MORTEN FENGER
Cand.merc.(scm.)

STATISTIK MED SAS®

– på den nemme måde med step-by-step cases, som alle kan forholde sig til.

Denne e-bog introducerer dig til markedets stærkeste statistikværktøj. SAS kan alt inden for analytics og er samtidig let at lære. Derfor er det bare med at komme i gang – og vejen starter i denne e-bog.

Copyright © 2012
SAS Institute A/S, Copenhagen Denmark
ISBN 978-87-984612-0-3

sas THE POWER TO KNOW.

Om dig

Denne e-bog er skrevet til dig, som enten har gennemført et indledende statistikkursus eller er i gang med det. Bogens case er boligmarkedet, fordi det er et emne, alle kan forholde sig til – uanset om du er studerende eller skal til at bruge SAS i dit job – og uanset om du er polit., farmaceut, mediciner, biolog, matematiker, erhvervsøkonom eller aktuar. Fællesnævneren er kendskabet til indledende statistik og en ambition om at lære SAS på en formiddag.

Om forfatteren

Denne e-bog er skrevet af Morten Fenger, cand.merc.(scm.) fra Aarhus Universitet (AU). Morten kender SAS som først studerende og siden underviser i AU's IKT-afdeling. Morten har været instruktor i fagene *Statistik I* og *II*, *Videnskabsteoretisk Metode II, Økonometri, Statistisk Simulering* og *Data Mining*. Morten arbejder i dag som research assistant på AU, hvor han anvender SAS til håndtering og analytics af "big data".

Om komitéen bag denne e-bog

SAS tager dig som læser alvorligt, og SAS' uddannelsesansvarlige Georg Morsing har derfor sammen med SAS' analyseekspert ph.d. Kaare Brandt Petersen sat kursen for denne e-bogs faglige indhold og approach. SAS Academics' projektleder Ane Gerken har under hele processen sørget for, at denne e-bog har fokus på at få dig let og hurtigt i gang med SAS.

Denne bog tager udgangspunkt i Gorm Gabrielsens introduktion til JMP* og introducerer SAS med den samme gennemgående case.

Har du kommentarer eller spørgsmål til denne e-bog eller andet inden for SAS, kan du altid skrive til *academic.dk@sas.com*, hvor vi kan hjælpe dig videre med SAS.

* JMP er et andet statistikprogram fra SAS med fokus på at skabe overblik over data før store analyser i fx SAS.

INDHOLD

1. EN KORT INTRODUKTION TIL SAS OG DENNE E-BOG 4
1.1. Hvad er SAS, og hvorfor er programmet så stort? 4
1.2. SAS på Windows og Mac 4
1.3. Introduktion til datasættet anvendt i denne e-bog 5
1.4. Hvordan er SAS bygget op? 6

2. GØR DIT DATASÆT KLAR TIL ANALYSER I SAS 7
2.1. Åbn SAS og importer dine data fra hvilken som helst kilde 7
2.1.1. Fortæl SAS, hvor dine data ligger: SAS libraries 9
2.1.2 Importér data fra Excel 11
2.2. Få overblik over dit datasæt 14
2.2.1 Datatyper i SAS 14
2.2.2. Beskrivende statistik 14
2.2.3. Visualisér data og tjek, om en variabel er normalfordelt 16
2.3. Tilpas dit datasæt til de analyser, du gerne vil lave 19
2.3.1. Rekodning af en variabel 20
2.3.2. Lav en dummyvariabel 24
2.3.3. Gruppering af en variabel 26
2.3.4. Beregning af en ny variabel 28
2.3.5. Transformering af en variabel 30
2.4. Begræns dit datasæt 31

3. DET MEST ANVENDTE STATISTIK I PRAKSIS 33
3.1. Test på gennemsnit (means) 34
3.1.1.Test for ét gennemsnit (One sample t-test) 34
3.1.2. Test af forskellen mellem to uafhængige gennemsnit (og varianser) 36
3.1.3. Test af to parvise stikprøver 39
3.1.4. Test flere gennemsnit med variansanalyse ('One-Way' ANOVA) 39
3.1.5. Test flere gennemsnit på to faktorer ('Two-Factor' ANOVA) 43
3.2. Test af andele (proportions) og antals-tabeller (Chi-i-anden-tests) 46
3.2.1. Test af en andel (= procent mellem 0% og 100% = tal mellem 0 og 1) 46
3.2.2. Test af forskellen mellem to uafhængige andele 47
3.2.3. Test for uafhængighed (Chi-i-anden-test) 50
3.3. Regressionsanalyse 54
3.3.1. Scatter plot og test af korrelation mellem to intervalvariable 54
3.3.2. Simpel lineær regression (regression med én forklarende variabel) 56
3.3.3. Multipel lineær regression (regression med flere forklarende variable) 58
3.3.4. Regression med mere avancerede variable 63
3.3.5. Logistisk regression 66
3.4. Fordelingstests 69
3.4.1. Binomialfordelingstest 69
3.4.2. Hypergeometrisk fordelingstest 70
3.4.3. Normalfordelingstest 71

4. INTRODUKTION TIL SAS-PROGRAMMERING 72
4.1. Automatisk kode, integreret hjælp og code completion 72
4.2. Kodeeksempler til denne e-bog – spar tid og dokumentér! 80

1. EN KORT INTRODUKTION TIL SAS OG DENNE E-BOG

Velkommen til denne e-bog, som er skrevet til dig, der gerne vil hurtigt i gang med at anvende SAS. Bogen er inddelt i 4 kapitler, som igen er inddelt i afsnit og underafsnit, jf. indholdsfortegnelsen på side 3. Her i kapitel 1 introduceres SAS og bogens gennemgående case og datasæt. I kapitel 2 importerer vi et datasæt fra Excel til SAS og laver et par ændringer i datasættet – SAS anvendes nemlig ofte til datahåndtering. I kapitel 3 udføres diverse statistiske analyser i SAS. Fokus er på at introducere teknikkerne og at komme omkring de mest anvendte statistiske metoder i praksis. Hvis du gerne vil gå direkte til kapitel 3, kan du finde e-bogens analysedatasæt på www.sasacademic.dk under fanen "Til undervisning" eller på www.sasdownload.dk Kapitel 4 giver en let introduktion til SAS-programmering og opsummerer denne e-bog på et par siders SAS-kode.

1.1. Hvad er SAS, og hvorfor er programmet så stort?

SAS er blandt verdens mest komplette statistik- og datahåndteringsprogrammer. SAS kan stort set alt inden for statistik og datahåndtering. Mange af verdens største virksomheder anvender SAS, og stort set alle danske C20-virksomheder er blandt SAS' kunder. Virksomheden bag SAS hedder SAS Institute og har hovedkontor i North Carolina i USA.

SAS er et stort, avanceret datahåndterings- og analysesystem, som startede med at blive udviklet allerede i 60'erne på *North Carolina State University*. Oprindeligt var det et forskningsprojekt til at analysere landbrugsdata, men projektet voksede hurtigt til andre forskningsfelter. Og heldigvis er der sammenfald mellem, hvad man lærer på universiteterne, og hvad virksomheder beskæftiger sig med, så SAS blev til en selvstændig virksomhed i 1976 og har siden da leveret dataanalytiske værktøjer til verdens største virksomheder inden for alle tænkelige områder som medicinalindustrien, banker, forsikringsselskaber, teleselskaber, universiteter og offentlige forvaltninger – i hele verden.

Som alle andre computerprogrammer med mere end 20 års historie har SAS udviklet sig fra et kodebaseret sprog til et moderne peg og klik-system. Alle funktioner er bevaret, og flere nye er kommet til. Det betyder, at du nu kan være i gang med at anvende SAS i praksis på en formiddag og to kopper kaffe. SAS-kodning lyder sværere, end det er – vi introducerer det i kapitel 4 og viser, hvordan det kan bruges som en alternativ måde at løse de samme opgaver, som vi vil gennemgå i peg og klik-SAS i kapitel 2 og 3.

De SAS-koder, der blev skrevet i SAS i 1968, er konvertible med de nyeste versioner af SAS. Der er bare blevet bygget oven på SAS-systemet lige siden, og der kommer stadig nye funktioner til, i takt med at forskningen udvikler det statistiske felt. Derfor fylder programmet lidt, og det tager lidt tid at installere. Til gengæld betyder det også, at de koder, du skriver i SAS i dag, med garanti også kan køre i fremtidige versioner af SAS.

Hvis du ikke allerede har installeret SAS, kan du finde en installationsvejledning på www.sasacademic.dk under fanen "Support".

1.2. SAS på Windows og Mac

SAS er bygget til at køre i Windows, fordi det er det, de fleste virksomheder anvender. For brugere, der foretrækker Mac, betyder det, at man først skal installere Windows-styresystemet på sin Mac. Man kan bruge softwaren *WMware Fusion* eller *Boot Camp* til dette. Når først dette er gjort, mangler man bare at installere SAS.

Det kan altså sagtens lade sig gøre at køre SAS på en Mac – alt i forhold til denne e-bog er faktisk udarbejdet på en Mac.

1.3. Introduktion til datasættet anvendt i denne e-bog

Denne e-bog tager udgangspunkt i noget, vi alle kan forholde os til, nemlig boligmarkedet. Uanset om man er mediciner, farmaceut eller økonom, så kan vi nok alle sætte os ind i, at der er sammenhænge mellem prisen på en bolig og fx størrelsen, antal værelser, herlighedsværdi etc.

I en søndagsavis engang i år 2000 var der en masse boliger til salg, og 69 af disse var huse nord og syd for København. Lad os for sjov skyld antage, at det er noget data, du har indsamlet og registreret, fordi du er på jagt efter en ny bolig i Københavnsområdet. Informationen om disse huse har du samlet i en stor tabel, hvor hver række har med én boligannonce at gøre. Hver kolonne kaldes en variabel, og et eksempel er, at hver boligannonce har en *kontantpris*. Sådan en tabel kaldes et datasæt i statistiksprog.

I de følgende analyser antager vi, at stikprøven af boliger er repræsentativ for alle huse nord og syd for København, dvs. vi antager, at det er et tilfældigt udpluk af disse boliger, der er til salg på denne givne søndag i år 2000.

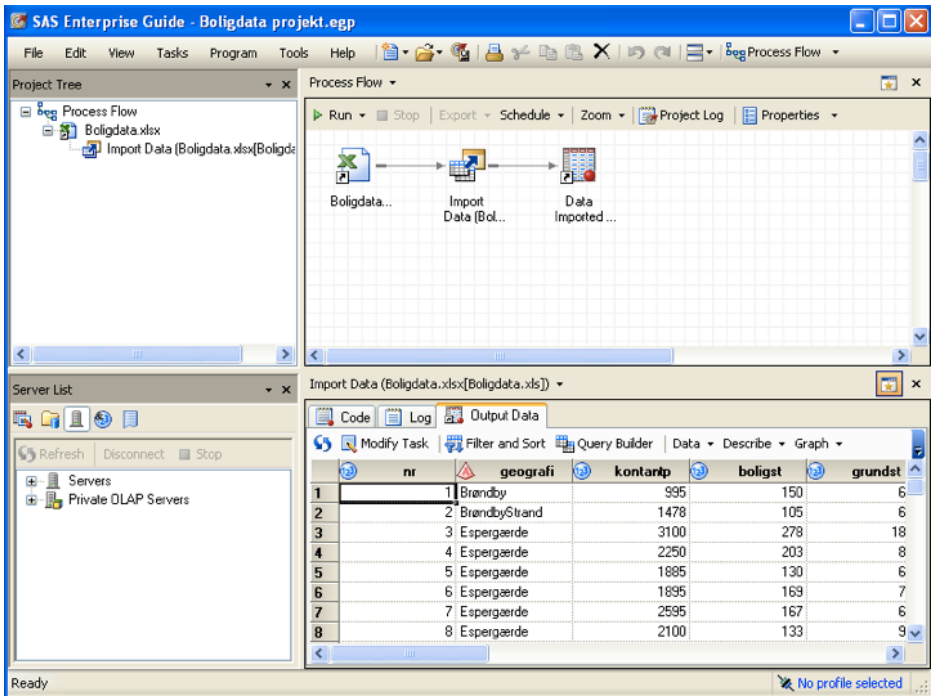
De forskellige variable i datasættet er:

nr	Nummeret på annoncen (tilfældig rækkefølge)
geografi	Kommunen, hvor huset ligger
kontantp	Kontantpris i 1.000 kr. (fx betyder 995 at huset koster 995.000 kr.)
boligst	Boligstørrelse i m^2
grundst	Grundstørrelse i m^2
vaerelse	Antal værelser i huset
omr	Områdevariablen angiver, om huset ligger nord (omr = 1) eller syd (omr = 2) for København
kvalitet	Udtryk for herlighedsværdien af huset vurderet mellem 1 (lav) og 5 (høj)
kvalitetgr	Gruppering af *kvalitet* i niveauerne: 1 (lav, dvs. *kvalitet* = 1 eller 2), 2 (mellem, dvs. *kvalitet* = 3) eller 3 (høj, dvs. *kvalitet* = 4 eller 5)

	nr	geografi	kontantp	boligst	grundst	vaerelse	omr	kvalitet	kvalitetgr
1	1	Brøndby	995	150	695	2	2	1	1
2	2	BrøndbyStrand	1478	105	630	3	2	3	2
3	3	Espergærde	3100	278	1815	6	1	2	1
4	4	Espergærde	2250	203	885	4	1	3	2
5	5	Espergærde	1885	130	627	3	1	3	2
6	6	Espergærde	1895	169	702	3	1	3	2
7	7	Espergærde	2595	167	694	3	1	4	3
8	8	Espergærde	2100	133	951	2	1	3	2
9	9	Espergærde	2700	255	1120	5	1	3	2
10	10	Greve	3995	111	3221	4	2	4	3
11	11	Greve	4200	179	736	6	2	5	3
12	12	Greve	1595	146	756	3	2	2	1
13	13	Greve	1450	144	766	3	2	1	1
14	14	Greve	1295	81	850	2	2	2	1
15	15	Greve	1895	165	788	3	2	3	2
16	16	Greve	1495	135	810	3	2	2	1
17	17	Greve	1795	182	795	4	2	2	1
18	18	Greve	3100	193	788	4	2	5	3
19	19	Greve	2195	183	758	3	2	3	2
20	20	Greve	1365	124	732	3	2	2	1
21	21	Greve	1975	185	771	4	2	2	1
22	22	Hørsholm	1669	121	384	3	1	3	2
23	23	Hørsholm	1795	121	812	3	1	3	2
24	24	Hørsholm	2496	160	2285	3	1	1	1
25	25	Hørsholm	2998	231	958	5	1	4	3
26	26	Hørsholm	2299	154	799	4	1	3	2
27	27	Hørsholm	1897	122	462	3	1	3	2
28	28	Hørsholm	2495	180	812	4	1	3	2
29	29	Hørsholm	3248	217	1825	3	1	3	2
30	30	Hørsholm	2438	177	747	4	1	4	3
31	31	Humlebæk	1745	143	805	4	1	2	1
32	32	Humlebæk	4700	236	1700	4	1	5	3
33	33	Humlebæk	2400	158	1204	8	1	2	1
34	34	Humlebæk	2625	181	925	4	1	4	3
35	35	Humlebæk	1850	151	1018	4	1	2	1
36	36	Humlebæk	1950	138	713	3	1	3	2
37	37	Humlebæk	2990	183	1167	6	1	4	3
38	38	Ishøj	1275	115	482	3	2	2	1
39	39	Ishøj	1585	181	656	6	2	1	1
40	40	Kokkedal	1668	135	708	3	1	3	2
41	41	Kokkedal	1495	127	784	3	1	3	1

1.4. Hvordan er SAS bygget op?

Nedenfor ser du SAS-brugergrænsefladen, som den fx kan se ud. Øverst er dropdown-menuerne, hvor alle funktioner findes. I øverste venstre hjørne er *Project Tree*, hvor man i en hierarkisk struktur kan holde overblik over sit SAS-projekt. I det store felt øverst til højre ses *Process Flow*-diagrammet. I *Process Flow* får man et godt overblik over sit projekt. Man kan se, hvad man har gjort med sine data og i hvilken rækkefølge. Det er virkelig smart til at bevare overblik og navigere rundt i projektet. I Excel har du mange forskellige regneark. I SPSS har du kun ét datasæt ad gangen og så en uoverskuelig række af outputs. I SAS har du det forkromede overblik via *Process Flow*, som automatisk genereres, når du arbejder i SAS. Som du måske kan gætte ud fra *Process Flow*-diagrammet nedenfor, har jeg importeret et Excel-datasæt i SAS. Hvis nu der kommer en ændring i Excel-filen, kan man automatisk importere et nyt, opdateret datasæt i SAS med et enkelt klik. Og hvis vi nu var på sidste side i denne e-bog og allerede havde lavet en masse analyser baseret på dette datasæt, så kunne de automatisk blive opdateret baseret på det nye datasæt med ét klik på *Run*-knappen i vores *Process Flow*-diagram. Det er altså ret smart, særligt i store projekter.



Nederst til venstre i skærmbilledet er der en oversigt over servere. Det er noget, man bruger, hvis man anvender SAS i store organisationer med data på diverse fællesdrev og servere med gigantisk beregningskraft. Man kan sagtens lukke den boks ved at klikke på det lille grå kryds til højre for *Server List* (den kan altid findes frem igen via dropdown-menuen *View*.)

SAS er et dynamisk program, og særligt feltet til højre vil skifte indhold, afhængigt af hvad man er i gang med. Højre halvdel af skærmen består typisk kun af et stort felt med det, man har fokus på, men her har jeg delt min skærm i to for at kunne vise både *Process Flow*-diagrammet og et datasæt samtidig (det er gjort via *View* > *Workspace* > *Layout*, hvis du absolut vil vide det, men det er ligegyldigt her).

I det store felt nederst til højre ses et eksempel på, hvordan et datasæt ser ud i SAS. Læg mærke til fanebladene i feltet nederst til højre. Klikker man på dem, ændrer feltet indhold. Du kan vælge at se koden, som kører inde bag ved i SAS-maskineriet. Du kan også se en *log* fra sidste kørte analyse/datahåndtering. Ingen af delene behøver du dog at forholde dig til i denne e-bog – vi snuser blot til det i kapitel 4.

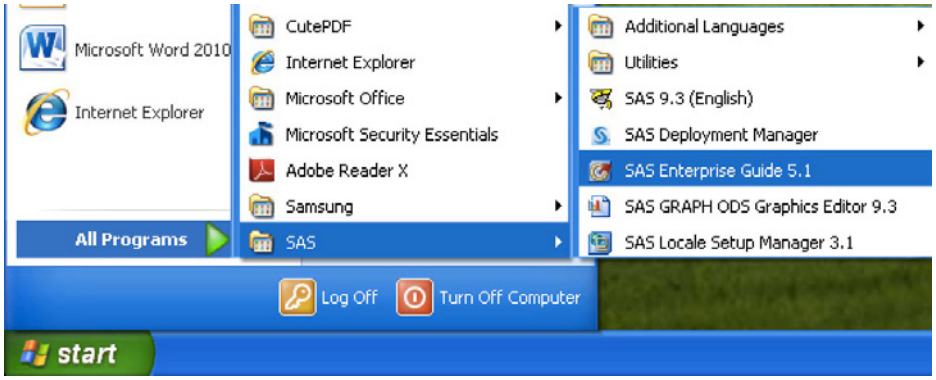
Stort set alt i denne e-bog kan laves fra dropdown-menuen *Tasks*, hvor både diverse datahåndtering, grafer og statistiske analyser kan laves. Du finder også mange af de samme valgmuligheder i "bjælken" over datasættet i skærmbilledet, og det er ofte via den bjælke, vi vil køre analyserne i denne guide. Lige om lidt skal vi prøve at importere det datasæt, vi skal arbejde med fra Excel, ligesom det er gjort i billedet ovenfor.

2. GØR DIT DATASÆT KLAR TIL ANALYSER I SAS

Før vi kan gå i gang, skal du selvfølgeligt have SAS installeret. Du kan finde installationsfiler og vejledning til installationen på www.sasacademic.dk. Denne e-bog antager i det følgende, at du allerede har SAS installeret.

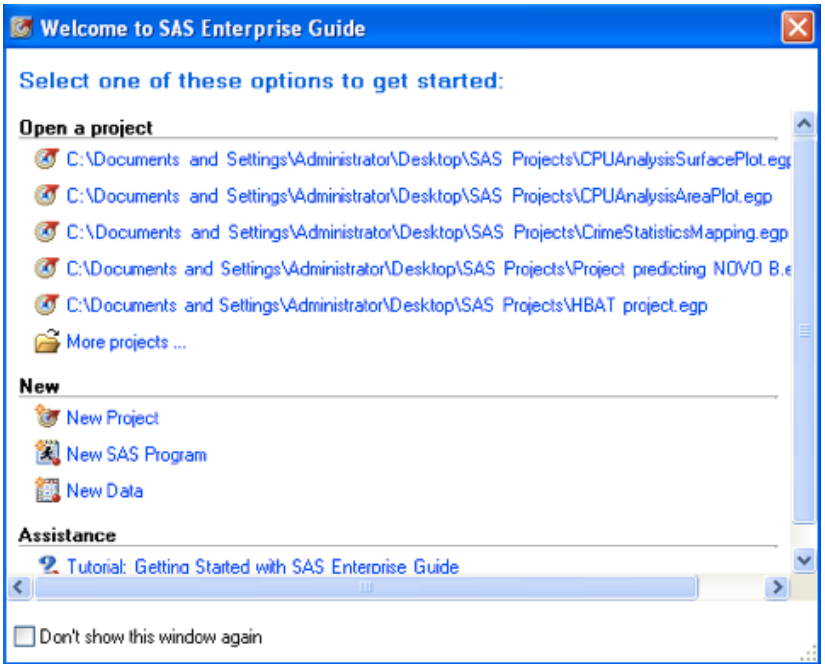
2.1. Åbn SAS og importer dine data fra hvilken som helst kilde

SAS startes som alle andre programmer via *Start* > *Programmer*, og derefter vælges *SAS* > *SAS Enterprise Guide* som nedenfor:

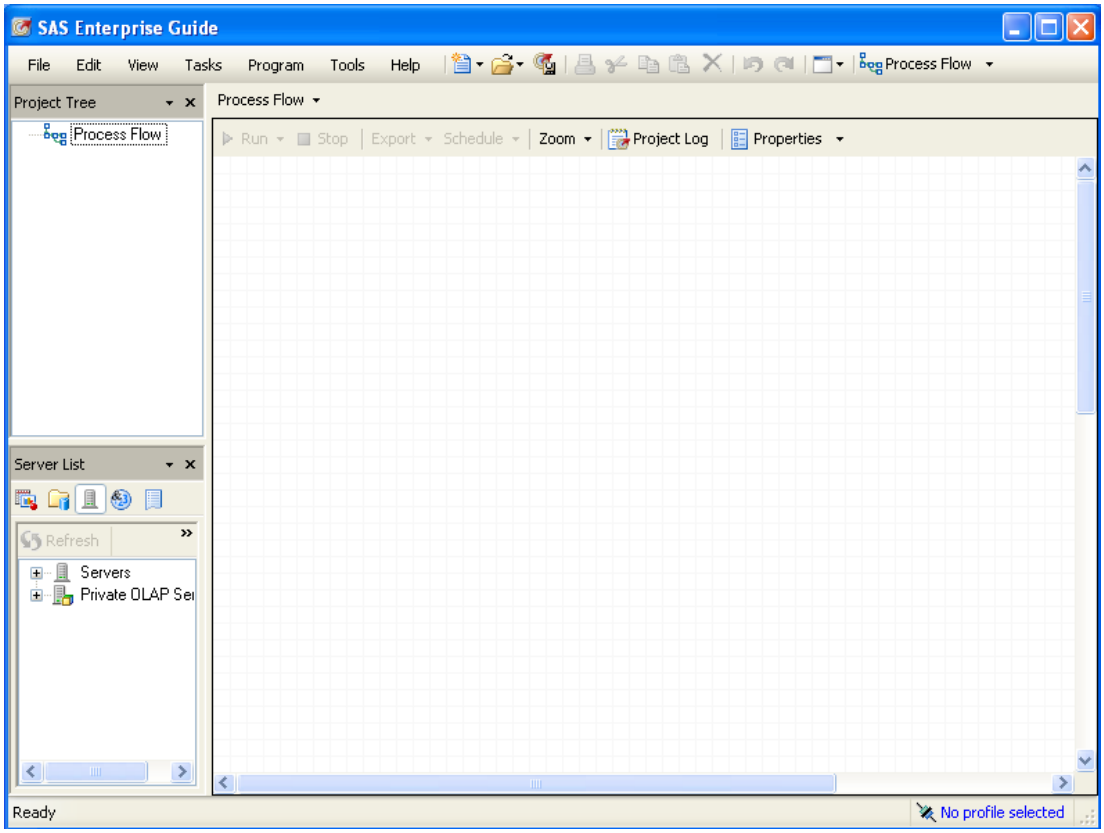


Når SAS åbnes, kommer velkomstskærmen frem. Der vil man kunne vælge sit projekt, næste gang man åbner SAS, men første gang man åbner SAS, er listen over tidligere SAS-projekter selvfølgelig tom.

Vi klikker på *New Project* cirka midt på velkomstskærmen nedenfor:

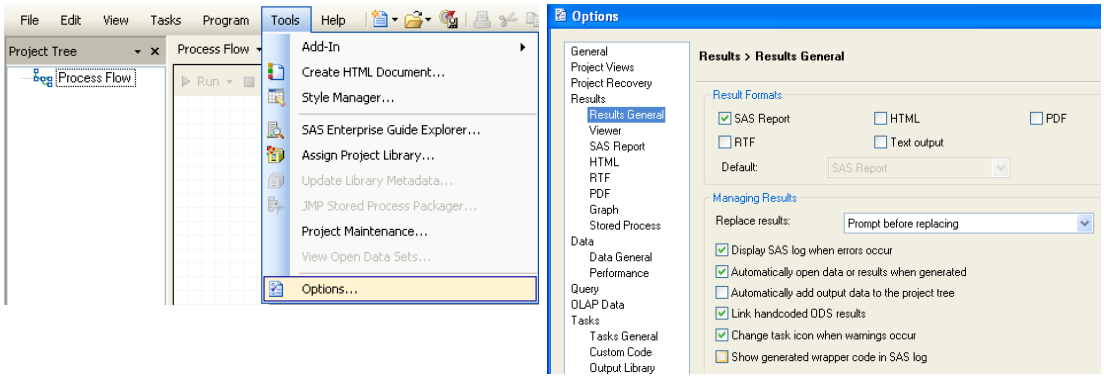


Så er vi klar med et nyt, tomt projekt!:



Før vi går videre, vil vi lige foretage en enkelt indstilling, som gør vores SAS-*logs* lettere at læse. Vi anvender egentlig ikke SAS-*logs* i hovedparten af denne e-bog for enkelhedens skyld, men bare lav indstillingen nedenfor alligevel – du kommer måske til at se på *log*s senere – måske bliver du bidt af SAS.

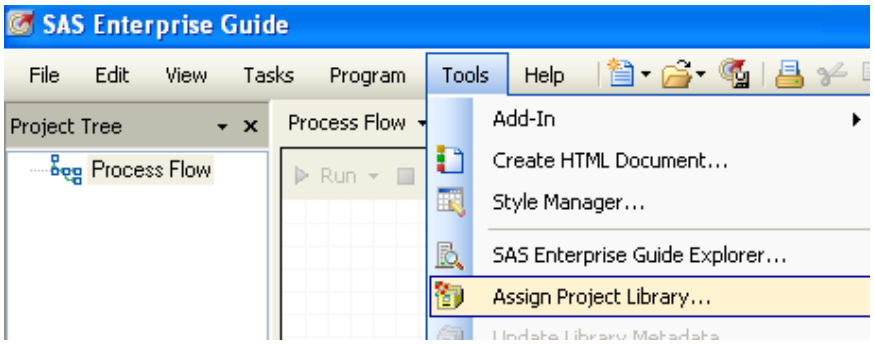
Vælg *Tools* > *Options*, vælg derefter *Results General* og fjern det nederste flueben i billedet nedenfor under valgmuligheden *Show generated wrapper code in SAS log*:



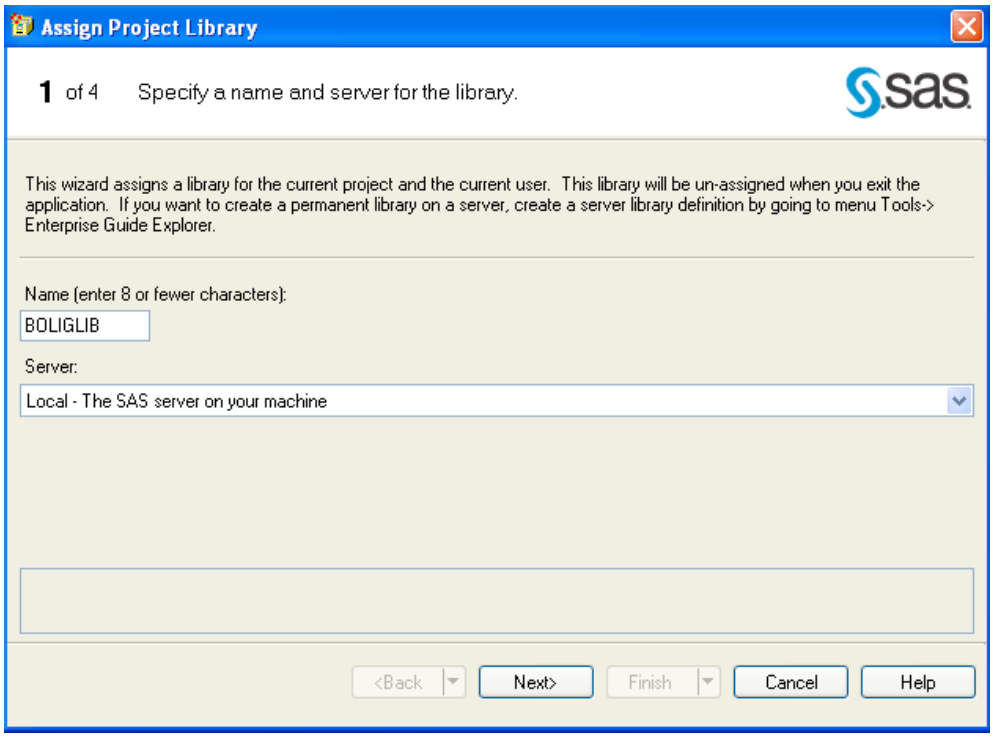
Det var det – nu er vores *log* lettere at læse, hvis vi skulle få brug for det.

2.1.1. Fortæl SAS, hvor dine data ligger: SAS libraries

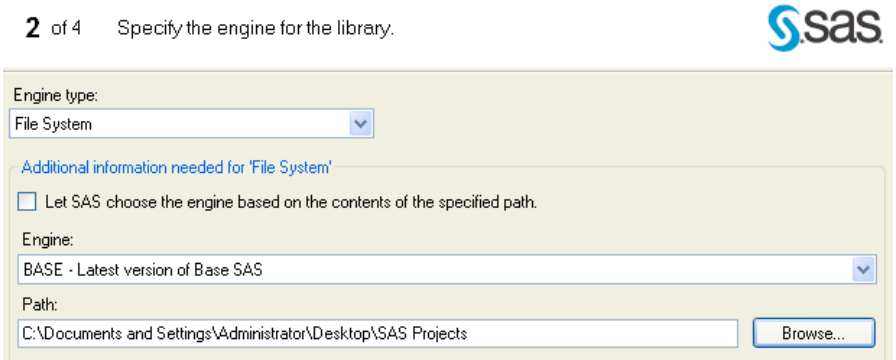
Det første, vi gør i SAS, er at fortælle SAS, hvor vores data fra vores projekt skal lægges. SAS kan køre på alt fra en almindelig computer til store, komplekse serverløsninger, så derfor er det ikke givet for SAS, at dine data ligger på dit C-drev eller dit skrivebord. SAS har derfor sit eget system til at holde styr på data, nemlig databiblioteker eller *libraries*, som det kaldes i SAS. Et SAS *library* er faktisk bare en forkortelse for en sti til dataene. Vi vil lave et *library*, som vi vil kalde *BOLIGLIB*. Det gøres ved at vælge *Tools* > *Assign Project Library*… som vist nedenfor:



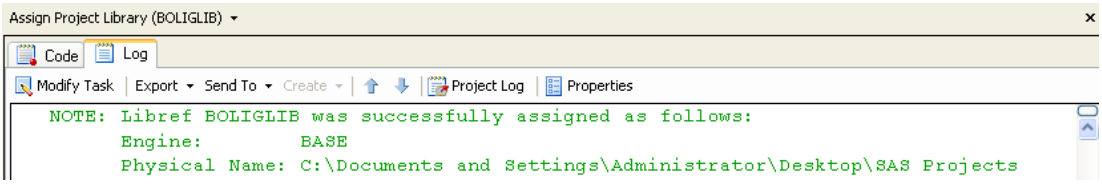
Herefter skal man navngive sit *library*. Vi kalder det *BOLIGLIB* og trykker derefter *Next*:



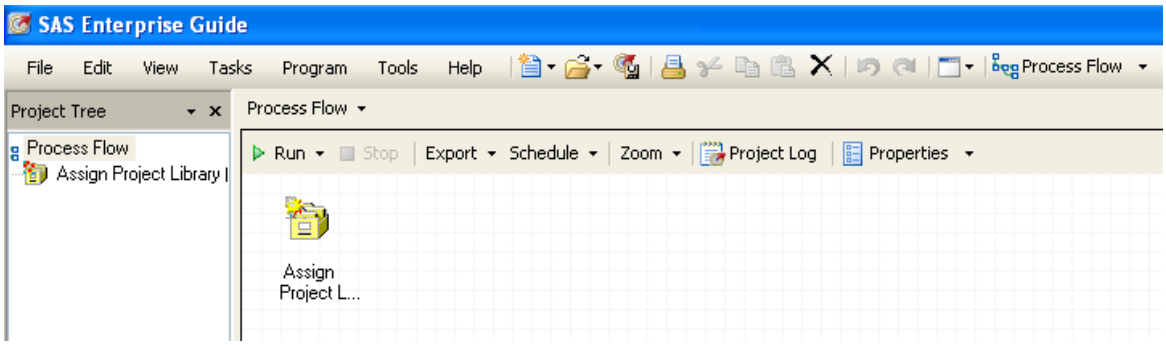
På næste skærmbillede nedenfor skal vi angive, hvor vi gerne vil gemme vores SAS-data. Her er valgt en mappe, som jeg har kaldt SAS Projects, som ligger på mit skrivebord i Windows. Man vælger et sted ved at trykke på Browse, og man navigerer dernæst frem til det ønskede sted. Tryk derefter på *Next*:



Tryk på *Next* igen og tryk derefter på Test Library. Hvis du har valgt en placering, du har skriverettigheder til, så vil SAS fortælle dig, at dit *library* er gyldigt (der står bare '*OK*'). Tryk nu på *Finish*, hvorefter der kommer en *log* frem som nedenfor. Loggen er i princippet ligegyldig, men hvis du vil, kan du skimme den igennem. Kig efter den grønne tekst, som er det vigtige. Og skulle der være rød skrift, kan du regne ud, at det betyder, at der er noget galt med koden, og loggen fortæller dig så, hvad der er galt (det kunne fx være, at du prøvede at åbne et datasæt, som du i mellemtiden havde flyttet). I vores *log* nedenfor er alt dog i sin skønneste orden, og det vil det også være i resten af denne e-bog:



På fanen *Code* kan du se den kode, som SAS genererer. Bare se helt bort fra fanerne *Code* og Log indtil videre. Vi snuser lidt til SAS-programmering i kapitel 4, men du behøver ikke vide noget som helst om det for at komme godt i gang med SAS i denne e-bog Skynd dig derfor at trykke på *Process Flow*-knappen på skærmen (Process Flow), og du vil kunne se, at vores SAS *library* er blevet oprettet:

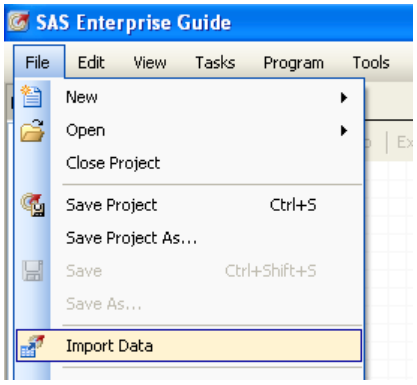


Vi er nu klar til at gå videre…

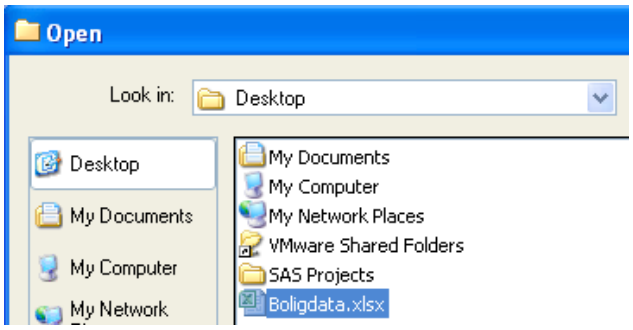
2.1.2 Importér data fra Excel

De fleste softwaremanualer starter med at give dig et datasæt i programmets eget dataformat. Sådan et datasæt kan du også finde på www.sasacademic.dk klar til brug. Men eftersom du nu læser denne SAS-intro-e-bog, er det nok, fordi du ikke har arbejdet så meget med SAS før. I Excel er du sikkert mere på hjemmebane, så derfor vil vi starte med at importere et Excel-regneark til SAS. Så kan du selv se, hvor let det er.

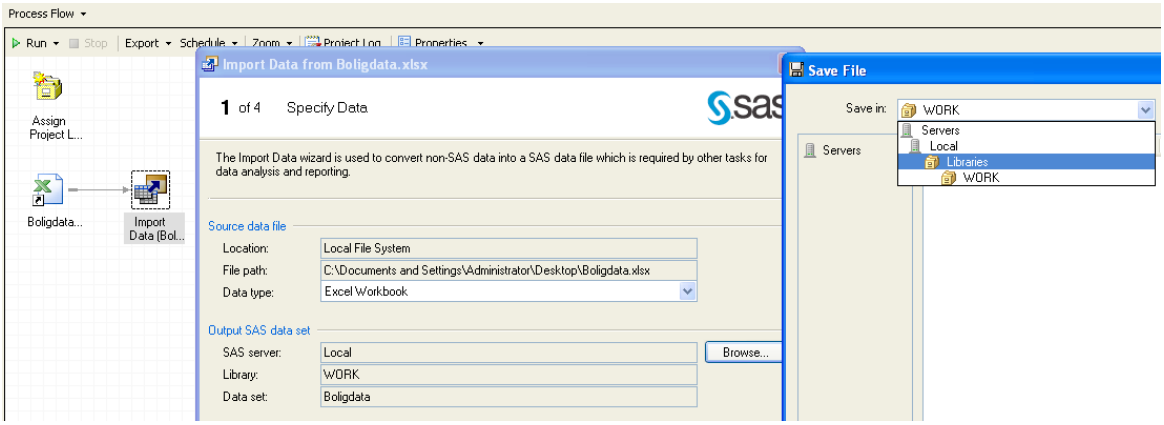
Vi vælger *File* > *Import Data* som nedenfor:



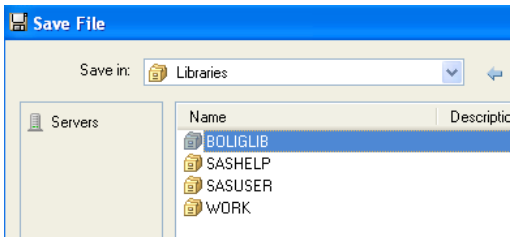
Derefter finder vi frem til vores Excel-fil *Boligdata.xlsx*, som her ligger på skrivebordet. Markér filen og tryk *Open*:



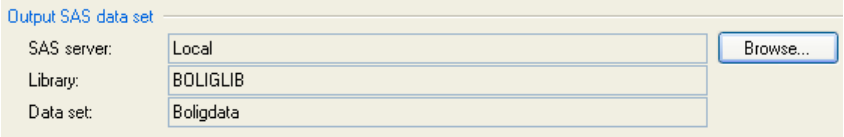
Herefter fremkommer dialogboksen nedenfor. Læg mærke til, at SAS allerede er i fuld gang med at illustrere dine handlinger i *Process Flow*-diagrammet ude til højre nedenfor. Lad os importere Excel-datasættet til vores *library*. Det gøres ved at trykke *Browse* i dialogboksen, og derefter vælge *Save in*: *Libraries* øverst til venstre…



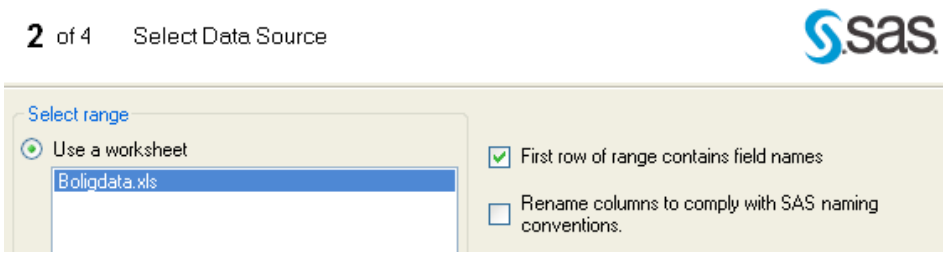
… hvorefter vores *library BOLIGLIB* vælges…



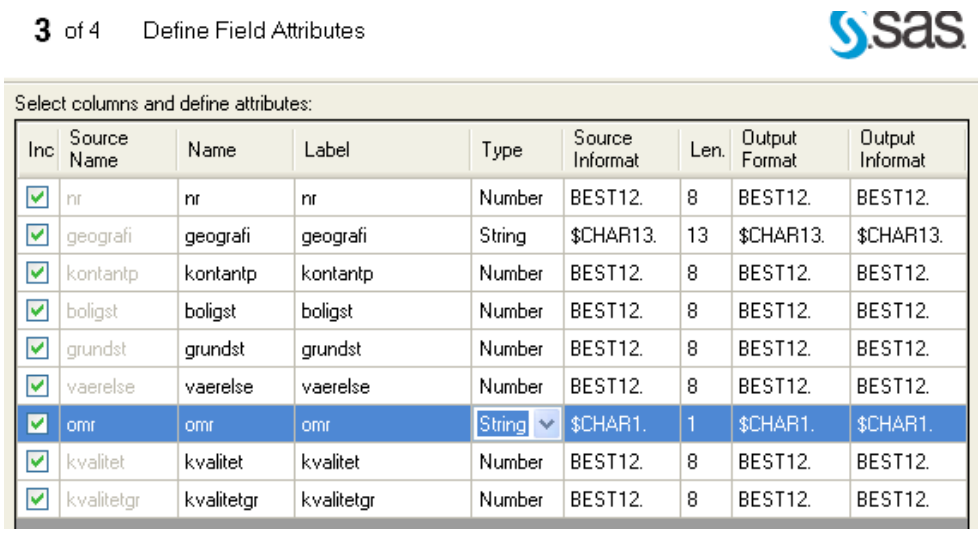
… og der trykkes *OK* > *Save*. Nu fremgår det af vores dialogboks, hvor det importerede data bliver gemt:



Tryk *Next*. På næste trin foreslår SAS selv at importere fra dataarket *Boligdata*, og at første række i Excel-regnearket indeholder variabelnavnene. Begge dele er fint, så vi laver ikke nogen ændringer her:

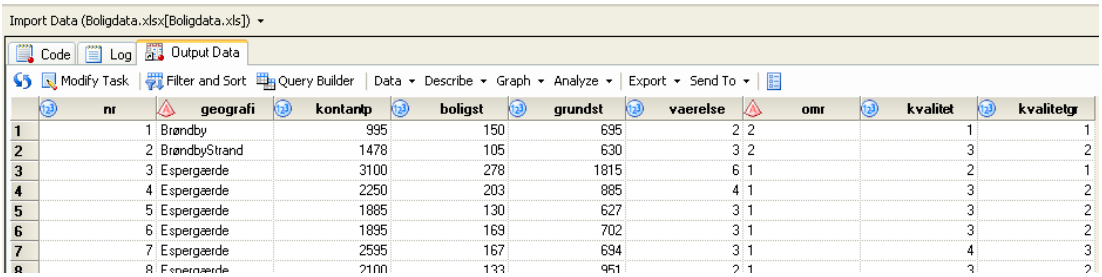


Tryk *Next* igen, og vi kan nu angive, om vores variable skal opfattes som tal i SAS (number) eller ikke som tal (string). Variablen *Geografi* er den eneste, som SAS ikke læser som et nummer. Variablen *omr* er godt nok også et tal, men der kunne lige så godt have stået *Nord/Syd* i stedet for *1* eller *2* i denne variabel (husk, at det er sådan, denne variabel er defineret – se evt. variabeloversigten på side 5). Derfor vælger vi at få SAS til at importere *omr*-variablen som *String*-type. Det gøres ved at vælge *Type*: *String* og derefter klikke *OK* til pop op-boksen, om SAS selv skal finde et passende format (pop op-boksen er ikke vist her):



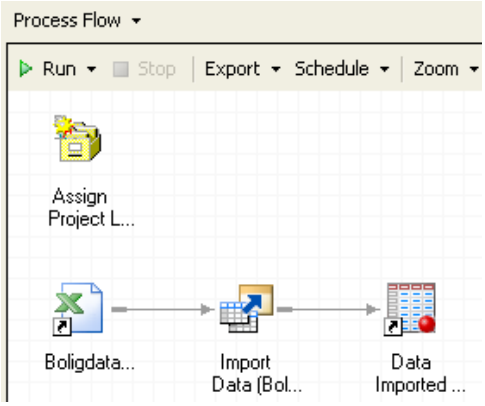
Inc	Source Name	Name	Label	Type	Source Informat	Len.	Output Format	Output Informat
☑	nr	nr	nr	Number	BEST12.	8	BEST12.	BEST12.
☑	geografi	geografi	geografi	String	$CHAR13.	13	$CHAR13.	$CHAR13.
☑	kontantp	kontantp	kontantp	Number	BEST12.	8	BEST12.	BEST12.
☑	boligst	boligst	boligst	Number	BEST12.	8	BEST12.	BEST12.
☑	grundst	grundst	grundst	Number	BEST12.	8	BEST12.	BEST12.
☑	vaerelse	vaerelse	vaerelse	Number	BEST12.	8	BEST12.	BEST12.
☑	omr	omr	omr	String	$CHAR1.	1	$CHAR1.	$CHAR1.
☑	kvalitet	kvalitet	kvalitet	Number	BEST12.	8	BEST12.	BEST12.
☑	kvalitetgr	kvalitetgr	kvalitetgr	Number	BEST12.	8	BEST12.	BEST12.

Vi kan nu trykke *Finish*, hvorefter datasættet nedenfor fremkommer (kun de første rækker er vist):



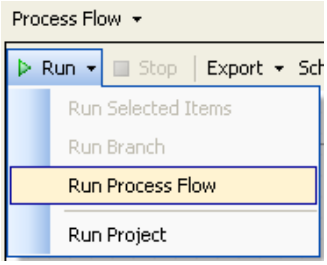
	nr	geografi	kontantp	boligst	grundst	vaerelse	omr	kvalitet	kvalitetgr
1	1	Brøndby	995	150	695	2	2	1	1
2	2	BrøndbyStrand	1478	105	630	3	2	3	2
3	3	Espergærde	3100	278	1815	6	1	2	1
4	4	Espergærde	2250	203	885	4	1	3	2
5	5	Espergærde	1885	130	627	3	1	3	2
6	6	Espergærde	1895	169	702	3	1	3	2
7	7	Espergærde	2595	167	694	3	1	4	3
8	8	Espergærde	2100	133	951	2	1	3	2

I *Process Flow*-diagrammet kan man tilsvarende se det importerede datasæt:



Så let var det. Velkommen til SAS.

OBS: Før du lukker SAS ned, så husk at gemme dit projekt. Når du skal arbejde med det igen, henter du projektet ind og kører det ved at trykke på 'Run' og dernæst 'Run Process Flow' via knappen øverst til venstre i 'Process Flow'-diagrammet. Det er vist nedenfor:



På den måde bliver alle dine indtastninger og analyser opdateret. I SAS-projektfilen ligger nemlig koden til alle handlinger foretaget i projektet. Smart ikke?

2.2. Få overblik over dit datasæt

Før man går i gang med at analysere data, er man nødt til at have overblik over sit datasæt. Det kan man få ved at beregne opsummerende tal for datasættets variable, fx gennemsnit af variable (underafsnit 2.2.2) samt ved at lave diverse grafer ud fra datasættets variable (underafsnit 2.2.3). Før vi går i gang med den beskrivende statistik, kommer der dog lige en kort intro af datatyperne i SAS og i statistik generelt (underafsnit 2.2.1).

2.2.1 Datatyper i SAS

SAS arbeider grundlæggende med to datatyper: tal eller tekst. Symbolerne er hhv. for tal og for tekst. Tekstvariable er typisk ord, fx kommune, men det kan også være tal, fx postnumre. Grunden til, at det kan være en god idé at klassificere nogle tal som tekst, er, at det fx ikke giver mening at beregne et gennemsnit af postnumre.

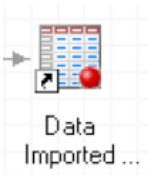
I statistik arbejder man ofte med tre datatyper: intervaldata (ting, der kan måles og vejes), ordinaldata (tal, der kan sættes i rækkefølge, men hvor afstanden mellem hvert trin ikke nødvendigvis er den samme), og nominel data (kategorier). Grunden til, at SAS kun har to datatyper, er, at det giver nogle friheder i forhold til datahåndteringen. Men det betyder så samtidig, at man selv skal holde tungen lige i munden i forhold til, hvilke variable man anvender som hhv. interval- eller ordinaldata i de analyser, man laver.

Datatype i SAS	**Datatype i statistisk teori**
Number	**Intervaldata (man kan beregne gennemsnit, fx gennemsnitlig løbstid i et motionsløb)**
	Ordinaldata (man kan kun rangordne, fx placering i et motionsløb)
String	**Numerisk data (man kan kun regne andele ud, fx andelen af løbere, der gennemførte motionsløbet på under 1 time)**

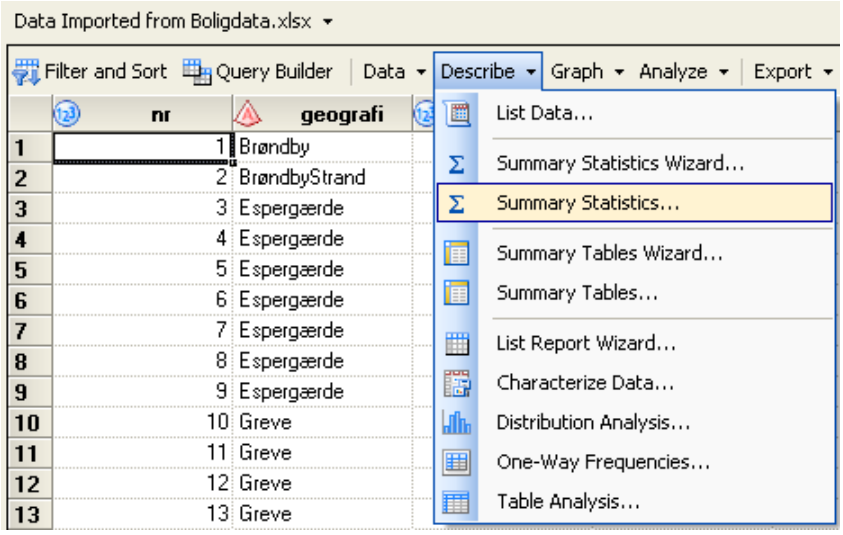
2.2.2. Beskrivende statistik

Når man får et nyt datasæt foran sig, er det altid godt at starte med at danne sig et overblik over det. Det gøres typisk ved at opsummere vores datasæt i nogle enkelte tal, som giver os overblik over hele datasættet og dets variable.

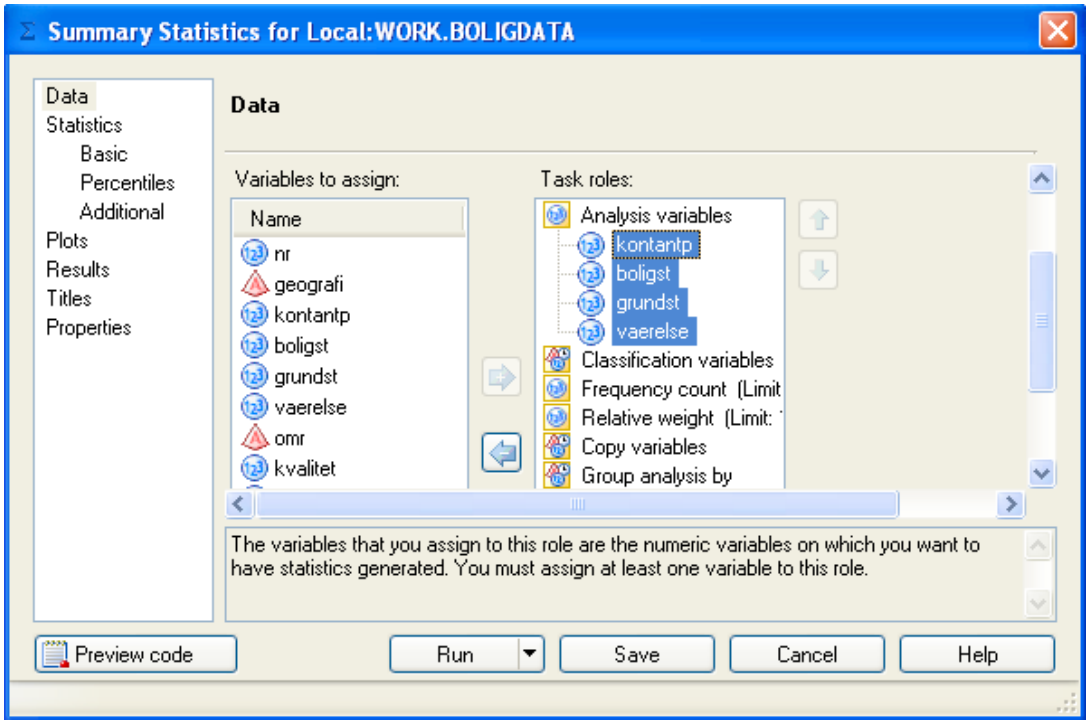
I SAS vil vi først åbne vores datasæt ved at dobbeltklikke på SAS-datasættet i *Process Flow*-diagrammet:



Herefter trykker vi på *Describe* i baren over datasættet, og vi vælger *Summary Statistics*, som kan beregne grundlæggende, beskrivende statistiske mål såsom gennemsnit, standardafvigelse, minimum- og maksimumværdier samt stikprøvestørrelsen:



Når *Summary Statistics* vælges, skal det angives, for hvilke variable man ønsker at få opsummeringer. I første omgang vælger vi alle intervalvariable, dvs. *kontantpris*, *boligst*, *grundst* og *vaerelse*:[1]

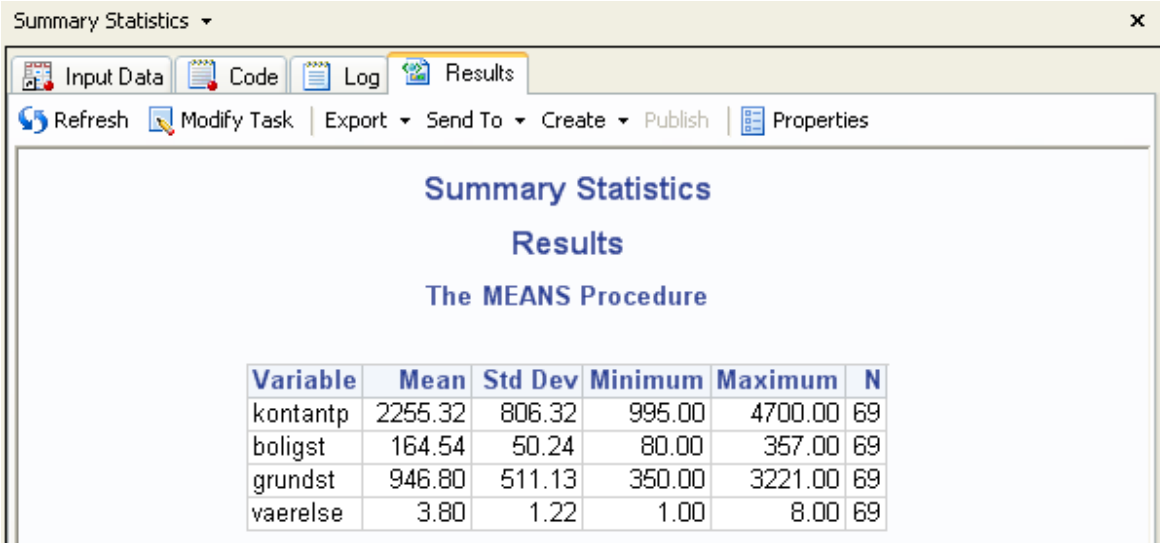


1. Variablene *kvalitet* og *kvalitetgr* er begge ordinale, da afstanden mellem en *rating* på 1 og 2 ikke nødvendigvis er den samme som afstanden mellem 4 og 5, da det jo er en subjektiv vurdering.

Under *Basics* kan vi specificere, hvilke beskrivende statistikker vi gerne vil have med på vores output (fx *Mean*, *Standard deviation* etc.), samt antal decimaler vi gerne vil have på vores tal. Vælg 2 decimaler. Resten behøver du ikke ændre på:

Tryk derefter *Run*, og følgende output kommer frem. Du skal måske lige vente et par sekunder, og mens du gør det, kan du nederst til venstre i skærmbilledet se status på SAS' beregninger:

Derefter fremkommer en fane, der hedder *Results* med vores output fra analysen. I outputtet nedenfor kan du aflæse grundlæggende beskrivende statistik om talvariablene:



Variable	Mean	Std Dev	Minimum	Maximum	N
kontantp	2255.32	806.32	995.00	4700.00	69
boligst	164.54	50.24	80.00	357.00	69
grundst	946.80	511.13	350.00	3221.00	69
vaerelse	3.80	1.22	1.00	8.00	69

Af outputtet ovenfor fremgår det bl.a., at gennemsnitsprisen på de 69 boliger er 2.255 mio. kr., og at den gennemsnitlige boligstørrelse på disse 69 boliger er 164,5 m^2. Prøv at se, om du selv kan fortolke de øvrige gennemsnitsværdier på samme måde.2

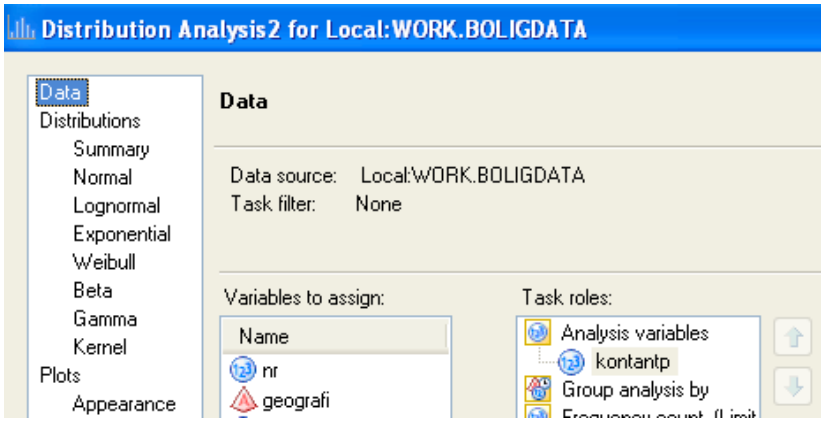
2.2.3. Visualisér data og tjek, om en variabel er normalfordelt

Ofte vil man gerne bruge statistik til at konkludere noget generelt ud over bare at beskrive sin stikprøve. Fx kan det jo være nærliggende at prøve, om man kan konkludere noget generelt om boligmarkedet for huse i Københavnsområdet i år 2000 ud fra vores datasæt. At generalisere fra en stikprøve virker måske lidt risikabelt, men det er ikke desto mindre lige præcis det, som er statistikkens hjemmebane. Tit er det enten for dyrt eller helt umuligt at undersøge alt.

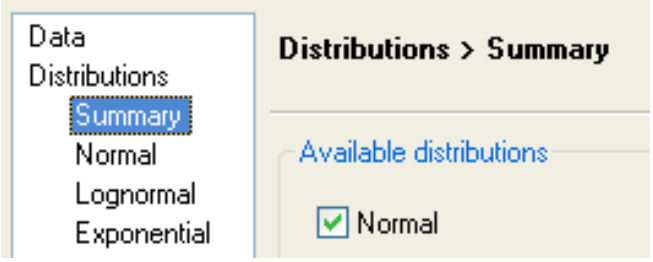
2. Grundstørrelse på ca. 950 m^2 og gennemsnitligt 3,8 værelser.

Histogram

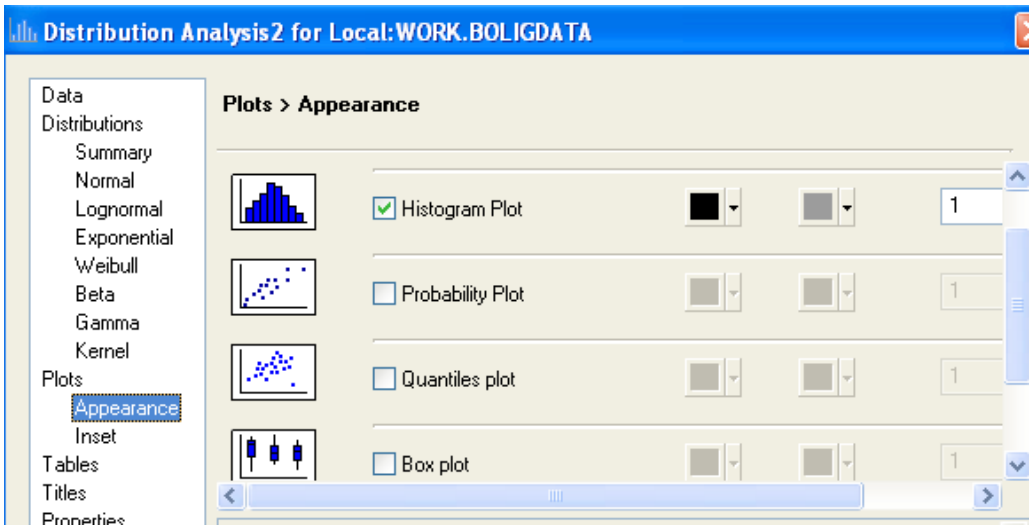
Histogrammer er et af de hyppigst anvendte værktøjer til at visualisere data. En forudsætning for mange statistiske tests er, at vores intervalvariable er normalfordelte. Det kan man også let undersøge ved at lave et histogram, hvor normalfordelt data skal være nogenlunde "klokkeformet". Man laver et histogram via dropdown-menuen *Tasks* > *Describe* > *Distribution Analysis*… Her trækkes fx variablen *kontantp* over under *Analysis variables*-punktet, som det er gjort nedenfor:



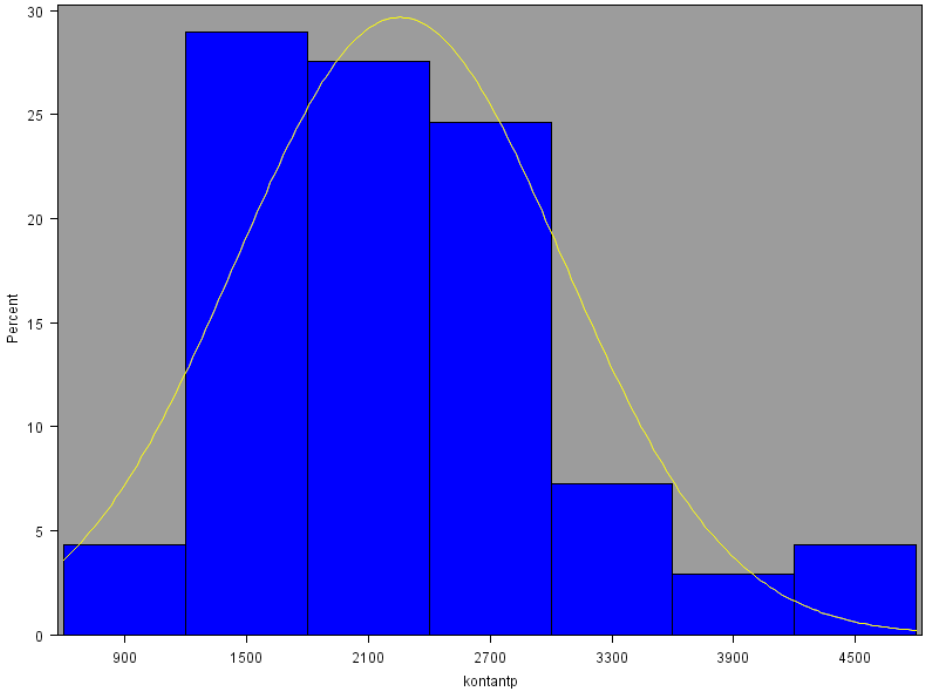
Klik herefter på *Distributions* ude til venstre i dialogboksen og sæt flueben ved *Normal*. Dette giver en normalfordelingskurve i histogrammet (når du ser normalfordelingskurven, giver det mening, hvad der menes med "klokkeformet"):



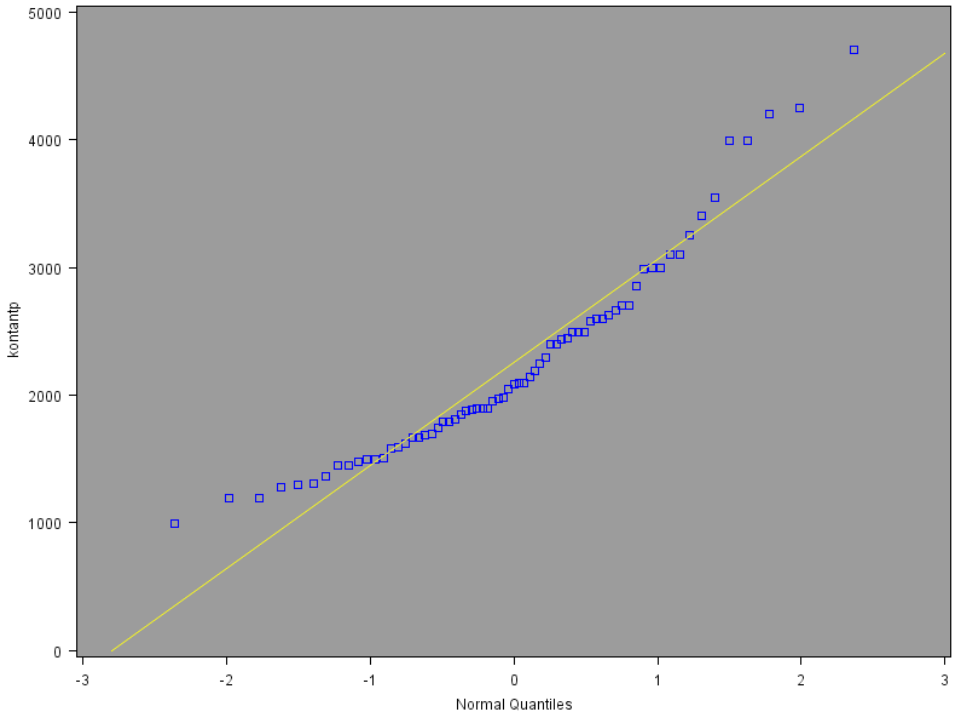
Derefter klikkes på *Plots* ude til venstre, og et histogram vælges:



Klik nu *Run*, og der kommer et output frem, og nederst i det finder du et histogram ligesom det nedenfor. Det viser, at boligpriserne er nogenlunde normalfordelte, men ikke helt perfekt. Der er fx en tendens til, at der er flere meget dyre huse, end der skulle være, hvis data var helt normalfordelt.



I dialogboksen ovenfor kunne man også have valgt et *Probability Plot* (PP plot), et *Quantiles Plot* (QQ plot) eller et *Box Plot*. PP og QQ plots viser grundlæggende det samme som histogrammet (Et *Box Plot* vender vi tilbage til senere). Nedenfor er vist et QQ plot. For at data er normalfordelt, skal den ligge tæt op ad den skrå linje



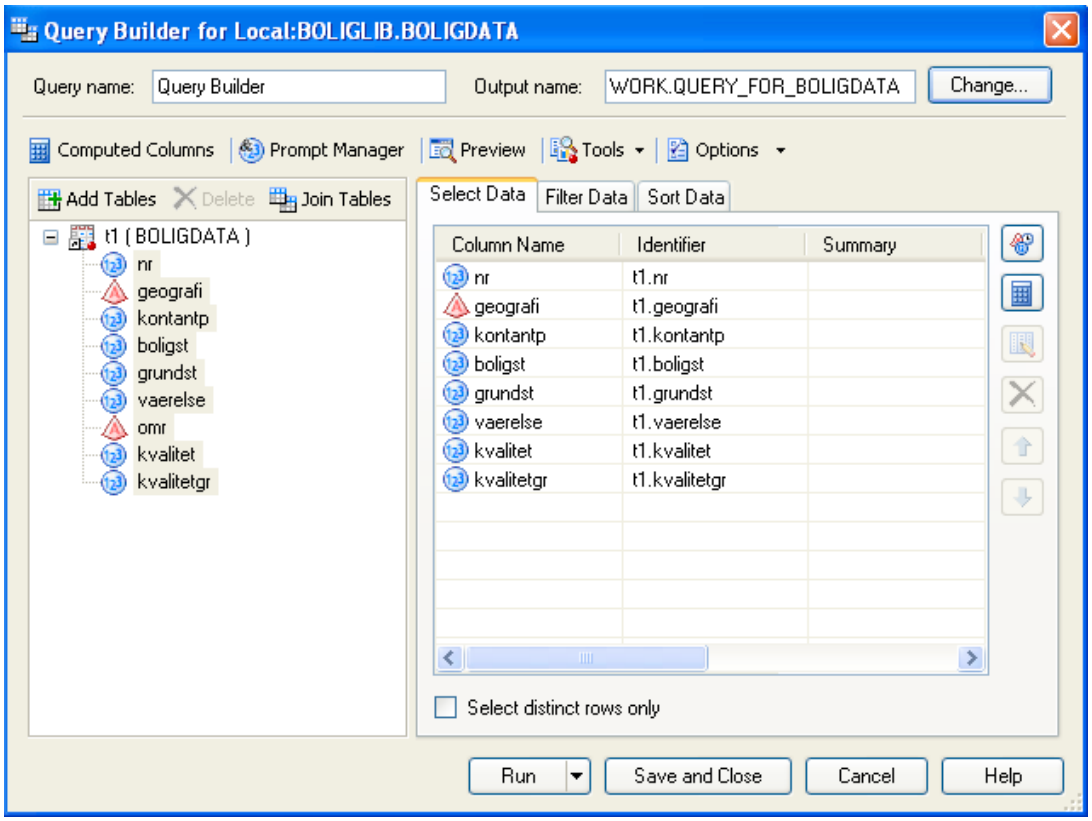
Som det ses af QQ-plottet, er der måske lidt problemer med, hvorvidt huspriserne er normalfordelte – særligt med de laveste af de højeste priser. Det var faktisk det samme, man kunne se på histogrammet tidligere, hvor der også er en tendens til, at der særligt i den øvre ende af prisskalaen er huse, der er så dyre, at normalfordelingen ikke passer helt perfekt. Man siger, at boligpriserne er højreskæve, hvilket bare betyder, at der er lidt flere observationer til højre på histogrammet, end man ville forvente, hvis observationerne fulgte normalfordelingskurven.

2.3. Tilpas dit datasæt til de analyser, du gerne vil lave

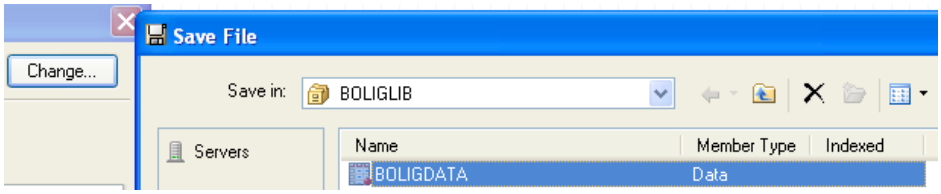
Ofte har man behov for at ændre lidt i sit datasæt, før man kan lave sine analyser. Man kalder dette 'datamanipulation', og det har altså ikke noget at gøre med at snyde med dataene. Det, man gør, er fx at tage en variabel og gruppere den. Det er fx allerede gjort i vores boligdatasæt med variablen *kvalitetgr*, som forenkler kvalitetsvurderingen af boligerne fra fem grupper til tre (1=lav, 2=middel, 3=høj). Vi vil dog give grupperne navne i stedet for numre bare for at vise, hvordan det gøres.

Når man ændrer i sit datasæt i SAS, så ændrer man ikke direkte i det originale datasæt, men man laver derimod en såkaldt *Query* i dialogboksen *Query Builder*. Dvs. man laver et såkaldt *dataudtræk* fra sit oprindelige datasæt, og så tilføjer man nogle beregninger og rekodninger baseret på de eksisterende variable. Dette resulterer i et nyt datasæt, som man så kan lave sine ønskede analyser ud fra. Det er ret smart, for på denne måde ændrer man ikke direkte i sit oprindelige datasæt, og man kan lave mange forskellige *dataudtræk* fra det samme originale og urørte datasæt. Og skulle det ske, at man havde brug for at opdatere noget i det originale datasæt, så bliver alle *dataudtræk*kene og de afledte analyser automatisk opdateret med ét klik. Det er altså virkelig smart – både for studerende, forskere og virksomheder, som gerne vil have fleksibilitet og datasikkerhed samtidig.

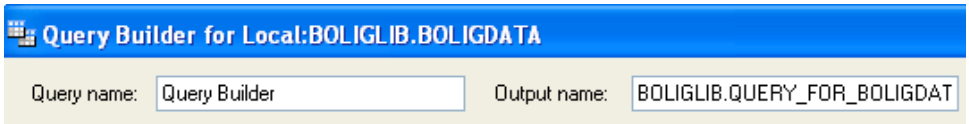
Når man laver et *dataudtræk*, starter man i *Query Builder* som nedenfor, og hvis man ønsker at bevare alle de originale variable, kan man starte med at markere alle variablene og trække dem over til højre. Det er gjort på billedet nedenfor, hvor alle variable bortset fra *omr*-variablen er trukket over:



Vi kan også lige specificere, at vores *Query* skal gemmes som et nyt datasæt i vores *library*. Det gøres ved at trykke på *Change* i øverste højre hjørne af dialogboksen ovenfor og vælge *Libraries* > *BOLIGLIB* og derefter trykke *Save* (SAS foreslår selv at kalde datasættet for QUERY_FOR_BOLIG*DATA*, hvilket er fint):



Derved bliver vores nye datasæt fra denne *Query* gemt i vores *library*, som det ses til højre nedenfor:

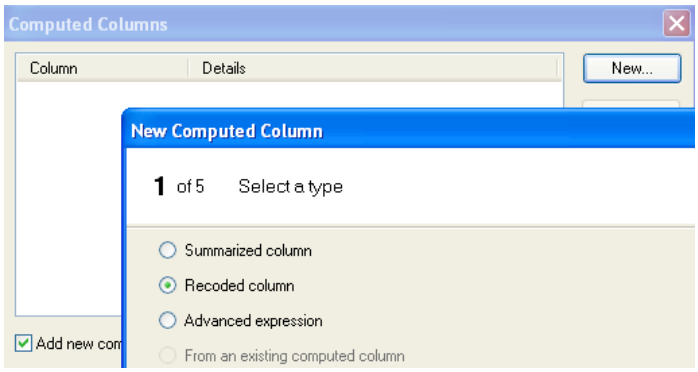


Hvis man trykkede *Run* nu, ville der blive oprettet et nyt datasæt som et *dataudtræk* fra det oprindelige datasæt. Det nye datasæt ville være helt identisk med det originale, bortset fra at *omr*-variablen ville være udeladt. Det ville jo være ret uinteressant at lave et identisk datasæt, så for at vise forskellige muligheder prøver vi lige at lave lidt tilføjelser og ændringer i vores *dataudtræk*. Disse tre eksempler er vist i de tre underafsnit. Vi vil prøve at rekode områdevariablen fra 1 og 2 til *Nord* og *Syd* (underafsnit 2.3.1) og på samme måde lave en dummyvariabel (2.3.2). Vi vil også gruppere *kontantpris*variablen i tre grupper: *lav*, *middel* og *høj* (2.3.3) og beregne kvadratmeterprisen for boligerne som en ny variabel i vores nye datasæt (2.3.4). Sluttelig vil vi prøve at lave en logaritmisk transformering af en variabel (2.3.5).

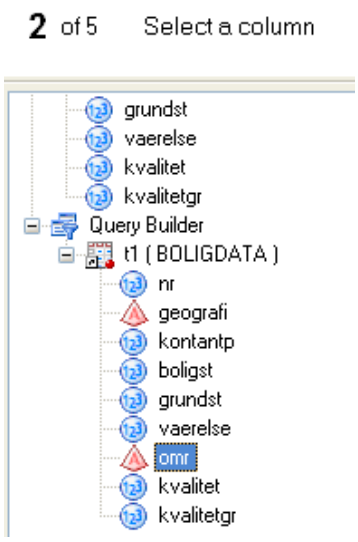
2.3.1. Rekodning af en variabel

Først vil vi ændre tallene *1* og *2* for områdevariablen til hhv. *Nord* og *Syd*.

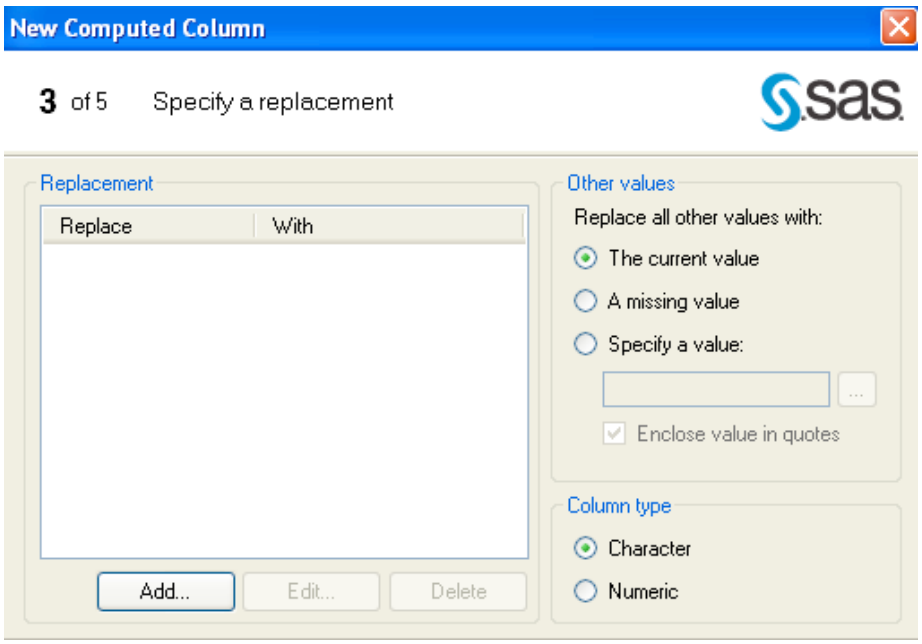
Vi trykker derfor på *Computed Columns* øverst til venstre i figuren ovenfor. Herefter trykkes der på *New…* og vælges *Recoded column* som vist nedenfor:



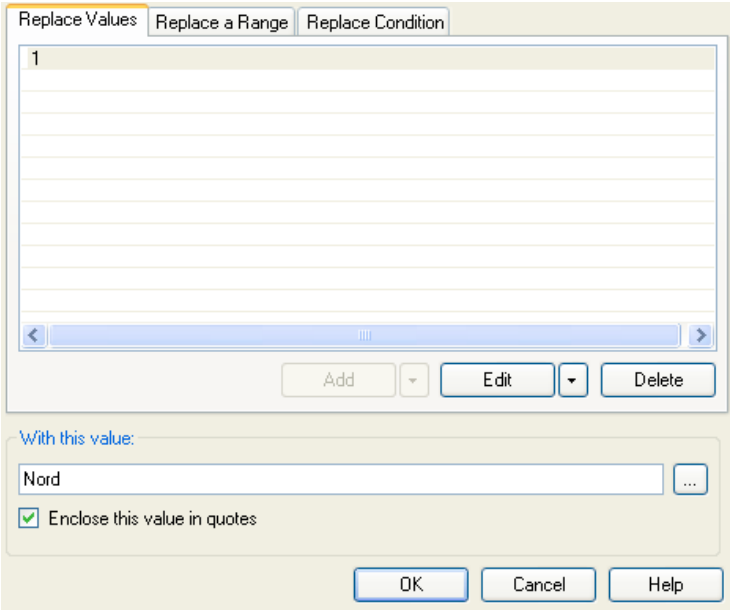
Da det er områdevariablen, vi først vil rekode, vælges denne som vist nedenfor. Man skal lige scrolle lidt ned for at kunne vælge den, for *omr* er ikke med i de øverste variable, da vi jo ikke har specificeret, at vi gerne vil have den med i det nye datasæt. Tryk derefter på *Next*:



Herefter trykkes der på *Add…* i dialogboksen nedenfor for at tilføje en omkodning:



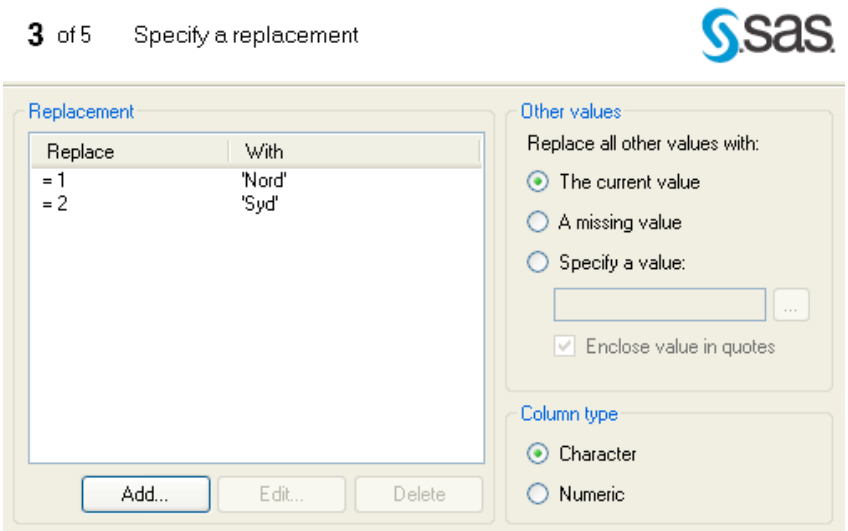
På den nye dialogboks, der kommer op, trykkes igen på den nye *Add*-knap, og derefter indtastes værdien *1* i det linjerede felt, og nederst skrives *Nord*. Dette vil erstatte område *1* med det mere sigende *Nord* i vores nye datasæt.



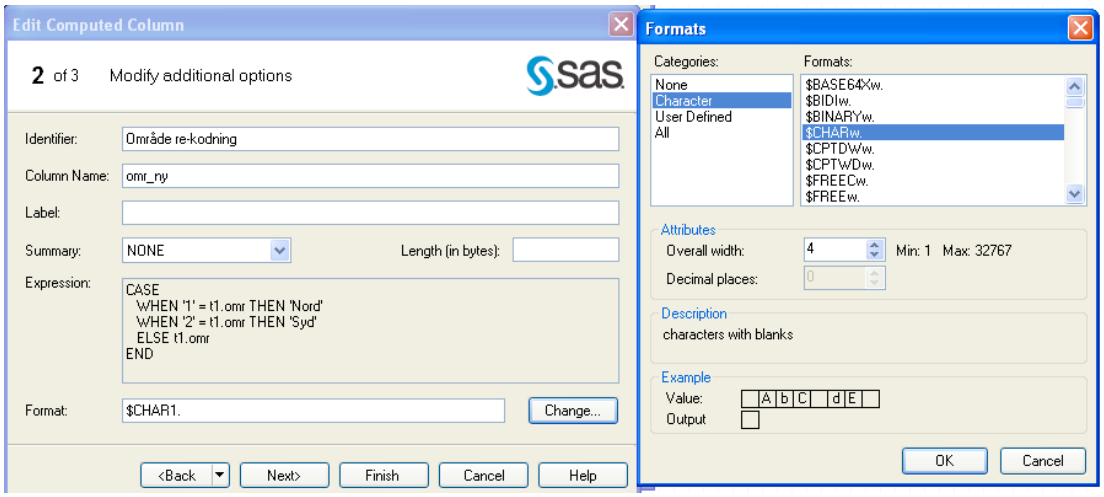
Det er vigtigt, at der er sat flueben under der, hvor *Nord* står, da det angiver, at denne nye rekodede variabel er en bogstavsvariabel og ikke en talvariabel, som den gamle variabel *omr* var det.

Tryk derefter på *OK*, og rekodningen fremgår af dialogboksen.

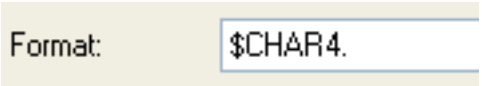
Proceduren gentages for 2 og *Syd* (på helt samme måde, så det er ikke vist igen). Når begge rekodninger er lavet, ser det ud som nedenfor:



Herefter trykkes på *Next*, og vores *Identifier* for rekodningsindtastningerne samt *Column Name* for den nye variabel navngives som nedenfor. Vi kalder *Identifier* for '*Område re-kodning*', og *Column Name* for '*omr_ny*':

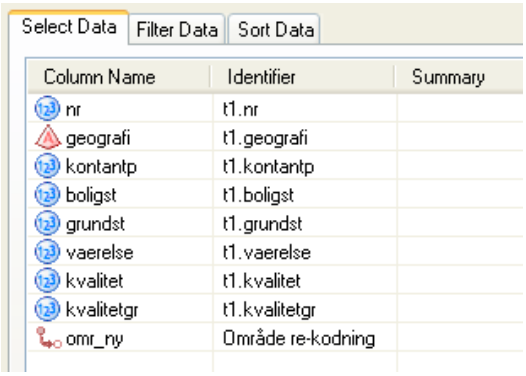


Her kommer lige en lille sidebemærkning om valgmuligheden *Format* ovenfor (der hvor der nu som standard er valgt *$CHAR1.*). SAS er ret avanceret, når det kommer til at håndtere forskellige tekst- og nummerformater. Grunden til, at det er smart, er, at man kan styre, hvor meget dataplads hver celle i et datasæt må fylde. Det betyder ikke noget i små datasæt som vores, men hvis man arbejder professionelt med kæmpe datasæt – fx i forbindelse med aktieanalyser, medicinske forskningsprojekter eller beregninger i et forsikringsselskab – så betyder det faktisk en hel del for, hvor hurtigt beregningerne kører. Glem alt om at prøve at huske disse formater – man finder ud af, hvad man skal bruge, når man får brug for det. Men her og nu kan du se, at der står *$CHAR1.* ovenfor, og det et character (bogstav)-format, hvilket er fint. 1-tallet betyder dog, at der kun er én dataplads, så derfor vil *syd* blive til **s** og *nord* til *n*. Det duer ikke, så vi skal lige have ændret det til flere datapladser. Tryk derfor *Change…* i billedet ovenfor og ændr *Attributes*: *Overall width* til 4 og tryk derefter *OK*. Nu kan du se, at formatet er ændret som nedenfor:

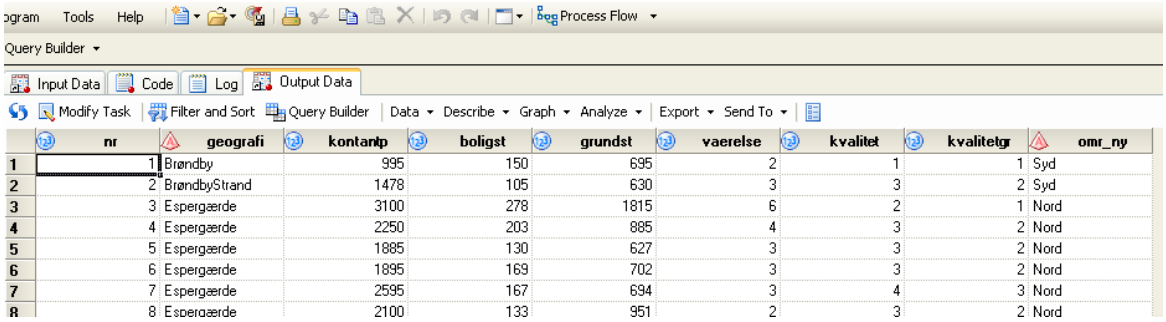


Tryk herefter på *Finish* > *Close*

Den nye rekodede variabel står nu nederst på listen over de variable, der vil være i vores nye datasæt:



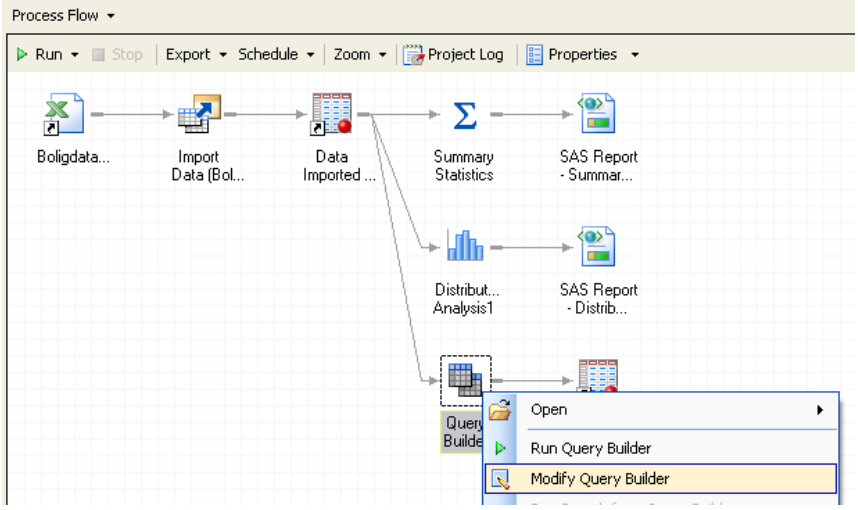
Hvis du trykker *Run* nu, kan du se den nye *omr_ny*-variabel i vores nye datasæt:



	nr	geografi	kontantp	boligst	grundst	vaerelse	kvalitet	kvalitetgr	omr_ny
1	1	Brøndby	995	150	695	2	1	1	Syd
2	2	BrøndbyStrand	1478	105	630	3	3	2	Syd
3	3	Espergærde	3100	278	1815	6	2	1	Nord
4	4	Espergærde	2250	203	885	4	3	2	Nord
5	5	Espergærde	1885	130	627	3	3	2	Nord
6	6	Espergærde	1895	169	702	3	3	2	Nord
7	7	Espergærde	2595	167	694	3	4	3	Nord
8	8	Espergærde	2100	133	951	2	3	2	Nord

Vi er dog ikke helt færdige med ændringer og tilføjelser til vores nye datasæt. Så derfor kan vi gå tilbage og ændre i vores *Query* ved at trykke på knappen *Modify Task*, som du ser lige over datasættet i venstre side af billedet ovenfor. På den måde kan vi komme tilbage til den *task*, som vi netop har udført (vores *Query*), og korrigere / lave tilføjelser.

Det samme kunne man gøre ved at trykke på *Process Flow*-knappen (allerøverst midt for i billedet ovenfor) og derefter højreklikke på *Query Builder*-ikonet og vælge Modify *Query Builder*, som det er vist nedenfor:



Med denne lille "SAS-sightseeing" er vi nu klar til at gå videre.

2.3.2. Lav en dummyvariabel

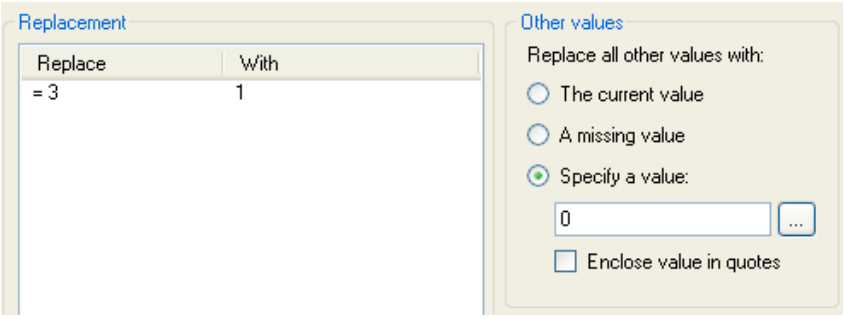
En dummyvariabel er en variabel, som enten er 0 eller 1. Her vil vi prøve at lave en dummyvariabel, som har værdien 1 for alle de boliger, som er i kvalitetsgruppe 3 (dvs. de boliger med den højeste herlighedsværdi) og værdien 0 for de andre kvalitetsgrupper.

Man laver en dummyvariabel efter samme fremgangsmåde som i eksemplet ovenfor (underafsnit 2.3.1). Det går bare lidt hurtigere, når man kun behøver at indtaste én betingelse i kodning. Nedenfor opsummeres de forskellige steps, man skal igennem.

I samme *Query Builder* som i forrige underafsnit vælges *Computed Columns* > *New…* > Recoded Column > *Next*.

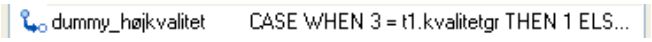
Herefter vælges variablen *kvalitetgr* (det er ligegyldigt, om du vælger den i den øverste eller nederste tabel). Vælg nu *Add… > Add* og skriv *3* som den værdi, der skal rekodes. I feltet *With this value* skrives *1* for at angive, at værdien 3 (høj herlighedsværdi) skal være 1 i vores dummyvariabel. Tryk nu på *OK*.

Nu mangler vi bare at specificere, at alle andre værdier af variablen *kvalitetgr* skal være 0. Det gøres til højre i billedet nedenfor ved under Other values at vælge *Specify a value* og skrive *0*. Med disse indstillinger får alle boliger med høj herlighedsværdi et 1-tal i den nye variabel, og alle øvrige boliger får et 0. Indstillingerne ser ud som nedenfor:

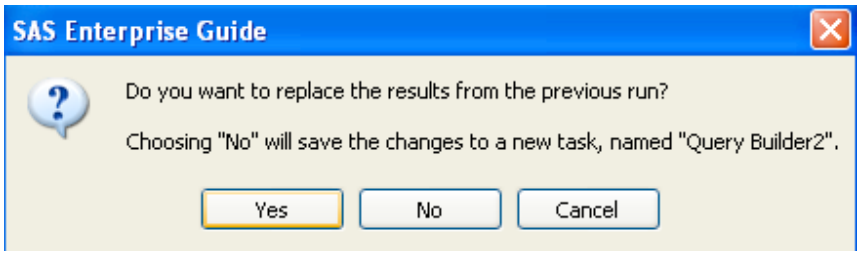


Tryk nu *Next* og navngiv derefter den nye variabel til fx *dummy_højkvalitet* (du kan bare bruge det samme navn til både *Identifier* og *Column Name*).

Tryk *Finish*, og den nye dummyvariabel fremgår af listen *Computed Columns*:



Tryk *Close*, og vi er klar til at gå videre. Hvis du trykker *Run* for at se resultatet (det behøver du ikke at gøre), kan du bare sige *Yes* til spørgsmålet, om du vil overskrive det tidligere resultat. Ellers kommer der et nyt datasæt, og det er jo ikke meningen:

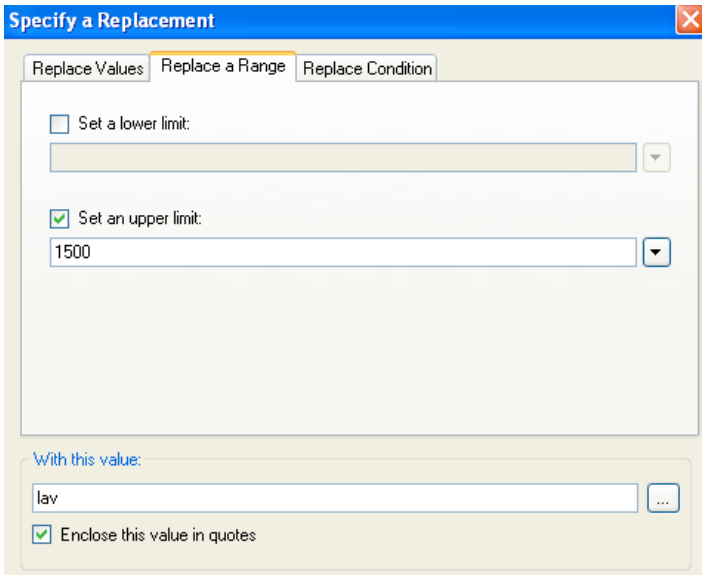


Skulle du ved en fejl komme til at sige No, er det ikke værre end, at du bare kan højreklikke på elementerne i *Process Flow*-diagrammet og vælge at slette dem. Bare husk, at det så er den gamle version, du får slettet.

2.3.3. Gruppering af en variabel

Herefter vil vi gruppere *kontantp* til en ny variabel, som enten er lav, *middel* eller høj, hvor alle huse under 1,5 mio. kr. kommer i prisgruppe lav, huse mellem 1,5 og 2,5 mio. kr. kommer i prisgruppe middel, og huse over 2,5 mio. kr. kommer i prisgruppe høj.

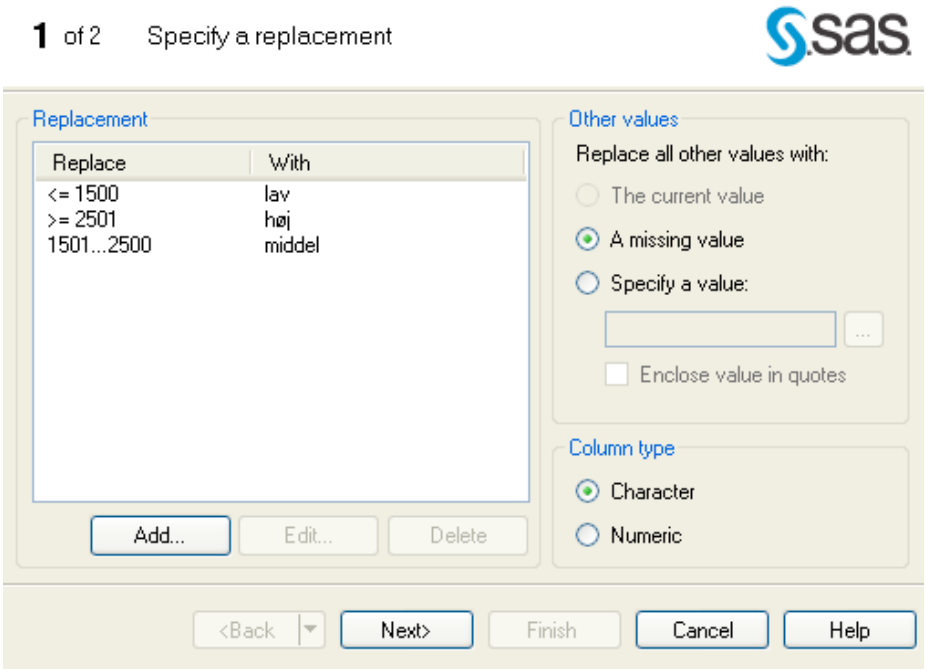
Dette gøres på næsten samme måde som rekodningen, så du skal igennem de samme indledende steps som ovenfor (start med samme *Query* som sidst og tryk på Computed Column-knappen, og herefter skal du vælge *kontantp* på listen over variablen). Det eneste, der her er lidt anderledes, er, at man efter at have trykket *Add…* første gang skal vælge *Replace a Range*, som det er gjort nedenfor:



Som det ses ovenfor, er første gruppe *lav* defineret ved værdier af *kontantp* under 1500 (altså 1,5 mio. kr.). Denne gang skal du selv huske at sætte flueben ud for *Enclose this value in quotes* under der, hvor du har skrevet '*lav*'.

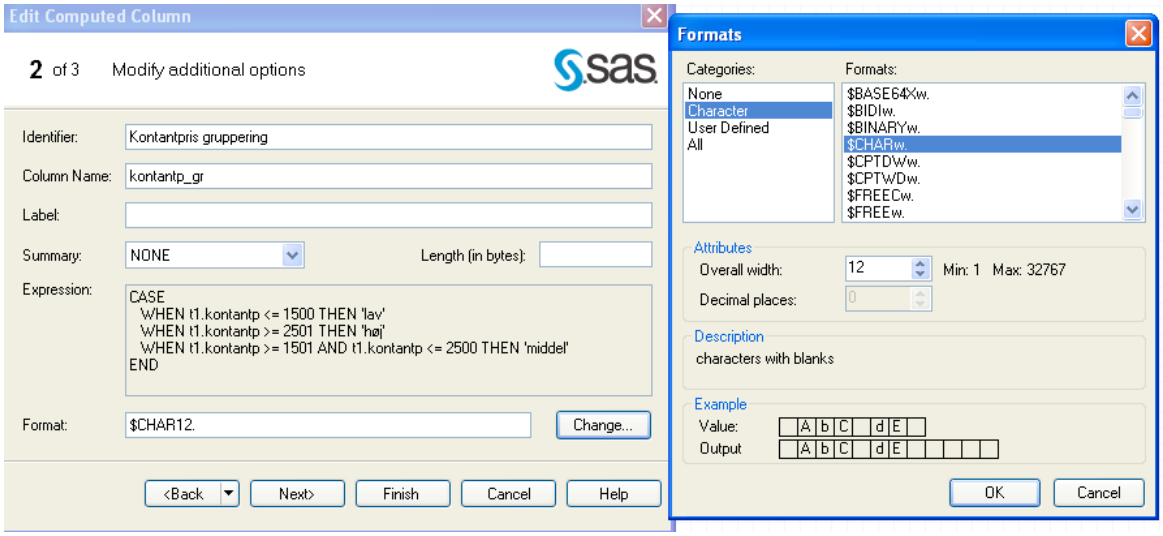
Tryk herefter *OK* og gentag for prisgruppe *middel* (*lower limit*: 1501 og *upper limit* 2500) og til sidst for prisgruppen *høj* (*lower limit*: 2501 og ingen øvre grænse).

Vi mangler bare lige at angive, at den nye variabel er en *Character*-variabel, fordi tallene for boligpriserne erstattes af en kategori. Dette angives nederst til højre i billedet nedenfor:



Tryk herefter på *Next* og navngiv den nye variabel som *kontantp_gr* og giv dine indstillinger et navn (*Identifier*), fx '*Kontantpris gruppering*'.

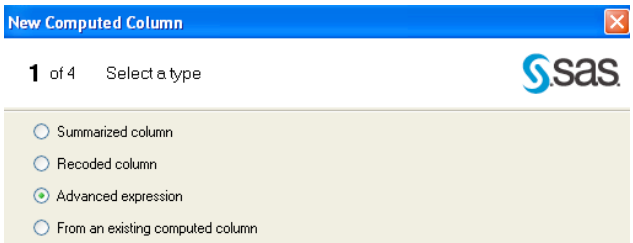
Denne gang er du nødt til at ændre formatet til et tekstformat. Tryk derfor *Change…* til højre for *Format*-linjen i det venstre billede ovenfor og vælg derefter *Character* > $CHARw. og tryk så *OK*. Nu er den nye områdevariabels format ændret til et tekstformat med plads til 12 bogstaver. Du kan også bare direkte skrive $*CHAR12.* i Format-feltet som på billedet nedenfor (husk punktummet efter 12-tallet, hvis du selv skriver det!). Hvorfor lige 12? Godt spørgsmål – 6 havde været nok ('middel' er det længste navn med 6 bogstaver), men man gider jo ikke tælle hver gang – vi har allerede sparet meget dataplads ved at tage stilling til formatet, som fx i Excel som standard er 256 datapladser! Og sparet dataplads betyder, at vores beregninger kører hurtigere:



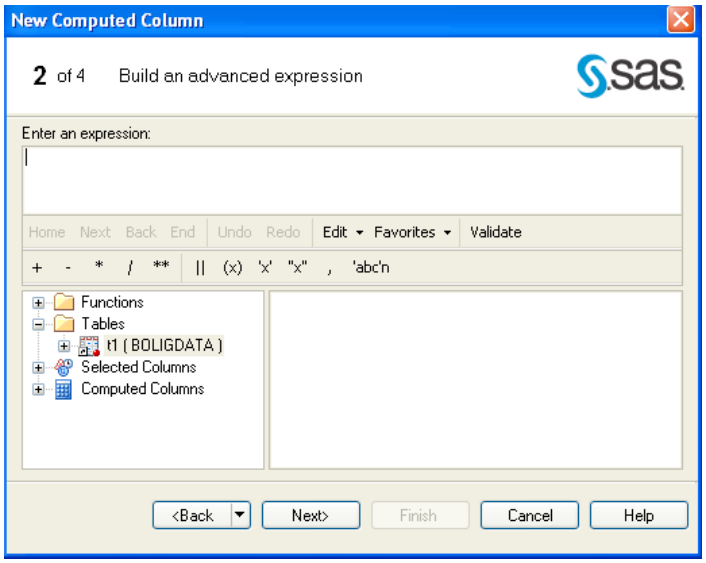
Tryk herefter på *Finish* > *Close*, og vores variabelgruppering er færdig.

2.3.4. Beregning af en ny variabel

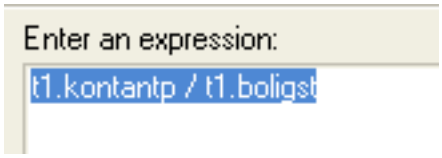
Vi vil nu beregne kvadratmeterprisen for de forskellige boliger, og kvadratmeterprisen vil så blive en del af datasættet som en ny variabel i en ny kolonne. Den beregnes som *kontantpris*en divideret med boligstørrelsen. Vi starter med at trykke på *Computed Columns* og vælger *Advanced expression* som gjort nedenfor:



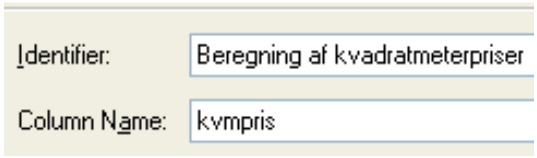
Tryk derefter på *Next*, og nedenstående formel-editor fremkommer:



Klik på det lille kryds til venstre for vores datasæt "*t1 (BOLIGDATA)*" og dobbeltklik så på variablen *kontantp*. Tryk dernæst på knappen med divider-symbolet (/) og dobbeltklik på variablen *boligst*. Din formel skulle gerne se ud som nedenstående nu:

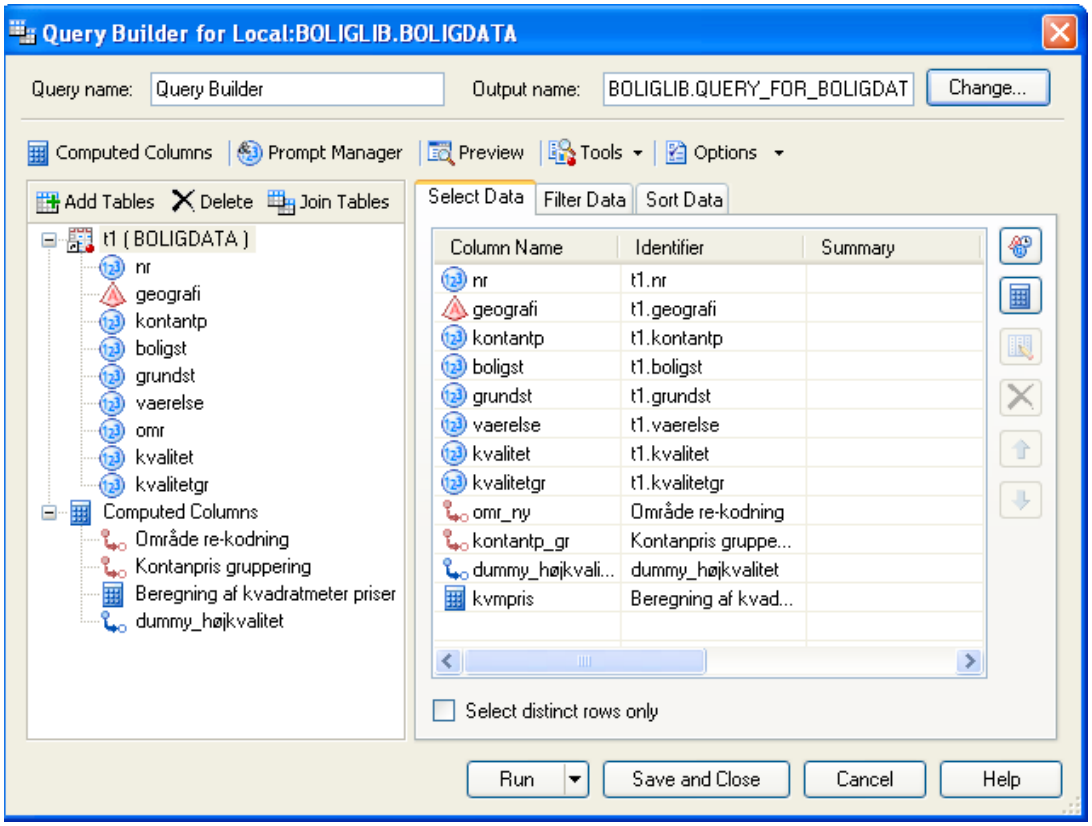


Tryk nu på *Next* og navngiv beregningen og den nye variabel som nedenfor:

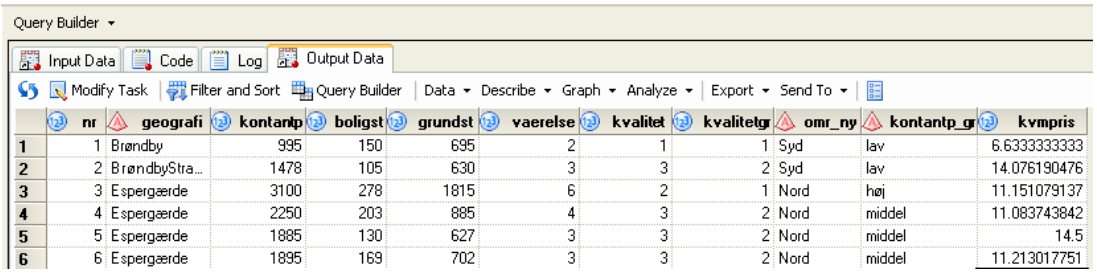


Tryk nu på *Finish* og derefter på *Close*.

Nu kan du se, at den nye beregnede variabel fremgår på listen af variable, der vil være med i det nye datasæt, som *Query Builder*'en laver:

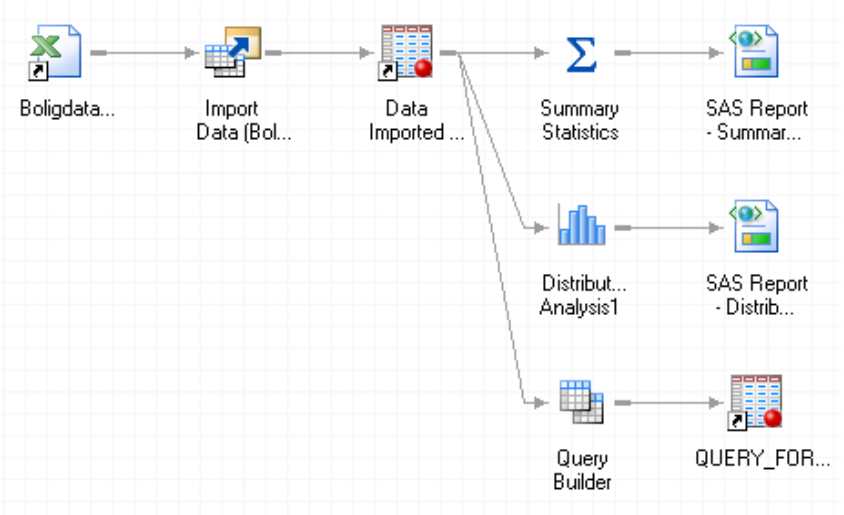


Hvis du trykker *Run* nu, så fremkommer et nyt datasæt, hvor de valgte og konstruerede variable er med (kun de første par rækker vises nedenfor):



	nr	geografi	kontantp	boligst	grundst	vaerelse	kvalitet	kvalitetgr	omr_ny	kontantp_gr	kvmpris
1	1	Brøndby	995	150	695	2	1	1	Syd	lav	6.6333333333
2	2	BrøndbyStra...	1478	105	630	3	3	2	Syd	lav	14.076190476
3	3	Espergærde	3100	278	1815	6	2	1	Nord	høj	11.151079137
4	4	Espergærde	2250	203	885	4	3	2	Nord	middel	11.083743842
5	5	Espergærde	1885	130	627	3	3	2	Nord	middel	14.5
6	6	Espergærde	1895	169	702	3	3	2	Nord	middel	11.213017751

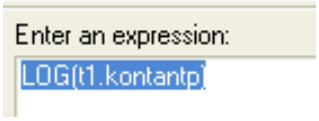
I *Process Flow*-diagrammet er det nye datasæt *Query for boligdata* også kommet frem nederst til højre. Det er det datasæt, vi vil bruge som udgangspunkt for analyserne senere i denne e-bog:



2.3.5. Transformering af en variabel

Hvis en variabel ikke er helt normalfordelt, som det fx er tilfældet for boligpriserne, så kan man vælge flere forskellige tilgange. Man kan fx prøve at 'transformere' variablen, så den bliver normalfordelt ved at tage den naturlige logaritme til variablen *ln(kontantp)*.

ln(kontantp) beregnes samme sted som beregningerne ovenfor. Formel kommer til at se sådan ud. *(LOG()* beregner den naturlige logaritme med grundtal e):

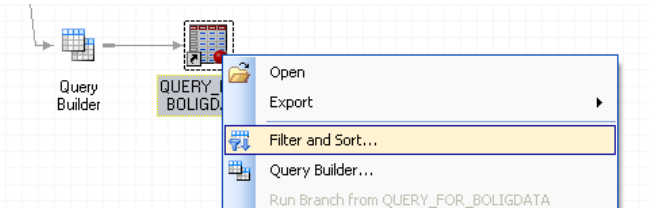


Hvis du prøver at beregne ln(kontantp), så sørg for, at du også har den originale variabel *kontantp* med over i dit nye datasæt, for det er den, vi for enkelhedens skyld vil bruge i resten af denne e-bog, selvom *ln(kontantp)* måske ville være mere korrekt.

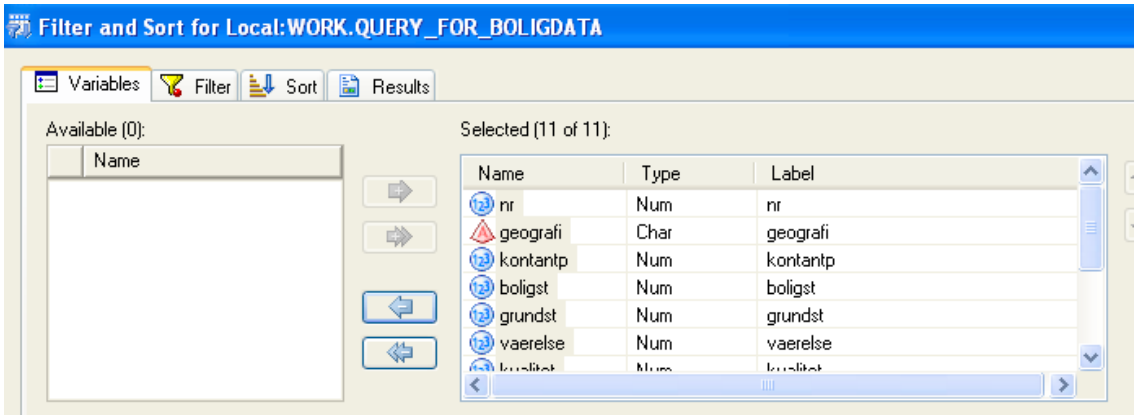
2.4. Begræns dit datasæt

Nogle gange er man kun interesseret i at analysere nogle dele af sit datasæt. Det kan fx være, at du har besluttet dig for, at du helst vil bo syd for København, da du netop har fået job i det område. Så ville det jo være smart at begrænse analysen af datasættet til kun at omfatte de boliger, der ligger i det område. Det kan let gøres i SAS ved at lægge et "filter" på datasættet.

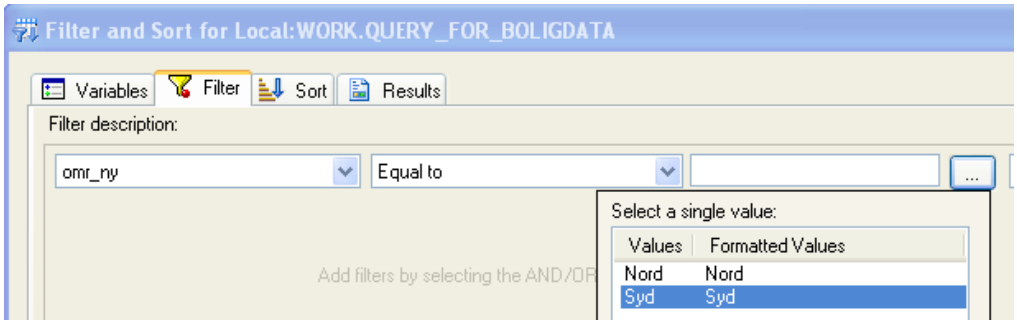
I *Process Flow*-diagrammet kan du højreklikke på det datasæt, du fik som resultat i forrige afsnit (hvis du ikke har fulgt forrige afsnit, så kan du finde datasættet på www.sasacademic.dk samme sted, som du fandt denne e-bog). Når du højreklikker på et datasæt, får du muligheden for at filtrere datasættet (*Filter and Sort…*):



Først trækker du alle variablene over til højre (brug evt. denne tast)



Derefter vælger du øverst i dialogboksen *Filter*-fanen og vælger *omr_ny*-variablen, som du angiver, skal være Equal to en given værdi. For at vælge, hvad *omr_ny* skal være equal to, trykker du på knappen ..., hvorefter valgmulighederne fremkommer. Her vælger du *Syd* som vist nedenfor (hvis man har flere/andre variable, der skal filtreres på, kan man vælge dropdown-boksen til venstre for knappen). Tryk derefter *OK*:

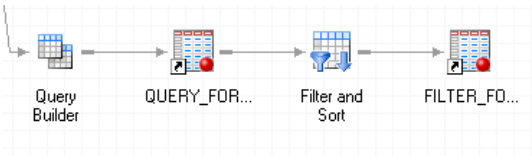


Tryk *OK* endnu en gang, og det filtrerede datasæt fremkommer, hvor kun huse syd for København indgår (kun de første rækker vises nedenfor):

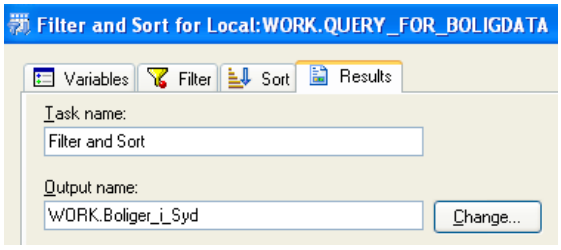
	nr	geografi	kontantp	boligst	grundst	vaerelse	kvalitet	kvalitetgr	omr_ny	kontantp_gr	kvmpris
1	1	Brøndby	995	150	695	2	1	1	Syd	lav	6.6333333333
2	2	BrøndbyStrand	1478	105	630	3	3	2	Syd	lav	14.076190476
3	10	Greve	3995	111	3221	4	4	3	Syd	høj	35.990990991
4	11	Greve	4200	179	736	6	5	3	Syd	høj	23.463687151
5	12	Greve	1595	146	756	3	2	1	Syd	middel	10.924657534
6	13	Greve	1450	144	766	3	1	1	Syd	lav	10.069444444
7	14	Greve	1295	81	850	2	2	1	Syd	lav	15.987654321
8	15	Greve	1895	165	788	3	3	2	Syd	middel	11.484848485
9	16	Greve	1495	135	810	3	2	1	Syd	lav	11.074074074

Man kunne også sortere det filtrerede datasæt – det kunne fx være, at man gerne ville se de billigste huse øverst og dyreste nederst. Det gøres via den tredje fane i dialogboksen fra *Filter* and Sort… (er ikke gjort her).

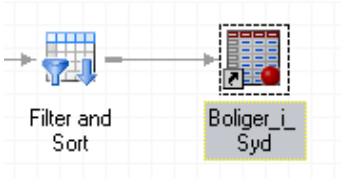
Det nye filtrerede datasæt hedder "*Filter_for_query_for_boligdata*" og fremgår af *Process Flow*-diagrammet:



Det navn er jo lidt klodset, så man kan ændre det ved at højreklikke på *Filter* and Sort og vælge *Modify Filter and Sort*. Der kan man øverst vælge fanen *Results*, hvor man har mulighed for at ændre navnet på det producerede datasæt. Her er det kaldet *Work.Boliger_i_Syd*, så *dataudtræk*ket ikke gemmes permanent, men i det midlertidige *library*, der hedder *WORK*:



Tryk derefter på *Save* > *OK*, og navnet er nu ændret:



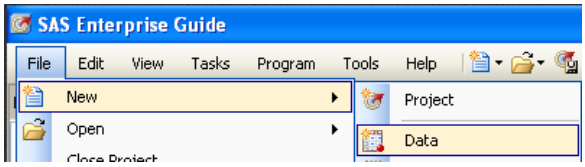
3. DET MEST ANVENDTE STATISTIK I PRAKSIS

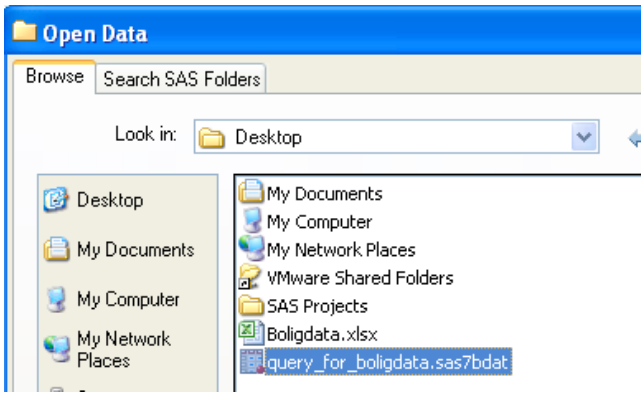
Noget af det, man ofte vil i statistik, er at teste, om nogle antagelser holder i praksis. Man kalder det hypotesetests. En statistisk hypotese formuleres som en påstand, som man enten kan forkaste eller ikke forkaste. Hvis man forkaster hypotesen, konkluderer man, at den var forkert. Hvis man ikke forkaster hypotesen, så siger man, at man ikke kan sige, at den var forkert – men altså heller ikke, at den er entydigt rigtig.

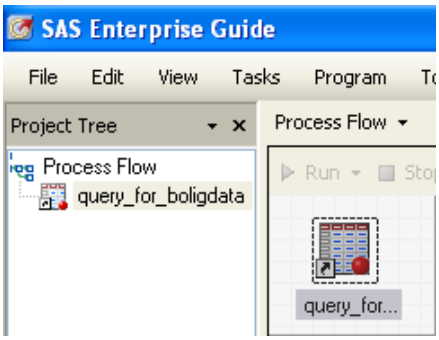
Ud over hypotesetests er den anden grundlæggende statistiske teknik at lave konfidensintervaller, og dem kommer dette kapitel også ind på.

En gruppe af hypoteser beskæftiger sig med test på gennemsnit, og blandt dem finder man t-tests og *ANOVA*-tests (afsnit 3.1). Hvis data er nominel (grupper), kan man ikke beregne gennemsnit. Derfor beregner man tit andele/procenter i stedet, og man kalder disse tests for *test på andele* eller *test på datatabeller* (begge dele behandles i afsnit 3.2). En anden ofte brugt teknik er regressionsanalyse. Fx giver det god mening at antage, at huspriser til en vis grad kan forklares af boligstørrelsen, og dette kan man lave en model for med en regressionsanalyse (afsnit 3.3). Nogle gange vil man gerne teste noget i forhold til en given statistisk fordeling. Sådanne tests kaldes fordelingstests, og denne e-bog viser med et par eksempler, hvordan disse laves i SAS for bl.a. binomialfordelingen og normalfordelingen (afsnit 3.4).

OBS: I resten af denne e-bog anvendes datasættet 'Query for Boligdata' til analyserne. Det er det datasæt, der er resultatet af afsnit 2.3. Datasættet 'Query for Boligdata' kan også downloades fra www.sasacademic.dk (samme sted, som du kan downloade denne e-bog) og kan åbnes direkte i SAS via 'File > New > Data' som vist nedenfor:





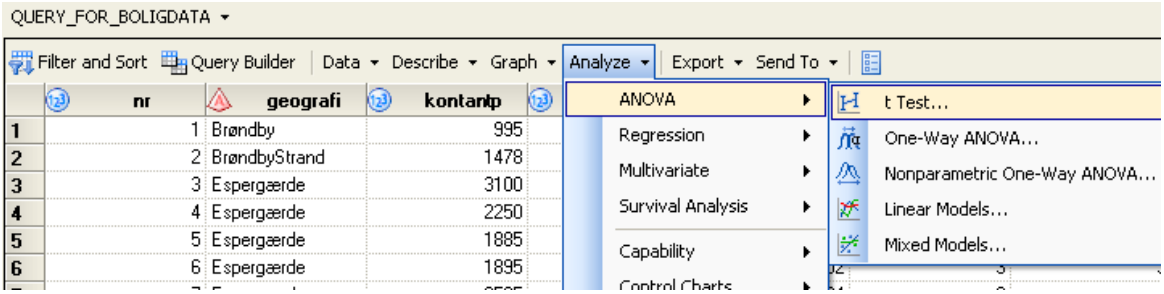


3.1. Test på gennemsnit (means)

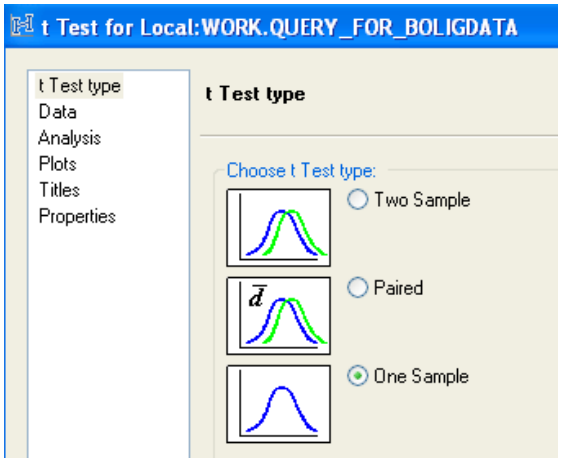
Dette afsnit viser først, hvordan man tester ét gennemsnit (underafsnit 3.1.1) og derefter test på forskellen mellem to uafhængige gennemsnit (3.1.2). Test på parvise stikprøver beskrives også (3.1.3). Når 3 eller flere grupper skal sammenlignes, anvendes *ANOVA*-tests (3.1.4), og hvis to faktorer skal testes samtidig, anvendes Two-Factor *ANOVA* (3.1.5). De enkelte tests introduceres yderligere i begyndelsen af deres egne underafsnit.

3.1.1.Test for ét gennemsnit (One sample t-test)

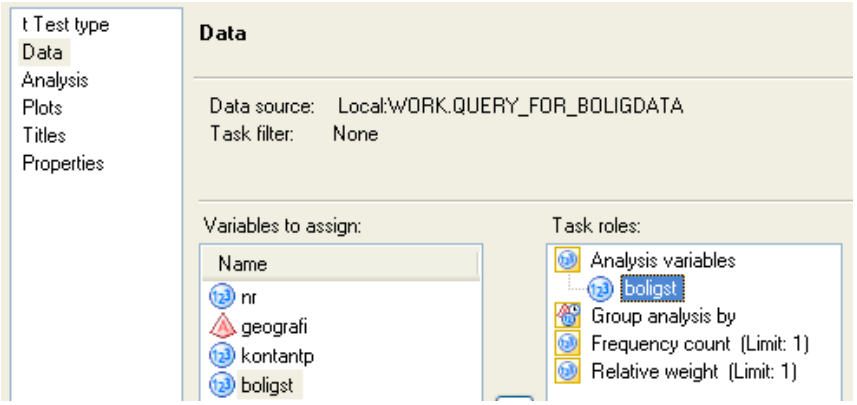
Lad os antage, at du har skimmet boligdatasættet igennem og har en fornemmelse af, at husenes størrelse gennemsnitlig er på 150 m². Denne antagelse kan du let teste i SAS. Du starter i *Process Flow*-diagrammet og dobbeltklikker på det datasæt, du gerne vil arbejde med – i vores tilfælde datasættet *Query* for *Boligdata*. Her vælger du menuen *Analyse* > *ANOVA* > *t Test*… som vist på billedet nedenfor:



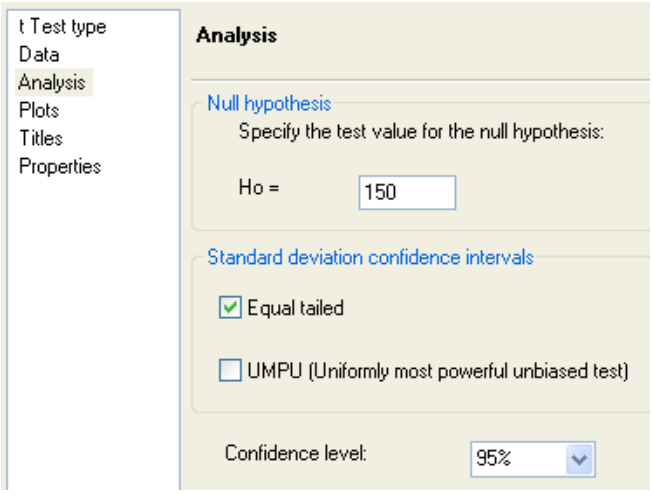
Vælg *One Sample* som testtype:



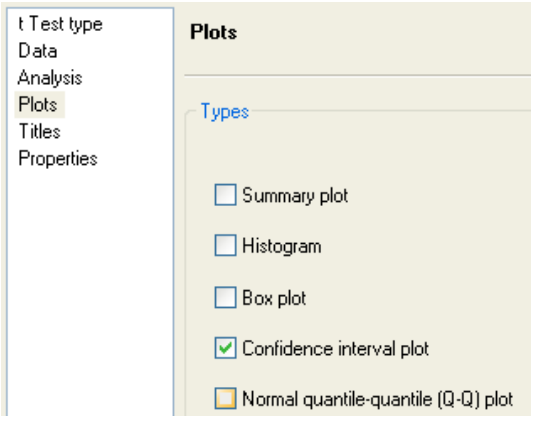
Klik derefter på *Data* og flyt variablen *boligst* over som *Analysis variables*:



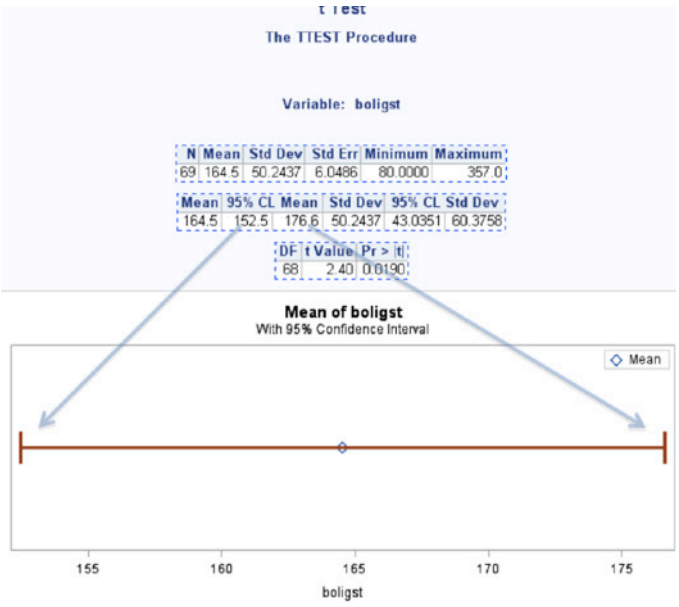
Klik derefter på *Analysis* ude til venstre og angiv, at den værdi, vi tester, er 150 (vi ville jo gerne teste, om boligerne gennemsnitlig er ca. 150 m^2). Det bliver formuleret som, at vores nulhypotese er H0: μ=150 med alternativhypotesen H1: μ. Som det fremgår på *Confidence level*, er niveauet som standard 95%, hvilket svarer til et signifikansniveau på 5% (α = 0,05).



Under *Plots* vælges *Confidence interval plot* som nedenfor:



Tryk derefter *Run*, og der kommer et output frem som nedenfor:



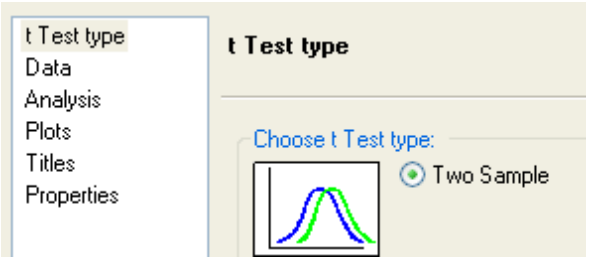
Af outputtet fremgår det, at den gennemsnitlige boligstørrelse er 164,5 (*mean*), og at p-værdien er 0,019 *(Pr>ltl)*. Da p-værdien er mindre end vores α-niveau på 0,05, forkastes H0, og vi konkluderer, at den gennemsnitlige boligstørrelse ikke er 150 m^2. Vores data indikerer endvidere, at gennemsnittet er højere end de 150m^2. 95% af konfidensintervallet for den gennemsnitlige boligstørrelse ligger jf. plottet ovenfor ca. mellem 152 m^2 og 177 m^2, så vi er 95% sikre på, at den sande gennemsnitsboligstørrelse for alle huse nord og syd for København ligger i dette interval. Mere præcist kan vi også aflæse konfidensintervallet i tabellen i outputtet, hvor der står, at den nedre grænse er 152,5, og den øvre grænse er 176,6. Hvis man vil regne det ud selv for at tjekke, at det passer, kan man sige 164,5 ± (2 · 6,0486), hvor de 6,0486 er standardafvigelsen af gennemsnittet fra outputtet *(Std.Error)*, og de 2 er en tommelfingerregel for konfidensintervallets bredde, når α=0,05 og testintervallet er tosidet.

3.1.2. Test af forskellen mellem to uafhængige gennemsnit (og varianser)

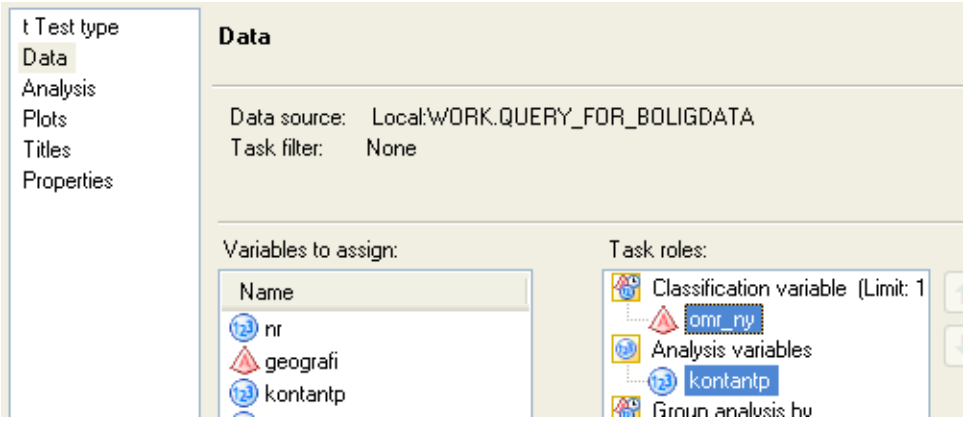
Man kunne godt få den tanke, at der måske er forskel på prisen af husene, der ligger hhv. nord og syd for København. I vores stikprøve fra boligmarkedet er der både huse fra nord og syd, så vi kan let teste, om der er forskel.

Start på samme måde som i forrige test med at åbne datasættet *Query for boligdata* og vælg *Analyze > ANOVA > t Test…*

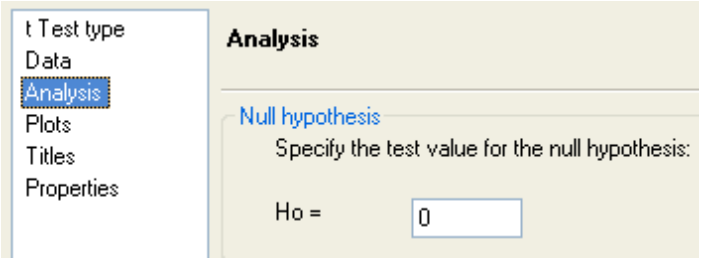
Standardindstillingen med *Two Sample* er helt fin i dette tilfælde, da vi jo ønsker at sammenligne to grupper – nord vs. syd.



Tryk derefter på *Data* ude til venstre og flyt variablen *omr_ny* over som klassifikationsvariabel og variablen *kontantp* over som analysevariabel, som det er gjort på billedet nedenfor:



Under *Analysis* ude til venstre ændrer vi ikke noget. At testværdien for analysen er 0 betyder, at vi vil undersøge, om forskellen på gennemsnitsprisen af huse i hhv. nord og syd er lig med 0 (altså at der ikke er nogen forskel på gennemsnittene). Dette er vores H_0-hypotese:



Under *Plots* ude til venstre vælger vi som i forrige eksempel *Confidence interval plot*. Tryk herefter på *Run*, og nedenstående output fremkommer:

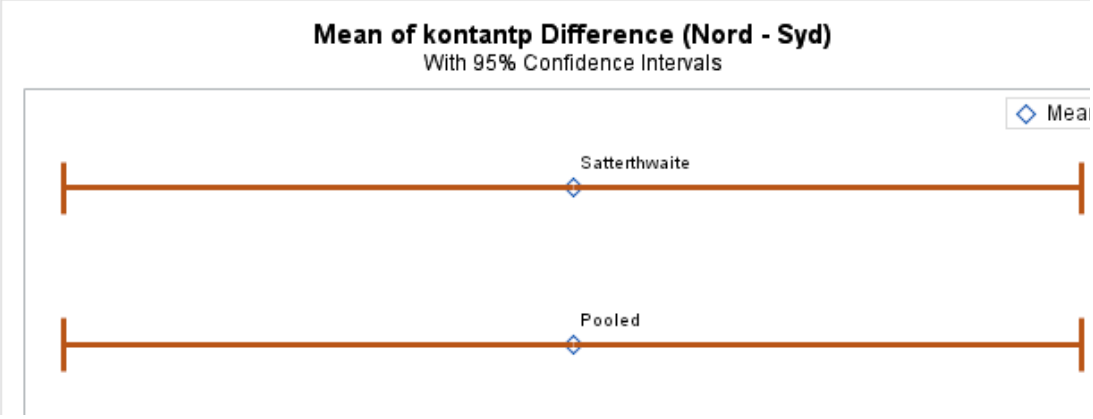
Variable: kontantp

omr_ny	N	Mean	Std Dev	Std Err	Minimum	Maximum
Nord	39	2429.1	788.5	126.3	1310.0	4700.0
Syd	30	2029.5	785.2	143.3	995.0	4200.0
Diff (1-2)		399.6	787.0	191.1		

omr_ny	Method	Mean	95% CL Mean		Std Dev	95% CL Std Dev	
Nord		2429.1	2173.5	2684.6	788.5	644.4	1016.2
Syd		2029.5	1736.3	2322.6	785.2	625.3	1055.5
Diff (1-2)	Pooled	399.6	18.0844	781.1	787.0	673.4	947.2
Diff (1-2)	Satterthwaite	399.6	17.8132	781.4			

Method	Variances	DF	t Value	Pr > |t|
Pooled	Equal	67	2.09	0.0404
Satterthwaite	Unequal	62.666	2.09	0.0405

Equality of Variances				
Method	Num DF	Den DF	F Value	Pr > F
Folded F	38	29	1.01	0.9933



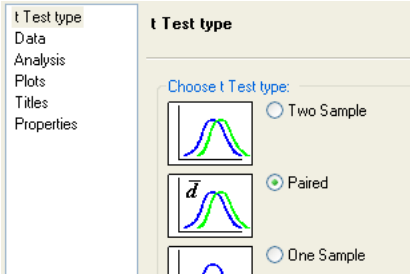
Før vi fortolker, om der er signifikant forskel mellem priserne i nord og syd, skal vi lige huske, at det er en forudsætning for t-testen, at variansen er ens mellem de to grupper af huse, vi tester. Det vil i praksis sige, at fx husene i nord ikke må svinge voldsomt mere i pris end husene i syd. Det tester SAS faktisk automatisk for os, og testresultatet er ovenfor i den nederste tabel, der hedder *Equality of Variances*. Varianstestens H_0-hypotese er, at varianserne er ens mellem de to grupper. Da p-værdien på 0,9933 er meget højere end vores signifikansniveau α=0,05, kan vi ikke forkaste denne H_0, og derfor konkluderer vi, at denne forudsætning er opfyldt.

Nu kan vi så gå videre til den hypotese, som vi oprindeligt ville teste: Er der forskel i gennemsnitsprisen mellem nord og syd. Da vi lige har fået bekræftet, at varianserne er ens, så aflæser vi testresultatet i den tredje tabel fra oven under metoden *Pooled* (betyder, at varianserne er *pooled* = lagt sammen, da de jo ikke er forskellige). P-værdien er 0,0404, hvilket er lige under vores signifikansniveau α=0,05, så derfor konkluderer vi, at der er prisforskel mellem gennemsnitsprisen på huse hhv. nord og syd for København. Da p-værdien er meget tæt på vores signifikansniveau, er konklusionen lidt usikker.

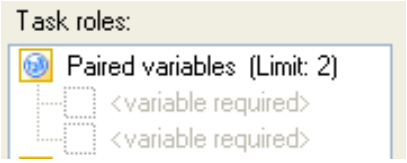
I den øverste tabel i outputtet ovenfor kan man se, at stikprøvegennemsnitsprisen for huse nord for København er 2,429 mio. kr., og at stikprøvegennemsnitsprisen for huse syd for København er 2,029 mio. kr. Dette betyder, at huse nord for København gennemsnitligt er dyrere end huse syd for København. Men her skal man være forsigtig med at begynde at overfortolke testresultatet. Der kan være mange årsager til, at husene i nord gennemsnitligt er dyrere end dem i syd. Måske er de gennemsnitligt større eller har gennemsnitligt højere herlighedsværdi end husene i syd. Det har vi jo slet ikke forholdt os til i denne enkle test. Det kommer vi tilbage til.

3.1.3. Test af to parvise stikprøver

Hvis man har parvise observationer – fx priser for de samme boliger i år 2000 og i dag – så kan man teste på forskellen. Det kalder man parvise stikprøver, fordi observationerne passer sammen to og to. Vores datasæt er ikke oplagt til at teste på parvise stikprøver, men det kan let gøres, hvis datasættet tillader det. Test på parvise stikprøver udføres samme sted som de to foregående tests:



I denne test skal man blot angive, hvilke to variable der hører sammen to og to:

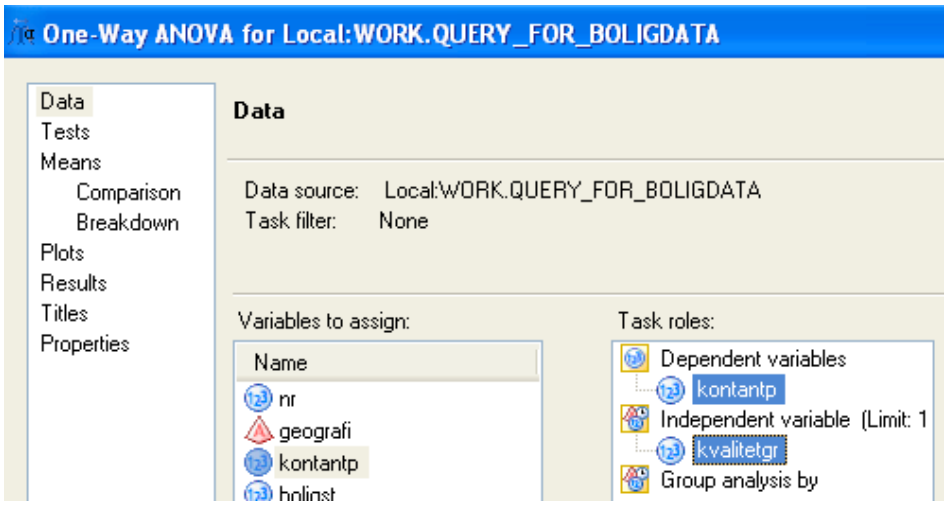


Hvis vi fx havde boligpriserne på de samme boliger i dag som en anden variabel, der hed *kontantp_nu*, så kunne man bruge den variabel sammen med variablen *kontantp* og derved teste, om priserne generelt havde ændret sig. Parvise tests er stærke rent statistisk, fordi de er virkelig gode til at spotte ændringer, som test for to uafhængige stikprøver ikke kan identificere.

3.1.4. Test flere gennemsnit med variansanalyse ('One-Way' *ANOVA*)

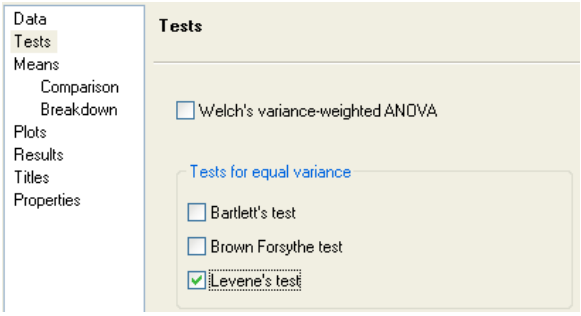
Hvis man vil teste flere end to gennemsnit, så er det one way *ANOVA*, man skal bruge. *ANOVA* er egentlig en forkortelse for *Analysis of Variance*, og det kan jo godt lyde lidt misvisende, når det er en test for gennemsnit, men sådan siger man altså. Vi vil her prøve at teste, om der er forskel i boligprisen mellem de tre forskellige kvalitetsgrupper af boliger, dvs. om herlighedsværdien påvirker boligprisen.

Denne *ANOVA*-test gennemføres ved først at åbne datasættet via *Process Flow*-diagrammet og derefter vælge *Analyze* > *ANOVA* > *One-Way ANOVA…* Derved fremkommer nedenstående dialogboks, hvor variablen *kontantp* vælges under *Dependent variables*, og variablen *kvalitetgr* vælges som *Independent variable*:



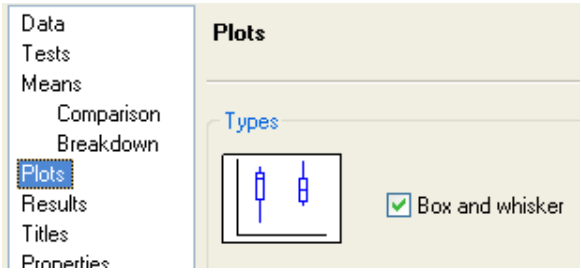
Man kan i princippet trykke *Run* med det samme, hvorefter resultatet fremkommer. Man skal dog lige huske at tjekke forudsætninger for testen først. De er 1) varianshomogenitet og 2) normalfordelte fejlled (residualer) fra *ANOVA*-testen.

- Varianshomogenitet kan fx testes med *Levene's test*, som man vælger under menuen *Tests*:



- Vi har allerede indirekte tjekket normalfordelingen af residualerne ved at undersøge, om *kontantp*-variablen var normalfordelt (se underafsnit 2.2.3), så det er godt nok her.

Ofte er det rart at få visualiseret resultatet af en *ANOVA*-test, så derfor vælger vi et *Box and whiskers*-plot under menuen *Plots*:



Tryk nu på *Run*, og du får dit output frem.

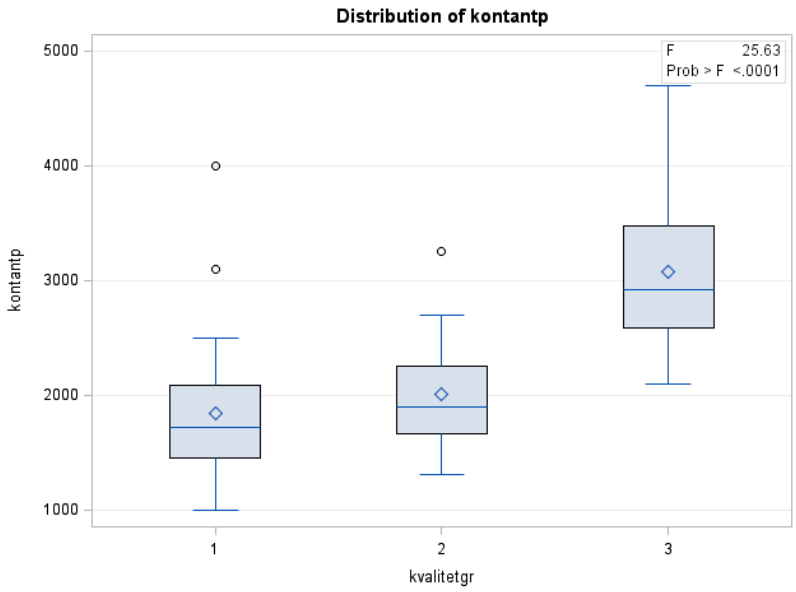
Først ser vi på *Levene's test* for varianshomogenitet. Vores H_0 er, at der er varianshomogenitet, altså at der ikke er problemer med forskellig varians af huspriser mellem de tre grupper af *kvalitetgr*-variablen. Da p-værdien på 0,3527 er langt over vores signifikansniveau på α=0,05, kan vi ikke forkaste H0 og antager derfor, at der er varianshomogenitet. Dvs. forudsætningen for *ANOVA*-testen er opfyldt:

Levene's Test for Homogeneity of kontantp Variance ANOVA of Squared Deviations from Group Means					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
kvalitetgr	2	1.017E12	5.086E11	1.06	0.3527
Error	66	3.171E13	4.804E11		

Nu kan vi aflæse selve *ANOVA*-testresultatet nedenfor. Som du kan se, er p-værdien under 0,0001, hvilket er langt under vores signifikansniveau på α=0,05, så derfor konkluderer vi, at der er signifikant forskel på prisen af boliger i de forskellige grupper:

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	2	19324980.42	9662490.21	25.63	<.0001
Error	66	24885130.56	377047.43		
Corrected Total	68	44210110.99			

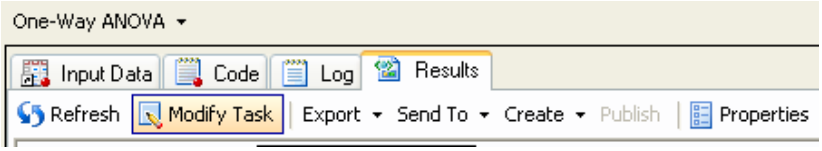
Det næste spørgsmål, som man så kan stille, er, hvordan boligprisen adskiller sig mellem de tre forskellige grupper af *kvalitetgr*. Her kan vi i første omgang se på vores *Box and whiskers*-plot nedenfor:

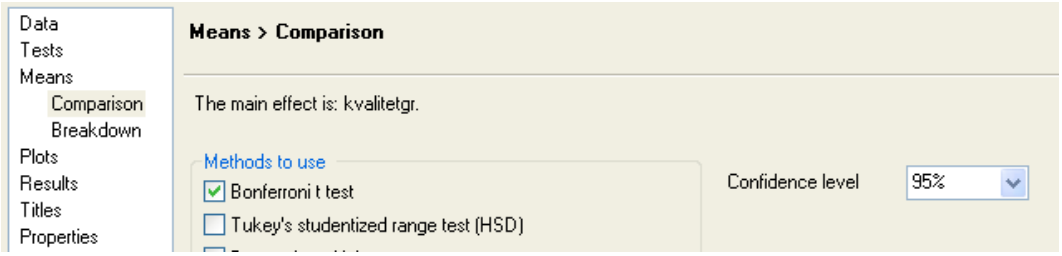


Plottet viser fordelingen af *kontantpris*en på de tre forskellige grupper af *kvalitetgr*. Denne type plot er god til at få et overblik over dataene, da den også giver overblik over spredningen af data på de enkelte grupper. Stregen igennem boksene er medianen og ikke gennemsnittet, men det giver stadig en god fornemmelse af, hvordan de forskellige grupper af huspriser ligger i forhold til hinanden.

Man kan fortolke plottet som, at det særligt er huse i *kvalitetgr* 3, der adskiller sig fra de to andre grupper ved at være markant dyrere, hvorimod det ikke er helt tydeligt at se, om forskellen mellem gruppe 1 og 2 er signifikant.

Hvis man vil have helt klare bud på, hvilke grupper der er signifikant forskellige i forhold til gennemsnitsprisen, kan man lave en *post-hoc*-test ved at gå tilbage og tilføje dette til testen. Det gøres fx ved at trykke på *Modify Task* i bjælken over outputtet og derefter vælge Bonferroni under menuen *Comparisons*, som det vises i de to nedenstående billeder:





Tryk derefter *Run* igen og bare sig *Yes* til dialogboksen, der kommer op og spørger, om det nye output skal erstatte det gamle.

Outputtet fremkommer nu igen, og denne gang med Bonferronis simultane konfidensintervaller nederst. Som du kan se, har SAS angivet, hvilke grupper der er signifikant forskellige fra hinanden mht. *kontantp*. Konklusionen er, at gruppe 3 er signifikant dyrere end både gruppe 1 og 2, men at gruppe 1 og 2 ikke er signifikant forskellige fra hinanden:

Comparisons significant at the 0.05 level are indicated by ***.				
kvalitetgr Comparison	Difference Between Means	Simultaneous 95% Confidence Limits		
3 - 2	1073.2	612.0	1534.4	***
3 - 1	1231.5	782.8	1680.1	***
2 - 3	-1073.2	-1534.4	-612.0	***
2 - 1	158.3	-273.5	590.0	
1 - 3	-1231.5	-1680.1	-782.8	***
1 - 2	-158.3	-590.0	273.5	

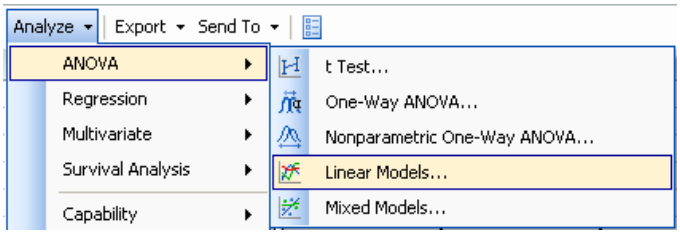
Alt i alt har vi fundet ud af to ting: 1) At boliger med en høj herlighedsværdi er signifikant dyrere end øvrige boliger, hvilket jo ikke er så overraskende. 2) At boliger med *lav* herlighedsværdi ikke er signifikant billigere end boliger med en gennemsnitlig herlighedsværdi, hvilket umiddelbart er lidt overraskende. Man skal dog være forsigtig med at konkludere for meget ud fra dette, for man kunne sagtens forestille sig, at boliger med beliggenheder med *lav* herlighedsværdi til gengæld har andre fordele, fx større bolig- og grundareal, hvilket jo så igen bringer deres boligpriser op. Om denne forklaring holder, kan du jo fx prøve at teste med endnu en One-Way *ANOVA*-test – nu ved du jo, hvordan man gør i SAS. (hint: gentag One-Way *ANOVA*-testen, men brug i stedet variablen *boligst* som den afhængige variabel).

3.1.5. Test flere gennemsnit på to faktorer ('Two-Factor' ANOVA)

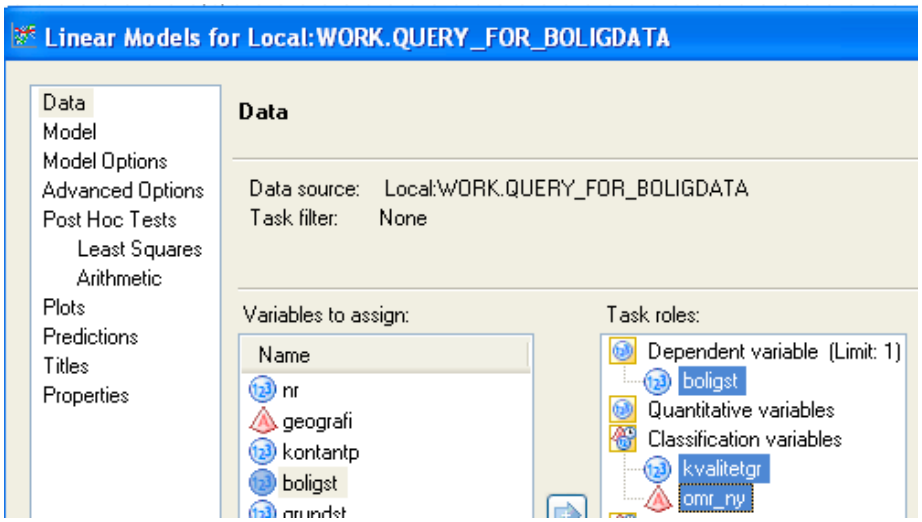
(Emnet for dette underafsnit er typisk ikke pensum på indledende statistikkurser, men er alligevel ret anvendeligt i praksis. Det kan dog sagtens springes over, hvis du har fået nok af ANOVA for nu. Så kan du evt. fortsætte med afsnit 3.2 på side 50).

Hvis du lavede den lille udfordring i slutningen af sidste afsnit, så er du nok kommet frem til, at med en almindelig One-Way *ANOVA*-test er der ikke signifikant forskel på boligstørrelserne mellem de forskellige kvalitetsgrupper af boliger. Men måske er sammenhængen mellem boligstørrelsen og kvalitetsgruppen forskellig mellem nord og syd. I 3.1.2 fandt vi jo ud af, at huse nord for København er signifikant dyrere end huse syd for København – men måske skyldes det jo bare, at de er større. Alt dette kan vi opsummere i én test, som samtidig vil være bedre til at spotte, om der reelt er forskel i boligstørrelsen mellem de tre kvalitetsgrupper, når vi samtidig tager højde for, om huset ligger nord eller syd for København.

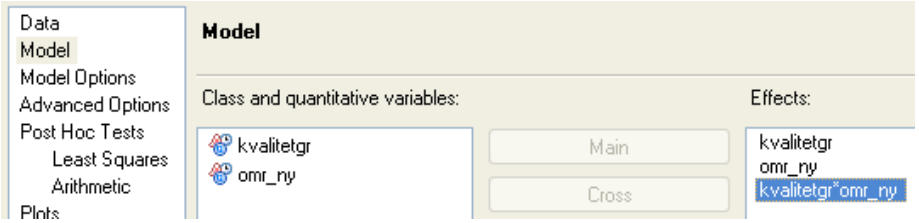
I SAS skal vi nu have fat i en *Linear Model*, og den analyse vælges via *Analyze* > *ANOVA* > *Linear Models…*:



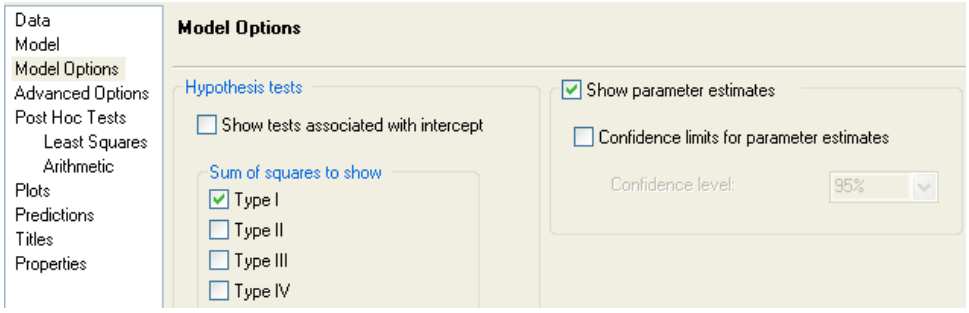
Her vælges *boligst* som den afhængige variabel, og *kvalitetgr* samt *omr_ny* vælges som klassifikationsvariable:



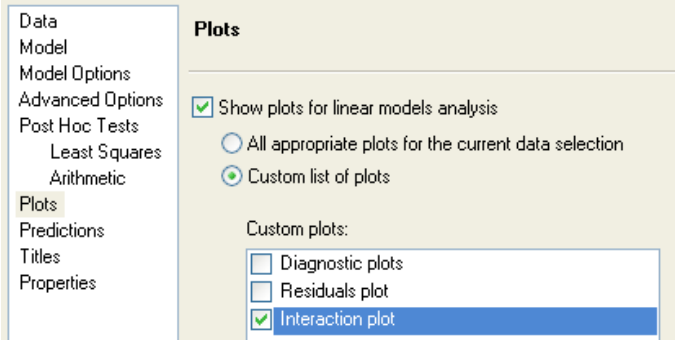
Under *Model* i menuen til venstre vælges både *kvalitetgr* og *omr_ny*, som trækkes over som Main Effects. Bagefter markeres de begge, og der trykkes på *Cross*. Derved oprettes en såkaldt interaktion mellem de to variable, hvilket vil gøre vores model fleksibel til at spotte, hvis variablen "*kvalitetsgr*" ikke har samme effekt på boligstørrelsen hhv. nord og syd for København:



Under *Model Options* fravælges *Type III* under *Sum of Squares to show*:



Under *Plots* har man flere muligheder for at få et virkelig godt overblik over residualerne fra testen, fx om de nu også er helt uafhængige på tværs af alle grupper. I praksis er det ret vigtigt, men her nøjes vi med at vælge et *Interaction plot*:



Nu er vi klar til at trykke *Run*, hvorefter outputtet fremkommer. Først i outputtet bliver det opsummeret, hvilke *levels* de to kategorivariable har, hhv. kvalitetsgruppe 1, 2 og 3, inden for hhv. område *Nord* eller *Syd*:

Class Level Information		
Class	Levels	Values
kvalitetgr	3	1 2 3
omr_ny	2	Nord Syd

Derunder kan vi se, at modellen samlet set er signifikant med en p-værdi på 0,0194, som ligger under vores signifikansniveau α=0,05. Dvs. der er altså signifikant forskel på boligstørrelsen mellem de tre kvalitetsgrupper og nord vs. syd, når det ses samlet:

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	5	32368.5219	6473.7044	2.93	0.0194
Error	63	139292.6375	2210.9942		
Corrected Total	68	171661.1594			

R-Square	Coeff Var	Root MSE	boligst Mean
0.188561	28.57803	47.02121	164.5362

Source	DF	Type I SS	Mean Square	F Value	Pr > F
kvalitetgr	2	7881.15340	3940.57670	1.78	0.1766
omr_ny	1	16009.55816	16009.55816	7.24	0.0091
kvalitetgr*omr_ny	2	8477.81036	4238.90518	1.92	0.1555

Hvorvidt den overordnede signifikans af testen kommer fra en signifikant forskel mellem nord og syd, eller om det skyldes forskel mellem de tre kvalitetsgrupper, kan vi se i den nederste tabel ovenfor. Deraf fremgår det, at:

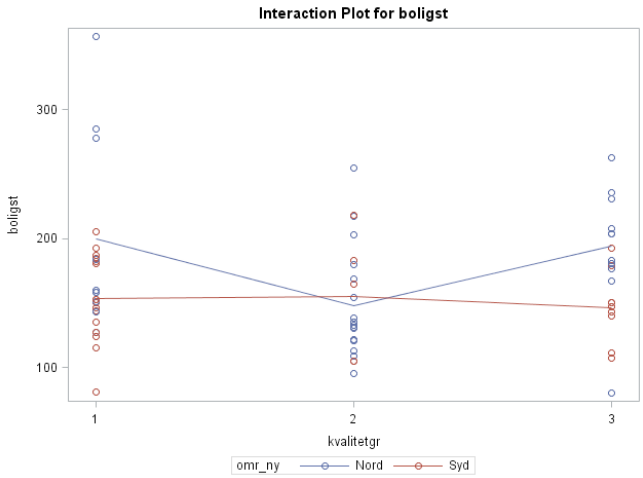
Forskellen i boligstørrelse mellem nord og syd er signifikant (p=0,0091, som er langt under α=0,05).

Forskellen i boligstørrelse mellem kvalitetsgrupperne er ikke signifikant (p=0,1766).

Interaktionen mellem område og kvalitetsgruppe er ikke signifikant (p=0,1555), hvilket kan fortolkes som, at der ikke er forskel på boligstørrelserne af de forskellige kvalitetsgruppers huse mellem hhv. nord og syd.

Vi kan altså stadig ikke sandsynliggøre, at der skulle være forskel på boligarealet af boliger i de tre forskellige kvalitetsgrupper.

Hvis vi ser på interaktionsplottet nedenfor, kan vi på den blå linje se, at der nord for København er en tendens til, at boliger i kvalitetsgruppe 2 er mindre end de øvrige. Det er på grund af interaktionsvariablen *kvalitetgr* * *omr_ny*, at de to linjer nedenfor ikke er parallelle. Det, plottet viser os, er lidt interessant, men vi husker på, at forskellen i boligstørrelserne fordelt på kvalitetsgrupperne ikke er statistisk signifikant, jf. vores p-værdier for *kvalitetsgr* ovenfor:



Nu har du en ret grundig viden om *ANOVA* i SAS og er klar til gå til næste afsnit om andele.

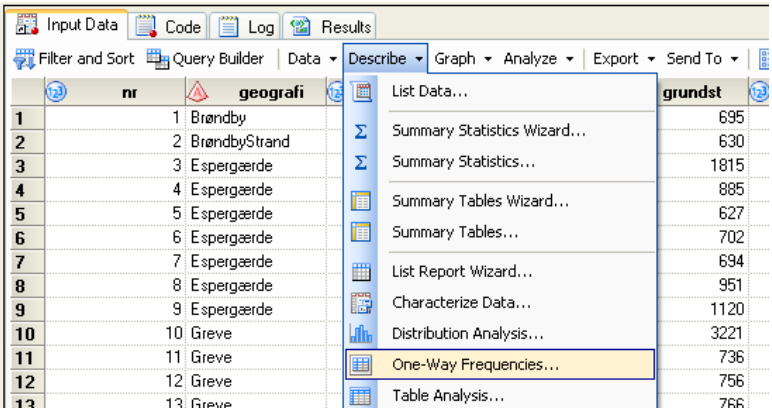
3.2. Test af andele (proportions) og antals-tabeller (Chi-i-anden-tests)

Hvis data er af den nominelle type (grupper), kan man ikke beregne gennemsnit. Derfor beregner man tit andele/procenter i stedet. Det kræver nogle andre teknikker, baseret på at tælle, hvor mange der er i hver gruppe. Test på én andel introduceres i underafsnit 3.2.1 og test på forskellen mellem to andele i 3.2.2.

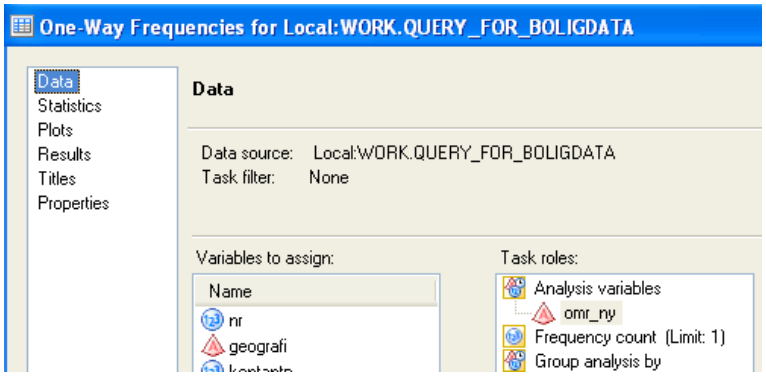
Hvis man gerne vil se, hvordan to nominelle variable er relaterede, laver man en antals-tabel (på engelsk *Cross tabulation* / *Contingency table*) på kombinationer af de to variable. Derefter kan man teste, om de to variable er uafhængige af hinanden. Denne teknik med at teste på antals-tabeller kalder man *Chi-i-anden-tests* (udtales Kí-i-anden). Et meget anvendt eksempel på en Chi-i-anden-test er en såkaldt uafhængighedstest, som vi prøver at lave i underafsnit 3.2.3.

3.2.1. Test af en andel (= procent mellem 0% og 100% = tal mellem 0 og 1)

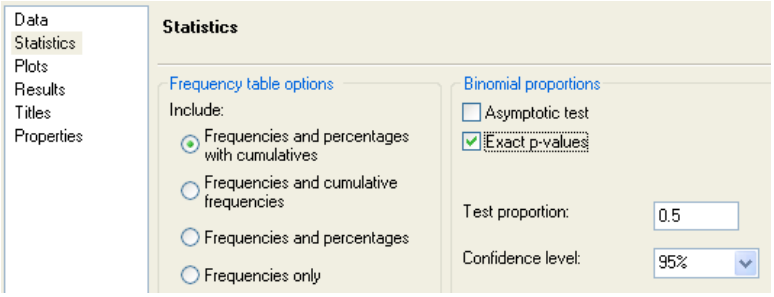
Et eksempel til test på én andel er, at vi fx kan prøve at teste, om andelen af boliger nord for København i vores datasæt er 50% (eller 0,5 udtrykt som et tal mellem 0 og 1). Det testes i SAS ved først at åbne vores datasæt *Query for boligdata* og derefter vælge *Describe* > *One-Way Frequencies…*:



Variablen *omr_ny* trækkes over som analysevariabel:



Derefter skal man specificere, at vores hypotese er, at andelen af boliger i nord er 50%. Det gøres under menuen *Statistics*, hvor der allerede står 0,5 som standard *Test proportion* og 95% som *Confidence level*. Begge dele er fint. Der sættes også flueben ved *Exact p-values*:



Tryk nu *Run*, og outputtet nedenfor fremkommer. Deraf fremgår det, at i vores datasæt er 56,5% af boligerne nord for København:

omr_ny	Frequency	Percent	Cumulative Frequency	Cumulative Percent
Nord	39	56.52	39	56.52
Syd	30	43.48	69	100.00

Det fremgår også, at p-værdien for en tosidet test på andelen *0,5* er 0,3356, hvilket er langt over vores signifikansniveau α=0,05. Derfor kan vi ikke afvise, at der generelt er lige mange huse til salg nord og syd for København (husk, at vi antog i 1.3, at dette datasæt er et tilfældigt udpluk af huse, der er til salg nord og syd for København):

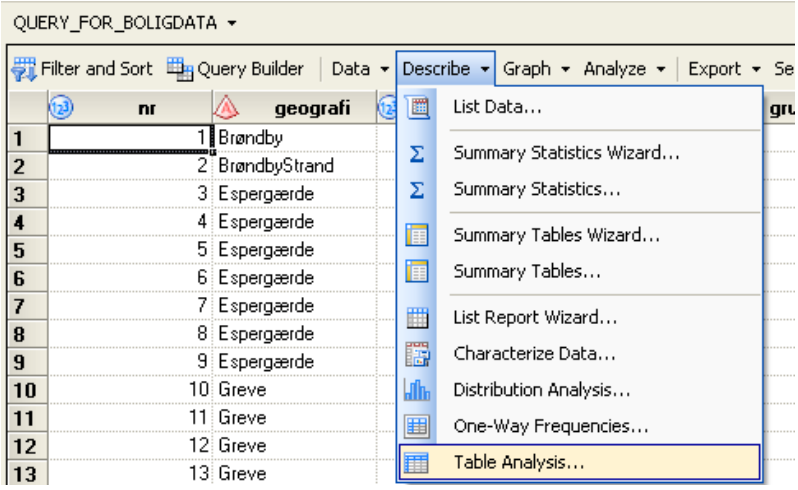
Exact Test	
One-sided Pr >= P	0.1678
Two-sided = 2 * One-sided	0.3356

Om en test er en- eller tosidet kommer an på, om vi på forhånd har belæg for at antage, at det teoretisk set kun kan falde til den ene side. Det har vi ikke her, så derfor anvendes tosidet test. SAS beregner dog begge p-værdier, som det ses ovenfor.

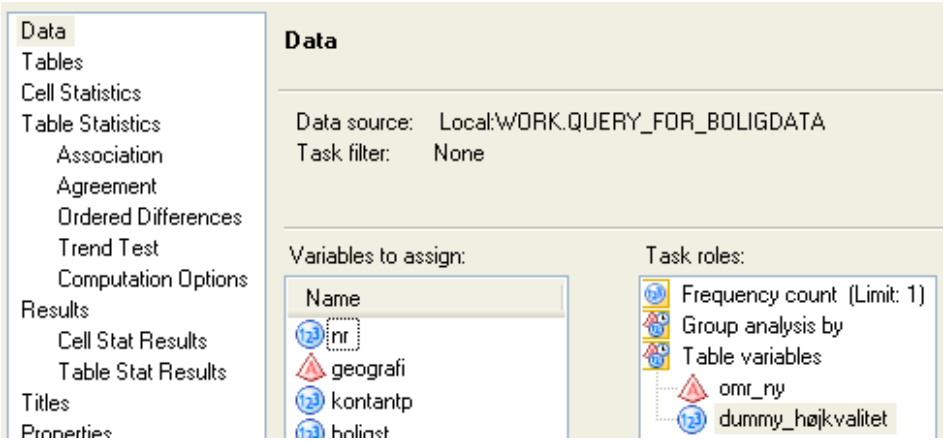
3.2.2. Test af forskellen mellem to uafhængige andele

Ofte kan det være interessant at se, om to andele er signifikant forskellige fra hinanden. Vi kunne også godt få en idé om, at der måske er flere boliger med høj herlighedsværdi nord for København i forhold til syd, målt som andelen af boliger, der har værdien *1* i dummyvariablen *dummy_højkvalitet* i nord vs. syd. Det kan vi selvfølgeligt også teste i SAS baseret på vores stikprøve, så vi kan få nogle facts i stedet for gætteri.

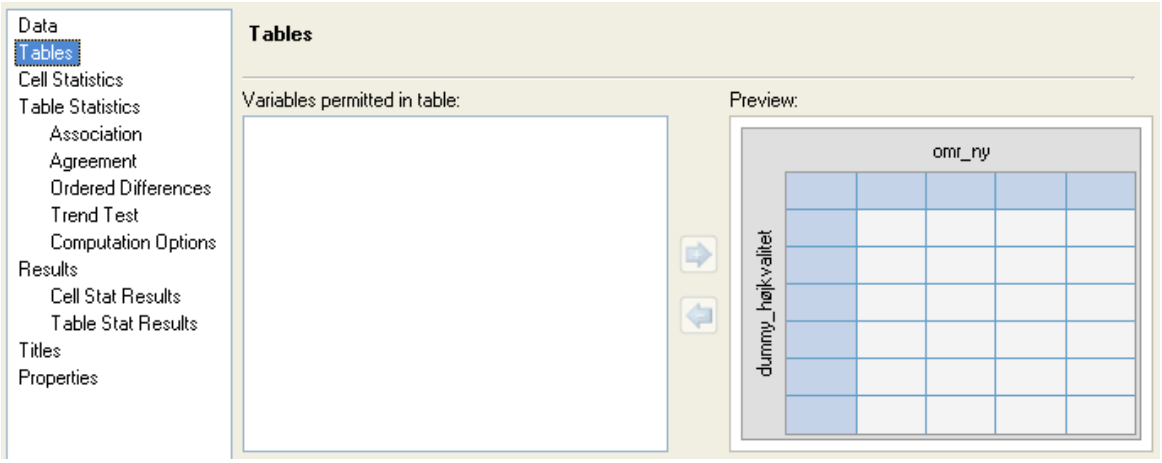
I SAS laves en datatabelanalyse. Det gøres ved først at åbne vores analysedatasæt *Query for boligdata* og derefter vælge *Describe* > *Table Analysis…*:



Her vælger vi variablene *omr_ny* og *dummy_højkvalitet* som *Table variables*:

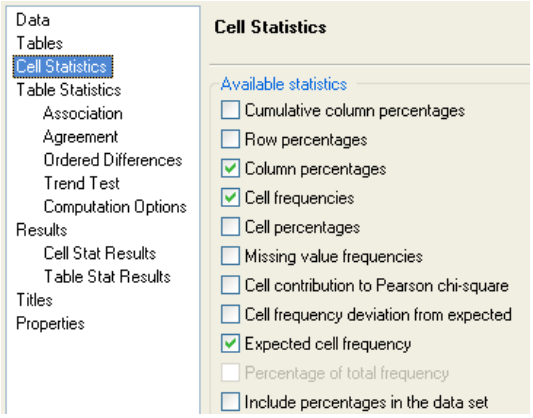


Under *Tables* ude til venstre trækkes først *omr_ny* over som kolonnevariabel og derefter *dummy_højkvalitet* over som rækkevariabel:

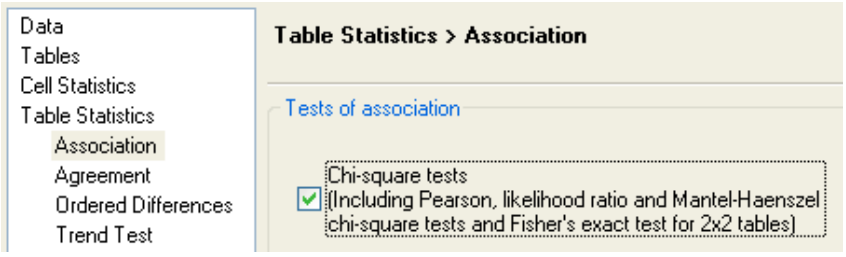


(Hvis du bruger en tidligere version af SAS end version 5.1, ser det lidt anderledes ud.)

Derefter vælges *Cell Statistics* ude til venstre, og *Expected cell frequency* tilvælges. Det er nemlig en forudsætning for denne testtype, at alle celler i tabellen har en forventet værdi på minimum 5 observationer:



Under *Association* vælges *Chi-square tests* (udtales [Ki]):



Der er selvfølgeligt mange andre valgmuligheder, men dette er nok for vores analyse. Derfor kan du nu trykke *Run*, hvorefter outputtet fremkommer. Øverst er en tabel, der opsummerer, hvordan dataene er fordelt. Fx kan vi se, at der i nord er 11 boliger i vores stikprøve med høj herlighedsværdi. De 11 udgør ca. 28% af alle 39 boliger i nord. Tilsvarende kan vi se, at der er 9 boliger med høj herlighedsværdi i syd, hvilket udgør 30%.

Table of dummy_højkvalitet by omr_ny				
		omr_ny		
		Nord	Syd	Total
dummy_højkvalitet				
0	Frequency	28	21	49
	Expected	27.696	21.304	
	Col Pct	71.79	70.00	
1	Frequency	11	9	20
	Expected	11.304	8.6957	
	Col Pct	28.21	30.00	
Total	Frequency	39	30	69

Længere nede i outputtet får vi vores p-værdi. På dette punkt er SAS rigeligt generøs – vi får mange forskellige tests for det samme. Vi ser bare på den øverste p-værdi *(Prob)* for *Chi-Square-testen*. Den er på 0,8706, dvs. langt over vores signifikansniveau på α=0,05:

Statistic	DF	Value	Prob
Chi-Square	1	0.0265	0.8706
Likelihood Ratio Chi-Square	1	0.0265	0.8707
Continuity Adj. Chi-Square	1	0.0000	1.0000
Mantel-Haenszel Chi-Square	1	0.0262	0.8715
Phi Coefficient		0.0196	
Contingency Coefficient		0.0196	
Cramer's V		0.0196	

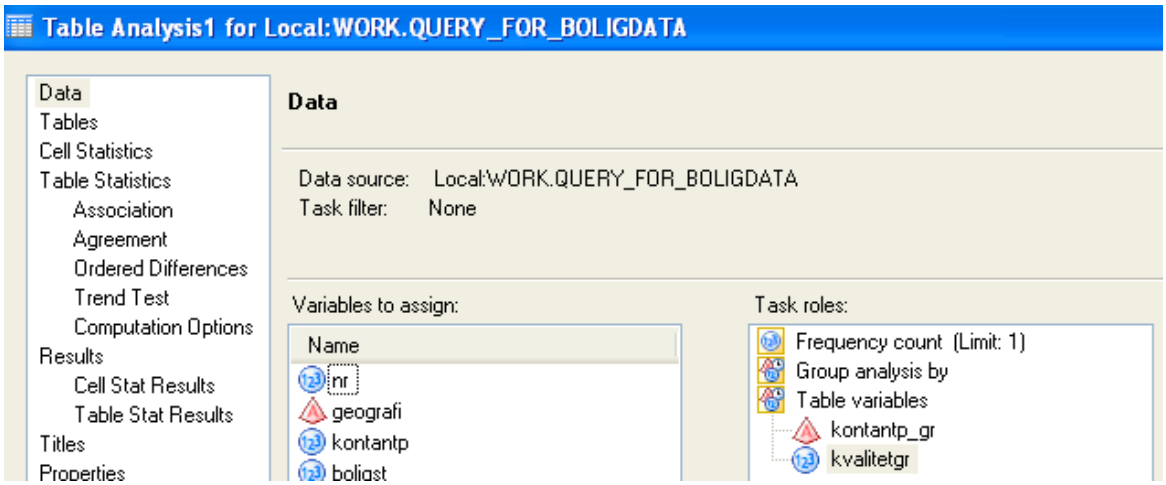
Konklusionen er derfor, at der altså intet statistisk belæg er for at konkludere, at andelen af boliger med høj herlighedsværdi er forskellig mellem nord og syd (forudsat, at vores stikprøve er repræsentativ for alle boliger nord og syd for København).

3.2.3. Test for uafhængighed (Chi-i-anden-test)

I forrige eksempel testede vi to andele via *Table Analysis* (underafsnit 3.2.2). På samme måde kan større antals-tabeller også testes. Vi kan fx teste, om der er en sammenhæng mellem, hvilken *kontantpris*gruppe en bolig ligger i på den ene side, og hvilken kvalitetsgruppe den ligger i på den anden side. Her vil man nok intuitivt forvente, at der er en sammenhæng: Boliger med høj herlighedsværdi vil nok også være i den dyre ende af skalaen − og omvendt for boliger med *lav* herlighedsværdi. En sådan test hedder en uafhængighedstest, fordi vores H0 er, at der er uafhængighed mellem de to variable. Hvis vi får en signifikant p-værdi i testen, så forkaster vi H0 og konkluderer, at der ikke er uafhængighed, altså at de to variable er indbyrdes forbundne på en eller anden måde.

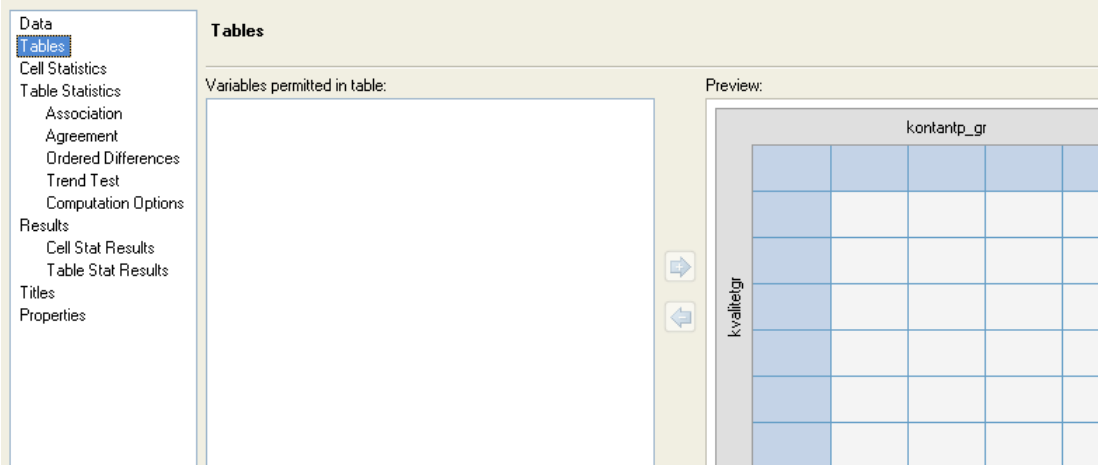
Lad os prøve at teste vores uafhængighedstest mellem *kvalitetgr* og *kontantp_gr* i SAS[3].

På samme måde som i forrige underafsnit åbnes datasættet *Query for boligdata*, og der vælges *Describe* > *Table Analysis…* Her vælges variablene *kvalitetgr* og *kontantp_gr* som tabelvariable som nedenfor:



3. Homogenitetstests laves på nøjagtig samme måde i SAS – det er kun en teoretisk forskel ift. uafhængighedstesten.

Herefter vælges *Tables* ude til venstre, og tabellen bygges op ved at trække *kontantp_gr* over som kolonnevariabel, og *kvalitetgr* over som rækkevariabel:



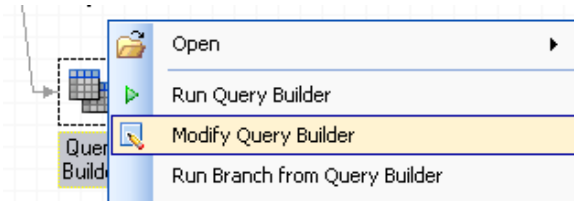
Herefter foretages de samme tilføjelser til indstillingerne som dem, der er billeder af i forrige underafsnit 3.2.2. (*Cell Statistics* > *Expected cell frequency* og *Association* > *Chi-square*-test). Tryk derefter *Run*, og nedenstående output kommer frem:

Table of kvalitetgr by kontantp_gr					
		kontantp_gr			Total
		høj	lav	middel	
kvalitetgr					
1	Frequency	2	10	14	26
	Expected	7.913	4.5217	13.565	
	Col Pct	9.52	83.33	38.89	
2	Frequency	3	2	18	23
	Expected	7	4	12	
	Col Pct	14.29	16.67	50.00	
3	Frequency	16	0	4	20
	Expected	6.087	3.4783	10.435	
	Col Pct	76.19	0.00	11.11	
Total	Frequency	21	12	36	69

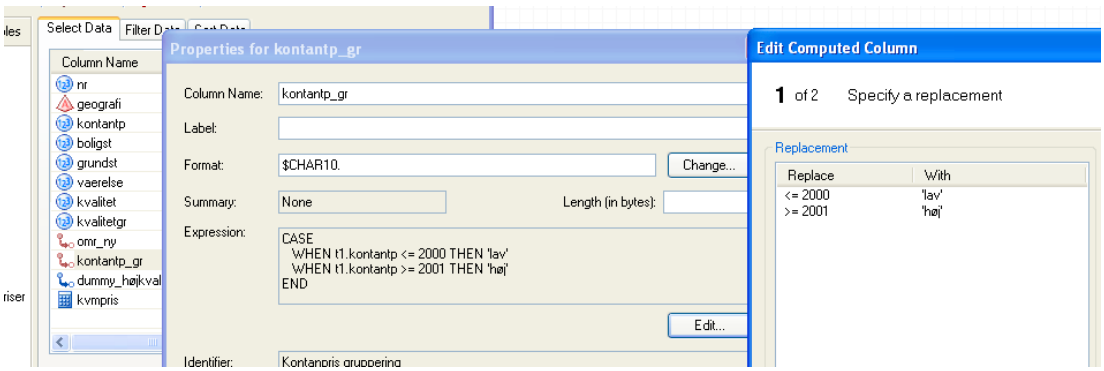
Hvis vi ser på *Expected cell frequencies* i tabellens rækker med *Expected*…, så kan vi se, at disse tal er under 5 for alle boligerne i den lave *kontantpris*gruppe. Da det er en forudsætning, at alle *Expected*-værdierne er over 5, kan vi ikke stole på testen nedenfor. Bemærk også, at SAS advarer os om det under tabellen med p-værdien fra *Chi-square*-testen:

Statistic	DF	Value	Prob
Chi-Square	4	40.9458	<.0001
Likelihood Ratio Chi-Square	4	41.2513	<.0001
Mantel-Haenszel Chi-Square	1	14.3318	0.0002
Phi Coefficient		0.7703	
Contingency Coefficient		0.6103	
Cramer's V		0.5447	
WARNING: 33% of the cells have expected counts less than 5. Chi-Square may not be a valid test.			

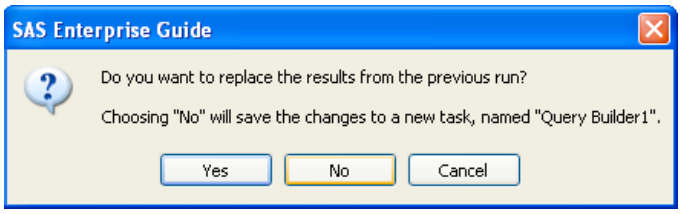
Løsningen er at ændre på kategorierne – der er åbenbart for få huse i boligkategorien af huse under 1,5 mio. kr. Derfor vil vi ændre på kodningen af *kontantpris*gruppevariablen *kontantp_gr* til kun at have to kategorier: *lav* for huse under 2 mio. og *høj* for huse over 2 mio. Denne ændring laves ved at gå tilbage til vores *Query* i *Process Flow*-diagrammet og ændre kodningen af variablen *kontantp_gr*. Først højreklikkes på *Query*-ikonet i *Process Flow*-diagrammet, og vi vælger *Modify Query builder*:



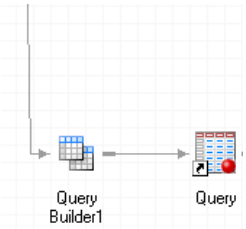
Derefter dobbeltklikker man på variablen *kontantpr_gr* og vælger *Edit…* i den næste dialogboks. Nu skal vi have ændret vores kategorier til at være som på billedet nedenfor, dvs. *lav* hvis mindre eller lig med 2000 og *høj* hvis større eller lig med 2001 (hvis du ikke kan huske, hvordan man gør, kan du se det i afsnit 2.3.3 på side 30):



Tryk herefter *Finish*, *OK* og *Run*. Nu bliver du spurgt, om du vil overskrive det tidligere resultat. Da alle vores forgående analyser er baseret på det tidligere datasæt, er det en god idé at svare *No* til det:



Derved vil vores redigerede *Query* resultere i et nyt datasæt. Det nye datasæt fremkommer også som et nyt ikon efter vores nye *Query* nederst i *Process Flow*-diagrammet:



Baseret på det nye datasæt prøver vi at gentage uafhængighedstesten fra før, men denne gang med den nye udgave af variablen *kontantp_gr*. Fremgangsmåden er nøjagtig som før. Outputtet bliver som nedenfor:

Table of kvalitetgr by kontantp_gr

		kontantp_gr		
		høj	lav	Total
kvalitetgr				
1	Frequency	8	18	26
	Expected	13.565	12.435	
	Col Pct	22.22	54.55	
2	Frequency	8	15	23
	Expected	12	11	
	Col Pct	22.22	45.45	
3	Frequency	20	0	20
	Expected	10.435	9.5652	
	Col Pct	55.56	0.00	
Total	Frequency	36	33	69

Statistics for Table of kvalitetgr by kontantp_gr

Statistic	DF	Value	Prob
Chi-Square	2	25.8951	<.0001
Likelihood Ratio Chi-Square	2	33.7071	<.0001
Mantel-Haenszel Chi-Square	1	19.8811	<.0001
Phi Coefficient		0.6126	
Contingency Coefficient		0.5224	
Cramer's V		0.6126	

Vi kan se, at forudsætningen om, at *Expected* skal være over 5 i alle celler af tabellen, nu er opfyldt. Ser vi på *Chi-Square-testen*, så er p-værdien under 0,0001, hvilket er langt under vores signifikansniveau α=0,05. Dvs. vi forkaster vores H_0 om uafhængighed og konkluderer, at der er <u>afhængighed</u> mellem prisgruppe og boligens kvalitetsgruppe. Dette matcher fint, hvad vi ville forvente at se.

Nu har vi godt styr på andels-tests og tests på datatabeller og er klar til at gå videre til et nyt emne.

3.3. Regressionsanalyse

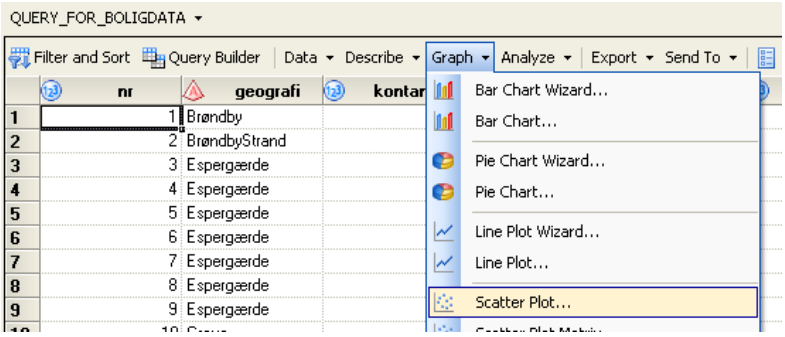
En anden ofte brugt gruppe af teknikker tester sammenhængen mellem en afhængig intervalvariabel og en eller flere uafhængige intervalvariable. Ideen her er, at måske kan én variabel forklares af en eller flere andre variable. Fx giver det god mening at antage, at huspriser til en vis grad kan forklares af boligstørrelsen. Det illustrerer vi med *scatter plot* og korrelationskoefficient i underafsnit 3.3.1, og i 3.3.2 laver vi en simpel lineær regression mellem disse to variable.

Det giver dog også god mening at antage, at *grundstø*rrelsen nok også spiller ind på boligprisen sammen med antal værelser og herlighedsværdien af boligen ligeså. Hvis man vil lave en udvidet regressionsmodel baseret på disse antagelser, så er det en multipel regressionsanalyse, man skal lave, og det prøver vi i underafsnit 3.3.3. *Regression*sanalyser arbejder primært med intervaldata, men kan også håndtere nominel data, og det ser vi et eksempel på i underafsnit 3.3.4, hvor vi også prøver at tilføje andre "eksotiske" variable til vores model (se evt. afsnit 2.2.1 på side 16 for en repetition af datatyper). I 3.3.5 prøver vi at lave en logistisk regression, som er noget lidt andet, hvor den afhængige variabel er numerisk med to kategorier (en *enten/eller*-variabel).

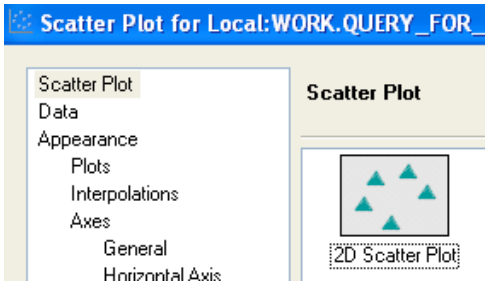
3.3.1. Scatter plot og test af korrelation mellem to intervalvariable

Hvis vi har to intervalvariable, kan de plottes i en graf på to akser – nøjagtig som du ville markere x- og y-koordinater på et ternet ark papir. Her kalder vi det bare et *scatter plot*, men det er altså det samme princip. Idéen med at lave sådan et plot er at se, om der er en sammenhæng mellem de to variable. Her vil vi prøve at lave et *scatter plot* med boligstørrelsesvariablen *boligst* på X-aksen og boligprisvariablen *kontantp* på Y-aksen.

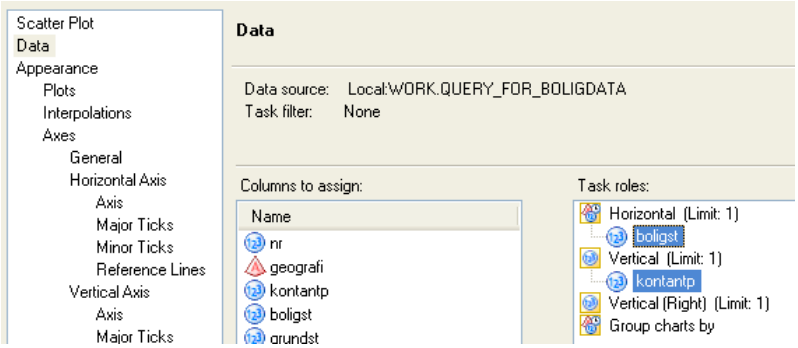
I SAS gøres det ved først at åbne vores datasæt *Query for boligdata* og derefter vælge *Graph* > *Scatter Plot…*, som det er gjort nedenfor:



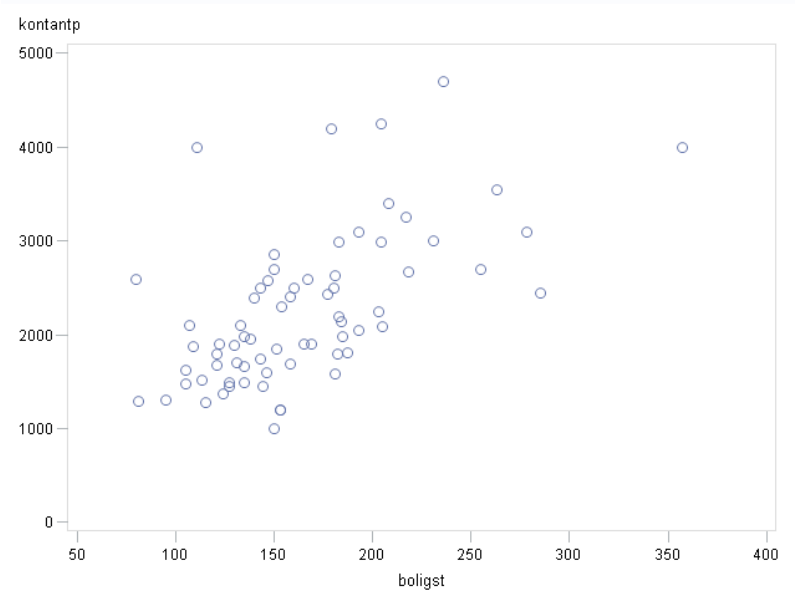
Derefter specificerer vi først, at det er et 2-dimensionalt *Scatter Plot*, vi vil lave:



Under *Data* ude til venstre flyttes variablen *boligst* over på *Horizontal*-aksen og variablen *kontantp* over på *Vertical*-aksen:

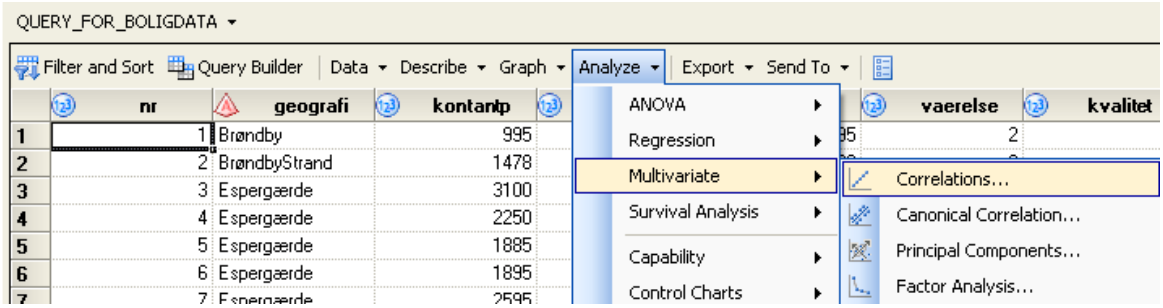


De resterende valgmuligheder er ligegyldige her – vi er klar til at trykke *Run*, hvorefter vores *Scatter Plot* fremkommer:

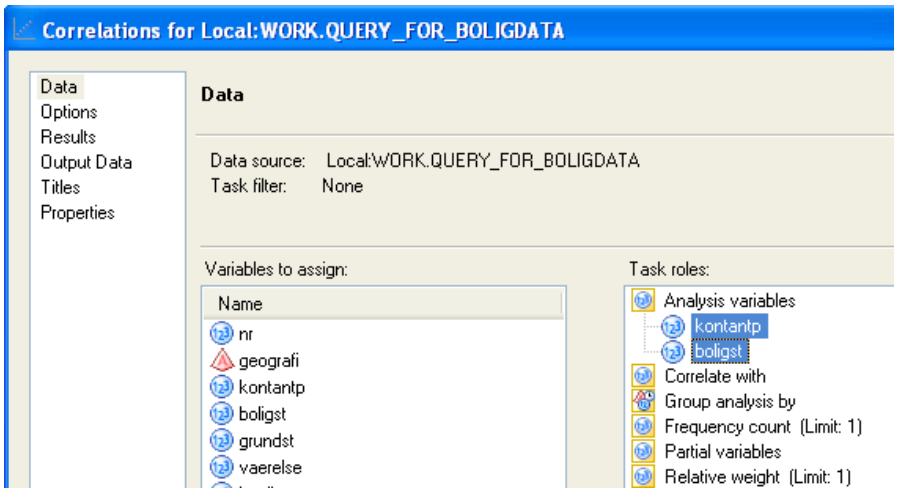


Scatter-plottet viser, at dataene ligger noget spredt, men der er dog en tendens til, at huspriserne stiger, når boligstørrelsen stiger. Hvorvidt der er en lineær sammenhæng, kan vi undersøge ved at teste korrelationen mellem de to variable.

Korrelationer testes i SAS ved igen at åbne vores datasæt *Query for boligdata*, og vælge *Analyze* > *Multivariate* > *Correlations*, som det er gjort nedenfor:



Derefter trækkes variablene *kontantp* og *boligst* over som *Analysis variables*:



Vi behøver ikke lave flere indstillinger og er derfor klar til at trykke *Run*, hvorefter nedenstående output fremkommer. Heraf fremgår det, at korrelationskoefficienten er 0,59408, hvilket skal ses i forhold til, at *0* ville betyde ingen relation mellem de to variable, og 1 ville betyde en perfekt lineær sammenhæng mellem de to variable.

Korrelationskoefficienten er her så høj, at sandsynligheden for, at det bare er en tilfældighed (p-værdien), er under 0,0001 dvs. langt under vores signifikansniveau på α=0,05, så vi konkluderer, at de to variable er signifikant positivt korrelerede. Dvs. at hvis den ene variabel fx stiger, så stiger den anden sandsynligvis også:

Pearson Correlation Coefficients, N = 69 Prob > |r| under H0: Rho=0		
	kontantp	boligst
kontantp	1.00000	0.59408
		<.0001
boligst	0.59408	1.00000
	<.0001	

Dette matcher jo ret godt, hvad vi intuitivt ville forvente.

3.3.2. Simpel lineær regression (regression med én forklarende variabel)

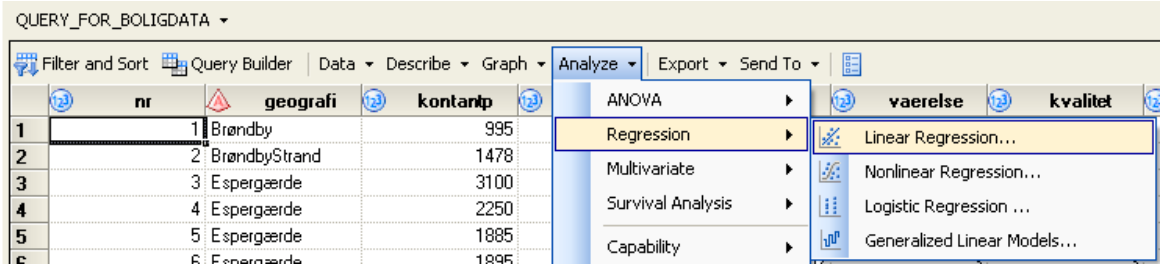
I sidste underafsnit fandt vi en signifikant korrelation mellem boligstørrelsen og boligprisen. Det er meget vigtigt at huske på, at en høj korrelation ikke nødvendigvis betyder, at ændringer i den ene variabel fører til ændringer i den anden. Det er den nok hyppigste fejlantagelse i anvendt statistik.

At en variabel leder til en anden kaldes kausalitet. Hvis du synes, kausalitet er et grimt ord, så bare tænk på det engelske cause (at forårsage noget), så giver det lidt mere mening.

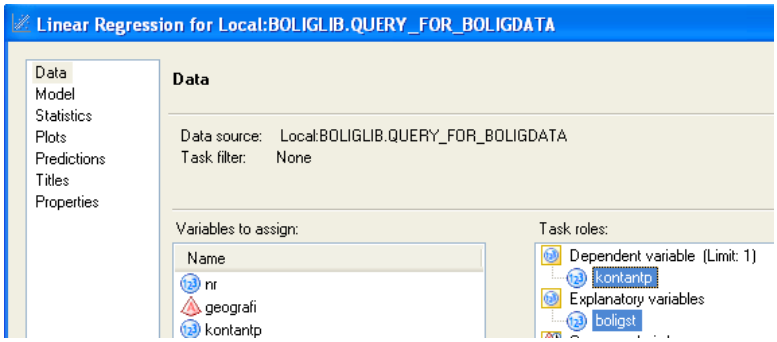
Hvis vi vælger at antage kausalitet mellem boligstørrelsen og *kontantpris*en, så skal vi begrunde det med en teoretisk sammenhæng og ikke, at vores data er højt korrelerede. I vores tilfælde vil vi antage, at et større boligareal medfører en højere boligpris, selvom der også kan være mange andre forklaringer, som også har effekt her (fx at store huse måske bygges i populære områder, er af højere *kvalitet* og er mere velholdte).

Fordi vi antager kausalitet, så giver det mening at lave en regressionsmodel, der undersøger, hvor meget boligprisen stiger, når boligstørrelsen stiger med fx 1 m^2. Modellen er selvfølgelig en grov forenkling af virkeligheden, men vi prøver alligevel for eksemplets skyld i SAS.

Start med at åbne datasættet *Query for boligdata* og vælg *Analyze* > *Regression* > *Linear Regression…*, som det er gjort nedenfor:



Træk derefter *kontantp* over som *Dependent variable* og *boligst* over som *Explanatory variable:*



Der er mange flere indstillingsmuligheder, men standardindstillingerne er *OK* for os, så vi er klar til at trykke *Run*, hvorefter vores output kommer frem:

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	1	15603372	15603372	36.54	<.0001
Error	67	28606739	426966		
Corrected Total	68	44210111			

Root MSE	653.42655	R-Square	0.3529
Dependent Mean	2255.31884	Adj R-Sq	0.3433
Coeff Var	28.97269		

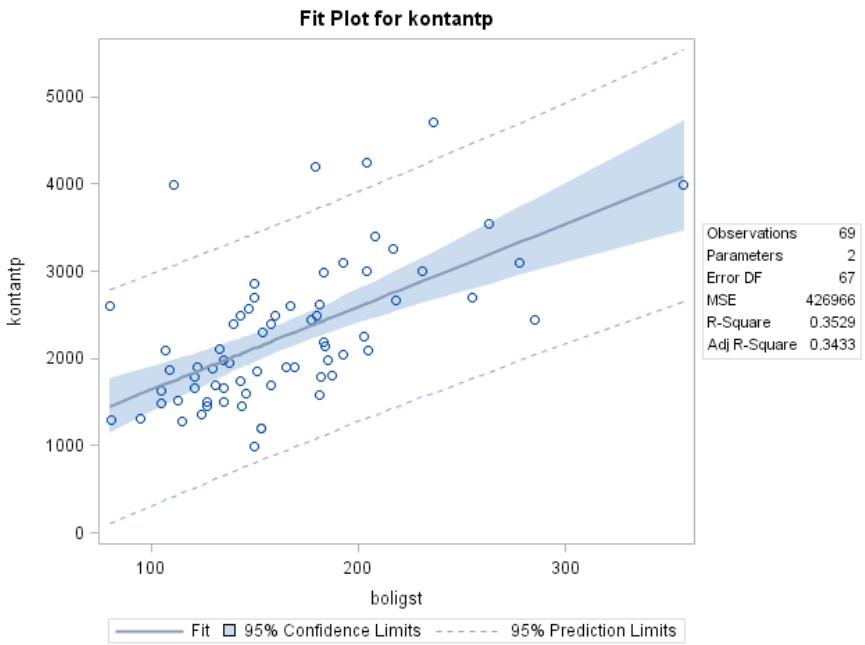
Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|
Intercept	1	686.63739	271.15216	2.53	0.0137
boligst	1	9.53396	1.57711	6.05	<.0001

Øverst ovenfor ser vi en ANOVA-test for, om modellen overordnet set er signifikant.4 Det er den med den lave p-værdi på under 0,0001, som jo er meget under vores signifikansniveau på α=0,05. Vi kan i den midterste tabel se, at boligstørrelsen kan forklare 35% af prisen på boliger (R2=0,3533). Den nederste tabel viser os, at i denne forenklede model for prisdannelsen på boliger resulterer én ekstra kvadratmeter i en prisstigning på 9.500 kr. Denne effekt er signifikant (p-værdien er igen under 0,0001).

4. Det virker måske lidt mærkeligt, at der bliver lavet en ANOVA-test i en regressionsanalyse, men sådan tester man overordnet modellen.

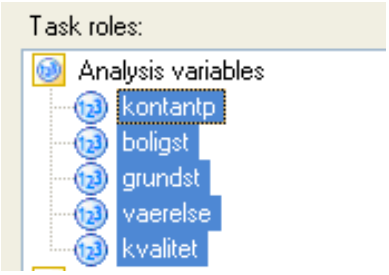
Nedenfor er et plot, som viser, hvordan SAS generaliserer fra vores stikprøve til populationen. Det grå skraverede område er et konfidensinterval for gennemsnitsprisen af boliger med den givne boligstørrelse. Det noget bredere forudsigelsesinterval, som ligger inden for vores *prediction limits* stiplede linjer, er det interval, hvori prisen på en tilfældig bolig med en given boligstørrelse med 95% sikkerhed vil ligge (et gennemsnit kan altid estimeres mere præcist end individuelle observationers værdier):



Forudsætningerne for regressionsanalysen burde vi for øvrigt have tjekket, før vi konkluderede på resultaterne ovenfor, men dem gemmer vi til næste afsnit – flere af forudsætningerne har nemlig at gøre med situationer, hvor man har mere end én forklarende variabel i sin regressionsanalyse, og det har man næsten altid i praksis. *Regression*sanalyse med mere end én forklarende variabel kaldes *multipel* regressionsanalyse, og det er emnet for næste underafsnit.

3.3.3. Multipel lineær regression (regression med flere forklarende variable)

Virkeligheden er ofte mere kompleks end som så, og regressionsmodeller med kun én forklarende variabel kommer næsten altid til at overfortolke på effekten af den uafhængige variabel på den afhængige. I vores eksempel med boligprisens afhængighed af boligstørrelsen kommer vi let til at overfortolke effekten af boligstørrelsen, fordi boligstørrelsen er korreleret med både *grundst*ørrelsen og antal værelser. Det kan ses ved at lave en korrelationstabel ligesom i underafsnit 3.3.1, men denne gang medtage de fem variable *kontantp, boligst*, *grundst*, *vaerelse* og *kvalitet* som *Analysis variables*:



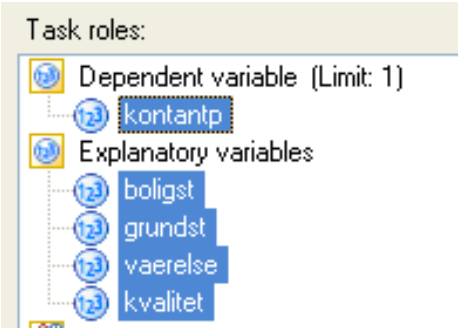
Korrelationstabellen ser ud som nedenfor, hvor de mange lave p-værdier indikerer signifikante korrelationer. Tabellen er symmetrisk, så det er ligegyldigt, om vi aflæser den øverste venstre eller den nederste højre "trekant". Ser vi på første række, kan vi se, at *kontantp* er signifikant korreleret med alle de fire øvrige variable. Det indikerer, at alle de øvrige variable kan bruges til at forklare variablen *kontantp*. Ser vi på de næste rækker, kan vi se, at de tre variable *boligst*, *grundst* og *vaerelse* også er signifikant korrelerede med hinanden. Variablen *kvalitet* er derimod ikke signifikant korreleret med de øvrige tre variable.

Pearson Correlation Coefficients, N = 69 Prob > |r| under H0: Rho=0					
	kontantp	boligst	grundst	vaerelse	kvalitet
kontantp	1.00000	0.59408	0.65371	0.48440	0.60861
		<.0001	<.0001	<.0001	<.0001
boligst	0.59408	1.00000	0.48060	0.67012	-0.04583
	<.0001		<.0001	<.0001	0.7085
grundst	0.65371	0.48060	1.00000	0.32398	-0.01890
	<.0001	<.0001		0.0066	0.8775
vaerelse	0.48440	0.67012	0.32398	1.00000	-0.01100
	<.0001	<.0001	0.0066		0.9285
kvalitet	0.60861	-0.04583	-0.01890	-0.01100	1.00000
	<.0001	0.7085	0.8775	0.9285	

Vi prøver at lave en ny regressionsmodel, som denne gang formuleres som, at *kontantpris*en afhænger af boligstørrelsen, *grundst*ørrelsen, antal værelser og boligens herlighedsværdi. Vi kan også skrive det som, at *kontantpris*en er en funktion af de øvrige variable:

kontantp = f(*boligst, grundst, værelser, kvalitet)*

I SAS gør vi, som vi gjorde i forrige underafsnit 3.3.2, men bare med flere forklarende/ uafhængige variable. Åbn først datasættet *Query for boligdata* og vælg derefter *Analyze* > *Regression* > *Linear Regression…* Vores *Dependent variable* er igen *kontantp,* og de fire øvrige ovenfor nævnte variable er *Explanatory variables*:



I princippet kan vi trykke *Run* nu og få vores output, men vi skal lige huske at tjekke vores forudsætninger også. Forudsætningerne for multipel regressionsanalyse er opsummeret nedenfor:

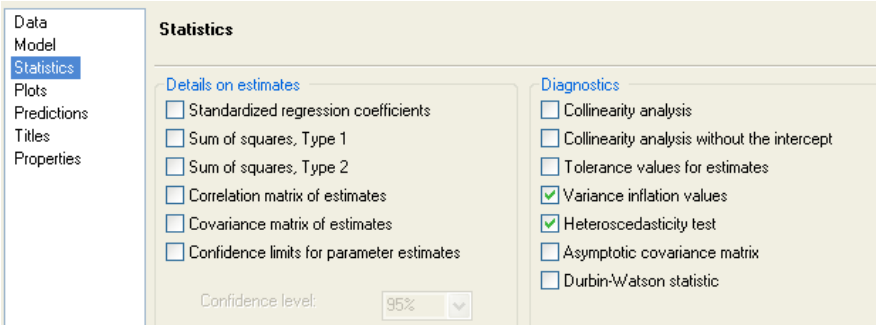
a. <u>Normalfordelte residualer:</u> På samme måde, som vi tjekkede *kontantp* med **histogram** og **QQ-plots** tilbage i underafsnit 2.2.3, skal vi her have tjekket **residualerne**.

b. <u>Ens varians af residualerne:</u> Tjek dette via *Statistics* > *Diagnostics* > *Heteroskedasticity test* samt et plot af **standardized residuals vs. predicted values**.

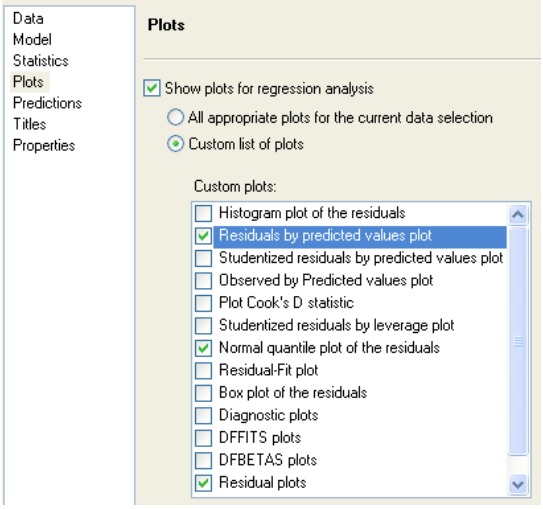
c. <u>De forklarende variable må godt være korellerede, men ikke for meget</u>. Det måles med *Variance inflation values*, som **skal være under 5**. Dem får man i SAS via *Statistics* > *Diagnostics* > *Variance inflation values*.

d. Residualerne skal være uafhængige af de forklarende variable, og der må ikke være trends: Tjek dette ved at se på **plots af residualerne** i forhold til de enkelte forklarende variable i modellen. Der må ikke være mønstre.

For at opsummere, hvad vi mangler at angive i SAS, mangler vi bare at sætte flueben ved de to valgmuligheder under *Statistics*-menuen og derefter at vælge plots. Det første gøres ved at vælge *Statistics* ude til venstre og derunder vælge *Heteroskedasticity test* og *Variance inflation values*, som det er vist nedenfor:

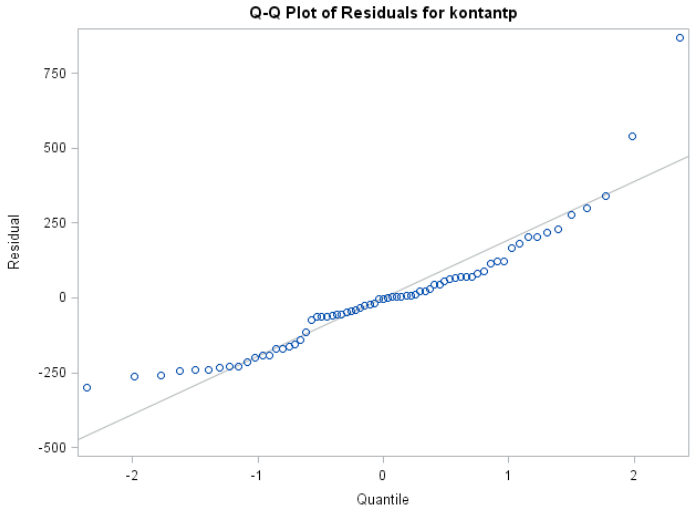


Plots vælges ved at vælge *Plots*-menuen. Som standard får man rigeligt med plots, og vi vil faktisk vælge lidt færre for overskuelighedens skyld. Vi vælger derfor *Residuals by predicted values plot*, *Normal quantile plot of the residuals* (QQ-plot) og *Residual plots*:



Nu er vi klar til at trykke *Run*, hvorefter vi får vores regressionsanalyse og forudsætningstjeks output. Lad os se på forudsætningerne først:

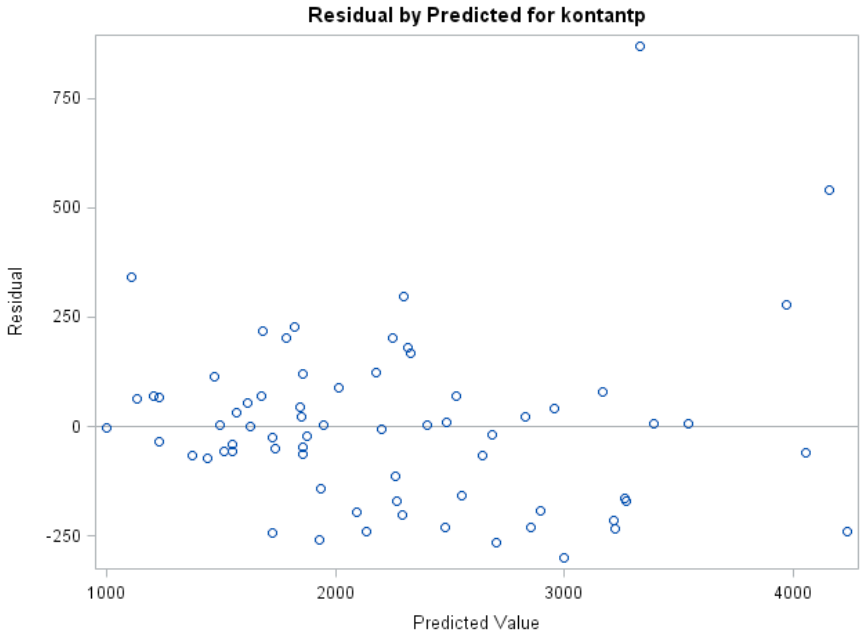
a. QQ-plottet af residualerne indikerer, at denne forudsætning faktisk ikke er opfyldt pga. nogle enkelte ekstreme observationer. Da det er boligmarkedet, kan disse yderpunkter karakteriseres som hhv. "håndværkertilbud" i den ende, hvor prisen er lavere end modellen forudsiger (husk, at vi ikke har en variabel med, der måler boligens stand), og fx kvalitetsbyggeri ud over det sædvanlige i den øvre ende (vores kvalitetsvariabel måler herlighedsværdien af husets beliggenhed, men ikke selve husets "liebhaverværdi"). Her går vi dog videre, som om forudsætningen var opfyldt (hvis du anvender logaritmen til boligprisvariablen *kontantp,* som vi omtalte i underafsnit 2.3.5, i stedet for *kontantp* i regressionsmodellen, så løser det faktisk problemet – prøv evt. selv!):



b. *Heteroskedasticity test* er det, som i fagbøger kaldes *White's* test, og det tester, om residualerne har nogenlunde ens varians for de forskellige værdier af den afhængige variabel *kontantp*. Udgangspunktet (H0) er, at der ikke er problemer. Outputtet fra testen er nedenfor og konkluderer, at der ikke er problemer, eftersom p-værdien er på 0,3120 som er højere end vores signifikansniveau på α=0,05:

Test of First and Second Moment Specification		
DF	Chi-Square	Pr > ChiSq
14	16.02	0.3120

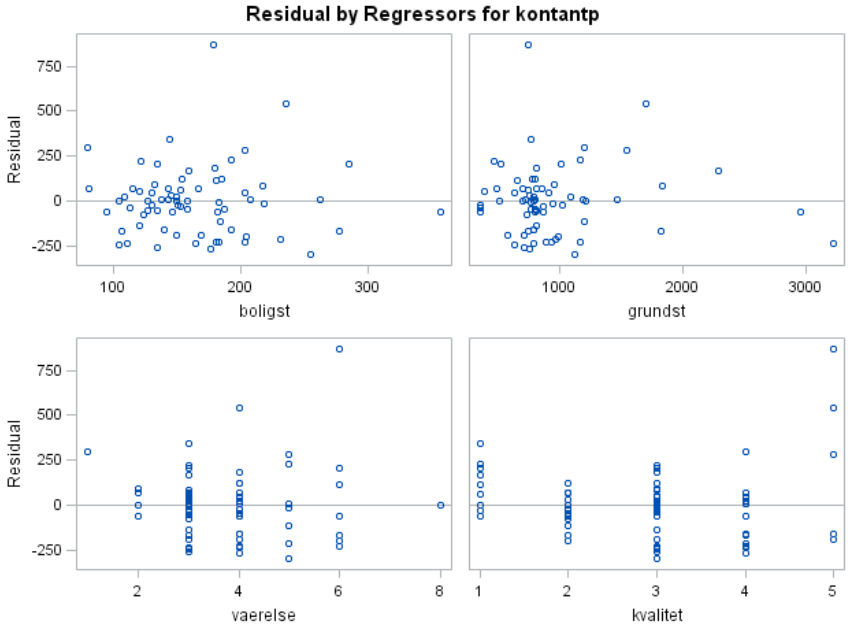
Vi kan også se det grafisk på plottet af de standardiserede residualer i forhold til de boligpriser, vores model estimerer:



Her kunne det måske godt se ud, som om variansen ændrer sig lidt i takt med, at Predicted Value stiger. Det skyldes givetvis de enkelte meget høje residuals, som forvrider billedet lidt. En løsning ville være at bruge ln(*kontantp*), og hvis du er flittig, kan du jo prøve at gøre det (det fikser faktisk alle potentielle forudsætningsproblemer). Her antager vi dog for enkelhedens skyld, at denne forudsætning også er opfyldt.

c. *Variance inflation*-værdierne er alle under 5, så den forudsætning er *OK* (se dem i *Parameter Estimates*-tabellen nedenfor).

d. Uafhængige residualer tjekkes i graferne nedenfor. Der er lidt problemer i forhold til *kvalitet*-variablen og måske også i forhold til nogle af de andre, men i det store hele vurderes det, at forudsætningen er tilstrækkeligt opfyldt:



Vi er nu klar til at aflæse regressionsresultatet. Nedenfor ses det i *ANOVA*-tabellen, at modellen overordnet set er meget signifikant med en p-værdi på under 0,0001, som er meget under vores signifikansniveau α=0,05. Vi kan også se, at vores nye model kan forklare 94% af prisdannelsen på boligerne i stikprøven (R2-værdien). I nederste tabel kan vi se, at alle de forklarende variable også er meget signifikante, alle med en p-værdi tæt på nul:

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	4	41649889	10412472	260.29	<.0001
Error	64	2560222	40003		
Corrected Total	68	44210111			

Root MSE	200.00866	R-Square	0.9421
Dependent Mean	2255.31884	Adj R-Sq	0.9385
Coeff Var	8.86831		

Parameter Estimates						
Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|	Variance Inflation
Intercept	1	-886.63643	109.45597	-8.10	<.0001	0
boligst	1	4.90613	0.70250	6.98	<.0001	2.11771
grundst	1	0.75071	0.05411	13.87	<.0001	1.30038
vaerelse	1	87.47838	26.80101	3.26	0.0018	1.81640
kvalitet	1	452.45268	21.52911	21.02	<.0001	1.00283

Parameterestimaterne ovenfor kan i forhold til effekten af boligstørrelsen på boligprisen aflæses som, at hvis boligstørrelsen stiger med 1 m^2, mens alle de øvrige variable forbliver konstante, så stiger boligens pris med 4.900 kr. – hvilket jo kun er knap det halve af de 9.500 kr., vi fandt i den simple lineære regressionsanalyse i foregående underafsnit 3.3.2. Resultatet i denne multiple regressionsanalyse er mere korrekt, og det viser bare vigtigheden af at huske at få alle relevante variable med.

De øvrige parameterestimater fortolkes som, at 1 m^2 øget grundareal resulterer i en forventet prisstigning på 750 kr., og at et ekstra værelse uden øget boligstørrelse resulterer i en forventet prisstigning ca. 87.000 kr. En ændring i boligens herlighedsværdi er associeret med en virkelig stor prisstigning på knap ½ mio. kr. for hvert forbedringstrin på skalaen fra 1 til 5. Så hvis man bor ved siden af en fabrik, der pludselig uventet lukkes og bliver lavet om til en smuk naturpark, så er det virkelig noget, der kan ses på boligpriserne i området!

Det er ret sjældent, at man har så høj en forklaringsgrad (R2), som vi har i dette eksempel. Men vi vil alligevel prøve, om vi kan forbedre modellen yderligere i næste underafsnit − og samtidig introducere nogle andre nyttige regressionsteknikker.

3.3.4. Regression med mere avancerede variable

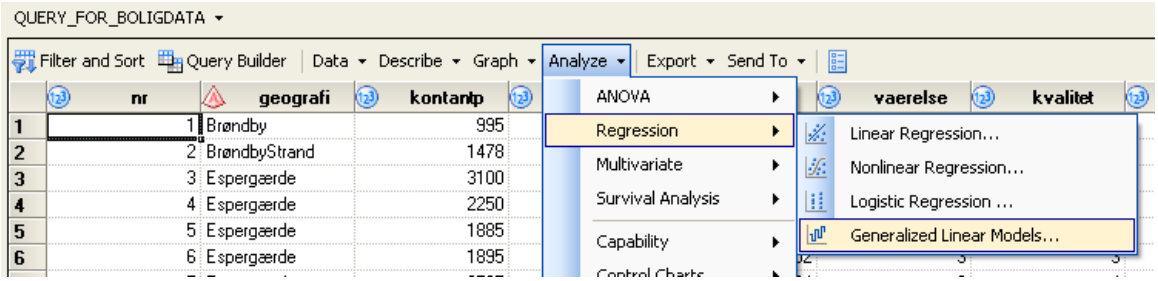
I dette underafsnit vil vi forsøge at forbedre modellen fra den multiple regressionsanalyse i forrige underafsnit 3.3.3. Vi vil prøve at tilføje tre nye typer variable til vores regressionsanalyse. De tre forskellige variable er:

i. **En nominel forklarende variabel:** Vi vil tilføje områdevariablen og se, om der er forskel på boligpriserne mellem nord og syd, når alle de andre variable også tages i betragtning.

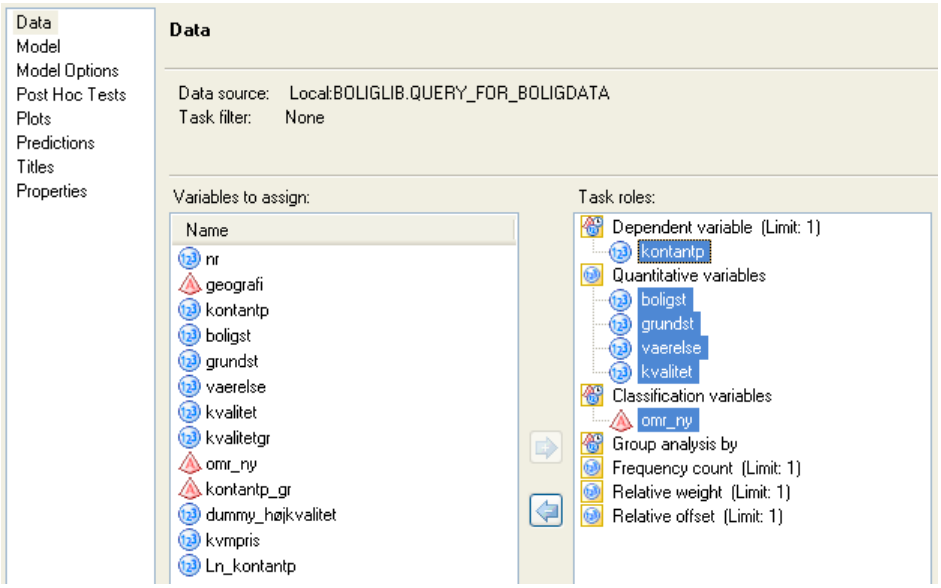
ii. **En kvadreret variabel**, dvs. en variabel ganget med sig selv. Det giver mulighed for, at en variabel kan have en enten aftagende eller stigende effekt. Vi vil prøve at medtage *grundst*grundst* (kan også skrives som *grundst*^{2}) for at undersøge, om prisen for en yderligere kvadratmetergrundstørrelse fx aftager, efterhånden som grundstørrelsen kommer over et vist niveau. Det kan jo være, at ens have kan blive så stor, at glæden ved den ekstra plads bliver trukket gevaldigt ned, når græsplænen skal klippes en gang om ugen i sommerhalvåret.

iii. **Interaktion** mellem boligstørrelsen og grundstørrelsen inkluderes også ved at gange dem sammen: *boligst*vaerelse*. Det kan selvfølgeligt gøres direkte i analysen i SAS. Idéen med at inkludere en interaktion er, at det giver mulighed for, at værdien af et ekstra værelse kan afhænge af boligstørrelsen. Måske er værdien af et ekstra værelse fx størst i små boliger, og det kan vi så teste med en interaktionsvariabel.

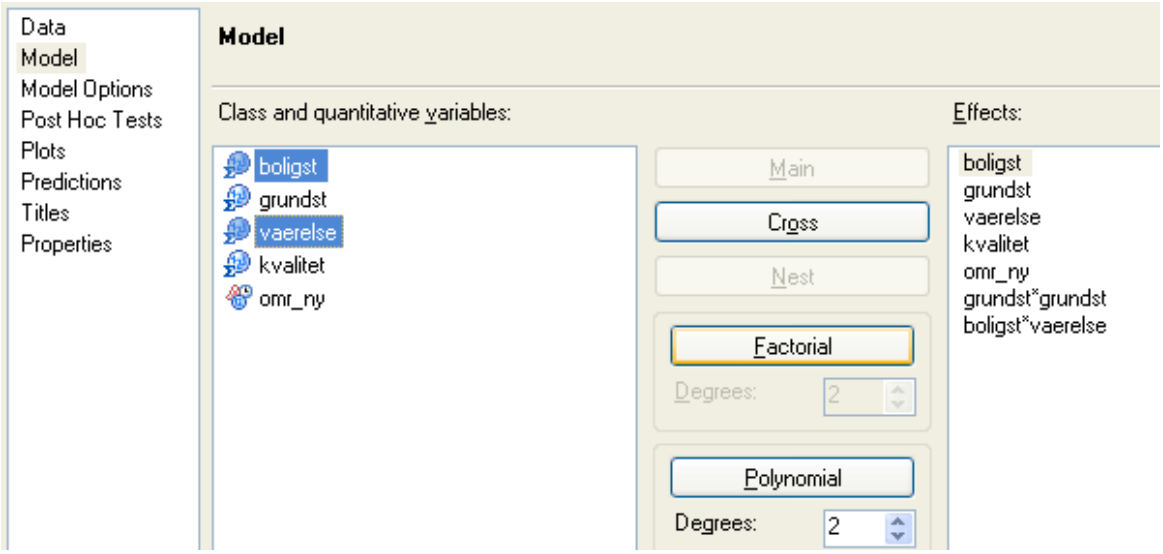
Vi er nu klar til at lave analysen i SAS. Det gøres ved at åbne vores datasæt *Query for boligdata* og denne gang vælge *Analyze* > *Regression* > *Generalized Linear Models*… som vist nedenfor. Dette er en anden analysetype i SAS, da vi denne gang også skal have den kategoriske variabel *omr_ny* med, og det tillader *Linear Regression*… ikke:



Herefter inkluderes de samme fem variable i analysen som i forrige underafsnit 3.3.3 under hhv. *Dependent variable* (*kontantp*) og *Quantitative variables* (*boligst*, *grundst*, *vaerelse* og *kvalitet*) samt denne gang også *Classification variable* (*omr_ny*):



Vi skal også specificere modellen, og det gøres under *Model*-valgmuligheden, hvor alle variablene først markeres, hvorefter der trykkes *Main*. Derefter markeres *grundst*, og der trykkes på *Polynomial*. Slutteligt markeres *boligst* og *vaerelse* samtidig (hold Ctrl-knappen på tastaturet nede, mens de vælges), og der trykkes *Cross* som vist på billedet nedenfor:



Vi behøver ikke at ændre mere – standardindstillingerne er *OK*. Tryk derfor på *Run*, hvorefter nedenstående output fremkommer:

Analysis Of Maximum Likelihood Parameter Estimates								
Parameter		DF	Estimate	Standard Error	Wald 95% Confidence Limits		Wald Chi-Square	Pr > ChiSq
Intercept		1	-844.373	259.1977	-1352.39	-336.354	10.61	0.0011
boligst		1	3.7153	1.8404	0.1081	7.3224	4.08	0.0435
grundst		1	0.9858	0.1921	0.6093	1.3623	26.34	<.0001
vaerelse		1	54.6564	63.2981	-69.4056	178.7183	0.75	0.3879
kvalitet		1	453.0193	21.3155	411.2416	494.7970	451.69	<.0001
omr_ny	Nord	1	-1.8590	48.8270	-97.5583	93.8402	0.00	0.9696
omr_ny	Syd	0	0.0000	0.0000	0.0000	0.0000	.	.
grundst*grundst		1	-0.0001	0.0001	-0.0002	0.0000	1.66	0.1977
boligst*vaerelse		1	0.2066	0.3735	-0.5254	0.9387	0.31	0.5801
Scale		1	190.2120	16.1919	160.9826	224.7486		

Som det fremgår af p-værdierne, er alle de tre tilføjede variable insignifikante. Derfor skal de fjernes fra modellen én ad gangen, indtil alle variable er signifikante. Variablen *vaerelse* bliver signifikant igen, når interaktionsvariablen bliver fjernet fra modellen – prøv selv at tjekke. Så vi ender faktisk på den samme model som i forrige underafsnit 3.3.3. Det vil i forhold til områdevariablen sige, at der ikke er signifikant prisforskel mellem huse nord og syd for København, når alle de øvrige aspekter af prisdannelsen på boligmarkedet tages med i betragtningen. At de andre nye variable også er insignifikante betyder, at også den marginale effekt af en ekstra kvadratmeter *grundst*ørrelse er konstant, og at prisen på et ekstra værelse ikke afhænger af boligstørrelsen.

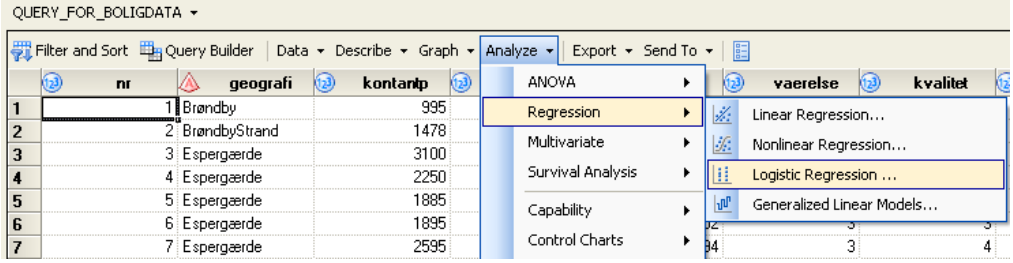
I outputtet kommer der også diverse plots, som alle indikerer, at forudsætningerne er opfyldte (da der ikke umiddelbart er mønstre i dataene i forhold til residualerne).

Nu ved du, hvordan regressionsanalyser laves i SAS.

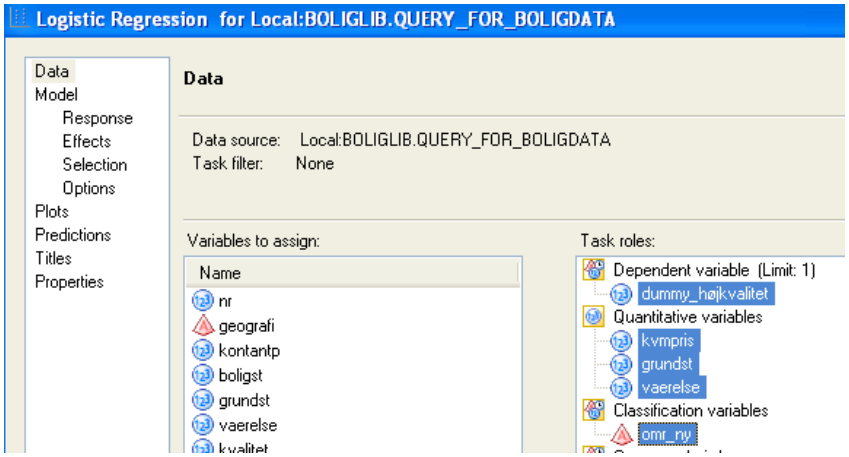
3.3.5. Logistisk regression

Logistisk regression er en teknik til at forudsige en *enten/eller*-variabel. I vores eksempel vil vi prøve at lave en model, der skal forudsige, om en bolig er i højkvalitetsgruppen eller ej. Hvis den er, så er variablen *dummy_højkvalitet* = 1. Hvis ikke, så er denne variabel = *0*. Vi vil bruge kvadratmeterprisen *kvm-pris* sammen med variablene *grundst*, *vaerelse* og *omr_ny* som forklarende variable i vores analyse.

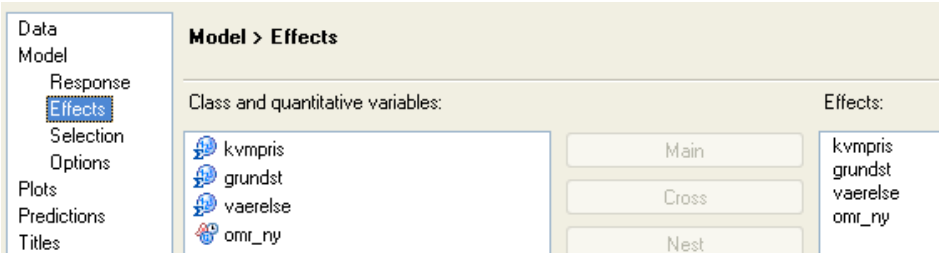
Logistisk regression vælges i SAS ved at åbne vores datasæt *Query for boligdata* og derefter vælge *Analyze* > *Regression* > *Logistic Regression…* som vist nedenfor:



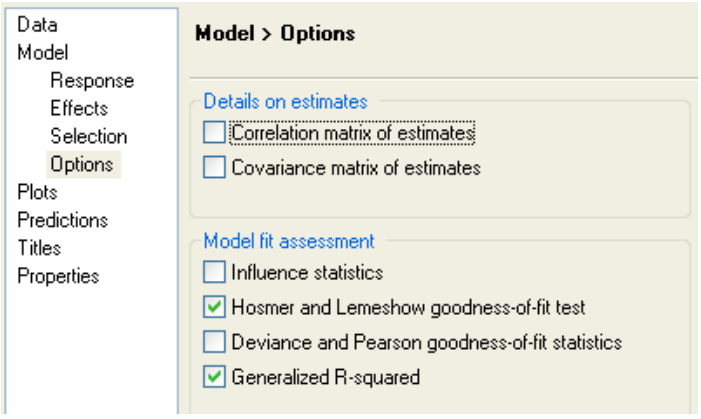
Vælg nedenstående variable til de på billedet angivne *Task roles*:



Træk alle variablene over som *Main* effects:



Under *Options* vælges *Hosmer and Lemeshow goodness-of-fit test* og *Generalized R-squared*:



Tryk nu *Run*, hvorefter outputtet fremkommer. Det består af et par tabeller og en lang stribe grafer. De første tabeller nedenfor viser, at: 1) AIC falder, når de forklarende variable inkluderes i modellen, hvilket betyder, at de forklarende variable faktisk kan forklare, hvorvidt en given bolig er i højkvalitetsgruppen eller ej. 2) R^2 ligger et sted mellem 48-69%, hvilket umiddelbart er udmærket. 3) Alle tre tests for modellens overordnede signifikans indikerer, at modellen samlet set er signifikant.

Model Fit Statistics		
Criterion	Intercept Only	Intercept and Covariates
AIC	85.079	47.102
SC	87.313	58.273
-2 Log L	83.079	37.102

R-Square	0.4864	Max-rescaled R-Square	0.6948

Testing Global Null Hypothesis: BETA=0			
Test	Chi-Square	DF	Pr > ChiSq
Likelihood Ratio	45.9765	4	<.0001
Score	31.2798	4	<.0001
Wald	13.7535	4	0.0081

Når vi ser på de enkelte forklarende variable i tabellen nedenfor, kan vi se, at kvadratmeterprisen er signifikant, og at antal værelser er næsten signifikant med en p-værdi på 0,0681. Men de to øvrige variable *grundst* og *omr_ny* er ikke signifikante:[5]

Analysis of Maximum Likelihood Estimates						
Parameter		DF	Estimate	Standard Error	Wald Chi-Square	Pr > ChiSq
Intercept		1	17.8670	4.9675	12.9369	0.0003
kvmpris		1	-0.9462	0.2640	12.8461	0.0003
grundst		1	0.000416	0.00103	0.1612	0.6880
vaerelse		1	-0.9334	0.5117	3.3276	0.0681
omr_ny	Nord	1	-0.0107	0.4978	0.0005	0.9828

5. Parameterestimaterne i logistisk regression kan ikke fortolkes ligeså enkelt som ved almindelig regression, så det vil vi overlade til din teoretiske statistikbog at introducere.

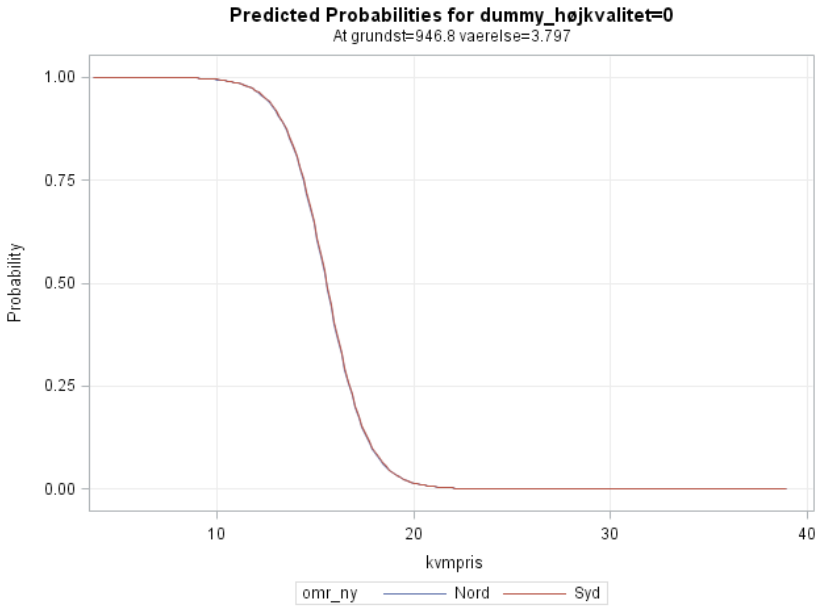
Hosmer and Lemeshow, som er testet nedenfor, er ikke signifikant, hvilket betyder, at det ikke kan afvises, at dataene er brugbare i en logistisk regression. Så det er jo godt for vores analyses brugbarhed:

Hosmer and Lemeshow Goodness-of-Fit Test		
Chi-Square	DF	Pr > ChiSq
13.1136	8	0.1080

Alle plots i outputtet (bortset fra sidste plot) bruges til at tjekke forudsætninger. Der må ikke være iøjnefaldende mønstre i dataene, hvilket vurderes *OK* i vores tilfælde.

Det sidste plot viser effekten af vores mest signifikante forklarende variabel *kvm-pris* på sandsynligheden for, at boligen er i højkvalitetsgruppen. Plottet viser, at når kvadratmeterprisen er over ca. 16.000 kr./m^2, så gætter modellen på, at boligen er i højkvalitetsgruppen (dvs. sandsynligheden for, at variablen *dummy_højkvalitet*=0 falder).

Plottet viser også, at kvadratmeterprisen har samme effekt nord og syd for København – fordi den blå og røde linje praktisk talt ligger lige oven i hinanden.



Det var logistisk regression på den lette måde i SAS.

3.4. Fordelingstests

Noget af det første, man skal igennem på mange indledende statistikkurser, er beregning af binomiale sandsynligheder, hypergeometriske sandsynligheder, normalfordelingssandsynligheder etc. I praksis er det ikke noget, man gør så meget, og derfor er disse funktioner pakket lidt af vejen i SAS, men de er stadig tilgængelige.

Nedenfor kommer et eksempel på tre forskellige fordelingstests i relation til vores boligdatasæt samt eksempler på, hvordan de beregnes i SAS.

3.4.1. Binomialfordelingstest

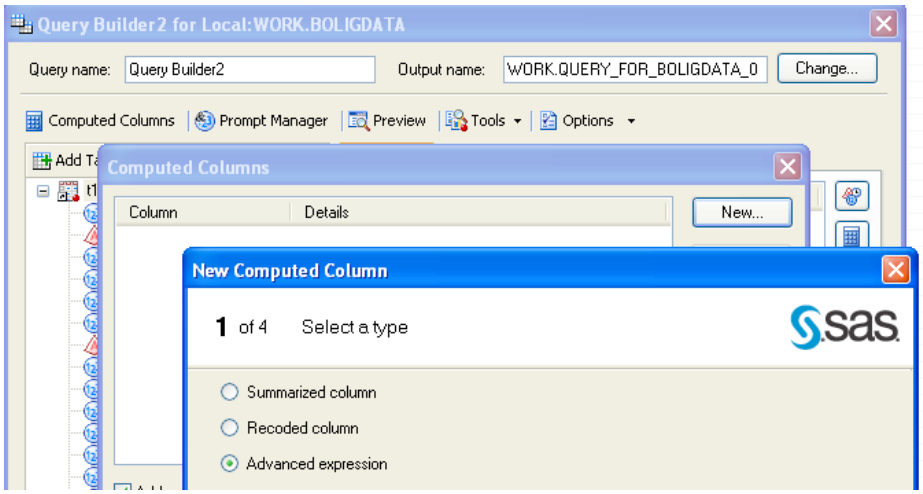
Et eksempel på en binomial sandsynlighed i dette boligdatasættet kunne være følgende: Antag, at du er på jagt efter et hus i Københavnsområdet. Før du lige åbner postkassen og tager boligavisen med ind til morgenkaffen, vil du lige beregne sandsynligheden for, at der denne givne søndag i år 2000 maks. er 10 boliger i avisen, som du kunne være interesseret i, ud af avisens 100 boligannoncer (du kan normalt nå at køre ud at se 10 boliger på en søndag). Du ved fra tidligere erfaringer, at du ved første øjekast kan lide ca. ét ud af fem huse, du ser i denne avis, altså 20% (eller 0,2 udtrykt som en sandsynlighed mellem 0 og 1), og sådan har det været, lige siden du begyndte at holde øje med boligannoncerne i denne avis. Antag også, at der er virkelig godt gang i boligmarkedet i år 2000, og at det derfor er nye boliger, der er i avisen hver søndag.

Opsummering af opgaven

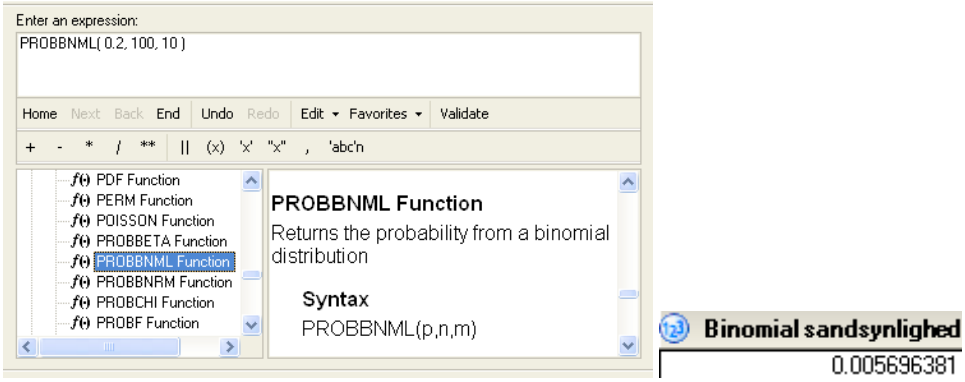
Først opstilles udtrykket for den binomiale sandsynlighed med sandsynligheden for, at du kan lide en given bolig $p = 0{,}2$ og stikprøvestørrelsen $n = 100$ (du tjekker de 100 tilfældige boliger, der netop er kommet til salg på denne givne søndag). Da du maks. kan nå at tjekke 10, er din testværdi x =10. Dette udtrykkes alt sammen i SAS i formlen *BIN(p , n , x)* og med de konkrete værdier som *BIN(0,2 , 100 , 10)*.

Løsning i SAS

I SAS kan du beregne hvad som helst via en avanceret formel-editor. Den finder du i dropdownmenuen *Tasks* > *Data* > *Query Builder…*, hvor du så klikker på *Computed Columns* i øverste venstre hjørne. Klik herefter på *New* og vælg *Advanced Expression*, ligesom det er gjort på billedet nedenfor. Klik herefter *Next*.



SAS kalder det ikke *BIN* men *PROBBNML (p, n, m)*, som du kan se af billedet nedenfor. Bemærk, at du sandsynligvis skal bruge punktum i stedet for komma, når du fx skriver sandsynligheden 0,2 (det afhænger af, hvordan din PC er indstillet, da SAS spejler dine lokale sprogindstillinger).



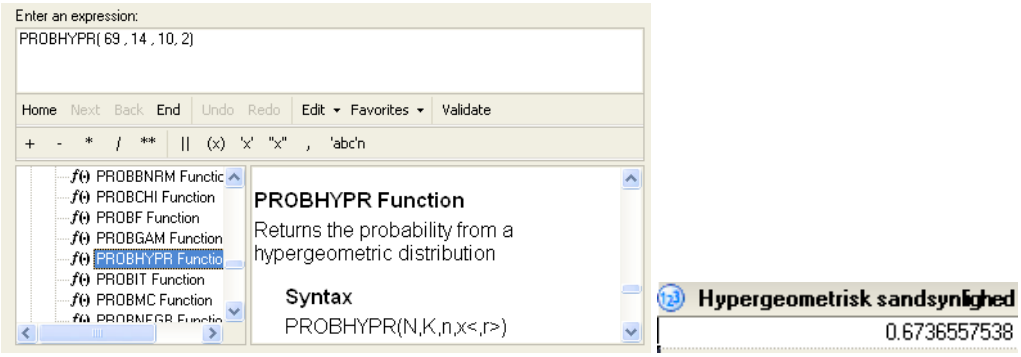
Klik herefter på *Next*, hvor du kan give beregningen et navn (kald den fx "Bin"), klik derefter *Finish* > *Close* og *Run*. Dette resulterer i den beregnede binomiale sandsynlighed på 0,005696381, eller ca. 0,6%. Det er med andre ord uhyre usandsynligt, at der kun er 10 eller færre huse, som du synes er interessante, i Søndagsavisen, som du nu vil ind og læse til morgenkaffen. Du må altså prioritere, hvilke huse du vil ud at se på.

3.4.2. Hypergeometrisk fordelingstest

Lad os antage, at der denne søndag er hele 69 huse, som har din interesse, og at du vil udvælge 10 tilfældige herfra. Du vil selvfølgeligt ikke køre ud at se det samme hus flere gange, og derfor er dette et eksempel på en såkaldt hypergeometrisk sandsynlighedsfordeling i stedet for en binomialfordeling. Du ved fra tidligere erfaringer, at du kan lide ca. ét ud af fem huse, som du kører ud og ser på, altså igen en sandsynlighed på 0,2 ligesom før. Dvs. at 20% af de 69 huse vil falde i din smag, altså 0,2*69=13,8 ≈ 14 af husene. Hvad er sandsynligheden for, at du i dag finder højst to huse, du kan lide, ud af de 10 huse, du tager ud for at tjekke?

Løsning i SAS

Svaret findes næsten på samme måde som ved binomialfordelingseksemplet tidligere. Den anvendte funktion hedder nu bare *PROBHYPR(N, K, n, x)*, hvor *N* er de 69 udvalgte boligannoncer, *K* er det forventede antal af dem, som du vil være virkeligt interesseret i, n er det antal boliger, du kan nå at tjekke i dag, og *x* er det antal, vi tester som øvre grænse. Dvs. den formel, du ender med at have indtastet, hedder *PROBHYPR(69, 14, 10, 2)*. Når du beregner den, giver det 0,67366, dvs. der er ca. 66% sandsynlighed for, at du finder maks. 2 huse, du virkelig godt kan lide i dag, ud af de 10 huse, du planlægger at køre ud at se på. Prøv at se, om du kan komme frem til samme resultat i SAS.



3.4.3. Normalfordelingstest

Lad os antage, at din bank har givet dig lov til at købe hus for maks. 1,7 mio. kr. Du ved fra tidligere, at gennemsnitsprisen på de huse, du er interesseret i, er ca. 2,3 mio. kr., og standardafvigelsen er 0,8 mio. kr. Hvis du i dag finder et hus, du virkelig godt kan lide, på din vej rundt for at se på dine udvalgte huse fra boligavisen, hvad er så sandsynligheden for, at du har råd til det givne hus?

Løsning i SAS

På samme måde som i de to foregående eksempler anvendes den avancerede formel-editor, hvor normalfordelingstesten indtastes som *PROBNORM(x)*. Da *PROBNORM*-formlen anvender standardnormalfordelingen (den med gennemsnit på 0 og standardafvigelse på 1), skal vores prisgrænse på 1,7 mio. kr. ”standardiseres”, siger man. Det gør man ved at trække gennemsnittet fra og dividere med standardafvigelsen, altså ved at sige 1,7 minus 2,3 = -0,6, som så divideres med 0,8, hvilket giver -0,75. Dvs. at det, du skal indtaste i formel-editoren, er *PROBNORM(-0,75)* (eller *PROBNORM ((1,7 – 2,3)/0,8))*, hvilket giver 0,2267. Der er altså knap 23% sandsynlighed for, at du har råd til det givne hus.

Normalfordelings sandsynlighed
0.2266273524

Hermed slutter vores rundtur i SAS-peg og klik. Har du mod på mere, så prøv at læse Kapitel 4 også, som åbner en helt ny verden af dataanalysemuligheder med SAS-programmering – der vil vi som eksempler se nærmere på, hvordan koden til nogle af de analyser vi har lavet via peg og klik, også kunne laves med et par linjers SAS-kode – og derefter evt. automatiseres via anden kode … Mulighederne er ubegrænsede!

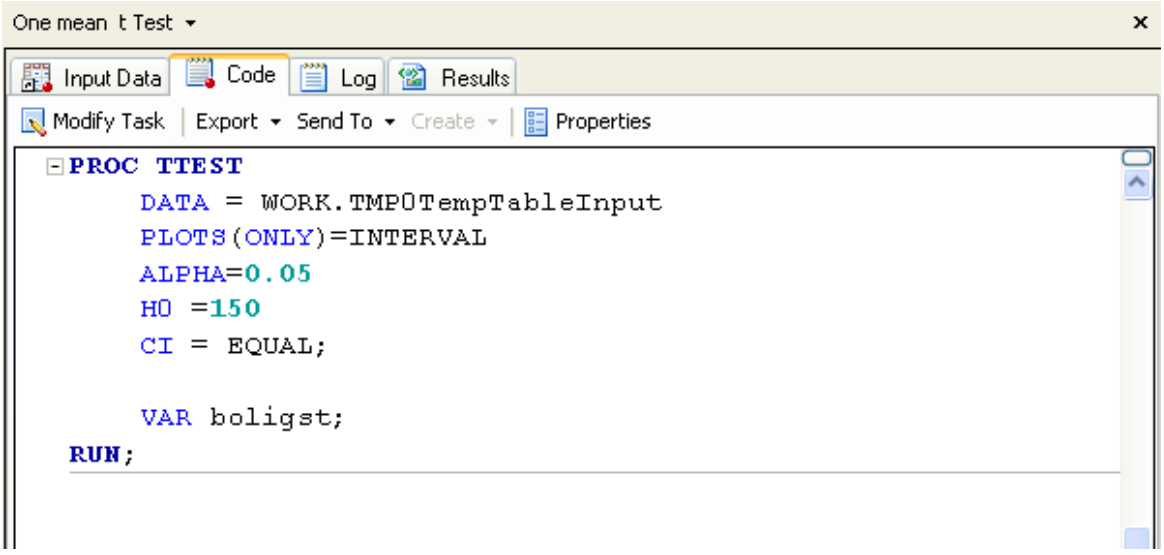
4. INTRODUKTION TIL SAS-PROGRAMMERING

SAS er baseret på programmeringskode, som kører inde bagved. Det er virkelig smart, for det betyder, at hvis dit datasæt bliver opdateret (fx ved en større eller ny stikprøve), kan du også opdatere dine analyser på ingen tid bare ved at vælge *Run* > *Run Process Flow* og så aflæse alle de opdaterede output. Så *Process Flow*-diagrammet giver både det forkromede overblik og betyder også, at du kan tænke i én SAS EG-projektfil til ét studie- eller arbejdsprojekt – og ikke mange forskellige datasæt.

Engang imellem har man dog brug for endnu mere automatisering af sin datahåndtering, og hvis du skulle få den slags behov, kan det godt betale sig, at du bruger lidt tid på at lære lidt SAS-kodning. Det er lidt som at lære et nyt sprog, hvor man dog kan komme langt med at kunne bare en lille smule, og det er også meget lettere at lære i dag end tidligere, fordi SAS automatisk foreslår ord og valgmuligheder. Alt dette ser vi nærmere på nedenfor.

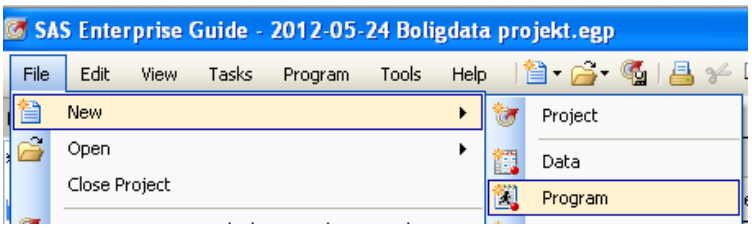
4.1. Automatisk kode, integreret hjælp og code completion

For at snuse lidt til SAS-programmering er koden til et par af de forskellige tests, vi har lavet i denne e-bog, tilføjet her som eksempler. Husk, at du også altid kan finde koden til de forskellige *task*s (=handlinger) i SAS på *Code*-fanen, som automatisk genereres, hver gang du kører en *task*. Den automatisk genererede SAS-kode er dog i de fleste tilfælde lidt mere kompleks, end den behøver at være – fordi den jo skal være meget generel og virke i alle tænkelige kombinationer. Nedenfor ser du noget af koden fra *Code*-fanen til vores første test i denne e-bog (test af ét gennemsnit fra underafsnit 3.1.1 på side 38). Man skal lige scrolle lidt ned for at kunne se SAS-koden nedenfor:

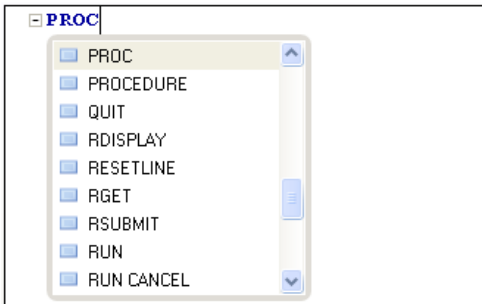


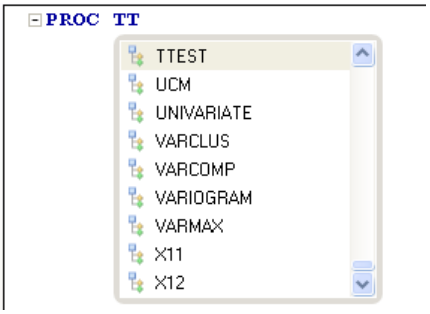
Det, du ser ovenfor, er den SAS-kode, der genererer et *One mean t Test*. Der ligger andet kode både over og under – kode, som ikke behøves, hvis du kører testen selv via SAS-kode. Det er det, vi nu vil prøve at gøre.

OK – nu har vi set koden. Lad os prøve at skrive koden selv i et nyt SAS-program inden i vores SAS-projekt. Åbn først et nyt 'program' via *File* > *New* > *Program*:



I det tomme vindue går du nu i gang med at skrive koden til t-testen. Koden på foregående side starter med *PROC*, som er en forkortelse for en procedure, som koden skal udføre. Det er for øvrigt fuldstændig lige meget, om du anvender store eller små bogstaver i koden:

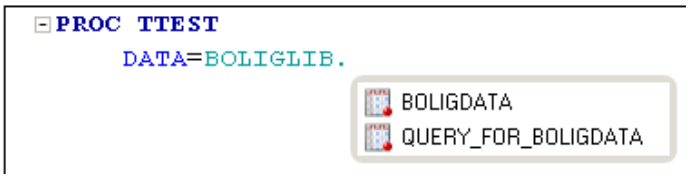




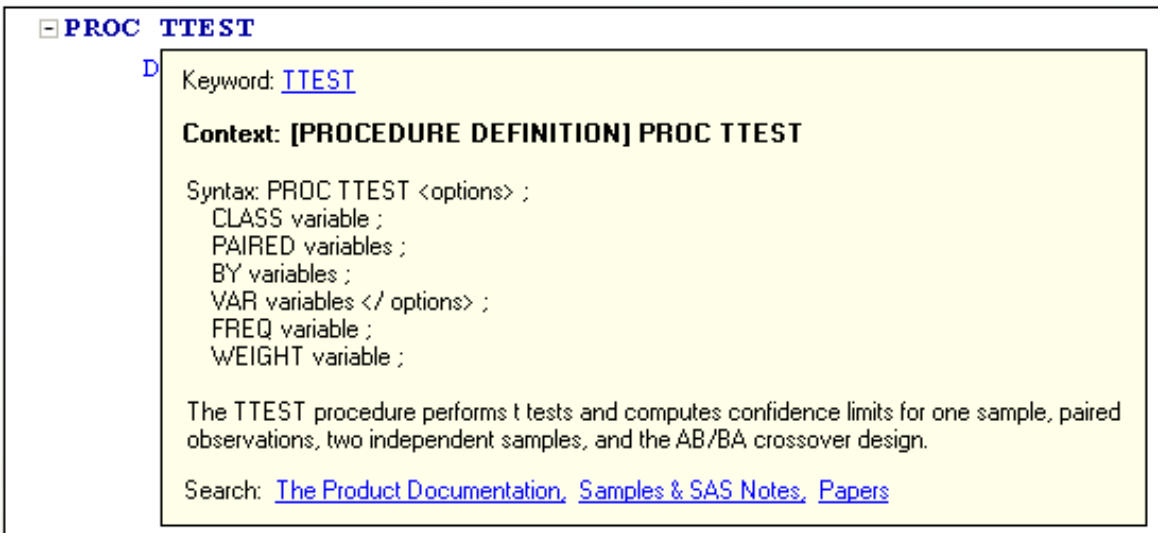
Som du kan se ovenfor og på din egen skærm, går SAS straks i gang med at gætte, hvad du prøver at kode. Det hedder *Code Completion* og er en stor hjælp til at lære at kode og undgå fejl i sin kode. Når du fortsætter med at skrive *TTEST*, ser du igen, hvor hjælpsom SAS er med kodningen. Du kan bare trykke på *Enter* eller på mellemrumstasten på dit tastatur, når SAS har gættet, hvad du gerne vil skrive.

Nu skal vi til at skrive *DATA* = <…>

I stedet for at skrive *DATA=WORK.*<…> kan vi jo passende anvende vores eget data *library BOLIGLIB*, for det er jo her, vores datasæt *Query*_For_*Boligdata* ligger (SAS anvender et midlertidigt datasæt – det er ikke relevant her). Når vi har skrevet vores *library* navn, fortæller SAS os, hvilke datasæt vi kan vælge imellem i *library*'et:



Hvis vi nu ikke havde forrige sides kode som skabelon, kunne man godt komme i tvivl om, hvad man skulle skrive som det næste. Her har SAS heldigvis integreret hjælp, ligesom du kender det fra Microsoft Office 2007 og frem – dvs. når du holder musen oven på de ord, som SAS genkender (alle dem, der er blå i koden), så fortæller SAS, hvordan koden skal struktureres. I billedet nedenfor er musen holdt oven på *TTEST*:



De første fire linjer, vi mangler at skrive, er under <*options*> efter *PROC TTEST* i syntaksen[6] ovenfor. Den femte linje vi mangler, er den med *VAR variables.* Resten har vi ikke brug for.

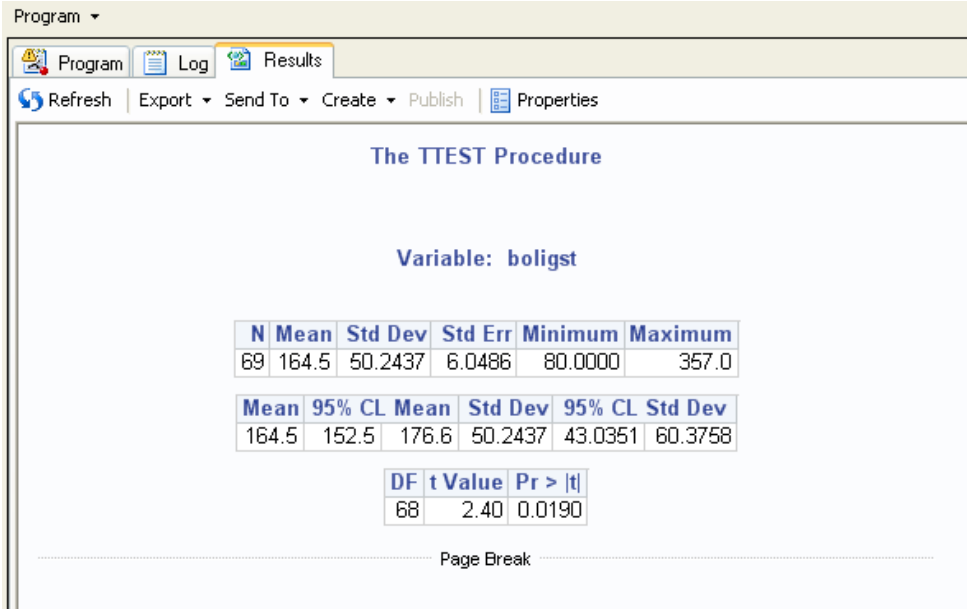
Vi skriver altså videre på koden, indtil den ser ud som nedenfor:

```
PROC TTEST
     DATA=BOLIGLIB.QUERY_FOR_BOLIGDATA
     PLOTS(ONLY)=INTERVAL
     ALPHA=0.05
     H0 =150
     CI = EQUAL;

     VAR boligst;
RUN;
```

6. Syntaks betyder kodestruktur.

Prøv at trykke på *Run*- knappen nu. Herefter fremkommer næsten det samme output, som du fik, da vi kørte vores oprindelige *One mean t-test*. Den eneste forskel er, at vores plot ikke er kommet med ud denne gang:

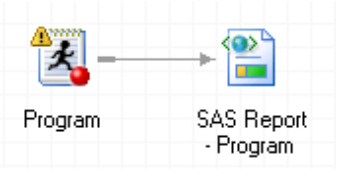


N	Mean	Std Dev	Std Err	Minimum	Maximum
69	164.5	50.2437	6.0486	80.0000	357.0

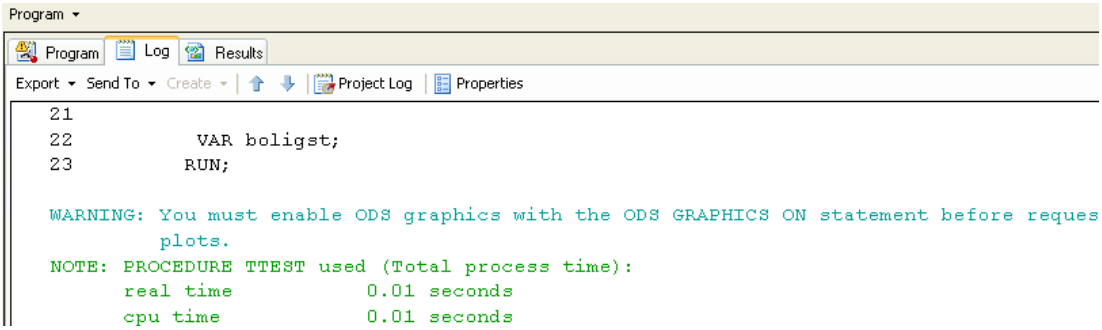
Mean	95% CL Mean		Std Dev	95% CL Std Dev	
164.5	152.5	176.6	50.2437	43.0351	60.3758

DF	t Value	Pr > |t|
68	2.40	0.0190

Tilbage i *Process Flow*-diagrammet kan du også se, at der nu er kommet et *Program* med på listen af *Tasks*, vi har udført:



SAS viser en lille advarselstrekant i højre hjørne af programmet ovenfor, og det har med vores manglende plots at gøre. For hvorfor kom de egentlig ikke med ud, når vi nu fik dem sidst? Det har med koden at gøre – vi glemte lige at få en lille ting i forhold til grafer med, da vi kopierede koden før, nemlig et såkaldt *ODS*-statement, som er SAS' kodesprog til at "kalde" grafer.[7] Vi kan også lige prøve at kigge i *loggen* fra vores program. En *log* er altid lidt omfattende, så måden at gribe den an på er at skimme den igennem for at se, hvor der står nogle kommentarer med andre farver end grøn. Det er gjort nedenfor:



Som du kan se i loggen ovenfor, så giver SAS os en *Warning*, hvor der ret tydeligt står, hvad vi skal gøre. Hvis du kigger i den originale kode fra vores *One mean t-test* genereret af SAS, kan du finde et sted, hvor der står *ODS GRAPHICS ON* først i koden og igen *ODS GRAPHICS OFF* i slutningen af koden. Lad os prøve at tilføje det til vores kode fra før.

7. At "kalde" i kodesprog betyder at bede computeren om at gøre et eller andet, fx lave grafer.

Resultatet skal se ud som nedenfor, hvor der altså skal stå *ODS GRAPHICS ON* før vores øvrige kode og *ODS GRAPHICS OFF* til sidst i koden. Det er ligegyldigt, om du bruger store eller små bogstaver:

```
ODS GRAPHICS ON

PROC TTEST
    DATA=BOLIGLIB.QUERY_FOR_BOLIGDATA
    PLOTS(ONLY)=INTERVAL
    ALPHA=0.05
    H0 =150
    CI = EQUAL;

    VAR boligst;
RUN;

ODS GRAPHICS OFF
```

Prøv at trykke på *Run*-knappen nu – så vil du blive introduceret til den nok hyppigste kodefejl i SAS: at man har glemt semikolon efter sin kode:

```
Program*   Log
Export ▾  Send To ▾  Create ▾  Project Log  Properties
15          ODS graphics ON
16
17          PROC TTEST
            ____
            22
17          PROC TTEST
            ____
            202
ERROR 22-322: Syntax error, expecting one of the following: ;, /.

ERROR 202-322: The option or parameter is not recognized and will be ignored.
```

Vi har glemt et semikolon efter vores *ODS*-statements. Det betyder, at SAS læser videre og forventer, at der kommer mere i forhold til *ODS*-kommandoen i stedet for at begynde på at læse en ny kommando ved *PROC* på næste linje. Denne fejl er en, som alle hopper i et par gange, når man begynder at kode SAS – ja, den er faktisk så almindelig, at der er dem, der påstår, at SAS er en forkortelse for *Sæt Altid Semikolon*! :-) Hvis du sammenligner forrige skærmbillede med dem tidligere, kan du også se, at teksten med *PROC TTEST* i forrige billede ikke længere er mørkeblå – dvs. ikke genkendt af SAS som en kommando. Loggen ovenfor indikerer, hvad problemet er (*SAS is expecting one of the following*: ; …), men det står ikke direkte.

Lad os skynde os at fikse det med to semikolonner – et efter hvert *ODS-statement*, som det er gjort nedenfor:

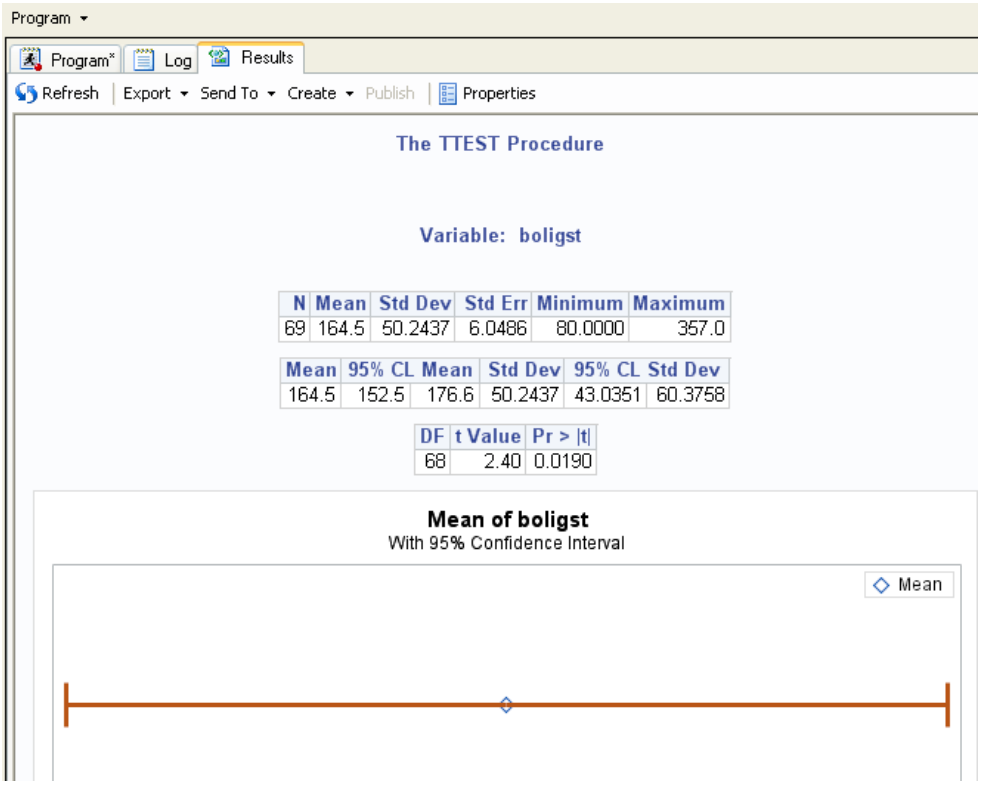
```
ODS GRAPHICS ON;

PROC TTEST
     DATA=BOLIGLIB.QUERY_FOR_BOLIGDATA
     PLOTS(ONLY)=INTERVAL
     ALPHA=0.05
     H0 =150
     CI = EQUAL;

     VAR boligst;
RUN;

ODS GRAPHICS OFF;
```

Prøv at trykke på *Run*-knappen nu, hvorefter vi får vores ønskede output frem – denne gang med grafen:



Nu er ovenstående graf jo ikke verdens mest interessante graf, men nu har du lært, hvordan du får grafer med i dit SAS-output. Grunden til, at grafer ikke kommer med af sig selv, er, at koden kører hurtigere uden. Og hvis det nu var et kæmpe datasæt med mange kommandoer og beregningstunge grafer, så er det jo praktisk nok, at man ikke skal vente på grafer, som man måske er ligeglad med, ikke?

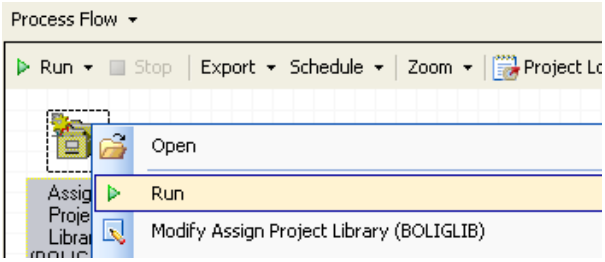
Prøv nu at gemme dit projekt (tryk evt. bare på *Ctrl+s* på dit tastatur for at gemme) og luk derefter SAS helt ned. Prøv så at åbne SAS igen og åbn dit program igen via *Process Flow*-diagrammet. Tryk derefter på *Run*-knappen. Herefter kommer loggen frem med en fejlmelding:

```
Program | Log
Export ▾ Send To ▾ Create ▾ | Project Log | Properties
15          ODS graphics ON;
16
17          PROC TTEST
18           DATA = BOLIGLIB.QUERY_FOR_BOLIGDATA
19           PLOTS(ONLY)=INTERVAL
ERROR: Libname BOLIGLIB is not assigned.
20           ALPHA=0.05
21           H0 =150
22           CI = EQUAL;
23
24           VAR boligst;
25          RUN;

NOTE: The SAS System stopped processing this step because of errors.
```

Problemet er, at SAS ikke kender dit *libname*. På din PC ville det måske være smart, hvis SAS huskede dine *libnames*, men i for store organisationer med utallige SAS-brugere på det samme SAS-system ville det være lidt problematisk, hvis alle brugere skulle anvende forskellige *libnames* altid, og hvis et *libname*, der blev genbrugt, ellers kunne pege forskellige steder hen.

Løsningen er dog enkel: Vi skal bare tilføje koden fra vores *libname statement* øverst i vores program. Vores SAS *library* var faktisk det allerførste, vi udførte i SAS i denne e-bog, tilbage i underafsnit 2.1.1 på side 9. Vi finder koden ved at køre vores *Assign Library task* igen, som er den øverste *task* i vores *Process Flow* diagram:



Herefter fremkommer fanerne *Code* og Log, hvor vi vælger kodefanen. Her står den kode, vi skal tilføje, øverst i vores program for altid at have vores *libname* defineret. Din sti til dataene "C:\..." ser helt sikkert anderledes ud – den peger jo på der, hvor du har gemt dine data, og det ville jo være sært, hvis du havde lagt dine data helt det samme sted, som jeg har. Derfor skal du ikke ændre i koden i dit *libname statement*, men kopiere det, som det er på din PC. Jeg viser bare, hvordan det ser ud på min skærm:

```
Assign Project Library (BOLIGLIB)
Code | Log
Modify Task | Export ▾ Send To ▾ Create ▾ | Properties
LIBNAME BOLIGLIB BASE "C:\Documents and Settings\Administrator\Desktop\SAS Projects" ;
```

Hvis du tager din *libname*-kode og kopierer ind øverst i dit program, ser det ud som nedenfor:

```
LIBNAME BOLIGLIB BASE "C:\Documents and Settings\Administrator\Desktop\SAS Projects" ;

ODS graphics ON;

PROC TTEST
     DATA = BOLIGLIB.QUERY_FOR_BOLIGDATA
     PLOTS(ONLY)=INTERVAL
     ALPHA=0.05
     H0 =150
     CI = EQUAL;

     VAR boligst;
RUN;
```

Prøv at køre din kode nu (tryk *Run*). Nu virker det bare, og du får dit output med det samme. Og det virker også, hvis du lukker og åbner SAS igen.

Nu kan vi jo prøve en lille variation. Prøv at ændre H_0-hypotesen til at teste, om boligstørrelsen er signifikant forskellig fra 160m^2. Det gøres bare ved at ændre de 150 i koden til 160 i stedet. Vi kan ved samme lejlighed prøve at ændre signifikansniveauet til 0.10 i stedet. Med disse ændringer ser koden ud som nedenfor:

```
PROC TTEST
     DATA=BOLIGLIB.QUERY_FOR_BOLIGDATA
     PLOTS(ONLY)=INTERVAL
     ALPHA=0.10
     H0 =160
     CI = EQUAL;

     VAR boligst;
RUN;
```

Prøv nu at køre testen (tryk *Run*) og aflæs resultatet – med en p-værdi på 0,4559 kan vi i dette tilfælde ikke afvise H0-hypotesen (outputtet er ikke vist her).

Tillykke – du har nu lavet et komplet funktionelt SAS-program skrevet i SAS-kode!

Selvfølgeligt har vi her kun vist dig toppen af isbjerget i forhold til SAS-kodning. Har du fået mod på mere, så har SAS lavet en lille bog, der meget passende hedder "*The Little SAS Book: A Primer*". Den er god til at komme i gang med kodning og er let læsning og kan bruges som opslagsværk og til øvning i grundlæggende SAS-kodning. Du kan bl.a. købe den på internettet via www.amazon.co.uk. Du kan også altid finde svar ved at google om SAS-kode – der er mange, der bruger SAS derude.

4.2. Kodeeksempler til denne e-bog – spar tid og dokumentér!

Alt det, vi har lavet i denne e-bog, kunne være lavet med et par kodestumper som dem på de følgende par sider. Koden her er skrevet noget kortere end den automatisk genererede kode for at vise, at det ikke behøver at være så svært.

Koden er faktisk også en virkelig god dokumentation for alt, hvad vi har gjort ved vores data – enhver kan gentage vores analyser med denne kode og få nøjagtig de samme resultater. På samme måde kan du vedhæfte din kode og dit datasæt til større projekter – så har du dokumenteret dit arbejde!

Vi starter som altid vores kode med et *Libname statement* og går derefter videre til at lave vores '*Query*_For_*Boligdata*'-datasæt. Det tog os masser af indtastningstid og mange sider i denne e-bog at lave den query – i SAS-koden nedenfor fylder vores *DATA*[8] step, som laver det samme datasæt, kun 11 linjer reel kode.

Resten af denne manual er skrevet i SAS-kode – man kan nemlig uden problemer skrive kommentarer til sig selv og sine samarbejdspartnere i SAS-koden, som du kan se på den grønne skrift i koden nedenfor (den mellem /* og */). Prøv at åbne et nyt, tomt SAS-program (*File* > *New* > *Program*) og indsæt kodeteksten nedenfor deri. Det eneste, du behøver at ændre, er stien til dine data i 2. linje nedenfor (vores *Libname statement*). God fornøjelse med SAS-kodningen – er du nået helt hertil, så kommer du helt sikkert også godt videre.

```
/* Først laver vi vores SAS Library reference: */

LIBNAME BOLIGLIB "C:\Documents and Settings\Administrator\Desktop\SAS Projects" ;

/* Herefter laver vi vores 'Query_For_Boligdata' datasæt med SAS kode.
   Den automatiske kode fra SAS peg-og-klik bruger SQL-kodning til query'en.
   Her bruger vi i stedet almindelig SAS kode. Læg mærke til at variablene
   Automatisk bliver defineret når vi nævner dem i koden (fx omr_ny = (…)): */

DATA BOLIGLIB.QUERY_FOR_BOLIGDATA;
        SET BOLIGLIB.BOLIGDATA;
        /* omr ny */
        IF omr = 1 THEN omr_ny = 'Nord';
        ELSE IF omr = 2 THEN omr_ny = 'Syd';
        /* kontantp gr */
        IF kontantp <= 1500 THEN kontantp_gr = 'Lav';
        ELSE IF kontantp >= 1501 AND kontantp <= 2500 THEN kontantp_gr = 'Middel';
        ELSE IF kontantp >= 2501 THEN kontantp_gr = 'Høj';
        /* dummy højkvalitet */
        IF kvalitetgr = 3 then dummy_højkvalitet = 1;
        ELSE dummy_højkvalitet = 0;
        /* kvmpris */
        kvmpris = kontantp / boligst;
        FORMAT kvmpris 8.2;
        /* Ln_kontantp */
         ln kontantp = (LOG(kontantp));
        FORMAT Ln_kontantp 8.2;
RUN;

/* Tænd for ODS grafik så vi får figurer med i vores outputs
   (grunden til at det ikke er standard, er at man kan spare tid
   uden grafikken ved store datasæt og mange beregninger) */

ODS GRAPHICS ON;
```

8. *DATA* eller *PROC* er de to ord, som påbegynder 99% af al SAS-kode! *DATA* steps bruges fx til at lave nye datasæt som i vores *Query*. Den automatisk genererede kode anvender dog SQL-kode, men her har vi gjort det samme med almindelig SAS-kode.

```
/* Test på ét gennemsnit */
PROC TTEST DATA=boliglib.query_for_boligdata
               PLOTS(ONLY)= INTERVAL
               ALPHA= 0.05
               H0 = 150
               CI = EQUAL;
        VAR boligst;
        TITLE1 "Test på ét gennemsnit";
RUN;


/* Test på to uafhængige gennemsnit */
PROC TTEST DATA=boliglib.query_for_boligdata
               PLOTS(ONLY)= INTERVAL
               ALPHA= 0.05
               H0 = 0
               CI = EQUAL;
        CLASS omr_ny;
        VAR kontantp;
        TITLE1 "Test af forskellen mellem to uafhængige gennemsnit";
RUN;


/* Test på tre eller flere gennemsnit (One-Way ANOVA) */
PROC ANOVA DATA=boliglib.query_for_boligdata;
        CLASS kvalitetgr;
        MODEL kontantp = kvalitetgr;
        MEANS kvalitetgr / HOVTEST=LEVENE BON ALPHA=0.05;
        TITLE1 "One-Way ANOVA test på prisforskellen mellem de tre kvalitetsgrupper";
RUN;


/* To-faktor ANOVA */
PROC GLM DATA=boliglib.query_for_boligdata   PLOTS=INTPLOT;
        CLASS kvalitetgr omr_ny;
        MODEL boligst = kvalitetgr omr_ny kvalitetgr*omr_ny;
        TITLE1 "To-faktor ANOVA test på forskellen mellem boligstørrelser for hhv.
kvalitetsgrupper og område";
RUN;


/* Test af én andel (om der er lige mange huse til salg i nord og syd) */
PROC FREQ DATA=BOLIGLIB.QUERY_FOR_BOLIGDATA;
        TABLES omr_ny / BINOMIAL(p=0.5);
        EXACT BINOMIAL;
        TITLE1 "Test af én andel";
RUN;


/* Test af om to andele er signifikant forskellige
  (om andelen af højkvalitets huse er ens nord og syd for København) */
PROC FREQ DATA=BOLIGLIB.QUERY_FOR_BOLIGDATA;
        TABLES dummy_højkvalitet * omr_ny / CHISQ;
        TITLE1 "Test af forskellen mellem to andele";
RUN;


/* Uafhængighedstest (Chi-square test) for om kvalitets gruppe og
   kontantpris gruppe er uafhængige */
PROC FREQ DATA=BOLIGLIB.QUERY_FOR_BOLIGDATA;
        TABLES kvalitetgr * kontantp_gr / chisq;
        TITLE1 "Uafhængigheds test (Chi-square test)";
RUN;


/* Scatter plot for kontantprisen ift. boligstørrelsen */
PROC GPLOT DATA=boliglib.query_for_boligdata;
        PLOT kontantp * boligst / FRAME ;
        TITLE1 "Scatter Plot mellem kontantprisen og boligstørrelsen";
RUN; QUIT;
```

```
/* Korrelations analyse mellem kontantprisen og boligstørrelsen */
PROC CORR DATA=BOLIGLIB.QUERY_FOR_BOLIGDATA;
        VAR kontantp boligst;
        TITLE1 "Korrelations analyse";
RUN;


/* Simple lineær regression hvor kontantprisen forklares ud fra boligstørrelsen */
PROC REG DATA=BOLIGLIB.QUERY_FOR_BOLIGDATA
        PLOTS(ONLY)=ALL;
        MODEL kontantp = boligst;
        TITLE1 "Simple lineær regression";
RUN;


/* Multipel korrelationsanalyse mellem kontantprisen og fire andre variabler */
PROC CORR DATA=BOLIGLIB.QUERY_FOR_BOLIGDATA;
        VAR kontantp boligst grundst vaerelse kvalitet;
        TITLE1 "Korrelations analyse med flere variable";
RUN;


/* Multipel lineær regression med kontantprisen og de fire andre variabler
   (Du kan evt. selv prøve at erstatte 'kontantp' med 'ln_kontantp' i koden nedenfor) */
PROC REG DATA=BOLIGLIB.QUERY_FOR_BOLIGDATA
                PLOTS(ONLY)=RESIDUALBYPREDICTED
                PLOTS(ONLY)=QQPLOT
                PLOTS(ONLY)=RESIDUALPANEL
                PLOTS(ONLY)=FITPLOT;
        MODEL kontantp = boligst grundst vaerelse kvalitet / VIF SPEC;
        TITLE1 "Multipel lineær regression";
RUN;


/* Multipel regression med dummy-variabel, kvadreret variabel og interaktionsvariabel */
PROC GENMOD DATA=BOLIGLIB.QUERY_FOR_BOLIGDATA PLOTS(ONLY)=ALL;
        CLASS omr_ny;
        MODEL kontantp = boligst grundst vaerelse kvalitet omr_ny
                        grundst*grundst boligst*vaerelse;
        TITLE1 "Regression med dummy-variabel, kvadreret variabel og interaktionsvariabel";
RUN;


/* Logistisk regression */
PROC LOGISTIC DATA=BOLIGLIB.QUERY_FOR_BOLIGDATA DESC PLOTS(ONLY)=ALL;
        CLASS omr_ny;
        MODEL dummy_højkvalitet (Event = '1') = kvmpris grundst vaerelse omr_ny / RSQUARE;
RUN;


/* Fordelingstests */
DATA BOLIGLIB.FORDELINGSTESTS;
        Binomial_sandsynlighed = PROBBNML( 0.2 , 100 , 10 );
        Hypergeometrisk_sandsynlighed = PROBHYPR( 69 , 14 , 10 , 2 );
        Normalfordelings_sandsynlighed = PROBNORM(( 1.7 - 2.3 ) / 0.8 );
        TITLE1 'Fordelingstests';
RUN;


/* Slå grafikken fra igen*/
ODS GRAPHICS OFF;
/* Det var det - pøj pøj fremover med SAS! */
```

Efter at have læst denne bog er du nu rustet til at gå i gang med at arbejde i SAS. Vil du gerne vide mere om mulighederne i SAS, kan du starte med følgende:

SAS Enterprise Guide - peg og klik:
SAS for Dummies
Stephen McDaniel and Chris Hemedinger

SAS programmering:
The Little SAS Book: A Primer
Lora Delwiche and Susan Slaughter

E-learningkurser tilgængelige via www.sas.com (gratis for studerende, undervisere og forskere):
SAS Programming I: Essentials
SAS Programming II: *Data* Manipulation Techniques

Skulle du have yderligere spørgsmål, er du velkommen til at kontakte os på
academic.dk@sas.com

SAS INSTITUTE A/S · WWW.SAS.COM/DK

Købmagergade 7-9 · DK-1150 København K · Telefon +45 70 28 28 70 · Kr. Kielbergsvej 3 · DK-8660 Skanderborg · Telefon +45 70 28 28 70

SAS er verdens førende leverandør af software og tjenesteydelser til business intelligence og business analytics. SAS er grundlagt i 1976, privatejet og uafhængig afeksterne økonomiske interesser. Flere end 45.000 virksomheder over hele verden får med SAS' innovative løsninger grundlag for at træffe bedre beslutninger hurtigere. SAS og alle SAS Institute Inc.'s produkter og ydelser er varemærker eller registrerede varemærker af SAS Institute Inc., Cary, NC, USA. ® indikerer registrering i USA og andre lande. SAS Institute A/S er et datterselskab af SAS Institute Inc., Cary, NC, USA. © Copyright 2011. 01055/DK/0911