPSS-Variable.

6. 10. 2008

MK- und UK-Felder mit mehreren Schlüsseln

Zur Verarbeitung von Mehrfachnennungen sind MK- und UK-Felder mit mehreren Schlüsseln möglich.

Ist nur ein Schlüssel angegeben, so stehen die Mehrfachnennungen in den Datensätzen hinter diesem Schlüssel .
Zum Beispiel

```
... frage2=1 3 5-9 ...
```

für die Wertezuweisung

```
-C1=MK1 (frage2)
```

oder

```
... ;frage7=1001001; ...
```

für die Wertezuweisung

```
-C1=UK1 (frage7)
```

Sind mehrere Schlüssel angegeben, so nimmt jeder Schlüssel eine einzelne Nennung auf.

Zum Beispiel:

```
... ;fr11=2;fr12=4;fr13=6; ...
```

für die Wertezuweisung

```
-C1=MK1 (fr11 fr12 fr13 fr14)
```

oder

```
... fr11=1;fr12=0;fr13=1; ...
```

für die Wertezuweisung

```
-C1=UK1 (fr11 fr12 fr13)
```


28. 9. 2008

Erweiterung der Syntax für Prototyp-Dateien

Die Prototyp-Funktionen wurden stark erweitert und die bisherige Syntax mit dem Ziel größerer Konsistenz überarbeitet.

Die alten Regeln werden weiter unterstützt, sollten aber möglichst nicht mehr verwendet werden. Detaillierte Auskunft gibt die Beschreibung in CNTW.PDF im Verzeichnis CNTW. Die zusammen mit CNTW ausgelieferten PTT-Dateien wurden entsprechend modernisiert.

Prototyp-Zeilen beginnen jetzt stets mit @:
 @GROUP, @INCLUDE, @MESSAGE, @PNAME, @CNTW-PROTOTYPE-FILE.

Das Statement @WINDOW ... wird zu
 @name WINDOW ...
 wobei die Variable @name die höchste Anzahl Eingaben zu den Feldern FLD des Fensters liefert.

Das Sektion-Statement lautet jetzt:
 [name] SX=1...n
 wobei für name ein beliebiger Text aus maximal 8 Zeichen möglich ist.

Zusätzlich zu den @-Variablen aus FNAME, FLD und WINDOW sind jetzt freie @-Variable möglich, die ihre Werte ganz analog zu den #-Makros erhalten. Während die #-Makros ihre Werte beim Einlesen der CNTA-Statements erhalten, werden die neuen @-Variablen beim Erstellen der CNTA-Statements durch CNTW gefüllt. Zum Beispiel:

```
@gewicht = @pgew * @proj
@title   TITLE='@HEAD'
```

@IF-Statements erlauben vollständige logische Ausdrücke, analog zu den @-Variablen. Sie erlauben auch @ELSEIF-Angaben, ebenfalls mit logischen Ausdrücken. @IF-Statements werden bei der Erstellung der CNTA-Statements wirksam.

@PROCESS ist ebenfalls mit vollständigen logischen Ausdrücken möglich. @PROCESS wird beim Einlesen der Prototyp-Datei durch CNTW wirksam.

@DO-Statements sind möglich in der Form:
 @DO @i=1...100
 auch diese werden beim Erstellen der CNTA-Statements ausgeführt.

Mit diesem Instrumentarium lassen sich Prototyp-Dateien ohne Benutzung von #-Makros erstellen, sodaß durchsichtigere CNTA-Statements entstehen.

1. 8. 2008

Hilfslinien in HOLECOUNT

Zur besseren Übersicht werden in den Auswertungen HOLECOUNT hinter den Spalten 5, 10 und 12 senkrechte Linien als Lesehilfen ausgegeben.

Bei der Ausgabe über PAGE werden diese Linien mit dem Zeichen | erzeugt. Dieses Zeichen lässt sich mit der Angabe

```
LCHAR = 'x'
```

durch ein beliebiges Zeichen x ersetzen.

24. 6. 2008

Fußtexte FOOT aus PAGE und PAGEP im Inhaltsverzeichnis INDEX

Es ist jetzt möglich, mit der Angabe

INDEX PAGEFOOT

auch die Fußtexte FOOT aus den Statements PAGE und PAGEP in die Inhaltsverzeichnisse aufzunehmen.

30. 5. 2008

Variablentexte in Tabellen unterdrücken

Es ist jetzt möglich, die Variablentexte von C-Variablen in den Kreuztabellen zu unterdrücken.

Dies geschieht mit den folgenden Angaben in XTAB:

SUPPRESS=VARLABELF	Variablentexte der C-Variablen im Filter unterdrücken
SUPPRESS=VARLABELC	Variablentexte der C-Variablen in den Köpfen unterdrücken
SUPPRESS=VARLABELR	Variablentexte der C-Variablen in den Zeilen unterdrücken
SUPPRESS=VARLABELS	Variablentexte der C-Variablen überall unterdrücken

20. 4. 2008

Farbige Linien in PAGEP

Farbige Linien sind jetzt möglich; außerdem ganz weiße Linien für farbig oder grau unterlegte Tabellen. Die Anzahl möglicher Linien wurde von 9 auf 16 erhöht.

Die Angabe von Linien in PAGEP geschieht auf folgende Weise:

$$L_i = s, d \left\langle \begin{array}{l} |, g \\ |, CO_i \\ |, G_i \end{array} \right\rangle \left\langle t \right\rangle$$

Diese Angaben ordnen den Kürzeln L1 ... L16 Linien zur Ausgabe mit den Statements XTAB, CODEBOOK, INDEX ... zu.

s Stärke der Linie in den Maßeinheiten von PAGEP.

d Distanz zwischen Linienmitte und Text. CNTA sorgt dafür, dass zwischen der Mitte der Linie einerseits und den Texten und Zählergebnissen andererseits mindestens dieser Abstand eingehalten wird.

g Grauton 0 ... 100, in dem die Linie ausgegeben werden soll. Die Angabe 100 sorgt für schwarze und 0 für weiße Linien. Ohne diese Angabe wird mit dem Wert 100 gearbeitet.

G_i Grauton G1 ... G16 aus diesem Statement PAGEP in der die Linie ausgegeben werden soll.

CO_i Farbe CO1 ... CO16 aus diesem Statement PAGEP in der die Linie ausgegeben werden soll.

t Typ 1 oder 2 der Linie. Typ 1 sorgt für einfache Linien und Typ 2 für doppelte.

Fehlt diese Angabe, so werden einfache Linien ausgegeben. Bei doppelten Linien besitzen die beiden Teillinien je ein Drittel der unter **s** angegebenen Stärke der Gesamtlinie.

Enthält PAGEP keine Angaben zu den Linien, so wird folgende Voreinstellung verwendet:

L1 = 0.05, 2.0, 100	L9 = 1.50, 3.0, 100
L2 = 0.10, 2.0, 100	L10 = 0.20, 2.0, 0
L3 = 0.20, 2.0, 100	L11 = 0.30, 2.0, 0
L4 = 0.30, 2.0, 100	L12 = 0.40, 2.0, 0
L5 = 0.40, 2.0, 100	L13 = 0.50, 2.0, 0
L6 = 0.50, 2.0, 100	L14 = 0.75, 2.0, 0
L7 = 0.75, 2.0, 100	L15 = 1.00, 2.5, 0
L8 = 1.00, 2.5, 100	L16 = 1.50, 3.0, 0

4. 4. 2008

Neuer Parameter STYLE in XTAB

Der Parameter `STYLE` umfasst die Funktionen der bisherigen Angaben `FONT` und `TEXT`. Diese werden weiter unterstützt, sollten aber möglichst nicht mehr verwendet werden.

Zusätzlich zu den bisherigen `FONT`- und `TEXT`-Angaben erlaubt `STYLE` die einfache Definition von Tonflächen für verschiedene Teile der Tabellen, als Grautöne oder als Farben.

Beispiel: `XTAB COLS=TOTAL:'Total':co16;C1007`
`ROWS=TOTAL:'Basis-Fälle = 100':co16;C6020[1].1;...5`
`STYLE=G5, COL(1):FT3`
`, ROW(1):FT3`
`LINES=L14`

Dieses Statement könnte folgende Tabelle erzeugen:

	Total	Beruf des Befragten					
		Arbeiter	Ange- stellter	Beamter	Freier Beruf/ Selbstän- diger	Hausfrau	in Ausbil- dung
Basis-Fälle = 100	7768	2235	3713	395	698	242	485
Anzahl Reisen							
1 Reise	55,6%	55,3%	56,2%	52,7%	52,4%	53,7%	61,0%
2 Reisen	14,7%	8,5%	17,4%	20,0%	19,1%	10,7%	14,2%
3 Reisen	4,1%	1,9%	5,2%	8,1%	4,7%	2,5%	3,3%
4 Reisen	1,0%	0,4%	1,2%	2,8%	1,9%	-	0,4%
5 und mehr Reisen	0,5%	-	0,7%	1,3%	0,9%	1,2%	0,2%

Die Angabe `STYLE=G5` unterlegt zunächst die ganze Tabelle mit dem Grauton `G5`.

Durch `COL(1):FT3,ROW(1):FT3` hinter `STYLE` werden die Spalten- und Zeilentexte der obersten Stufe in der fetten Schrift `FT3` ausgegeben.

`LINES=L14` sorgt für die weißen Trennlinien innerhalb und um die Tabelle herum.

Die Angabe `:co16` in `ROWS` und `COLS` unterlegt die erste Zeile und Spalte mit blauer Farbe.

Überschreiben von LOGOS

Wurden im Statement `PAGEP` mit der Angabe `LOGOS` auch Logos ausgegeben, so erscheinen Kopftexte `HEAD`, Fußtexte `FOOT` und Blattnummern `NR` rechts oder links der Logos. Diese Texte können jetzt auch die Logos überschreiben. Dazu ist hinter die Positionsangabe der Buchstabe `K` zu stellen. Zum Beispiel:

```
PAGEP NR='Seite',TRK HEAD='Projekt 1421',LK
```

28. 2. 2008

Farbangaben in PAGEP

Zur Einfärbung von Schriften, Tonflächen und Linien können im Statement PAGEP Farben definiert werden:

< COi = r , g , b >















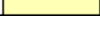

Diese Angaben ordnen den Kürzeln CO1 ... CO16 Farben im RGB-System zu.

rFarbwert 0 ... 255 für rot.

g Farbwert 0 ... 255 für grün.

b Farbwert 0 ... 255 für blau.

Liegen keine Angaben zu den Farben vor, so wird folgende Voreinstellung verwendet:

CO1 = 180, 0, 0		CO9 = 0, 170, 0	
CO2 = 255, 0, 0		CO10 = 0, 255, 0	
CO3 = 255, 120, 120		CO11 = 120, 255, 120	
CO4 = 255, 180, 180		CO12 = 180, 255, 180	
CO5 = 150, 150, 0		CO13 = 0, 0, 180	
CO6 = 200, 200, 0		CO14 = 120, 120, 180	
CO7 = 255, 255, 0		CO15 = 120, 120, 255	
CO8 = 255, 255, 180		CO16 = 180, 180, 255	

Diese Farben sind im Statement XTAB überall dort möglich, wo bisher Grautöne angegeben werden konnten, also hinter ROWS, COLS, PRINT und INSERT.

So unterlegt das Statement

```
XTAB ROWS=TOTAL:'Basis':CO16;C1
      COLS=TOTAL:CO16;C21
      PRINT=ABS,CO12 ...
```

die erste Zeile und Spalte hellblau und die Zählergebnisse hellgrün.

28. 2. 2008

Zusätzliche Grautöne

Anstelle von nur 9 Grautönen sind im Statement PAGEP jetzt die Angaben G1 ... G16 möglich.

Kreuztabellen lassen sich jetzt ganz oder in Teilen mit Tonflächen hinterlegen:

```
XTAB TEXT=G6, STUBHEAD:CO4 ...
```

12. 2. 2008

Folgestatments zu XTAB für Wertezuweisungen

Hinter XTAB und XADD werden Folgestatements eingeführt.

Sie ermöglichen die Manipulation der Zählergebnisse vor ihrer Aufbereitung in den Tabellen. Dies geschieht nach Verrechnung von numerischen Ausdrücken, XADD und XADDB, aber vor der Sortierung durch SORT und Unterdrückung von Zeilen oder Spalten durch SUPPRESS. Nur Absolutwerte können verändert werden, nicht aufbereitete Werte wie Prozente, Index oder Rangzahlen.

Wie alle Folgestatements beginnen sie mit einem Strich in der ersten Position. Sie gelten nur für das davor liegende XTAB und werden in der eingegebenen Reihenfolge abgearbeitet.

Um in späteren XTAB-Statements auf Werte davor stehender Tabellen zugreifen zu können, werden Variable V0 ... V99 eingeführt, die Zeilen und Spalten zwischenspeichern können. Auf die Werte dieser Variablen können spätere XTAB Statements zugreifen.

Folgende Wertezuweisungen sind als Folgestatements möglich:

```
XTAB ...
-V1=R(7)      Zeile 7 im Variable V1 speichern
-V9=C(-1)     die letzte Spalte im Variable V9 speichern
```

Die Daten aus diesen Variablen lassen sich in die Tabelle des laufenden XTAB übertragen:

```
XTAB ...
-R(5)=V9      die Zeile oder Spalte aus V9 in die Zeile 5 der laufenden Tabelle stellen
-C(1)=(10 20 30 40 50)  die Spalte 1 der laufenden Tabelle mit konstanten Werten füllen
```

Zusätzlich zu überschreibenden Wertezuweisungen sind arithmetische Operationen möglich:

```
XTAB ...
-R(7)+=V9     Variable V9 zu Zeile 7 hinzuaddieren
-C(1)-=V99    Variable V99 von Spalte 1 subtrahieren
-R(14)*=100   alle Werte der Zeile 14 mit 100 multiplizieren
-C(4)/=C(3)   alle Werte der Spalte 4 durch entsprechende Werte der Spalte 3 dividieren.
               Division durch 0 führt zum Ergebnis 0.
```

12. 2. 2008

Folgestatments ROUND zu XTAB für Rundungen

Es wurden Folgestatements für XTAB zur Rundung von Zählergebnissen eingeführt. Sie verändern die Werte vor ihrer Aufbereitung als Prozente, Index oder Rangzahlen in den Tabellen.

```
XTAB ...
-ROUND(z s)=100  die Absolutwerte der Zeilen z und Spalten s werden auf 100 gerundet, d.h.
                 die beiden letzten Stellen erscheinen als 00. Die Angaben z und s folgen den
                 aus PRINT bekannten Regeln:
                 R(5) C(10..-1 A) ...
```

```
XTAB ...
-ROUND=100      hier wird die ganze Tabelle ohne Randbedingungen gerundet.
```

12. 2. 2008

Folgestatments PROSUM für XTAB zur Anpassung und Rundungen

Es wurden Folgestatements für XTAB zur Hochrechnung und Rundung von Zählergebnissen eingeführt. Sie verändern die Werte vor ihrer Aufbereitung als Prozente, Index oder Rangzahlen in den Tabellen.

Ein XTAB-Statement ohne PROSUM-Statements könnte folgende Tabelle liefern:

	Gesamt		Bayern		Niedersachsen	
Basis in Tsd	63 785		9 214		4 526	
Männer	1 164	44,7%	161	44,4%	81	44,3%
Frauen	1 438	55,3%	202	55,6%	102	55,7%

Die erste Zeile enthält Bevölkerungswerte in Tausend und beiden folgenden Zeilen Werte aus der Stichprobe.

Das Folgestatement

`-PROSUM-ROWS (R (2 3) C (1..3))=R (1)`

zum gleichen XTAB erstellt daraus folgende Tabelle durch Hochrechnung:

	Gesamt		Bayern		Niedersachsen	
Basis in Tsd	63 785	100,0%	9 214	100,0%	4 526	100,0%
Männer	28 534	44,7%	4 087	44,4%	2 003	44,3%
Frauen	35 251	55,3%	5 127	55,6%	2 523	55,7%

In den Spalten 1 ... 3 werden die Summen der Zeilen 2 und 3 gebildet.

Die Werte der Zeilen 2 und 3 werden nun mit einem Faktor multipliziert, sodaß ihre Summe mit dem Wert der ersten Zeile übereinstimmt. Für die erste Spalte ist dies der Faktor

$$24,514 = 63785 / 2602 = 63785 / (1164 + 1438)$$

Das Folgestatement

`-PROSUM-ROWS (R (2 3) C (1..3))=R (1) ROUND=100`

erzeugt eine Tabelle mit Hochrechnung und anschließender Rundung mit dem Faktor 100:

	Gesamt		Bayern		Niedersachsen	
Basis in Tsd	63 785	100,0%	9 214	100,0%	4 526	100,0%
Männer	28 500	44,7%	4 100	44,5%	2 000	44,2%
Frauen	35 300	55,3%	5 100	55,4%	2 500	55,2%

Zunächst wird auch hier die Hochrechnung wie in der vorigen Tabelle ausgeführt. Die neuen Werte werden jedoch zusätzlich auf 100 gerundet und so korrigiert, dass ihre Summe mit dem gerundeten Wert der ersten Zeile übereinstimmt. Diese Korrektur wird so vorgenommen dass sich in den Zeilen 2 oder 3 die Prozentwerte möglichst wenig ändern.

Mit den Statements

`-PROSUM-ROWS (R (2 3) C (1..3))=R (1) ROUND=100`

`-ROUND (R (1) C (1..3))=100`

wird auch die erste Zeile noch auf 100 gerundet:

	Gesamt		Bayern		Niedersachsen	
Basis in Tsd	63 800	100,0%	9 200	100,0%	4 500	100,0%
Männer	28 500	44,7%	4 100	44,6%	2 000	44,4%
Frauen	35 300	55,3%	5 100	55,4%	2 500	55,6%

Das Folgestatement

`-PROSUM-ROWS (R (2 3) C (1..3))=R (1) ROUND=(100, 50, 10) R (1), 0.2`

liefert eine Tabelle mit Hochrechnung und Rundung, bei der zusätzlich versucht wird, Prozentwerte soweit wie möglich zu erhalten:

	Gesamt		Bayern		Niedersachsen	
Basis in Tsd	63 785	100,0%	9 214	100,0%	4 526	100,0%
Männer	28 540	44,7%	4 080	44,3%	2 010	44,4%
Frauen	35 250	55,3%	5 130	55,7%	2 520	55,7%

Zunächst wird auch hier die Hochrechnung wie in der vorigen Tabelle ausgeführt. Danach werden die neuen Werte nacheinander mit den Faktoren 100, 50 und 10 gerundet und so korrigiert, dass ihre Summe mit dem gerundeten Wert der ersten Zeile übereinstimmt.

Wegen der Angabe `ROUND= ... R(1), 0.2` wird der größte der drei Rundungsfaktoren verwendet, bei dem die Prozentwerte sich um weniger als 0.2 von den Prozentwerten ohne Rundung unterscheiden. In diesem Beispiel wird der Rundungsfaktor 10 gewählt, um die prozentuale Abweichung unter der angegebenen Schwelle zu halten. Prozentuierungsbasis ist dabei die erste Zeile.

9. 1. 2008

Ausgabe von Schlüsseln in Felder AK, DK, MK und UK bei leeren Datenwerten

Sind für eine Wertezuweisung mit den Zielfeldern AK, DK, MK oder UK leere Datenwerte zu übertragen, so werden ohne die Angabe `KKB` keine Daten ausgegeben. Mit der Angabe `KKB` werden auch bei fehlenden Daten Schlüssel ausgegeben.

Besitzt zum Beispiel die Variable C7 in der Wertezuweisung

`-MK1 ()=C7 KKB`

keine Angabe, so wird dafür

`C7=`

ausgegeben.

Die Wertezuweisung

`-DK1 (Alter)=N9 KKB`

erzeugt die Ausgabe

`Alter=`

wenn die Variable N9 den Wert Z0 für *keine Angabe* enthält.

6. 1. 2008

Erweiterte Aufbereitung der Zählergebnisse in Kreuztabellen

Bisher konnten die Zählergebnisse bereits mit einer Textangabe 't' hinter PRINT aufbereitet werden. Zum Beispiel stellt

```
PRINT=ABS, ' (& EUR) ' ...
```

das Zählergebnis 105 als (105 EUR) in die Tabelle.

Die Angabe 't' kann auch mehrere Zeichen & enthalten. Die Verarbeitungsregeln dazu wurden jetzt erweitert: Jedes Zeichen & wird nun durch ein einzelnes Zeichen des Zählergebnisses ersetzt, von rechts beginnend. Besitzt das Zählergebnis mehr Zeichen als & in 't' enthalten sind, so nimmt das äußerste linke & die noch verbleibenden Zeichen auf. Besitzt das Zählergebnis dagegen weniger Zeichen, so verschwinden die überzähligen & in der Ausgabe.

Mit der Angabe

```
PRINT=ABS, ' (&.&&& EUR) ' ...
```

wird das Zählergebnis 123 als (123 EUR) ausgegeben, der Wert 9324 als (9.234 EUR) und der Wert 120456 schließlich als (120.456 EUR).

Mit

```
PRINT=ABS, 1, ' (&.&&&&& EUR) ' ...
```

wird das Zählergebnis 2345.55 zunächst auf eine Nachkommastelle gerundet und danach als (2.345,6 EUR) ausgegeben.

26. 12. 2007

C-Variable mit 9 999 Merkmalen

Die Anzahl Merkmale pro C-Variable wurde von 4 000 auf 9 999 erweitert.

18. 11. 2007

Erleichterte Auswahl von Statementdateien in CNTW

Bisher wurde das Fenster zur Auswahl von Statementdateien zunächst auf das Verzeichnis CNTWORK gestellt und als Dateierweiterung die in der jeweiligen Prototypdatei festgelegten Vorgaben verwendet, zum Beispiel STM.

Die folgende Angabe in der Datei CNTW.INI legt davon abweichende Regeln fest:

STATEMENTS = Dateinamen

Mit der Eintragung

```
STATEMENTS=D:\PROGRAMME\*.PRO
```

wird das Fenster zur Dateiauswahl auf das Verzeichnis D:\PROGRAMME gestellt und nur Dateien mit der Erweiterung PRO angezeigt.

Die Angabe

```
STATEMENTS=D:\PROGRAMME\S*.PRO *.STM *.*
```

stellt ebenfalls auf das Verzeichnis D:\PROGRAMME, zeigt aber nur Dateien mit der Erweiterung PRO an, die mit dem Buchstaben S beginnen. Außerdem lassen sich Dateien mit der Erweiterung STM auswählen und durch *.* auch ganz beliebige Dateien.

Die Eintragung

```
STATEMENTS=S*.PRO *.STM *.*
```

leistet das Gleiche wie die vorige, jedoch bleibt das Fenster zur Dateiauswahl auf das Verzeichnis CNTWORK eingestellt.

7. 11. 2007

Nummernzeichen # als Separatorzeichen in CSV-Dateien

In den Statements INPUT-EXT, OUTPUT-EXT, FETCH-FILE, PAGE und PAGEP kann mit der Angabe

```
SEP=NUMBER
```

das Nummernzeichen # zum Separatorzeichen für CSV-Dateien erklärt werden.

17. 9. 2007

Zusätzliche Wertezuweisungen

Es ist jetzt möglich, Werte aus T-Variablen und alphanumerischen Fall-Identifikationen in N-Variable und numerische Datenfelder zu übertragen. Zum Beispiel:

```
ADD-PROC
-N1=T7
-F2 (1, 2) =ID
```

Dabei müssen die Werte der T-Variablen und Fall-Identifikationen nach den Regeln der D-Felder aufgebaut sein.

Umgekehrt können Werte aus N-Variablen und numerischen Fall-Identifikationen in T-Variable und alphanumerische Datenfelder übertragen werden. Zum Beispiel:

```
ADD-PROC
-T1=N7
-T2=ID
-AS1 (3) =N1
```

Dabei werden die numerischen Werte auf gleiche Weise aufbereitet wie für D-Felder.

Schließlich lassen sich mit Werten aus N-Variablen Merkmale direkt setzen. Enthält die Variable N3 im Statement

```
-C1=N3
```

den Wert 7, so wird das Merkmal C1.7 auf 1 oder *erfüllt* gesetzt und die übrigen Merkmale von C1 auf 0 oder *nicht erfüllt*. Enthält N3 den Wert Z0 oder einen Wert außerhalb der für C1 definierten Merkmale, so werden alle Merkmale von C1 gelöscht.

Auch numerische Ausdrücke sind dabei möglich:

```
-C1=N1+N2+N3
```

bildet durch Addition der drei numerischen Variablen einen Merkmalswert, setzt das entsprechende Merkmal der Variablen C1 auf 1 und löscht alle übrigen Merkmale auf 0.

Eine solche Wertezuweisung lässt sich auch auf ausgewählte Merkmale der Variablen C20 beschränken. Das Statement

```
-C20 (5 . . 7) =N3
```

setzt das Merkmal C20.5 auf 1, wenn N3 den Wert 1 enthält. Die Merkmale 6 und 7 werden gelöscht und die übrigen Merkmale nicht verändert.

Das Statement

```
-C20=101
```

schließlich setzt das Merkmal C20.101 auf 1 und löscht die sonstigen Merkmale.

Hier ist eine Änderung der CNT-Syntax zu beachten:

Bisher verstand das Programm die Konstante 101 als Kette von Merkmalswerten 0 und 1. Das vorige Statement setzte daher die Merkmale 1 und 3 auf 1. Um dieses Verfahren zu erhalten, ist jetzt das Zeichen ° vor die Merkmalswerte 0 und 1 zu stellen:

```
-C20=°101
```

Außerdem lassen sich mit der Angabe OLD im Statement OPTIONS die bisherigen Regeln für alte Statementdateien reaktivieren.

17. 9. 2007

Klasseneinteilung zu numerischen Werten

Die Werte von N-Variablen lassen sich jetzt auf einfache Weise in Klassen einteilen:

Enthält zum Beispiel die Variable N10 Altersangaben als Zahlen 0 ... 99, so setzt das Statement

```
-C10 = N10(0 30 60 100)
```

folgende drei Altersklassen in der Variablen C10:

C10.1 0 bis 29 Jahre

C10.2 30 bis 59 Jahre

C10.3 60 bis 99 Jahre

Enthält die Variable N10 den Wert Z0 für *keine Angabe*, so liefert N10(0 30 60 100) das Ergebnis Z0: Keines der Merkmale in C10 wird gesetzt. Das Gleiche geschieht, wenn N10 einen Wert kleiner als 0 oder größergleich 100 besitzt.

Mit diesem Klassen-Operanden lässt sich auch rechnen. Zum Beispiel setzt

```
-C10 = 2 + N10(0 30 60 100)
```

die Merkmale 3 bis 5 der Variablen C10.

Schließlich kann man damit auch numerische Variable füllen. Zum Beispiel stellt

```
-N21 = 2 + N10(0 30 60 100)
```

die Werte 3 bis 5 oder Z0 in die Variable N21.

2. 9. 2007

Erweiterte Ausgabemöglichkeiten der Zählergebnisse

Die Statements PAGE und PAGEP wurden erweitert, so dass die Zählergebnisse als zusätzliche Dateien in den Formaten PDF, DIF, CSV und XML ausgegeben werden können.

Außerdem ist es möglich, in einem Auswertungslauf mehrere dieser Formate auszugeben.

Enthält die Auswertung ein INDEX-Statement, so werden daraus in PDF-Dateien Lesezeichen/Bookmarks zu den Tabellen erzeugt und in XML-Dateien Hyperlinks.

Das XML-Format liefert besser formatierte Tabellen in Excel und steht jetzt auch für die Datenübertragung per Knopfdruck von CNTW an Excel zur Verfügung.

2. 9. 2007

Zusätzliche Logo-Verzeichnisse

Das Programm sucht die Logo-Dateien jetzt in weiteren Verzeichnissen in dieser Reihenfolge:

1. Im Verzeichnis der REP-Datei, wenn eine solche vorhanden ist. Neu vom Programm CNTW angelegte Auswertungen mit dem Namen ~N~.REP befinden sich stets im Verzeichnis CNTWORK.
2. Im Verzeichnis der Statementdatei.
3. Im dem durch die Angabe LOGOS= in der Datei CNTA.INI oder CNTW.INI festgelegten Verzeichnis.
4. Im Programmverzeichnis CNTW oder CNTA.

16. 2. 2007

Zählergebnisse als PDF-Datei ausgeben

Bisher ließen sich die Zählergebnisse nur über CNTW als PDF-Datei ausgeben.

Jetzt kann auch CNTA solche PDF-Dateien erstellen. Dazu wurde in den Statements PAGE und PAGEP der folgende Parameter eingeführt:

< PDF = | dateiname | >
| NO |

Diese Angabe stellt die Auswertungsergebnisse der folgenden Statements im PDF-Format in die Datei *dateiname*. Durch ein späteres PAGE- oder PAGEP-Statement kann diese Angabe verändert werden.

dateiname Name der Datei, die die Auswertungsergebnisse der folgenden Statements aufnehmen soll.

Datenamen mit Leerzeichen oder Kommas sind in Hochkommas ' einzuschließen. Zum Beispiel:
 PDF='C:\STUDIE 105.PDF'

Enthält der Dateiname das Zeichen &, so wird für jede Tabelle eine neue Datei erstellt und das Zeichen & durch eine laufende Nummer 1, 2, ... ersetzt. Mehrere zusammenstehende &-Zeichen erzeugen dabei eine laufende Nummer mit einer entsprechenden Anzahl Ziffern. Die Angabe

PAGE PDF=C:\TAB&&.PDF

stellt die erste Tabelle in die Datei *TAB01.PDF*, die zweite Tabelle in *TAB02.PDF*, die dritte in *TAB03.PDF* usw.

NO setzt die Angabe PDF aus früheren PAGE- oder PAGEP-Statements außer Kraft.

3. 1. 2007

Mehrfachnennungen in MS- und MK-Feldern von OUTPUT-EXT

Im Statement OUTPUT-EXT wurde der folgende Parameter neu eingeführt:

< MSTD = NS >

Diese Angabe beeinflusst die Ausgabe von Mehrfachnennungen in MS- und MK-Feldern. Ohne MSTD=NS erscheinen Mehrfachnennungen mit Von-Bis-Abkürzungen in der Form:
 1 3-7 10

Mit MSTD=NS werden alle Nennungen einzeln aufgeführt:

1 3 4 5 6 7 10

30. 12. 2006

Unterstreichungsstrich _ in den Texten

Soll eine Textzeile einen Unterstreichungsstrich _ enthalten, so sind dafür zwei direkt hintereinander stehende Unterstreichungsstriche __ anzugeben, um es von den Soll-Trennstellen _ für FIT zu unterscheiden. Zum Beispiel erzeugt das Statement

'Frage 1__a'

den Text *Frage 1_a*

30. 12. 2006

Nummernzeichen # in den Texten

Soll eine Textzeile ein Nummernzeichen # enthalten, so sind dafür zwei direkt hintereinander stehenden Nummernzeichen ## anzugeben, um es von Makronamen zu unterscheiden. Zum Beispiel erzeugt das Statement

```
'Marke ##3'
```

den Text *Marke #3*

15. 11. 2006

Erweiterung der Randgewichtung mit WEIGHT

Das Statement WEIGHT hat eine neue Optimierungsfunktion zur Ermittlung von Randgewichten erhalten. Diese Funktion versucht bei problematischen Wichtungsaufgaben sehr kleine Gewichte nahe bei Null zu vermeiden. Damit wird die Wichtungsqualität bei stark disproportionalen Untersuchungen wesentlich verbessert. Zur Auswahl der Zielfunktion wurde der neue Parameter FUNCTION eingeführt.

Gleichzeitig wurde die Anzahl der möglichen Randbedingungen pro Wichtungslauf deutlich erhöht.

< **FUNCTION = n** >

Diese Angabe bestimmt das Wichtungsverfahren für Randgewichtungen.

FUNCTION=1 ist ein innovatives Optimierungsverfahren, das mit einer aufwendigeren Zielfunktion sehr kleine Gewichte in der Nähe von Null vermeidet. Einzelne größere Gewichte sind hin und wieder als Ausreißer zu erwarten, lassen sich jedoch mit MAX=CLIP unter Verzicht auf etwas Effizienz eliminieren.

FUNCTION=2 arbeitet mit der üblichen quadratischen Zielfunktion. Bei stark disproportionalen Stichproben ist die Qualität der Gewichte meistens schlechter als mit FUNCTION=1. Die Zählergebnisse der gewichteten Daten können sich dann zwischen FUNCTION=1 und FUNCTION=2 statistisch signifikant unterscheiden.

Bei unproblematischen Wichtungsaufgaben mit wenigen kleinen Gewichten unterscheiden sich die Ergebnisse von FUNCTION=1 und FUNCTION=2 nicht sehr. Mit wachsender Disproportionalität zwischen Ist- und Sollwerten nehmen die Unterschiede deutlich zu.

FUNCTION=3 benutzt ebenfalls eine quadratische Zielfunktion, versucht aber mit heuristischen Methoden sehr kleine Gewichte zu vermeiden. Dies ist das ursprüngliche von CNTA benutzte Verfahren.

Fehlt die Angabe FUNCTION, so wird FUNCTION=1 angenommen.

Ausführliche Beschreibungen zu diesen Funktionen finden sich im CNTA-Handbuch oder mit CNTW unter *Hilfe / CNTA-Statements ...*

Zur besseren Bewertung der Wichtungsergebnisse werden in den Zählprotokollen neben der Standardabweichung der Gewichte jetzt die beiden Werte *Effizienz 1* und *Effizienz 2* ausgewiesen.

Dies sind Zahlenwerte zwischen 0 und 100. Je näher sie bei 100 liegen, desto besser das Wichtungsergebnis im Sinne der jeweiligen Zielfunktion.

Für die neuen Optimierungsfunktionen wurden die Angaben MAX und MIN erweitert:

MAX = CLIP

Diese Angabe senkt die größten Fallgewichte auf Kosten der Effizienz ab. Dafür wird maximal 1% Effizienz geopfert.

MIN = CLIP

Diese Angabe hebt die kleinsten Fallgewichte auf Kosten der Effizienz an. Dafür wird maximal 1% Effizienz geopfert.

1. 11. 2006

Erweiterung der DIF- und Excel-Übergabe

Sämtliche Auswertungsergebnisse - XTAB, CODEBOOK, INDEX, HOLECOUNT und TEXT - lassen sich jetzt aus CNTW durch Knopfdruck an EXCEL übertragen.

Außerdem lassen sich alle Auswertungsergebnisse auch aus den Statements heraus in CSV- und DIF-Dateien ausgeben. Dazu wurden in den Statements PAGE und PAGEP die neuen Parameter CSV und DIF eingeführt. Zum Beispiel stellt das Statement

```
PAGEP CSV=resultate.csv
```

die Auswertungsergebnisse aller folgenden Statements im CSV-Format in die Datei *resultate.csv* und

```
PAGEP DIF=resultate.dif
```

im DIF-Format in die Datei *resultate.dif*.

In den XTAB-Statements entfallen dafür die Parameter DIF und DIFS.

Für die CSV-Ausgabe werden als Separatorzeichen das Semikolon und als Texterkennungszeichen die Anführungsstriche " verwendet. Mit der neuen Angabe

```
< SEP = | 's' | < , 't' >
        | TAB |
```

in den Statements PAGE und PAGEP lassen sich abweichende Separatorzeichen *s* und Texterkennungszeichen *t* vorgeben. Mit TAB wird das Tabulatorzeichen (ANSI/ASCII 9) zum Separatorzeichen.

15. 10. 2006

Variablenbeschreibungen in OUTPUT-EXT durch VNAME

Die Angabe VNAME erzeugt bisher bei SPSS-Ausgaben die SPSS-Syntax-Datei mit den Definitionen der ausgegebenen Variablen und Daten. Jetzt ist VNAME auch in den Formaten TRIPLES und AUTO ... möglich:

AUTOASCII ... In diesen Formaten erstellt VNAME die Variablenbeschreibungen im CSV-Format, mit dem Semikolon als Separatorzeichen und den Anführungszeichen " zur Texterkennung. Eine solche Datei könnte zum Beispiel folgendermaßen aussehen:

```
C10; "Beilage bemerkt" ; ; ; M1(4,1)
; ; 1 ; "ja"
; ; 2 ; "nein"
C11; "Wie viele Beilagen erhalten"; ; ; M1(5,4)
; ; 1 ; "9 - 12"
; ; 2 ; "6 - 8"
; ; 3 ; "3 - 5"
; ; 4 ; "1 - 2"
; ; 5 ; "keine"
C12; "Wie viele Beiträge gelesen" ; ; ; M1(6,1)
; ; 1 ; "fast alle Beiträge"
; ; 2 ; "etwa drei Viertel"
```

Die Angabe VNAME ist bei den Formaten AUTOASCII ... nicht zwingend erforderlich. Zur leichteren Verarbeitung der erstellten Datei durch EXCEL sollte als Dateierweiterung CSV verwendet werden.

SPSS Hier ist die Angabe VNAME zwingend erforderlich. Es werden Variablendefinitionen in Form der SPSS-Statements DATA LIST, VARIABLE LABELS und VALUE LABELS ausgegeben. Zur späteren Verarbeitung als Syntax-Datei durch SPSS sollte die Dateierweiterung SPS verwendet werden.

TRIPLES Hier ist die Angabe VNAME zwingend erforderlich. Die Variablendefinitionen werden im TRIPLE-S-Format der Version 1.1 ausgegeben.

15. 10. 2006

Neues Ausgabeformat TRIPLES in OUTPUT-EXT

Das neue Ausgabeformat TRIPLES erstellt analog zu SPSS eine Datendatei FNAME und eine Datei VNAME mit einer Beschreibung der ausgegebenen Variablen im Triple-S-Format. Siehe www.triple-s.org.

Die Organisation der Ausgabedatei erfolgt automatisch: Position, Länge und Typ der Datenfelder legt das Programm selbstständig fest. Dabei werden die Werte der vorhandenen Variablen in die externe Datei ausgegeben.

Mit Folgestatements hinter OUTPUT-EXT lassen sich Reihenfolge und Auswahl der Variablen sowie externe Feldtypen verändern. Die Parameter CSTD, DSTD, NSTD und VSTD verändern die Ausgaberegeln. Die Angabe HEAD in OUTPUT-EXT erzeugt eine <TITLE>-Angabe in der Datei VNAME.

15. 10. 2006

Feldlängen bei SPSS und AUTOASCII ... in OUTPUT-EXT

Ohne besondere Angaben reserviert das Programm in den Formaten SPSS und AUTOASCII ... in den Ausgabesätzen so viel Platz, wie die in den Variablen vorkommenden Werte gerade benötigen. Bei C-Variablen mit Mehrfachnennungen wird außerdem dafür gesorgt, dass sich die vorhandenen Nennungen auch ausgeben lassen.

Jetzt ist es jetzt möglich, die Größe der externen Datenfelder für N- und T-Variable vorzugeben. Zum Beispiel:

```
OUTPUT-EXT AUTOASCII ...  
-N17 D=6  
-T117 A=30
```

Hier wird für die Variable N17 ein externes Datenfeld im D-Format verlangt, das mindestens sechsstellige Zahlen aufnehmen kann. Variable mit längeren Werten führen zu Fehlermeldungen bei der Datenausgabe.

Für die Variable T117 wird ein Datenfeld im A-Format angelegt, mit Platz für 30 Zeichen. Auch hier führen längere Texte zu Fehlermeldungen bei der Datenausgabe.

Zusätzlich ist im Statement OUTPUT-EXT die Angabe VNAME verfügbar:

VSTD Mit dieser Angabe erfolgt die Platzeinteilung nicht anhand der Eingabedaten sondern nach der Variablendefinition: Die Datenfelder werden so gewählt, dass auch der größtmögliche Variablenwert Platz findet. Damit wird der Aufbau der Ausgabedatei unabhängig von den zufälligen Eingabewerten.

Bei C-Variablen sorgt VSTD nur für Einfachnennungen. In den Folgestatements von OUTPUT-EXT kann zu den Variablen jedoch die gewünschte Anzahl von Nennungen vorgegeben werden. Zum Beispiel reserviert das Folgestatement von OUTPUT-EXT

```
-C10 M=3
```

Platz für drei Nennungen der Variablen C10 im M-Format.

15. 6. 2006

größere Zeilenanzahl LINES im Statement PAGE

Die maximale Zeilenanzahl LINES wurde von 32 000 auf 1 000 000 erhöht.

Der höchstmögliche Wert wird zusätzlich durch die Eintragung MAXWORK in den Dateien CNTA.INI oder CNTW.INI begrenzt: Die durch LINES und POS vorgegebene Anzahl von Zeichen pro Blatt darf nur die Hälfte des durch MAXWORK vorgegebenen Arbeitsbereichs verbrauchen.

29. 5. 2006

Folgestatements zu OUTPUT-INT für die Datenerfassung mit CNTD

Die statistischen Fälle einer internen Datei lassen sich mit dem Erfassungsprogramm CNTD anzeigen, bearbeiten und erfassen:

- w Die Dateneingabe erfolgt direkt in die Variablen.
- w Regeln zur Prüfung der Eingabewerte bei der Erfassung können vorgegeben werden.
- w Sprungbedingungen ermöglichen eine von den Eingabewerten abhängige Filtersteuerung.
- w Automatische und halbautomatische Vercodung offener Texte.
- w CNTD kann auch ohne CNTW-Dongle arbeiten. Dann ist jedoch nur die Neuerfassung für die hinter START-CNTD aufgeführten Variablen möglich.

Die Regeln zur Datenerfassung werden durch Folgestatements von OUTPUT-INT vorgegeben, die mit folgendem Statement einzuleiten sind:

```

-START-CNTD < | ID | = nr > <AUTO> < LANGm > < STATUS = | Nn | >
              | ID1 |          | Cn          |
              | ID2 |          |
              | ID3 |          |
              | ID4 |          |
    
```

nr Anzahl Zeichen der ID-Stufe für die Erfassung. Numerische ID-Stufen erlauben bis zu 9 Zeichen, alphanumerische so viele, wie bei der Definition der ID-Stufe festgelegt.

Fehlt diese Angabe für eine ID-Stufe, so wird die größtmögliche Zeichenanzahl verwendet.

AUTO Mit dieser Angabe werden die Fall-Identifikationen in CNTD durch automatisches Weiterzählen

zu Beginn jedes neuen Falles ermittelt. Bei mehrstufigen Fall-Identifikationen gilt dies nur für die unterste Stufe.

LANGm Mit der Angabe *m* wird die Sprache der Variablentexte ausgewählt, die in CNTD anzuzeigen sind.

Damit ist es möglich, den Variablen unterschiedliche Texte für die Datenerfassung und Auswertung zu geben. Fehlt die Angabe LANG, so werden die Texte zu der niedrigsten vorhandenen Nummer *m* in CNTD angezeigt.

Cn Diese Variable nehmen den Erfassungsstatus der Fälle auf:

Nn Sie erhalten den Wert 1, wenn der Fall vollständig mit CNTD erfasst wurde und den Wert 0, wenn die Erfassung vorzeitig beendet wurde.

Hinter START-CNTD werden mit weiteren Folgestatements die Variablen und ihre Prüfungsregeln für die Datenerfassung ausgewählt. Die hier angegebenen Variablen werden in die Datei OUTPUT-INT übernommen. Ihre Reihenfolge hinter START-CNTD legt auch die Reihenfolge für die Erfassung fest.

Auch vor dem Statement START-CNTD können Variable zur Ausgabe in die interne Datei angegeben werden. Für die Neuerfassung sind diese jedoch nur verfügbar, wenn sie noch einmal hinter START-CNTD erscheinen.

Die Variablen für die Datenerfassung sind in folgender Form auszuwählen:

```
-Cn < v < .. b > > < m : z > < R >
```

Dieses Statement wählt eine C-Variable zur Erfassung aus.

v oder *v .. b*

Diese Angabe bestimmt die Anzahl zulässiger Nennungen der Variablen *Cn*:
v ist die Mindestanzahl Nennungen und *b* die Höchstanzahl.

Wird nur *v* angegeben, so sind genau *v* Merkmalsnummern einzugeben.

Fehlen beide Angaben, so ist nur ein Merkmal erlaubt.

m : z Mit dieser Angabe lässt sich die Erfassung der direkt hinter diesem Statement liegenden Variablen überspringen. *m* ist dabei eine Merkmalsnummer von *Cn* und *z* ein Sprungziel hinter dem laufenden Statement. Die Erfassung verzweigt zum Sprungziel *z*, wenn das Merkmal *m* für die Variable *Cn* eingegeben wurde. Siehe unten *Filter und Sprünge*.

R Diese Angabe erzwingt die Eingabe von Werten (required).
Ohne *R* kann die Variable leer bleiben, auch wenn Mindestnennungen mit *v* vorgegeben sind.

Zum Beispiel:

-C900 4:C9	Es kann nur ein Merkmal erfasst werden, keine Angabe ist möglich. Nach Eingabe von Merkmal 4 wird mit Variable C9 fortgefahren.
-C80 [3] 1..6	1 bis 6 Nennungen oder keine Angabe sind zulässig.
-C19 3	3 Nennungen oder keine Angabe sind erlaubt.
-C9 3 R	Es sind genau 3 Merkmale einzugeben.
-C1..5 R	Alle zwischen C1 und C5 definierten Variablen verlangen genau eine Angabe.
-C10 [3]..4	Die Variablen C[3] und C10[4] werden mit einer oder keiner Angabe erfasst.

-Nn <v <.. b>> <R>

Dieses Statement wählt eine N-Variable zur Erfassung aus.

v oder *v..b*

Diese Angabe legt die zulässigen Eingabewerte für die Variable *Nn* fest:
v gibt den kleinsten Wert an und *b* den größten. Wird nur *v* angegeben, so ist nur dieser Wert zulässig. Fehlen beide Angaben, so sind Eingabewerte zwischen 1 und dem größtmöglichen Wert der Variablen erlaubt.

R Diese Angabe erzwingt die Eingabe von Werten (required).
Ohne *R* kann die Variable auch leer bleiben.

In diesem Beispiel soll die Variable N10 unter DEFS mit -N10:3 definiert sein:

-N10	Werte zwischen 1 und 999 oder keine Eingabe sind möglich.
-N10 -10..10	Werte zwischen -10 und +10 oder keine Eingabe sind zulässig.
-N10 987	Nur der Wert 987 oder keine Eingabe sind erlaubt.
-N10 R	Es muß ein Wert zwischen 1 und 999 eingegeben werden.

-Tn <v <.. b>> <R>

Dieses Statement wählt eine T-Variable zur Erfassung aus.

v oder *v..b*

Diese Angabe legt die Länge der Eingabetexte für die Variable *Tn* fest.
v ist die Mindestlänge der Eingabewerte und *b* die Höchstlänge.
Ist nur *v* angegeben, so müssen alle Eingabewerte genau diese Länge besitzen. Ohne diese Angaben muß die Eingabelänge zwischen einem Zeichen und der definierten Länge der Variablen liegen.

R Diese Angabe erzwingt die Eingabe von Werten (required).
Ohne *R* kann die Variable auch leer bleiben.

Zum Beispiel:

-T101	Eingaben bis zur Länge der Variablen sind möglich oder keine Eingabe.
-T101 2..10	Die Eingaben müssen aus 2 bis 10 Zeichen bestehen oder leer sein.

- T101 5 Genau fünf Zeichen oder keine Eingabe sind zulässig.
- T101 4 R Eine Eingabe aus vier Zeichen ist zwingend erforderlich.

DO-Statements

DO-Statements hinter `START-CNTD` dienen in erster Linie der Erfassung von Gitterfragen. Eine solche Frage könnte im Fragebogen so aussehen:

Kinder im Haushalt			
	männlich	weiblich	Alter
1. Kind	1	2	∴ ∴
2. Kind	1	2	∴ ∴
3. Kind	1	2	∴ ∴
4. Kind	1	2	∴ ∴
-

Die folgenden CNTA-Statements erzeugen eine dazu passende Erfassungsdatei:

```
DEFS
-C1[5]:2 'Geschlecht' 1='männlich' 2='weiblich'
-N1[5]:2 'Alter'

OUTPUT-INT ...
-START-CNTD
-DO <1>=1..5
-C1[<1>]
-N1[<1>]
-EDO
-C2
```

Die Dateneingabe zu dieser Frage erfolgt zeilenweise in der Reihenfolge `C1[1] N1[1] C1[2] N1[2] ...`. Die Taste `÷` im numerischen Block beendet dabei die DO-Schleife vorzeitig und fährt mit der Variablen `C2` fort. Die Taste `Enter` beendet die laufende Zeile vorzeitig und fährt mit dem Anfang der nächsten Zeile fort.

Filter und Sprünge

Abhängig von bereits eingegebenen Werten, kann die Erfassung für weitere Variable übersprungen werden. Dazu lassen sich Sprungbedingungen und Sprungziele in den Folgestatements hinter `START-CNTD` angeben. Zum Beispiel:

```
OUTPUT-INT FNAME=1022.INT
-START-CNTD
-C1 1:C3           Die Angabe 1:C3 bewirkt nach Eingabe des Merkmals 1 den
                   Sprung zur Variablen C3.
-GOTO END IF C1.Z0 Beendet die Erfassung des laufenden Falles, wenn keine Eingabe
                   zur Variablen C1 vorliegt.
-C2
-C3
```

GOTO-Statements

GOTO-Statements lassen sich überall in die Folgestatements von `START-CNTD` einfügen. Sie beginnen am Zeilenanfang mit der Angabe:

```
-GOTO sprungziel
```

Ohne zusätzliche Angaben führt dieses Statement zu einem unbedingten Sprung zum *sprungziel* sobald die Erfassung dieses Statement erreicht.

Die GOTO-Statements können auch mit einer Bedingung versehen werden in der Form:

```
-GOTO sprungziel IF logischer Ausdruck
```

In diesem Fall wird nur dann beim *sprungziel* fortgefahren, wenn die Bedingung im *logischer Ausdruck* erfüllt ist. Hier sind alle Angaben möglich, die im Abschnitt *Logische Ausdrücke* beschrieben sind. Zum Beispiel:

```
OUTPUT-INT FNAME=1022.INT
-START-CNTD
-C1 1:C3
-GOTO C3 IF C1.1|..4
-GOTO C5 IF C1.5|..8
-C2           Diese Variable wird nur erfasst, wenn C1 nicht die Werte 1 ... 8 besitzt.
-GOTO C6
-C3           Diese Variable wird nur erfasst, wenn C1 die Werte 1 ... 4 besitzt.
-GOTO C6
-C5           Diese Variable wird nur erfasst, wenn C1 die Werte 5 ... 8 besitzt.
-C6           Hier wird in jedem Fall fortgesetzt.
```

Sprungziele

Sprungziele müssen immer hinter der Zeile liegen, von der aus gesprungen werden soll.

Als Sprungziele sind folgende Angaben möglich:

w Name einer Variablen, die in den Folgestatements hinter der laufenden Zeile erscheint.

w END oder ENDE zum vorzeitigen Beenden der Eingabe eines statistischen Falles.

w Ein Sprunglabel, bestehend aus bis zu 10 beliebigen Zeichen ungleich END oder ENDE.

Ein solches Sprunglabel lässt sich überall in die Folgestatements einfügen, muß aber am Zeilenanfang mit den Zeichen `->` beginnen. Zum Beispiel:

```
->frage4
```

Beispiel: Leere Erfassungsdatei für CNTD erstellen

```
DEFS
-C1:2   'Geschlecht' 1='Männer' 2='Frauen'
-N1:2   'Alter'
-T1:30  'Name des Befragten'

OUTPUT-INT FNAME=demo.int
-START-CNTD ID=4 AUTO
-T1 R
-N1 14 .. 99
-C1
```

Das Statement DEFS und seine Folgestatements definieren verschiedene Variable.

Das Statement OUTPUT-INT erzeugt die interne Datei für die Datenerfassung. Da kein Statement INPUT-EXT und INPUT-INT vorliegt, enthält die Erfassungsdatei Variablendefinitionen und Erfassungsregeln aber keine statistischen Fälle.

Das Statement START-CNTD stellt Angaben zur Datenerfassung mit CNTD in die Ausgabedatei. Die Angabe ID=4 verlangt maximal vier Zeichen lange Fall-Identifikationen. AUTO sorgt bei der Erfassung für automatische Vergabe der Fall-Identifikation.

Die Folgestatements hinter START-CNTD wählen die Variablen T1, N1 und C1 für die Neuerfassung in der hier angegebenen Reihenfolge aus.

Hinter -T1 sorgt R dafür, dass der Name des Befragten zwingend einzugeben ist.

Zu -N1 sind nur Altersangaben zwischen 14 und 99 möglich.

Da bei N1 und C1 die Angabe R fehlt, können diese Variablen auch leer bleiben.

11. 5. 2006

Ausgabe von Modalwerten in XTAB

Hinter ROW und COL können Modalwerte zu C- und N-Variablen mit den Funktionen MODL und MODU angefordert werden. Diese beiden Funktionen liefern nur dann unterschiedliche Ergebnisse, wenn eine Variable mehrere Modalwerte besitzt: MODL zeigt den kleinsten und MODU den größten davon. Zum Beispiel

```
XTAB ROW=TOTAL; C10; MODL (C10)
      PRINT=ABS
      PRINT1=MOD, '*', F7 ...
```

Dieses Statement stellt die Modalwerte von C10 hinter den Einzelwerten von C10 in die Tabelle. Die Angabe MOD hinter PRINT1 markiert den Modalwert mit einem Stern, falls die Variable mehrere Modalwerte besitzt. Außerdem wird in diesem Fall der Text des Textelements (7) als Fußnote unter der Tabelle ausgegeben.

Hinter DEFS können zum Beispiel in der Form

```
DEFS
...
-MODL      'Modalwert'
-MODU      'Modalwert'
```

Standardtexte für diese Funktionen vorgegeben werden.

11. 5. 2006

Ausgabe von Perzentilen in XTAB

Die Funktion PERCTL liefert hinter ROWm und COL Perzentilwerte zu C- und N-Variablen.

Zum Beispiel stellt das Statement

```
XTAB ROW=TOTAL; C10; PERCTL (C10, 90) ...
```

die 90-Prozent-Perzentile der Variablen C10 in die Tabelle.

Hinter DEFS kann zum Beispiel in der Form

```
DEFS
...
-PERCTL    'Perzentil'
```

ein Standardtext für diese Funktion vorgegeben werden.

1. 4. 2006

Ausgabe von Makros in CODEBOOK und OUTPUT-INT

Bisher mussten die Zeichen # vor den auszugebenden Makros weggelassen werden; jetzt sind sie anzugeben.

Statt: -MACROS col5 col7..col10

jetzt: -MACROS #col5 #col7..#col10

25. 3. 2006

OUTPUT-EXT mit AUTOASCII, AUTOANSI, AUTOEBCDIC und KEYLINE

In OUTPUT-EXT mit der Angabe SEP sind jetzt auch die Datenfelder AK, DK, MK und UK zulässig. Dabei können die Schlüssel dieser Felder durch die Angaben KEYLINE oder KEYLINES in die ersten Datensätze der Ausgabedatei gestellt werden.

Dies gilt auch für die Datenformate AUTOANSI, AUTOASCII und AUTOEBCDIC.

28. 2. 2006

Bessere Feldlängenermittlung bei OUTPUT-EXT mit AUTOASCII ... SPSS

Hier wurde bisher in den Ausgabesätzen zu den einzelnen Variablen Platz für den größtmöglichen Wert reserviert, den die Variable nach Definition annehmen kann. Für Mehrfachnennungen von C-Variablen musste die maximal zulässige Zahl von Nennungen in Folgestatements von OUTPUT-EXT angegeben werden.

Jetzt werden vor der Platzvergabe in den Datensätzen die tatsächlich vorhandenen Variablenwerte ermittelt und nur der dafür nötige Platz reserviert. Mehrfachnennungen werden automatisch ausgegeben, es sind dazu keine Angaben in Folgestatements mehr nötig.

Dadurch werden die externen Datenfelder vom Typ E für numerische Werte ohne Vorzeichen überflüssig.

25. 2. 2006

OUTPUT-EXT mit AUTOASCII, AUTOANSI, AUTOEBCDIC auch als CSV-Datei

In den Datenformaten AUTOANSI, AUTOASCII und AUTOEBCDIC lassen sich jetzt durch die Angabe SEP auch CSV-Dateien erstellen.

23. 2. 2006

OUTPUT-EXT mit AUTOANSI

Zusätzlich zu AUTOASCII wurde das neue Format AUTOANSI zur Ausgabe von Textdaten im ANSI-Format eingeführt.

12. 2. 2006

Erweiterung des Parameters KEYLINE in INPUT-EXT und FETCH-FILE

Liegen mehrere Datensätze pro Fall vor, so kann jetzt wahlweise mit einer einzigen Schlüsselzeile für alle Satzarten oder mit einer eigenen Schlüsselzeile für jede Satzart gearbeitet werden:

KEYLINE Diese Angabe bestimmt, dass der erste Datensatz Schlüssel für die Wertezuweisungen durch AK-, DK-, MK- und UK-Felder enthält. Die Datenwerte der Felder müssen in der gleichen Spalte wie ihr Schlüssel stehen. Eine Datei zum Statement

```
INPUT-EXT KEYLINE ID=DK(fall) ...
```

könnte folgendermaßen beginnen:

```
Fall;Geschlecht;Alter
001;      2;    35
002;      1;    47
```

Werden durch die Angabe RC mehrere Datensätze pro Fall angefordert, so gilt die Schlüsselzeile für alle Datensätze. Die Datei zum Statement

```
INPUT-EXT KEYLINE ID=DK(fall) RC=DK(satz)1,2 ...
```

könnte folgendermaßen beginnen:

```
Fall;Satz;Marke;bekannt
001;  1;  17;    1
001;  2; 112;    2
002;  1;  79;    2
```

KEYLINES Wie KEYLINE erklärt diese Angabe die ersten Zeilen der Datei zu Schlüsselzeilen. Sie verlangt jedoch für jede Satzart eine eigene Schlüsselzeile. Die Datei zum Statement

```
INPUT-EXT KEYLINES ID=DK(fall) RC=DK(satz)1,2...
```

könnte folgendermaßen beginnen:

```
Fall;Satz;Geschlecht;Alter
Fall;Satz;Marke;bekannt
001;  1;      2;    35
001;  2;   17;    1
002;  1;      1;    47
002;  2; 112;    2
```

2. 2. 2006

Erweiterungen zu CODEBOOK

Zu den T-Variablen wird die Zeichenanzahl des kürzesten und des längsten Textes der Variablen ausgewiesen.
Zum Beispiel:

```
T30: 20 Name des Befragten
      Fälle mit Angaben    311 59.7%
      Fälle ohne Angaben   210 40.3%
      kleinste Länge        4
      größte Länge         18
```

Bei C-Variablen erhalten die Angaben Z0 ... Z3 einen erläuternden Text:

```
C10: 4 Alter des Befragten
      1 18-25 Jahre    130 25.0%
      2 26-40 Jahre   104 20.0%
      3 41-60 Jahre    78 15.0%
      4 61+ Jahre     171 32.8%
      Z0 keine Angabe  38  7.3%
      Z1 eine Angabe  483 92.7%
```

11. 11. 2005

Verarbeitungsläufe ohne INPUT-EXT und INPUT-INT

Es sind jetzt CNTA-Läufe ohne die Statements INPUT-INT und INPUT-EXT möglich.

Damit können mit DEFS Variable definiert und durch OUTPUT-INT auf eine interne Datei ohne statistische Fälle ausgegeben werden; zum Beispiel für die Datenerfassung mit CNTD.

10. 11. 2005

Schlüssel in den externen Datenfeldern AK DK MK UK ohne Hochkommas

Die Schlüssel in diesen Datenfeldern können jetzt auch ohne Hochkommas eingegeben werden. Zum Beispiel

```
-N1=DK(abc) statt -N1=DK('abc')
```

Nur wenn der Schlüssel selbst Klammern (oder) enthält, ist er weiterhin mit Hochkommas zu umgeben.

Zum Beispiel -N1=DK('abc()')

25. 6. 2005

Texterkennungszeichen in CSV-Dateien

Die Statements INPUT-EXT, OUTPUT-EXT und FETCH-FILE können CSV-Dateien verarbeiten. Dabei ist es jetzt auch möglich, mit Texterkennungszeichen eingerahmte alphanumerische Werte zu verarbeiten.

Zum Beispiel liest das Statement

```
INPUT-EXT SEP=',','"' ...
```

eine CSV-Datei, mit Kommas als Trennzeichen zwischen den Datenwerten.

Darüber hinaus dient das Zeichen " als Texterkennungszeichen, das am Anfang und Ende alphanumerischer Werte für AS- und AK-Felder stehen kann. Damit wird es möglich mit Kommas, obwohl dies die Separatorzeichen sind.

30. 4. 2005

Neue Aufrufparameter für CNTW

Für den Aufruf von CNTW aus der Kommandozeile wurden neue Parameter eingerichtet. Damit kann zum Beispiel ein Editor-Programm CNTW direkt aufrufen, dabei eine REP-Datei und PTT-Datei vorgeben sowie den Namen der gerade editierten Statementdatei als @-Variable in diese REP-Datei stellen.

reportdatei Es kann der Name einer Reportdatei mit der Dateierweiterung REP angegeben werden.

Zum Beispiel wird mit
`CNTW C: LA2195. REP`
 das Programm CNTW gestartet und sofort die Auswertung L2195.REP aktiviert.

Dateinamen mit Leerzeichen sind mit Anführungsstrichen " oder ' zu versehen. Zum Beispiel:
`CNTW "C: \Neue Studien\ LA 219. REP"`

Ist eine hier angegebene Reportdatei nicht vorhanden, so wird sie von CNTW beim Aufruf neu angelegt.

Fehlt in der Angabe *reportdatei* der Pfad, so wird dafür das Verzeichnis CNTWORK verwendet. Fehlt der Parameter *reportdatei*, obwohl andere Parameter angegeben sind, so wird dafür die Auswertungsdatei ~N~.REP im Verzeichnis CNTWORK verwendet.

prototyp Es kann der Name einer Prototypdatei mit der Dateierweiterung PTT angegeben werden. Diese Angabe wird verwendet, wenn für diesen Aufruf eine neue REP-Datei zu erstellen ist.

Zum Beispiel wird mit
`CNTW C: LA2195. REP C: \CNTW TAB. PTT`
 das Programm CNTW gestartet. Ist die Datei C:LA2195.REP dabei noch nicht vorhanden, so wird sie neu eingerichtet und mit der Prototypdatei C: \CNTW TAB.PTT versehen. Ist die Datei C:LA2195.REP dagegen schon vorhanden, so bleibt die Angabe *prototyp* wirkungslos, auch wenn die Auswertungsdatei eine abweichende Prototyp-Angabe besitzt.

Fehlt in der Angabe *prototyp* der Pfad, so wird dafür das Verzeichnis CNTW verwendet. Ist eine neue REP-Datei einzurichten und fehlt die Angabe *prototyp*, so wird dafür die Datei CNTA.PTT aus dem Verzeichnis CNTW verwendet.

-run Dieser Parameter kann in Groß- oder Kleinschrift eingegeben werden. Er sorgt dafür, dass nach dem Start des Programms ohne weitere Eingabe sofort die Zählung mit der *reportdatei* ausgeführt wird. Zum Beispiel:
`CNTW C: LA2195. REP - RUN`

@-variable Am Ende dieser Parameter sind eine oder mehrere @-Variable mit Gleichheitszeichen und Werten

möglich. Zum Beispiel
`CNTW LA2195. REP CNTA. PTT - RUN @STMTS=LA2195. STM @NPUT=LA2195. INT`
 Dann werden die Werte LA2195.STM und LA2195.INT der @-Variablen @STMTS und @INPUT in die *reportdatei* geschrieben, bevor diese von CNTW eingelesen wird.

Mit dieser Beispielzeile könnte ein Editorprogramm CNTW aufrufen, nachdem die Statementdatei LA2195.STM gespeichert wurde. CNTW würde danach die Auswertung mit dieser Datei sofort beginnen und anschließend ohne weiteren Eingriff die Ergebnisse vorzeigen.

28. 4. 2005

Wertezuweisungen aus leeren Datenfeldern

Bei einer Wertezuweisung von Texten in der Form

-T1=A1(1,5) SKB

verhindert die Angabe SKB die Ausführung des Statements, wenn das Datenfeld A1(1,5) nur Leerzeichen enthält.

Die Angabe SKB ist jetzt auch bei der Zuweisung von Bedingungs- und numerischen Werten möglich.

Sie lässt das Zielfeld oder die Zielvariable unverändert, wenn der Eingabewert nur aus Leerzeichen besteht oder bei AK-, DK und UK-Feldern keine Werte im Datensatz vorhanden sind.

Zum Beispiel:

-C10=MK() SKB

-N20=DS(7) SKB

-N30=D(10,4) SKB

31. 3. 2005

Erweiterung der Funktionen MEAN, MED ... in XTAB

Neu eingeführt wurden die folgenden Funktionen für ROW und COL:

MAX(Cn) Größtes erfülltes Merkmal der Bedingungsvariablen Cn.

MAX(Nn) Größter Wert der numerischen Variablen Nn.

MIN(Cn) Kleinstes erfülltes Merkmal der Bedingungsvariablen Cn.

MIN(Nn) Kleinster Wert der numerischen Variablen Nn, der von Z0 (keine Angabe) verschieden ist.

MED(Nn) Median M aus den Werten der numerischen Variablen Nn, die von Z0 verschieden sind.

Darüber hinaus ist es jetzt möglich, alle Funktionen gleichzeitig in ROW und COL zu benutzen.

30. 3. 2005

Verbesserte Ausgabe numerischer Werte in externe Dateien

Bei der Ausgabe numerischer Werte in externe Felder vom Typ D, DK und DS wurden Nachkommastellen bisher ohne Dezimalzeichen *Punkt* oder *Komma* ausgegeben.

Jetzt werden Zahlenwerte mit Nachkommastellen in der Regel mit Dezimalzeichen ausgegeben.

Dafür wurde im Statement OUTPUT-EXT der Parameter DSTD folgendermaßen erweitert:

```
< DSTD = | LB | < , | '.' | > >
          | LZ | | ',' |
          | ND |
```

Diese Angabe beeinflusst die Ausgabe numerischer Werte in die externe Datei.

Ohne die Angabe DSTD in OUTPUT-EXT wird mit DSTD=LZ gearbeitet.

'.' ',' Diese Angabe entscheidet über das Dezimalzeichen *Punkt* oder *Komma*, mit dem Nachkommastellen in die externen Felder vom Typ D, DK und DS ausgegeben werden.

LB Hiermit werden die Zahlen in D-Feldern mit führenden Leerzeichen ausgegeben.

Diese Angabe wirkt auf alle auszugebenden D-Felder in den Folgestatements von OUTPUT-EXT und auf die Angaben ID=(...) in OUTPUT-EXT selbst.

Die Satz-Kennzeichen RC=(...) werden dagegen stets mit führenden Nullen ausgegeben.

LZ Diese Angabe gibt die Zahlen in D-Feldern mit führenden Nullen aus.

ND Mit dieser Angabe werden alle numerischen Werte ohne Nachkommastellen ausgegeben und dafür gegebenenfalls gerundet. In den einzelnen Wertezuweisungen kann diese Regel ausgeschaltet werden, indem im Ausgabefeld die Anzahl gewünschter Nachkommastellen vorgegeben wird.

Zum Beispiel stellt die Wertezuweisung

```
-D1 (10, 5) 2=N1
```

auch dann zwei Nachkommastellen in das D-Feld, wenn OUTPUT-EXT mit DSTD=ND arbeitet.

29. 3. 2005

Schlüssel für externe Datenfelder AK DK MK und UK in eigenen Datensätzen

In CSV-Dateien mussten die Schlüssel für die AK-, DK-, MK- und UK-Felder bisher direkt vor den Datenwerten und einem Gleichheitszeichen stehen. Zum Beispiel in der Form:

```
ID=0001;alter=34; ...
ID=0002;c2=5 ;alter=25; ...
```

Jetzt ist es auch möglich, die Schlüssel in eigenen Datensätzen am Anfang der Datei anzugeben, in der gleichen Spalte wie die dazugehörigen Datenwerte. Eine solche Datei könnte folgendermaßen beginnen:

```
ID; c2; alter; ...
0001;   ;    34; ...
0002;  5;    25; ...
```

Dazu genügt im Statement INPUT-EXT oder FETCH-FILE die Angabe
KEYLINE

Die AK-, DK-, MK- und UK-Felder ändern sich dafür nicht.

10. 3. 2005

Neue Statements für Prototyp-Dateien

Zur Bereinigung der Prototyp-Syntax sollten folgende Änderungen beachtet werden:

- w Das Statement GROUP= wird durch @GROUP ersetzt.
- w Das Statement CENTRAL wird durch @GROUP CENTRAL ersetzt.
- w Das Statement WINDOW= wird durch @WINDOW ersetzt.
- w Das Statement PTT-TITLE = wird durch @PNAME ersetzt.
- w Das Statement II-FNAME wird durch FNAME V ersetzt.
- w Das Statement MESSAGE wird durch @MESSAGE ersetzt.
- w Das Statement PTT-TITLE wird durch @GROUP CENTRAL ersetzt.

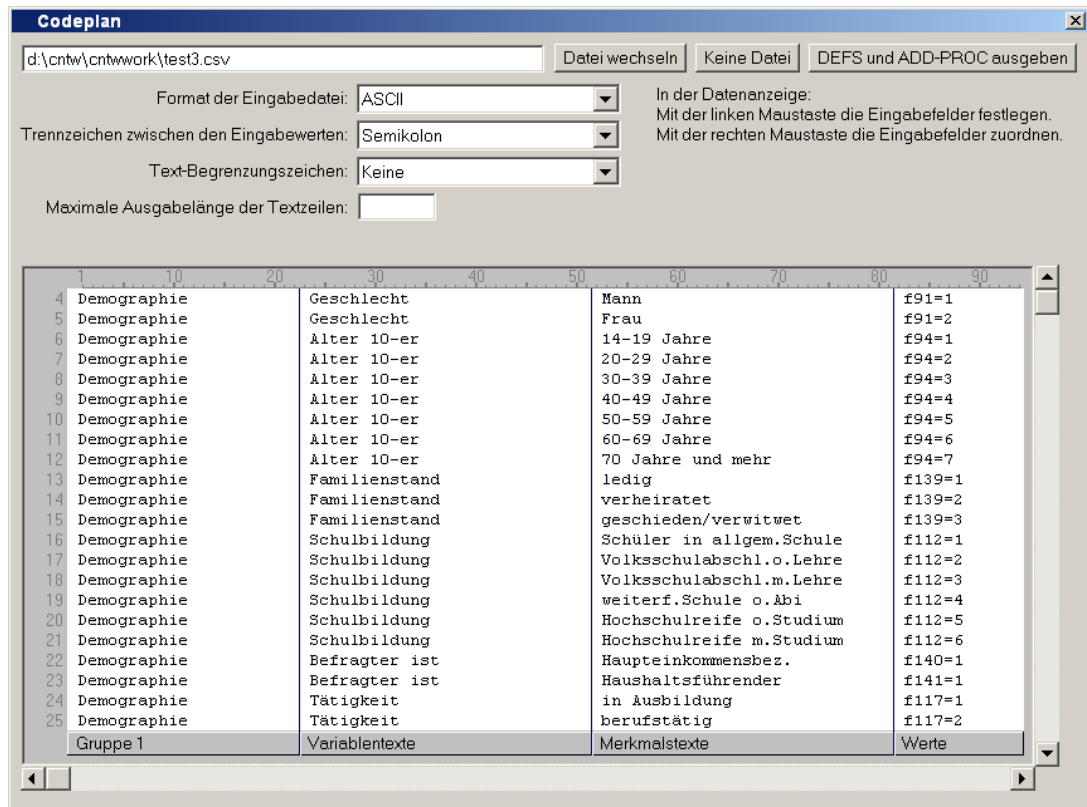
Die alten Schreibweisen werden von CNTW weiter akzeptiert, sollten aber nicht mehr verwendet werden.

Das neue Statement

```
@vname FNAME 'Text' C <= @name > < DEF = Dateinamen > < NONAME > < FIRST >
```

erzeugt Bildschirmfenster zur Anzeige und Untersuchung von Codeplan-Dateien im CSV-Format. Insbesondere kann dieses Fenster aus den Codeplan-Daten automatisch Variablendefinitionen DEFS sowie Wertezuweisungen ADD-PROC erzeugen und in CNTA-Statement-Dateien stellen.

Das Fenster eines FNAME-C-Statements könnte folgendermaßen aussehen:



In der oberen Zeile dieses Fensters wird die gewünschte Codeplan-Datei ausgewählt. Der untere Teil des Fensters zeigt die Datensätze dieser Datei an.

In den Datenfeldern oberhalb dieser Anzeige können die verschiedenen Eigenschaften dieser Datei eingestellt werden. Zum Beispiel das Dateiformat ASCII oder ANSI, das Trennzeichen für CSV-Dateien, Textbegrenzungszeichen usw.

Nach jeder Änderung in diesen Feldern wird die Anzeige der Datensätze entsprechend angepaßt. Mit Hilfe von Mausfunktionen wird die Textanzeige in Spalten untergliedert. Danach lässt sich festlegen, wie die Daten der einzelnen Spalten zur Erstellung von Variablendefinitionen und Wertezuweisungen zu verwenden sind.

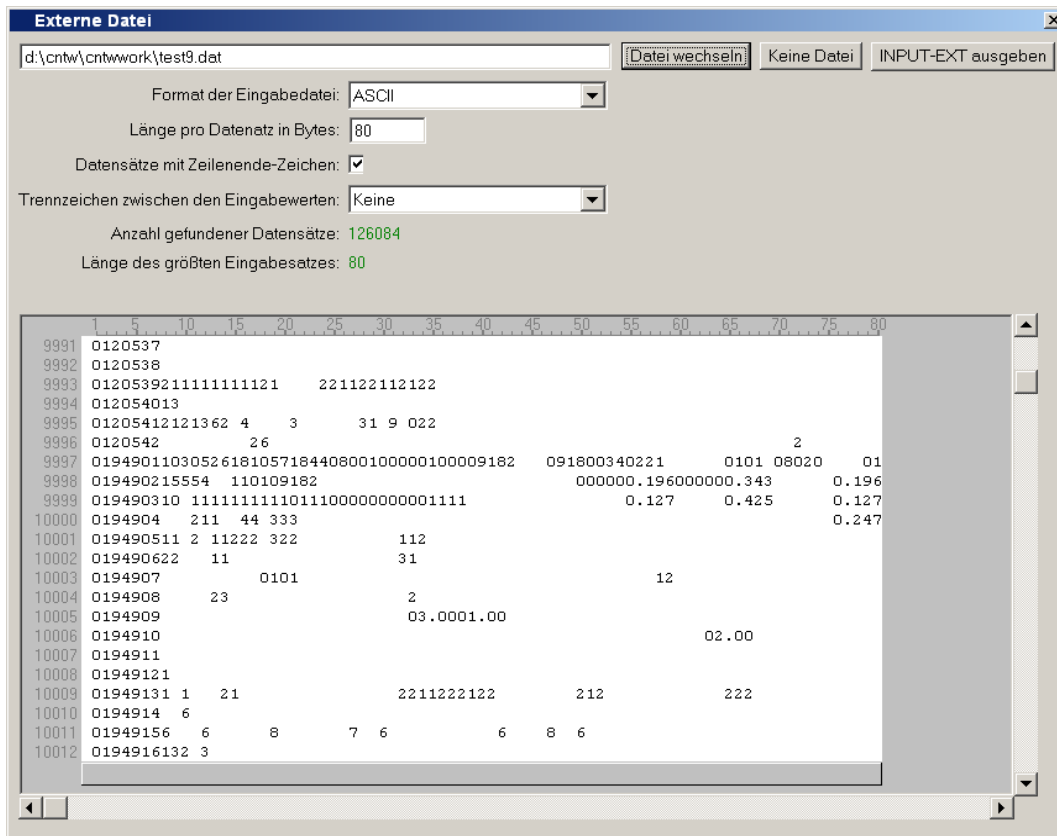
Mit der Schaltfläche *DEFS und ADD-PROC ausgeben* werden aus diesen Daten automatisch Statements in eine Statement-Datei gestellt. Insbesondere kann CNTW für Wertezuweisungen auch einige Feldbeschreibungen außerhalb der CNT-Syntax korrekt interpretieren.

Zur Bestimmung der Statementdatei ist hinter C eine @-Variable anzugeben, die auf ein FNAME-S-Statement zeigt. In die dort ausgewählte Datei werden die DEFS- und ADD-PROC-Statements gestellt.

Das neue Statement

`@vname FNAME 'Text' E <= @sname > < DEF = Dateinamen > < NONAME > < FIRST >`

erzeugt Bildschirmfenster zur Anzeige und Untersuchung externer Dateien. Ein FNAME-E-Fenster könnte folgendermaßen aussehen:



In der oberen Zeile dieses Fensters wird die gewünschte externe Datei ausgewählt. Der untere Teil des Fensters zeigt die Datensätze dieser Datei an.

In den Datenfeldern oberhalb dieser Anzeige können die verschiedenen Eigenschaften einer solchen externen Datei eingestellt werden. Zum Beispiel das Dateiformat ASCII, ANSI, COLBIN usw., die Satzlänge, End-Of-File-Zeichen, Trennzeichen der CSV-Dateien usw.

Nach jeder Änderung in diesen Feldern wird die Anzeige der Datensätze entsprechend angepasst. Auf diese Weise lassen sich die Eigenschaften von externen Dateien mit unbekanntem Aufbau meist schnell und sicher erkennen.

Außerdem kann dieses Fenster zu den angezeigten Daten das passende INPUT-EXT-Statement erzeugen und in eine CNTA-Statement-Datei stellen.

Dazu ist hinter E eine @-Variable anzugeben, die auf ein FNAME-S-Statement zeigt.

In die damit ausgewählte Datei wird das Statement INPUT-EXT geschrieben.

In der mit CNTW ausgelieferten Prototyp-Datei CNTA.PTT werden diese beiden neuen Statements bereits verwendet.

14. 2. 2005

Neue Voreinstellungen für externe F- G- I- und M-Felder

Bisher war für F- und G-Felder eine Längenangabe zwingend erforderlich.
Fehlt diese, so wird jetzt mit der Länge 1 gearbeitet. F1(10) ist daher gleichwertig mit F1(10,1).

Wurden bei externen Datenfeldern vom Typ I keine Längen angegeben, so wurde der Wert 2 für die Längen angenommen. I1(5) war gleichwertig mit I1(5,2,2).

Jetzt wird für fehlende Längen 1 angenommen; I1(5) ist daher gleichwertig mit I1(5,1,1).

Darüber hinaus wird die erste Längenangabe als Voreinstellung für die zweite verwendet.

Daher ist I1(5,2) gleichwertig mit I1(5,2,2).

Schließlich kann die Reihenfolge der ersten und zweiten Längenangabe vertauscht werden:

I1(5,2,4) ist gleichwertig mit I1(5,4,2).

Ganz analog dazu wurden die Voreinstellungen für die M-Felder erweitert.

9. 2. 2005

CODEBOOK-Ausgabe an EXCEL

CNTW kann jetzt auch CODEBOOK-Auswertungen aus der *Anzeige der Ergebnisse* direkt an EXCEL übergeben.

22. 01. 2005

Neue Dateiformate BINARY1 und BINARY2

In den Statements INPUT-EXT, OUTPUT-EXT und FETCH-FILE sind jetzt neben den bisherigen Angaben ANSI, ASCII, COLBIN ... die beiden folgenden Formate möglich:

BINARY1 Diese Dateien bestehen aus ein bis vier Bytes langen Dualzahlen.
Positionen und Längen in den externen Datensätzen sind in Bytes anzugeben.
Es sind die externen Datenfelder vom Typ B, I, F und G möglich.

BINARY2 Diese Dateien bestehen aus zwei und vier Bytes langen Dualzahlen.
Positionen und Längen in den externen Datensätzen sind in Feldern anzugeben, jedes Feld zu zwei Bytes. Es sind die externen Datenfelder vom Typ B, I, F und G möglich.

15. 1. 2005

Auswahl einzelner Merkmalsnummern in MK- und MS-Feldern

Bei externen Eingabefeldern im MK- und MS-Feldern ist es jetzt möglich, aus Mehrfachnennungen einzelne Merkmale herauszugreifen.

Enthält der Datensatz zum Beispiel die Werte

```
frage2=1 3 5-9
```

so liest die Wertezuweisung

```
-C1=MK1('frage2') alle Merkmalsnummer 1, 3 und 5-9 in die Variable C1 ein.
```

Die Wertezuweisung

```
-C1=MK1('frage2'(4)) dagegen stellt nur den vierten Eingabewert, also das Merkmal 6 in die Variable C1.
```

Enthält der Datensatz die Werte

```
; ; 1 3 5-9
```

so liest die Wertezuweisung

```
-C1=MS1(3(4))
```

die Merkmalsnummer 6 in die Variable C1 ein.

8. 12. 2004

Verbesserte Behandlung von Nachkommastellen in externen Datenfeldern

Bisher wurden bei der Ausgabe von D- DK- und DS-Feldern die Nachkommastellen entfernt. Jetzt werden sie mit Dezimalzeichen ausgegeben.

Im Statement OUTPUT-EXT entscheidet die erweiterte Angabe DSTD, ob die D-Felder mit führenden Nullen oder führenden Leerzeichen ausgegeben und wie Nachkommastellen aufbereitet werden.

28. 8. 2004

Erweiterung der Nachkommastellen von N-Variablen

Bei der Definition von N-Variablen darf die Anzahl von Nachkommastellen genau so groß werden, wie die Zahl der Dezimalstellen. Folgende Definitionen sind jetzt gültig:

```
DEFS
```

```
-N1 : 9, 9
```

```
-N2 : 3, 3
```

Diese Angaben wurden bisher als fehlerhaft angesehen.

24. 8. 2004

Unterdrückung von Fehlermeldungen im Zählprotokoll

Im Statement `OPTIONS` kann die Ausgabe von Fehlermeldungen 500 ... im Zählprotokoll mit folgenden Angaben beeinflusst werden:

```
< MSG = | NO
          | < FETCH : | YES | > < , MISSING : | YES | > < , RANGE : | YES | > > |
          | NO          |      | NO          |      | NO          |      |
```

NO Mit `MSG=NO` werden die PRT-Meldungen, die TST-Meldungen und die Fehlermeldungen 500 ... 599 zu den eingelesenen statistischen Fällen unterdrückt; siehe PRT-Statement, TST-Statement und im Abschnitt *Fehlermeldungen*.

FETCH Durch `MSG=FETCH` wird die Ausgabe der Fehlermeldung 518 *Datensatz fehlt in FETCH-FILE* im Zählprotokoll gesteuert. `FETCH:YES` sorgt für die Ausgabe der Meldung und `FETCH:NO` verhindert sie. Fehlt die Angabe `FETCH:NO` so wird die Fehlermeldung 518 ausgegeben.

MISSING Durch `MSG=MISSING` wird die Ausgabe der Fehlermeldung 525 *Schlüssel nicht im Datensatz* im Zählprotokoll gesteuert. Diese Meldung erscheint, wenn ein Datensatz zu einem Datenfeld vom Typ AK, DK, MK oder UK keinen passenden Schlüssel enthält. `MISSING:YES` sorgt für die Ausgabe der Meldung und `MISSING:NO` verhindert sie. Fehlt die Angabe `MISSING:YES` so wird die Fehlermeldung 525 nicht ausgegeben.

RANGE Durch `MSG=RANGE` wird die Ausgabe der Fehlermeldung 524 *Merkmal passt nicht zum Zielfeld* im Zählprotokoll gesteuert. `RANGE:YES` sorgt für die Ausgabe der Meldung und `RANGE:NO` verhindert sie. Fehlt die Angabe `RANGE:YES` so wird die Fehlermeldung 524 nicht ausgegeben.

Wenn die Fehlermeldungen 500 ... durch `MSG`-Angaben unterdrückt werden, enthält der Operand `ERR` trotzdem die entsprechenden Fehlernummern. Auch die Fehlerstatistiken `ESUM` und `ESUMS` werden unverändert ausgegeben.

Die Angabe `NOPRT` im `OPTION`-Statement entfällt. Sie wird durch `MSG=NO` ersetzt.

3. 7. 2004

Externe Datenfelder AK DK MK und UK mit Schlüssel

In CSV-Dateien ist es jetzt möglich, Datenwerte zu verarbeiten, die an beliebiger Stelle im Datensatz stehen und mit einem davor stehenden Schlüssel gekennzeichnet sind. Zwischen Schlüssel und Datenwert ist ein Gleichheitszeichen erforderlich.

Ein Datensatz mit diesen neuen Datenfeldern und dem Semikolon als Separatorzeichen könnte folgendermaßen aussehen:

```
ID=0001;RC=1;frage4=1 3-5 10;frage5=100101;alter=55;name=Schmitt
```

Mit dem Statement

```
INPUT-EXT ID=DK('ID') RC=DK('RC') SEP=';' ...
```

wird dieser Datensatz eingelesen.

Durch die Wertezuweisung

```
-C1=MK1('FRAGE4')
```

werden in der Variablen C1 die Merkmale 1, 3, 4, 5 und 10 gesetzt.

Mit

```
-C2=UK1('FRAGE5')
```

werden in der Variablen C2 die Merkmale 1, 4 und 6 gesetzt.

Die Wertezuweisung

```
-N1=DK1('alter')
```

stellt den Zahlenwert 55 in die Variable N1.

Durch

```
-T1=AK1('name')
```

wird der Textwert *Schmitt* in die Variable T1 gestellt.

Die Reihenfolge der Angaben im Datensatz ist beliebig, da die Werte mit Hilfe des davorstehenden Schlüssels gefunden werden. Bei den Schlüsseln wird nicht zwischen Groß- und Kleinschrift unterschieden.

Zusätzliche Leerzeichen werden ignoriert.

So liefert der folgende Datensatz die gleichen Ergebnisse wie der obige:

```
alter = 55 ; name = Schmitt ; RC = 1 ; ID = 0001 ; frage4 =1 3 -5 10 ; frage5 = 100101
```

In den Wertezuweisungen dieser neuen Felder kann der Schlüssel entfallen, wenn auf der anderen Seite des Gleichheitszeichens eine Variable steht:

Bei T1=AK1() wird 'T1' zum Schlüssel.

Entsprechend wird in Ein- und Ausgabestements für ID=DK() der Schlüssel 'ID' verwendet, für

KEY=DK() der Schlüssel 'KEY' und für RC=DK() der Schlüssel 'RC'.

3. 7. 2004

Erweiterung der MS-Felder

Mehrfachnennungen können aus MS-Feldern eingelesen werden, wenn sie direkt hintereinander stehen und durch das Separatorzeichen SEP voneinander getrennt sind.

Zum Beispiel setzt die Wertezuweisung
- C1=MS1(1, 4)

aus dem Datensatz

07; 23; ; 9; ...

die Merkmale 7, 9 und 23 der Variablen C1, wenn SEP=' ; ' das Separatorzeichen ist.

Dabei legt die Angabe 4 in MS1(1, 4) fest, dass maximal 4 Mehrfachnennungen einzulesen sind.

Jetzt lassen sich Mehrfachnennungen auch zwischen zwei Separatorzeichen stellen, durch Leerzeichen voneinander getrennt. Dabei sind auch Abkürzungen möglich, mit Von-Bis-Werten und Bindestrich dazwischen.

Enthält der Datensatz zum Beispiel die Werte

; 3-6 1 11;

so setzt die Wertezuweisung

- C1=MS1(2)

die Merkmale 1, 3, 4, 5, 6 und 11 der Variablen C1.

Dieses Datenformat wird durch die fehlende Längenangabe in MS1(2) ausgewählt.

3. 7. 2004

Erweiterung der US-Felder

Mehrfachnennungen können aus US-Feldern eingelesen werden, wenn sie direkt hintereinander stehen und durch das Separatorzeichen SEP voneinander getrennt sind.

Zum Beispiel setzt die Wertezuweisung
- C1=US1(1, 4)

aus dem Datensatz

1; 0; ; 1; ...

die Merkmale 1 und 4 der Variablen C1, wenn SEP=' ; ' das Separatorzeichen ist.

Dabei legt die Angabe 4 in US1(1, 4) fest, dass maximal 4 Mehrfachnennungen einzulesen sind.

Jetzt lassen sich Mehrfachnennungen auch zwischen zwei Separatorzeichen stellen.

Enthält der Datensatz zum Beispiel die Werte

; 10 1;

so setzt die Wertezuweisung

- C1=US1(2)

die Merkmale 1 und 4 der Variablen C1.

Dieses Datenformat wird durch die fehlende Längenangabe in US1(2) ausgewählt.

17. 6. 2004

Anzeige von Makrowerten im Zählprotokoll

Es ist jetzt möglich, im Zählprotokoll in den Statements an jeder beliebigen Stelle die Werte von Makros anzuzeigen. Das geschieht mit Hilfe des DUMP-Statements.

So könnte im Zählprotokoll direkt hinter dem Statement

```
DUMP MACRO=#rows
```

der Wert des Makros #rows in folgender Form erscheinen:

```
#ROWS: 'ROWS=TOTAL;C117;C128;C195'
```

7. 6. 2004

Operand ERR zur Abfrage von Fehlerbedingungen

In den Wertezuweisungen von ADD-PROC, UPD-PROC und OUTPUT-EXT kann mit Hilfe des neuen Operanden ERR abgefragt werden, ob die Eingabe aus einem externen Datenfeld fehlerfrei abgelaufen ist.

Der Operand ERR wird zu Beginn jeder Wertezuweisung auf 0 gesetzt. Läuft die Wertezuweisung fehlerfrei ab, so behält ERR diesen Wert. Wird hingegen ein Fehler gefunden, so enthält ERR anschließend die Nummer 500 ... 599 der entsprechenden Fehlermeldung.

Mit den Statements

```
- N1=D1( 10, 3)  
- I F ERR > 0  
- N1=0  
- E I F
```

erhält die Variable N1 den Wert 0, wenn das Feld D1(10,3) einen fehlerhaften Wert besitzt. Ohne das IF-Statement hätte N1 im Fehlerfall den Wert Z0.

Mit den Statements

```
- N1=D1( 10, 3)  
- N2=ERR
```

erhält die Variable N2 den Wert 0, wenn das Feld D1(10,3) einen korrekten Wert besitzt, andernfalls die Nummer 500 ... 599 der entsprechenden Fehlermeldung.

Mit dem Statement

```
- PRT ERR
```

lässt sich der Wert des Operanden ERR im Zählprotokoll ausdrucken.

15. 3. 2004

Erweiterung der Zeilen- und Spaltenangaben zu IROWCOL in XTAB

Es ist jetzt möglich, die Zeilen- und Spaltennummern für IROWCOL auch nach SUPPRESS und SORT anzugeben. Zum Beispiel bildet

```
PRI NT=I ROWS1COLS- 1
```

Indexwerte mit der ersten Zeile und der letzten Spalte als Basis, jeweils nach SUPPRESS und SORT.

Die Basiszeilen und -spalten lassen sich auch über Klassenbuchstaben A{ } ... Z{ } festlegen.

Mit dem Statement

```
XTAB ROW=A{ C1. Z123} ; B{ C1} ; A{ C2. Z123} ; B{ C2}
      COL=TOTAL ; C3
      PRI NT=R( B) PCOLA
```

werden die Werte aus den Zeilen B{C1} auf die erste davor liegende Spalte A{C1.Z123} prozentuiert und die Werte der Spalten B{C2} auf die erste davor liegende Spalte A{C2.Z123}.

8. 3. 2004

Dateinamen wahlweise mit oder ohne Hochkommas

Dateinamen hinter FNAME in den verschiedenen Ein- und Ausgabestatements können jetzt auch ohne Hochkommas angegeben werden. Das erste Leerzeichen beendet dabei den Dateinamen.

Enthält ein Dateiname selber Leerzeichen, so ist er weiterhin in Hochkommas einzuschließen.

30. 12. 2003

Erweiterung der Variablen und Merkmale

Die maximale Anzahl von Merkmalen der C-Variablen wurde von 255 auf 4 000 erhöht.

Die maximale Variablennummern für C-, N- und T-Variable wurden von 9999 auf 99999 erweitert. Entsprechend sind Textelemente bis zu (99999) möglich.

Die maximale Anzahl von Indices wurde für C-, N- und T-Variable von [9999] auf [99999] erhöht.

22. 12. 2003

Unterdrückung des Nullzeichens bei Signifikanzvergleichen

Für die t-Tests mit MTT und MTU im Statement XTAB werden die Buchstaben a ... z für signifikante Unterschiede in die Tabellen gestellt.

Wenn keine signifikanten Unterschiede bestehen, wird das Nullzeichen NCHAR aus dem PAGE-Statement ausgegeben.

Es ist jetzt möglich, die Ausgabe des Nullzeichens zu unterdrücken.

Dazu ist im Parameter PRINT hinter SIG die Angabe ZB erforderlich. Zum Beispiel:

```
XTAB  CCL=TOTAL; C500
      ROW=TOTAL; MTT{ MEAN( C520) }
      PRINT1=SIG, 5, ZB
```

17. 11. 2003

FETCH-FILE-Dateien ohne KEY

Bisher verlangte das Statement FETCH-FILE stets die Angabe KEY= ...

Jetzt kann auf KEY verzichtet werden. Dann arbeitet FETCH-FILE analog zu INPUT-EXT ohne ID:

Die aus der FETCH-FILE-Datei eingelesenen Fälle werden - beginnend bei 1 - automatisch durchnummeriert. Diese Nummer ersetzt dann den KEY.

Damit wird es möglich, mehrere externe Dateien in einem Verarbeitungslauf zusammenzuführen, die keine ID- und KEY-Angaben besitzen, wohl aber Daten zu den einzelnen Fällen in gleicher Anzahl und Reihenfolge enthalten.

Zum Beispiel:

```
INPUT-EXT  FNAME='datei1.ext'  SIZE=80,EOL
FETCH-FILE FNAME='datei2.ext'  SIZE=80,EOL
DEFS
-C1..2:20
ADD-PROC
-C1=M1(1,2)
-C2=FETCH(ID,M1(1,2))
```

Mit den Statements INPUT-EXT und FETCH-FILE werden zwei externe Dateien angefordert.

Da INPUT-EXT keine ID-Angabe besitzt, werden die eingelesenen Fälle mit 1 ... durchnummeriert.

Diese Nummern werden als ID verwendet.

Da FETCH-FILE keine KEY-Angabe enthält, werden auch diese Fälle mit 1 ... durchnummeriert.

Diese Nummern werden entsprechend als KEY verwendet.

Mit dem Statement

```
-C1=M1(1,2)
```

erhält die Variable C1 Werte aus der INPUT-EXT-Datei und mit

```
-C2=FETCH(ID,M1(1,2))
```

wird die Variable C2 mit Werten aus der FETCH-FILE-Datei versehen.

Die Angabe ID hinter FETCH erhält ihre Werte aus der Numerierung von INPUT-EXT.

Sie sorgt dann dafür, dass dem ersten Datensatz aus INPUT-EXT der erste Datensatz aus FETCH-FILE zugewiesen wird, dem zweiten der zweite und so weiter für alle folgenden Fälle.

17. 11. 2003

Größere Anzahl von FETCH-FILE-Dateien

Die maximale Anzahl von FETCH-FILE-Dateien eines Verarbeitungslaufs wurde von bisher 10 auf 30 erhöht. Es sind also jetzt gleichzeitig die Angaben FETCH-FILE, FETCH-FILE1 ... FETCH-FILE29 möglich.

10. 11. 2003

Neue Schreibweise der Statements in Prototyp-Dateien

Zur Vereinheitlichung der Schreibweise sind die Prototyp-Statements

GROUP, MESSAGE, K-VARIABLE, PINCLUDE, WINDOW

zukünftig mit @ davor zu schreiben:

@GROUP, @MESSAGE, @K-VARIABLE, @PINCLUDE, @WINDOW

Die Gleichheitszeichen hinter diesen Schlüsselworten entfallen.

Das Statement

PTT-TITLE

wurde umbenannt in:

@PNAME

Die alten Schreibweisen werden weiterhin akzeptiert, sollten aber zukünftig nicht mehr verwendet werden.

10. 11. 2003

Erweiterung des Statements @PINCLUDE

Bisher verlangte dieses Statement einen festen Dateinamen, zum Beispiel

@PINCLUDE KUNDEN.INC

Jetzt ist es auch möglich, anstelle des Dateinamens eine @-Variable aus einem FNAME- oder II-FNAME-Statement zu verwenden. Zum Beispiel

@PINCLUDE @KUNDEN

Sobald diese @-Variable durch Bildschirmeingaben einen neuen Wert erhält, wird die PTT-Datei neu eingelesen. @PINCLUDE liest dabei die neue Include-Datei ein.

Eine solche @-Variable muß, vor dem Statement @PINCLUDE definiert sein. Außerdem darf diese Definition nicht in einer Section liegen.

26. 9. 2003

Neue Grundeinstellung für TEXT in XTAB

Fehlt in den XTAB-Statements die Angabe TEXT, so wurde bisher mit TEXT=KEEP gearbeitet: Zu lange Texte wurden abgeschnitten oder über die Ränder hinweg gedruckt.

Diese Grundeinstellung wurde jetzt auf TEXT=FIT geändert: Zu lange Texte werden aufgebrochen und in die verfügbaren Spaltenbreiten eingepasst.

9. 8. 2003

Prozentuierung auf Zeilen- und Spaltennummern nach SUPPRESS und SORT

Bisher war es nur möglich, mit PROW oder PCOL auf Zeilen- und Spaltennummern vor eventueller Sortierung durch SORT oder Unterdrückung durch SUPPRESS zu prozentuieren.

Jetzt ist es möglich, die Basiszeilen oder Basisspalten nach SUPPRESS oder SORT anzugeben.

Das geschieht in der Form:

PRINT = ... PROWS_z ... PCOLS_s

wobei *z* die Zeilennummer und *s* die Spaltennummer nach Sortierung und Unterdrückung ist.

Wie bisher sind die Zeilen- und Spaltennummern 1 ... 32 767 für die Zählung vom Anfang der Tabelle und

-1 ... -32 767 für die Zählung vom Ende der Tabelle zulässig.

Das Statement

```
XTAB SUPPRESS=FROW SORT=ROWS1
      PRINT=PROWS1 . . .
```

entfernt wegen SUPPRESS=FROW zunächst die erste Tabellenzeile und sortiert danach wegen SORT=ROWS1 die Zeilen nach den Werten der ersten Spalte.

PRINT=PROWS1 prozentuiert nun alle Zähler der Tabelle auf die Werte der neu entstandenen ersten Zeile.

9. 8. 2003

Prozentuierung auf Zeilen und Spalten mit Klassenmarkierung A{ } ... { }

Es ist jetzt möglich, die Prozentuierungsbasen nicht nur als Zeilen- oder Spaltennummern anzugeben, sondern auch über Klassenbuchstaben A{ } ... Z{ }.

Mit dem Statement

```
XTAB ROW=A{ C1. Z123} ; B{ C1} ; A{ C2. Z123} ; B{ C2}
      PRINT=R( B) PROWA
```

werden die Werte aus den Zeilen B{C1} auf die erste davor liegende Zeile A{C1.Z123} prozentuiert und die Werte der Zeilen B{C2} auf die erste davor liegende Zeile A{C2.Z123}.

Durch PROWA wird von der aktuellen Zeile nach oben in der Tabelle die erste Zeile mit der Klasse A{ } gesucht und als Prozentuierungsbasis verwendet. Mit der Angabe PROW-A wird nach unten in der Tabelle die erste Zeile mit der Klasse A{ } gesucht und als Basis benutzt.

Zur Spaltenprozentuierung gelten ganz analoge Regeln mit den Angaben PCOLA ... PCOLZ.

13. 7. 2003

CHI2: Zweidimensionaler Chi-Quadrat-Test

Dieser χ^2 -Test vergleicht Zeilen- und Spaltenbedingungen einer Tabelle oder Teiltabelle miteinander. Die beteiligten Tabellenzähler werden in der Ausgabe mit frei wählbaren Zeichen markiert. Als Ergebnis wird der χ^2 -Wert und der dazugehörige Signifikanzwert in Prozent ausgegeben und zwar hinter einem Fußnotentext unterhalb der Tabelle.

Der Signifikanzwert gibt den kleinsten Fehler an, bei dem die Hypothese *die Zeilen- und Spaltenbedingungen sind statistisch unabhängig* abzulehnen ist: Kleine Werte weisen auf signifikant abhängige Bedingungen hin, große lassen auf unabhängige Bedingungen schließen.

Dieser Chi-Quadrat-Test verlangt lediglich eine Eintragung hinter PRINT ... PRINT3 im XTAB-Statement:

PRINTi = R(...) C(...) CHI2 <,'t'> <,Fi> <,FTi> <,Gi>

Mit den Angaben R(...) und C(...) sind die Zeilen und Spalten auszuwählen, die an dem Test teilnehmen sollen. Fehlen solche Angaben, so werden alle Zeilen- oder Spaltenbedingungen untersucht.

Alle am Test beteiligten Tabellenzellen werden mit den Zeichen aus der Druckmaske 't' gekennzeichnet. Fehlt eine solche Druckmaske, so wird dafür das Zeichen x verwendet.

Der ermittelte Chi-Quadrat-Wert und der dazugehörige Signifikanzwert werden hinter der Fußnote Fi in der Form *Chi-Quadrat = x.xx Signifikanz = y.yy%* ausgegeben. Ist kein Fußnotentext Fi angegeben, so erscheint dafür der Text:

x) *Chi-Quadrat-Test dieser Tabellenwerte: Chi-Quadrat = x.xx Signifikanz = y.yy%*

Beispiel

```
XTAB  ROW=TOTAL:ABS;C1
      COL=TOTAL:ABS;C2
      PRINT=R(2..-1)C(2..-1)PROW1,1,%
      PRINT1=R(2..-1)C(2..-1)CHI2
```

Dieses Statement könnte folgende Tabelle liefern:

Die Angabe PRINT1 sorgt für einen Chi-Quadrat-Test der Werte der Zeilen 2 ... 6 und Spalten 2 ... 6. Die daran teilnehmenden Tabellenzellen werden in der Position PRINT1 mit dem Zeichen x markiert.

Das Ergebnis dieses Tests wird unterhalb der Tabelle als Fußnote ausgewiesen.

9. 3. 2003

Mehrsprachige Variablentexte

CNTA kann jetzt einer Variablen oder einem Textelement gleichzeitig mehrere Texte zuweisen, zum Beispiel in verschiedenen Sprachen oder als Kurz- und Langtexte.

Dazu ist mit den Statements LANG, LANG1 ... LANG9 die Sprache auszuwählen, die für die dahinter stehenden Statements gelten soll. Diese Sprachschlüssel LANG ... LANG9 können beliebig oft und an jeder Stelle eines Programms erscheinen. Es bleibt dem Anwender überlassen, welche Bedeutung er den einzelnen Sprachschlüsseln zuordnen will.

So weisen die Statements

```

DEFS
- LANG
- C1: 2 ' Geschl echt ' 1=' männl i ch' 2=' wei bl i ch'
- N1: 2 ' Al t er '
- TOTAL ' Gesamt '
- LANG1
- C1 ' sex' 1=' mal e' 2=' f ermal e'
- N1 ' age'
- TOTAL ' t ot al '
    
```

den Variablen C1, N1 und dem Operator TOTAL unter dem Sprachschlüssel LANG deutsche Texte und unter dem Sprachschlüssel LANG1 englische Texte zu. Da die Statements LANG und LANG1 innerhalb der Folgestatements von DEFS erscheinen, müssen sie mit einem waagerechten Strich beginnen.

In den Auswertungen können mit den Statements LANG ... LANG9 Texte aus unterschiedlichen Sprachen angefordert werden. Nach der obigen Variablendefinition liefern die Statements

```

LANG
XTAB ROW=C1 CCL=TOTAL
LANG1
XTAB ROW=C1 CCL=TOTAL
    
```

zunächst eine Tabelle mit deutschen Texten und dahinter die gleiche Tabelle mit englischen Texten.

Ist kein Statement LANG ... LANG9 angegeben, so wird mit dem Sprachschlüssel LANG gearbeitet.

9. 3. 2003

Signifikanzmarkierungen in Kreuztabellen alphabetisch sortieren

Bisher wurden die Markierungszeichen a ... z in den Kreuztabellen stets nach ihren Signifikanzwerten sortiert. Jetzt ist es möglich, diese Markierungszeichen auch alphabetisch sortiert auszugeben.

Dazu ist im Parameter PRINT von XTAB hinter SIG die Angabe A erforderlich. Zum Beispiel

```

XTAB PRINT=SIG, A ...
    
```

14. 02. 2003

I-Format in COLBIN-Dateien

In externen Dateien sind jetzt auch Datenfelder im I-Format möglich.
Dies gilt sowohl für die Eingabe über INPUT-EXT als auch für die Ausgabe auf OUTPUT-EXT.

14. 02. 2003

K-Format für OUTPUT-EXT mit AUTOCOLBIN

Bei der Ausgabe von C-Variablen im K-Format wurden bei OUTPUT-EXT mit AUTOCOLBIN oder AUTOQUANTUM bisher auch die Lochpositionen X und Y belegt.
Jetzt werden nur noch die Positionen 1 ... 10 verwendet.

11. 02. 2003

Numerische Werte im F-Format für OUTPUT-EXT mit AUTOCOLBIN

Es ist jetzt möglich, die Werte von N-Variablen auch im F- oder G-Format in externe Dateien mit AUTOCOLBIN oder AUTOQUANTUM auszugeben.

Dies kann zum Beispiel mit der Angabe

`NSTD=F`

im Statement OUTPUT-EXT festgelegt werden oder auch über das Folgestatement

`-N10 F`

von OUTPUT-EXT.

12. 11. 2002

Neues Dateiformat QUANTUM in OUTPUT-EXT

Das Statement OUTPUT-EXT ist jetzt auch in der Lage, mit der Angabe QUANTUM anstelle der Datenformate ASCII, COLBIN ... Quantum-Dateien zu erzeugen.

Zusätzlich lässt sich mit der Angabe AUTOQUANTUM eine Quantum-Datei erzeugen, bei der der Spaltenplan automatisch von CNTA erstellt wird, ganz analog zu AUTOASCII, AUTOCOLBIN usw.

12. 11. 2002

Vereinfachungen bei COPY in OUTPUT-EXT

Beim Kopieren externer Dateien mit der Angabe COPY in OUTPUT-EXT musste die Eingabedatei aus INPUT-EXT bisher das gleiche Datenformat (ASCII, COLBIN ...) besitzen wie die Ausgabedatei. Dies ist jetzt nicht mehr erforderlich. Zum Beispiel kopieren die Statements:

```
INPUT-EXT FNAME='test1.ext' COLBIN SIZE=80
OUTPUT-EXT FNAME='test2.ext' ASCII SIZE=80,EOL COPY
```

die Eingabedatei in die Ausgabedatei und setzen gleichzeitig das Datenformat COLBIN in das ASCII-Format um.

24. 7. 2002

Ausgabe von PDF-Dateien aus CNTW

Die Auswertungsergebnisse von CNTA konnten bisher schon als PDF-Dateien ausgegeben werden. Jetzt werden die PDF-Dateien automatisch mit Lesezeichen (Bookmarks) versehen, sofern die CNTA-Ausgabe aus dem Statement INDEX ein Inhaltsverzeichnis der Tabellen enthält.

30. 6. 2002

Neue Referenzpunkte für START im XTAB- und TEXT-Statement

Der Parameter START besaß bisher die folgenden Möglichkeiten zur relativen Positionierung:

RE Referenzpunkt ist das Ende der vorigen Ausgabe aus XTAB oder TITLE.

RM Referenzpunkt ist die bisher maximalen Ausgabe aus XTAB oder TITLE auf dem Blatt.

RS Referenzpunkt ist der Anfang der vorigen Ausgabe aus XTAB oder TITLE.

Neu hinzugekommen sind die folgenden Möglichkeiten:

RB Durch diese Angabe wird der Anfang der Zählerwerte der vorigen XTAB-Ausgabe zum Referenzpunkt. In der x-Richtung ist dies das rechte Ende der Spalte mit den Zeilentexten, in der y-Richtung das untere Ende der Spaltenüberschriften.

RT Durch diese Angabe wird der Tabellenanfang der der vorigen XTAB-Ausgabe zum Referenzpunkt. In der x-Richtung ist dies der linke Anfang der vorigen Tabelle, in der y-Richtung der obere Anfang der Spaltenüberschriften, hinter eventuellen TITLE- oder FILTER-Angaben.

Darüber hinaus sind für die Positionsangaben x und y jetzt auch negative Werte zulässig. Zum Beispiel:

```
START=RM 2, - 5
```

22. 5. 2002

Anzahl TAB-Angaben in XTAB vergrößert

Bisher erlaubte das Statement XTAB maximal 255 Angaben zum Parameter TAB.
Diese Grenze ist jetzt auf 32 767 angehoben.

5. 4. 2002

Makroausdrücke

Aus Makros lassen sich mit logischen und numerischen Operatoren komplexe Ausdrücke bilden. Diese werden in den unten beschriebenen *Wertezuweisungen mit Makroausdrücken* sowie in den Statements #IF und PROCESS verwendet.

Zum Beispiel stellt das Statement

```
#c #= #a + #b
```

die Zahl 5 in das Makro #c, sofern Makro #a den Wert 2 und Makro #b den Wert 3 enthält.

Dagegen liefert das Statement

```
#c #a + #b
```

die Zeichenfolge 2 + 3 im Makro #c.

Das Statement

```
#! F (#a + #b) > 10
```

sorgt dafür, dass die dahinter stehenden Statements nur dann eingelesen werden, wenn die Summe der Makrowerte aus #a und #b größer als 10 ist.

Makroausdrücke werden aus den folgenden Operanden zusammengesetzt:

#name Ein beliebiges Makro, zum Beispiel: #anz

k Numerische Konstante, zum Beispiel: 314. 26. Sie kann aus maximal 9 Ziffern bestehen.

Als Dezimalzeichen ist der Punkt zu verwenden, das Komma wird nicht akzeptiert.

Die Schreibweisen .9 und 9. sind unzulässig, 0.9 und 9.0 sind korrekt.

text Vergleichswert aus beliebigen Zeichen ohne Leerstellen dazwischen, zum Beispiel: abc1

'text' Vergleichswert aus beliebigen Zeichen auch mit Leerzeichen dazwischen, in Hochkommas eingeschlossen. Zum Beispiel: ' abc 1234'

Die folgenden Funktionen sind ebenfalls als Operanden möglich:

#def Für ein beliebiges Makro #anz liefert #def (#anz) den Wert 0, wenn #anz nicht definiert ist, also in den davor liegenden Statements noch keinen Wert erhalten hat.

Ist #anz dagegen definiert, so liefert #def (#anz) den Wert 1.

#num Für ein beliebiges Makro #anz liefert #num(#anz) den Wert 1, wenn #anz einen numerischen Wert besitzt. Dagegen liefert #num(#anz) den Wert 0, wenn #anz nicht numerisch oder nicht definiert ist, also in den davor liegenden Statements keinen Wert erhalten hat.

Diese Operanden lassen sich durch die folgenden Operatoren miteinander verrechnen:

	Beispiel
+ - positives und negatives Vorzeichen	- 314 oder +3. 14
+ Addition	#anz+75
- Subtraktion	#anz- 3
* Multiplikation	#anz*2
/ Division	#anz/ 5
** Potenzierung	#anz**2

Besitzt ein Makro hierbei einen nicht numerischen Wert, so wird dafür mit dem Wert 0 gerechnet.

Eine Division durch 0 führt zum Ergebnis 0.

Die Operanden können auch durch die folgenden Operatoren miteinander verglichen werden.

		Beispiel
=	gleich	#anz = #max
<	kleiner	#ort < ki el
>	größer	#name > ' abc 123'
<=	kleiner oder gleich	#anz <= 13. 4
>=	größer oder gleich	#anz >= 13. 4
^= oder ¬=	ungleich	#anz ^= 0
¬< oder ^<	nicht kleiner	#anz ^< 10
¬> oder ^>	nicht größer	#anz ^> 10

Das Ergebnis eines solchen Vergleichs ist die Zahl 1 für *wahr*, wenn die Vergleichsbedingung erfüllt ist oder die Zahl 0 für *falsch*, wenn sie nicht erfüllt ist.

Ist einer der zu vergleichenden Werte nicht numerisch, so werden die beiden Werte zeichenweise verglichen und zwar in der Reihenfolge des ASCII-Codes. Siehe dazu im Abschnitt *Code-Tabellen* die Reihenfolge der Zeichen in der *Zeichenzuordnung im ASCII-Code*.

So liefert der Makroausdruck

```
270<' 31'
```

den Wert 1, da das Zeichen 2 aus 270 in der ASCII-Tabelle vor dem Zeichen 3 aus '31' liegt.

Sind beide zu vergleichenden Werte numerisch, so werden die Zahlenwerte miteinander verglichen.

Der Makroausdruck

```
170<21
```

liefert den Wert 0, da die Zahl 170 größer als 21 ist.

Die Operanden lassen sich mit den folgenden logischen Operatoren zu komplexen Ausdrücken zusammenfügen:

		Beispiel
&	und	#anza & #anzb
oder !	oder	#anza #anzb
¬ oder ^	nicht	^#anz

Das Ergebnis einer solchen logischen Operation ist die Zahl 1, wenn die Bedingung erfüllt ist oder die Zahl 0, wenn sie nicht erfüllt ist.

Besitzt ein Makro einer solchen Operation den Wert 0 oder nur Leerzeichen, so wird dafür *falsch* oder 0 angenommen. Enthält das Makro eine Zahl ungleich 0 oder Textzeichen, so wird dafür *wahr* oder 1 verwendet.

Jeder Makroausdruck kann bis zu 255 Operatoren und Operanden besitzen. Er darf praktisch unbegrenzt viele Klammern (und) enthalten. Die Operatoren eines Ausdrucks werden in dieser Prioritätenfolge abgearbeitet:

hohe Priorität	()	Klammern um Teilausdrücke
	+ -	Vorzeichen Plus und Minus
	**	Potenzierung
	* /	Multiplikation und Division
	+ -	Addition und Subtraktion
	< > ...	Vergleichsoperatoren
	¬ ^	Negation
niedrige Priorität	&	und
	!	oder

Stehen mehrere Operatoren gleicher Priorität ohne Klammern hintereinander, wird von links nach rechts ausgewertet. So wird der Ausdruck

$\#a1+\#a2/10+(\#b-10)*\#id$

auf folgende Weise aufgelöst:

Zunächst wird der Quotient $\#a2/10$ ermittelt und zwischengespeichert. Danach wird, der Klammern wegen, die Differenz $\#b-10$ gebildet. Diese wird mit dem Wert aus $\#id$ multipliziert und das Ergebnis zwischengespeichert. Schließlich werden die gespeicherten Summanden zum Wert von $\#a1$ addiert.

5. 4. 2002

Wertezuweisung mit Makroausdrücken

Die oben beschriebenen Makroausdrücke erlauben eine weitere Form der Wertezuweisung.

Durch die Zeichen $\#=$ hinter dem Makronamen wird die dahinter stehende Zeichenfolge als Makroausdruck interpretiert. So stellen zum Beispiel die Statements:

```
#a 3
#b 5
#c #= (#a + #b) * 2
```

die Zahl 16 in das Makro $\#c$.

Die Statements

```
#a 3
#b 5
#c (#a + #b) * 2
```

stellen dagegen die Zeichenfolge $(3 + 5)*2$ in das Makro $\#c$.

Anstelle der Wertezuweisung $\#=$ sind auch die folgenden Angaben möglich:

```
#+ für Addition
#- für Subtraktion
#* für Multiplikation
#/ für Division
```

Diese Angaben verrechnen den Wert des Makroausdrucks mit dem bisherigen Wert des Zielmakros.

So addiert das Statement

```
#c #+ 2
```

die Zahl 2 auf den bestehenden Wert des Makros $\#c$ und speichert das Ergebnis im Makro $\#c$.

Als Voraussetzung dafür muß das Makro $\#c$ bereits einen numerischen Wert enthalten.

Besitzt $\#c$ keinen numerischen Wert, so wird dieses Statement mit einer Fehlermeldung beantwortet.

5. 4. 2002

Ausdruck mit nur einem Makro

Im einfachsten Fall besteht ein Makroausdruck nur aus einem Makro ohne Operatoren.

Ein solcher Ausdruck liefert den Wert 0, wenn das Makro einen nicht numerischen Wert besitzt, sonst aber den numerischen Wert selbst.

So füllt das Statement

`#x #= #abc`

das Makro `#x` mit dem Wert 0, wenn das Makro `#abc`:

w den Wert 0 besitzt,

w einen nicht numerischen Wert enthält oder

w nicht definiert ist.

Besitzt `#abc` dagegen einen numerischen Wert, so wird dieser unverändert in das Makro `#x` übertragen.

5. 4. 2002

Neue Statements: #IF #ELSE #EIF

Die Statements #IF, #ELSE und #EIF entscheiden anhand von Makrowerten, ob die dahinter liegenden Statements eingelesen oder ungelesen übersprungen werden.

Im Gegensatz dazu werten die -IF Statements die Variablenwerte der einzelnen statistischen Fälle aus und entscheiden darüber, ob die dahinter liegenden Statements für den jeweiligen Fall verarbeitet oder übersprungen werden.

Folgende Angaben sind möglich:

#IF Makroausdruck

Dabei können beliebig komplizierte Ausdrücke verwendet werden, wie sie im Abschnitt *Makros: Makroausdrücke* beschrieben sind.

Beim Einlesen der Statements wird der Makroausdruck ausgewertet.

Ist das Ergebnis *wahr*, so werden die hinter #IF liegenden Statements normal verarbeitet.

Ist das Ergebnis hingegen *falsch*, so werden alle Statements, die zwischen dem #IF und dem dazugehörigen #EIF oder #ELSE liegen, nicht verarbeitet. Zum Beispiel:

```
#I F ( #a > 2 ) & ( #b > 10 )
DO <1>=1. . #a
XTAB COL=C10 ROW=#r ow<1>
EDO
#E I F
```

Enthält das Makro #a einen Wert größer 2 und das Makro #b einen Wert größer 10, so werden die Statements zwischen #IF und #EIF eingelesen. Ist diese Bedingung dagegen nicht erfüllt, so werden erst wieder die Statements hinter #EIF eingelesen

Das zu einem #IF gehörige #EIF-Statement ist zunächst das erste hinter #IF liegende #EIF.

Die dazwischen liegenden Statements bilden den Wirkungsbereich des #IF-Statements.

#IF-Statements lassen sich jedoch auch schachteln:

Im Wirkungsbereich eines #IF kann ein weiteres #IF-Statement liegen.

Dieses zweite #IF wird ebenfalls durch ein dahinter liegendes #EIF beendet.

Das #EIF zum zweiten #IF liegt vor dem #EIF zum ersten #IF. Zum Beispiel:

```
#I F
  . . .
  #I F
    . . .
    #E I F
  . . .
#E I F
```

Die Statements im Wirkungsbereich eines solchen zweiten, geschachtelten #IF-Statements werden nur ausgeführt, wenn die logischen Ausdrücke in beiden #IF-Statements *wahr* sind.

Eine solche Schachtelung darf bis zu zehn Stufen tief sein.

Das Statement PROCESS hat eine ähnliche Wirkung wie #IF, kann aber nicht geschachtelt werden.

#EIF

Dieses Statement beendet ein vorausgehendes #IF-Statements.

#ELSE

Dieses Statement beendet den Wirkungsbereich des #IF und sorgt dafür, dass die hinter dem #ELSE stehenden Statements nur eingelesen werden, wenn der Makroausdruck im #IF *nicht* erfüllt ist. Der Wirkungsbereich des #ELSE wird durch das nächste #EIF beendet. Zum Beispiel:

```
#I F #sprache = D
XTAB TITLE=' Rangrei he nach Net t or ei chwei t e' ...
#ELSE
XTAB TITLE=' Ranki ng on cover age' ...
#E I F
```

Enthält das Makro #sprache das Zeichen D, so wird das hinter #IF stehende XTAB-Statement mit einem deutschen TITLE-Text eingelesen und verarbeitet, das XTAB-Statement hinter #ELSE jedoch übersprungen.

Enthält das Makro #sprache einen Wert ungleich D, so wird das Statement mit deutschem Text übersprungen und dafür das Statement hinter #ELSE verarbeitet.

#ELSE IF Makroausdruck

Dieses Statement beendet den Wirkungsbereich eines davor stehenden #IF oder #ELSE IF und sorgt dafür, dass die dahinter stehenden Statements nur eingelesen werden, wenn die Makroausdrücke der davor stehenden #IF und #ELSE IF *nicht* erfüllt sind und der Makroausdruck des laufenden #ELSE IF dagegen erfüllt ist. Zum Beispiel:

```
#I F #sprache = D
XTAB TITLE=' Rangrei he nach Net t or ei chwei t e' ...
#ELSE I F #sprache = E
XTAB TITLE=' Ranki ng on cover age' ...
#ELSE
Fehl er : Mak ro ' sprache' mi t fa lschem Wert
#E I F
```

Enthält das Makro #sprache das Zeichen D, so wird das hinter #IF stehende XTAB-Statement mit einem deutschen TITLE-Text eingelesen und verarbeitet. Die Statements zwischen #ELSE IF und #EIF werden dagegen ausgelassen.

Enthält das Makro #sprache das Zeichen E, so wird nur das XTAB-Statement mit dem englischen Text eingelesen und die übrigen Statements übersprungen.

Besitzt das Makro #sprache einen Wert ungleich D oder E, so wird die Zeile *Fehler: Makro ...* eingelesen und alle übrigen Statements ausgelassen.

Diese Zeile enthält kein gültiges Statement, produziert also die Fehlermeldung 100 und führt zum vorzeitigen Ende des Programmlaufs.

12. 3. 2002

Intel-Integer in F-, G- und I-Feldern

Bisher mussten die Dualzahlen in den F-, G- und I-Feldern "normal" gespeichert sei, das heißt die höchstwertigen Bits in den linken Bytes. Jetzt ist auch die umgekehrte Speicherungsform möglich, mit den höchstwertigen Bits in den rechten Bytes. Dazu erhalten die Statements INPUT-EXT und OUTPUT-EXT den folgenden Parameter:

< INTEGER = | NORMAL | >
 | REVERSE |

Mit diesen Angaben wird festgelegt, wie die Dualzahlen der externen Datenfelder vom Typ F, G und I in der Datei gespeichert sind.

Fehlt die Angabe INTEGER, so wird INTEGER=NORMAL angenommen.

NORMAL Für Dualzahlen von 2 ... 4 Byte Länge enthalten die linken Bytes werden die höherwertigen Bits.

REVERSE Für Dualzahlen von 2 ... 4 Byte Länge enthalten die rechten Bytes die höherwertigen Bits.