Brukermanual for microdata.no

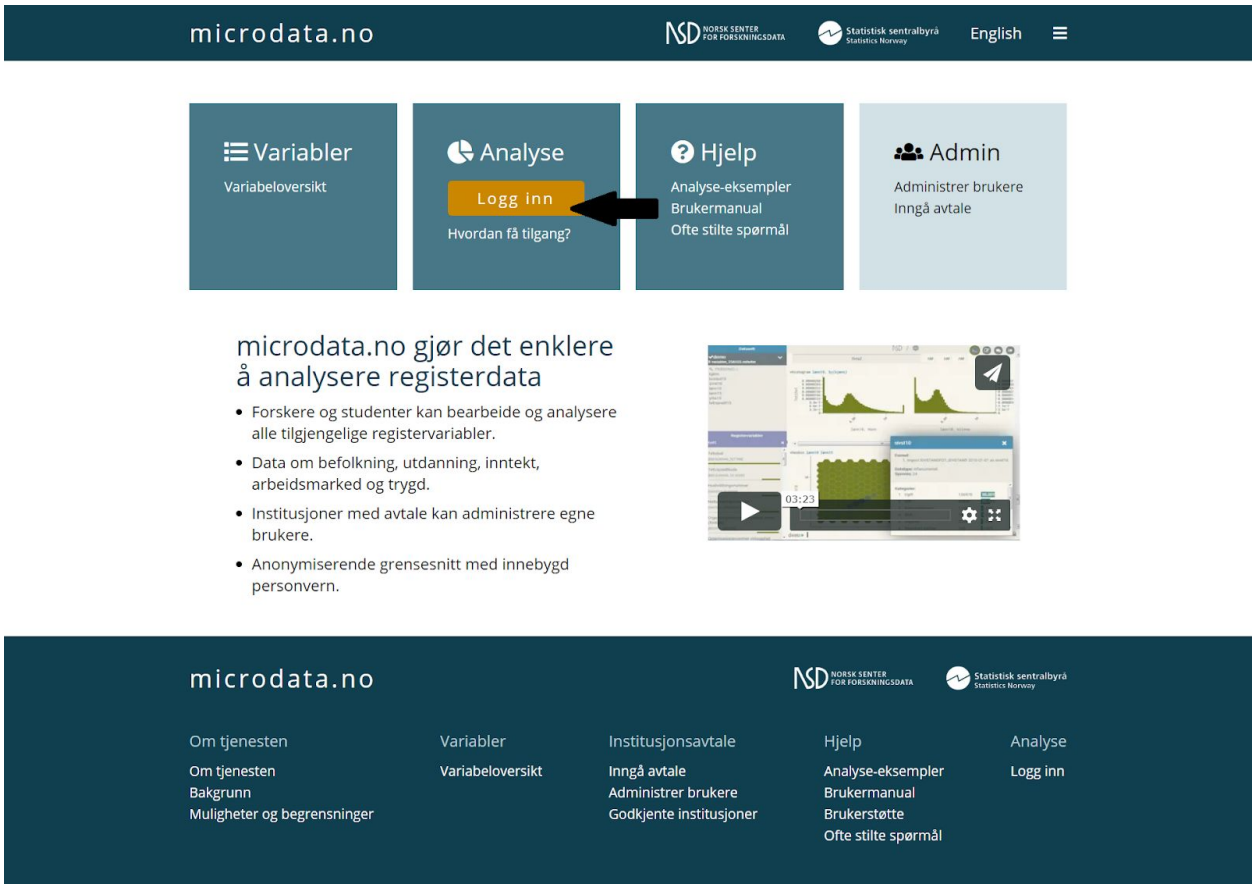
Innholdsliste

1. Om brukergrensesnittet

Microdata.no er et nettbasert analysesystem. Det anbefales å bruke nettlesere som Chrome og Firefox for best mulig brukeropplevelse. Internet Explorer og Edge vil kunne gi feil ved at skjermen blir blank og/eller at det ikke går an å logge seg inn, og anbefales derfor ikke.

Innlogging i microdata.no gjøres via følgende nettside:

http://microdata.no/



Det som møter deg etter innlogging er en nettside bestående av et analyseområde. Dette brukes til å utforske variabler og teste ut kommandokjøringer:

En kan klikke på variabler for å gjøres seg kjent med innholdet, verdiformat, gyldighetsperiode m.m. Gjennom kommandoer som kjøres enkeltvis kan variabler importeres til egne datasett for videre bearbeiding og analyser. I tillegg til deskriptive statistiske muligheter, kan en dessuten

foreta avanserte statistiske operasjoner som regresjonsanalyser m.m. Se kapittler 1.1-1.4 for mer informasjon om arbeidsområdet.

Etter å ha utforsket og gjort seg kjent med data en ønsker å benytte i analyser, anbefales det på det sterkeste å benytte skript-funksjonaliteten for å systematisere analysearbeidet. Dette gjelder særlig om en har til hensikt å foreta en mer omfattende analyse (utover å bare lage enkel deskriptiv statistikk for et fåtall variabler). Bruk av skript har mange fordeler fremfor å utelukkende jobbe i analyseområdet gjennom bruk av enkeltkommandoer:

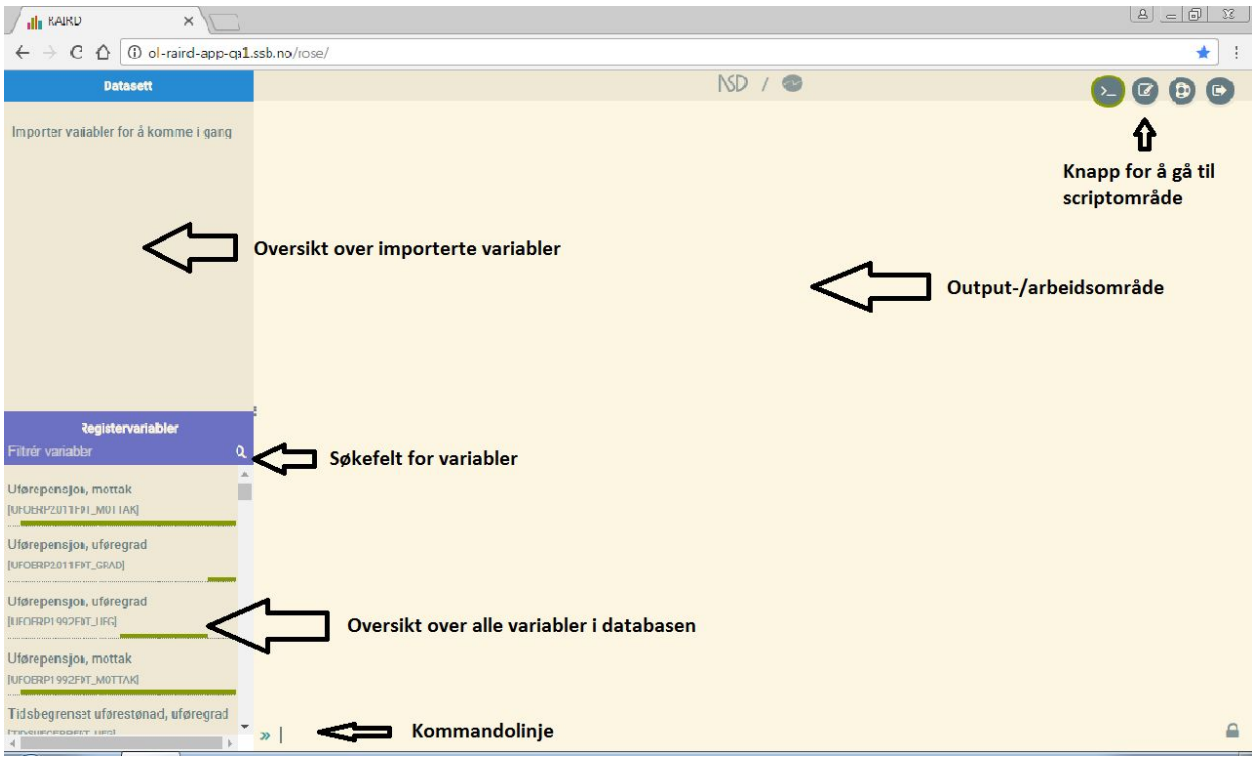
a) En kan sette opp kommandosekvenser som kan kjøres i én operasjon. Resultatet av skriptkjøringer vises fortløpende etterhvert som kommandoer kjøres, og kjøringen stoppes dersom feil oppdages i skript (syntax- eller logiske feil)
b) Kommandosekvenser kan redigeres og kjøres på nytt
c) Mye enklere å identifisere hvor eventuelle feil oppstår i en lang rekke med kommandosekvenser
d) Fungerer som dokumentasjon/logg av arbeidet
e) Forskjellige arbeidsøkter kan lagres med egne navn og senere kalles frem
f) Innholdet i et skript kan kopieres over til egne dokumenter for ekstra backup

Arbeid en allerede har utført i analyseområdet kan enkelt overføres til et skript for videre redigering. Det kan gjøres på to måter:

- I skriptvinduet finnes det en menyknapp helt oppe til venstre. Der kan en velge “Nytt skript med historikk fra kommandolinjen”.
- I arbeidsområdet kan en skrive kommandoen `history`. Dette frembringer alle kommandoer en har kjørt i en enkelt liste som kan kopieres på vanlig måte (ctrl+c) og limes inn i et skriptvindu

For mer informasjon om bruk av skript, se kapittel 1.5.

1.1 Analyseområdet

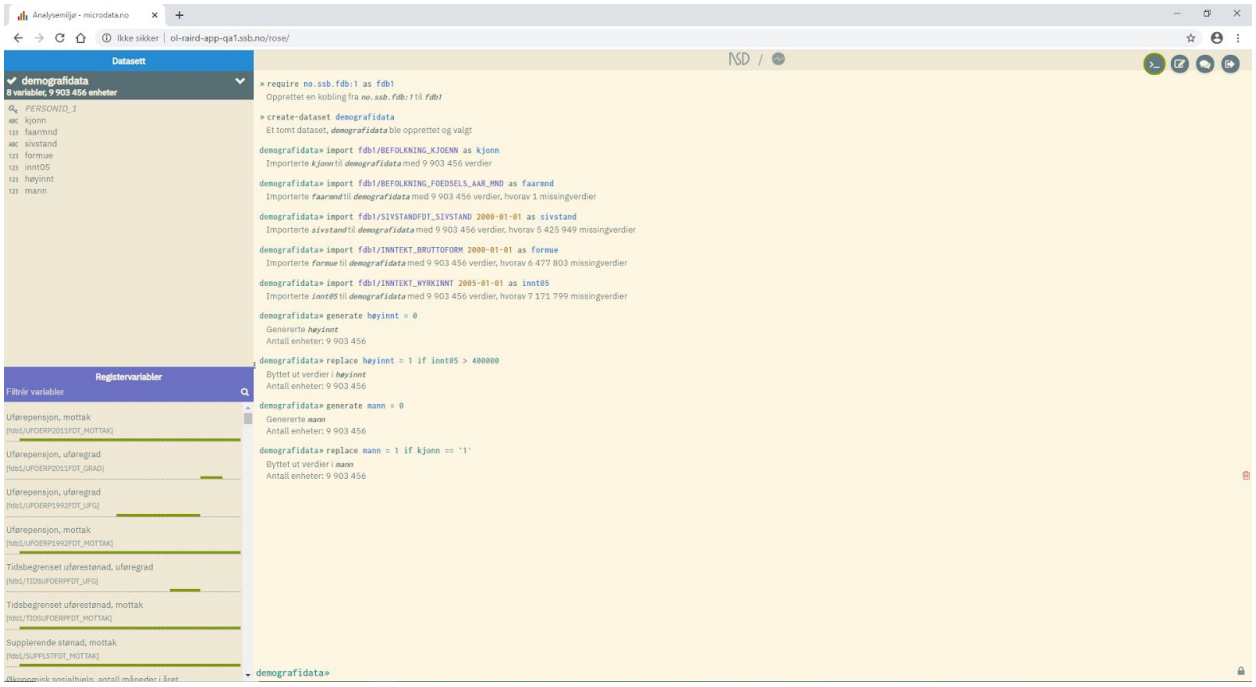


Analyseområdet består av følgende:

- Arbeidsområdet (største feltet helt til høyre)
- Oversikt over tilgjengelige variabler (feltet nederst til venstre - feltet vil være blankt helt til en kobler seg mot databanken vha. kommandoen `require`, jfr. kapittel 2.1)
- Oversikt over variabler hentet/importert til egne arbeidsdatasett (feltet øverst til venstre)
- Kommandolinjen (nederst i resultatområdet)

Egne arbeidsdatasett bygges opp ved å hente (importere) og om nødvendig bearbeide (og utvikle nye) variabler fra datakatalogen. Datasett blir lagret i brukerens arbeidsområde og forsvinner ikke uten at brukeren selv sletter dem. Se kapittel 2 for hvordan en oppretter datasett og importerer variabler.

Etter at analyseområdet er blitt fylt opp med importerte variabler, kan det se slik ut:



I eksempelet over er det opprettet et datasett med navnet “demografidata” som har 8 variabler og 9 903 456 enheter (individer). Blant variablene finner en identifikasjonsnøkkelen PERSONID_1. Dette er en systemvariabel som alltid følger med ved import av variabler. Den angir en unik personidentifikator som brukes som koblingsnøkkel. Systemet kobler automatisk sammen variabler (“left join”) dersom en bare skal bruke data på personnivå, og da trenger en ikke forholde seg til denne variabelen.

I enkelte tilfeller kan identifikasjonsnøkkelen PERSONID_1 være aktuell å bruke. Det er mulig å lage oppsummerende data basert på hendelser over tid gjennom kommandoen `collapse` og selv styre på hvilket enhetsnivå en ønsker at de aggregerte data skal være tilgjengelig (dette blir beskrevet nærmere i kapittel 3.4). Kommandoen `merge` gjør det dessuten mulig å koble sammen ulike datasett som brukeren har opprettet. Også her kan nevnte identifikasjonsnøkkel være aktuell å spesifisere (se kapittel 2.8).

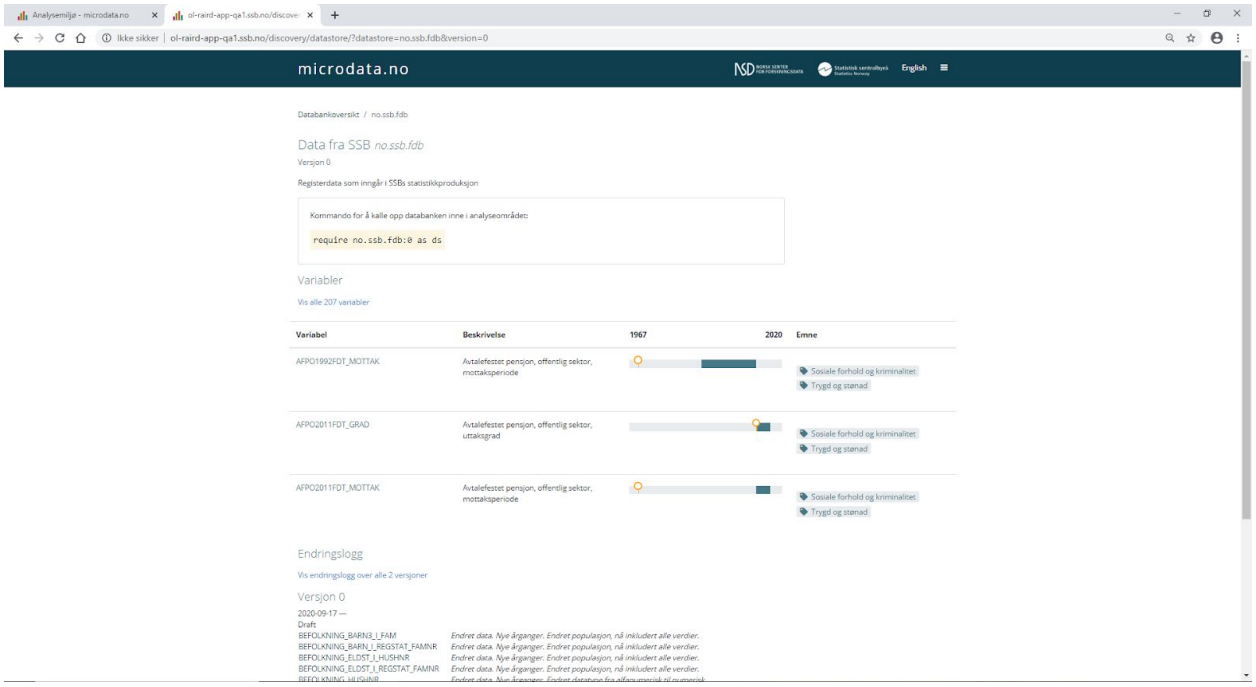
Merk at både `collapse` og `merge` åpner for hhv. aggregering og kobling på ulike nivå. Systemet krever imidlertid at koblings-/aggregeringsnøkkelen innehar et minimum antall verdikategorier (over 1000). F.eks. kan en ikke aggregere eller koble datasett på kjønnsnivå, sivilstandsnivå eller bostedsnivå. Til det er antallet kategorier for lavt.

Arbeidsområdet viser en logg for hvilke kommandoer som har vært utført og hvilket svar man får, det være seg tabeller, figurer og annen feedback.

Helt nederst står det nå "demografidata>>" i stedet for ">>", for å markere at en nå står i datasettet "demografidata". Om en oppretter flere datasett vil vinduet øverst til venstre inneholde flere variabeloversikter tilsvarende den som gjelder for "demografidata". For å arbeide på ulike datasett kan en skifte gjennom å bruke kommandoen `use <datasett>`. Da vil ledeteksten i kommandolinjen endre navn til det datasettet en har forflyttet seg til.

1.2 Variabler og variabelbeskrivelser

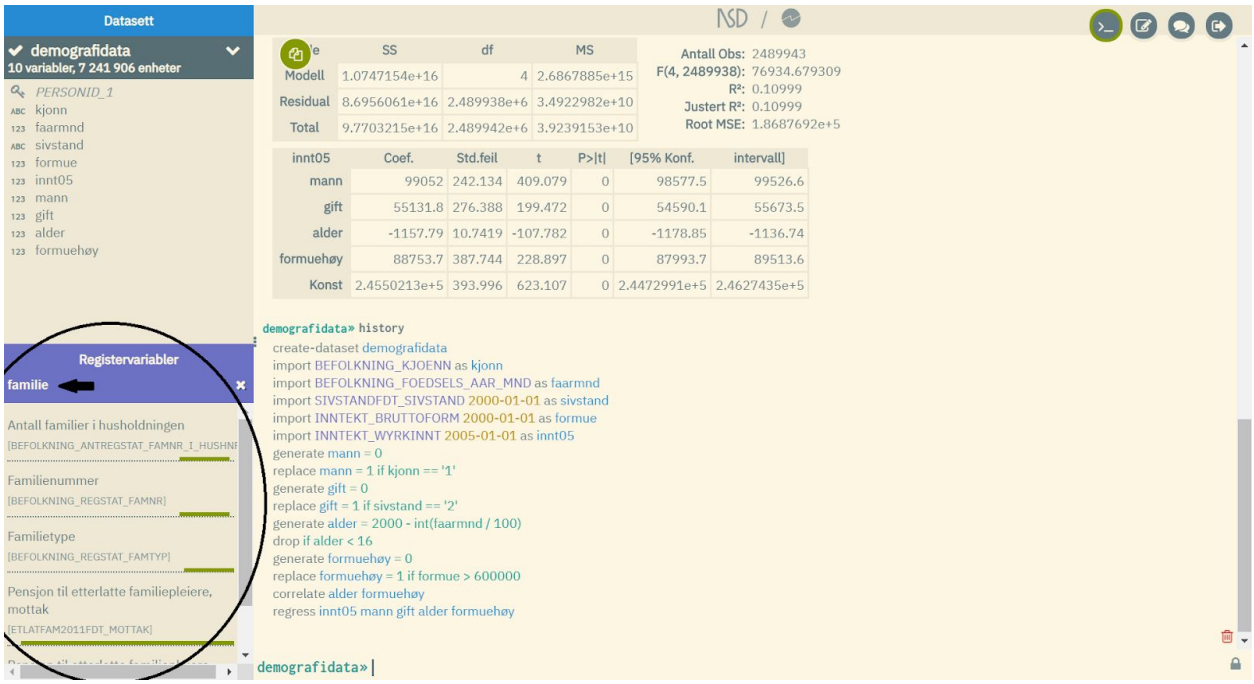
Analysesystemet microdata.no har en lang rekke demografiske, utdanningsrelaterte, økonomiske, sysselsettingsrelaterte og trygderelaterte variabler i databasen. Disse kan en få en oversikt over gjennom å utforske variabeloversikten en finner på åpningssiden (https://microdata.no/discovery):



Variabellisten kan ekspanderes slik at den viser samtlige variabler i databanken (over 200 i SSBs databank). Det jobbes med å forbedre oversikten med blant annet et søkefelt og tematisk inndeling. Dette vil komme innen relativt kort tid.

Nederst i variabellisten finnes det også en oversikt over de ulike versjonene av databanken, og hva som er blitt endret. Ved å klikke på versjonsnavnene, vil en komme til en ny side som viser alle variablene for denne versjonen.

Som illustrasjonen nedenfor viser, finnes det også en fullstendig variabeloversikt nederst til venstre inne i selve analyseområdet. Her kan en filtrere variabellisten ved å skrive inn deler av et variabelnavn i søkefeltet. Filteret fungerer både mot variabelbeskrivelse og selve navnet. På den måten blir det lettere å finne variabelen en ønsker.



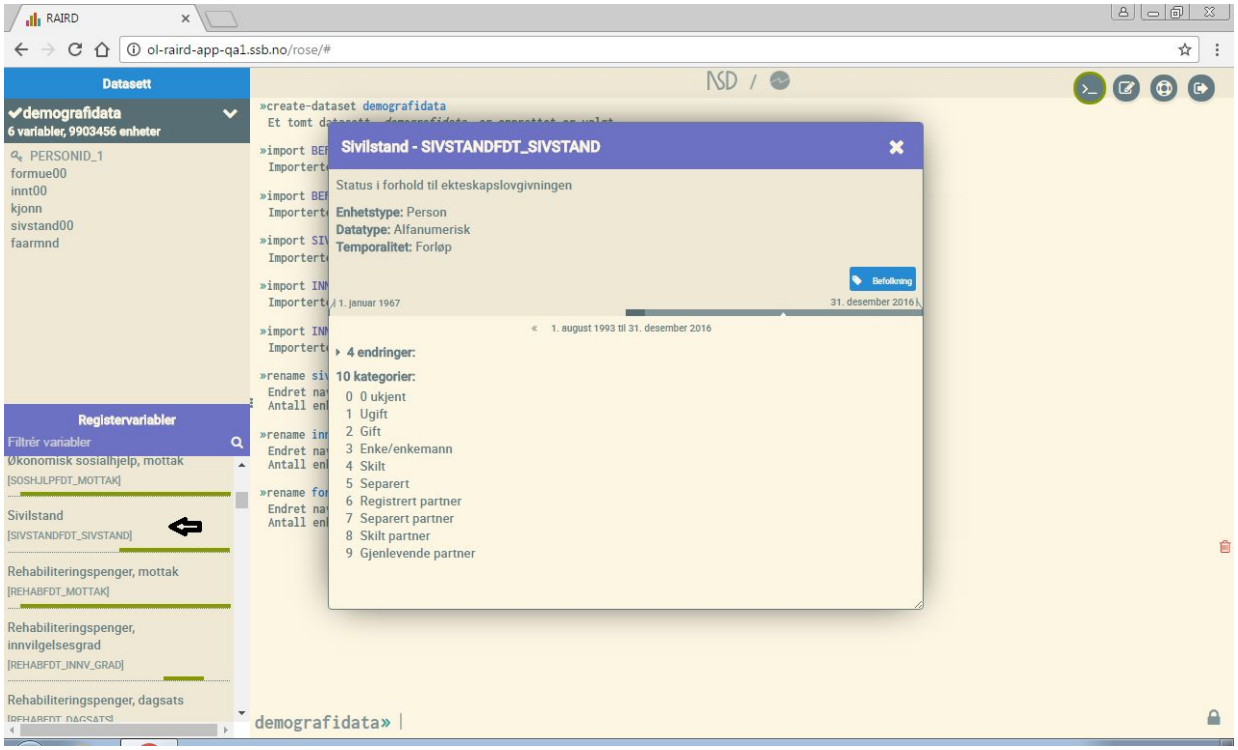
Alle variabler vises med en tilhørende tidslinje som markerer gyldighetsperioden(e), dvs. hvilket tidsspenn som er dekket. Variabler i microdata.no er tredimensjonale - de inneholder tid. Ved å "klikke" på variabler i listen kan en hente frem deskriptiv statistikk og annen informasjon som variabeltype o.l.

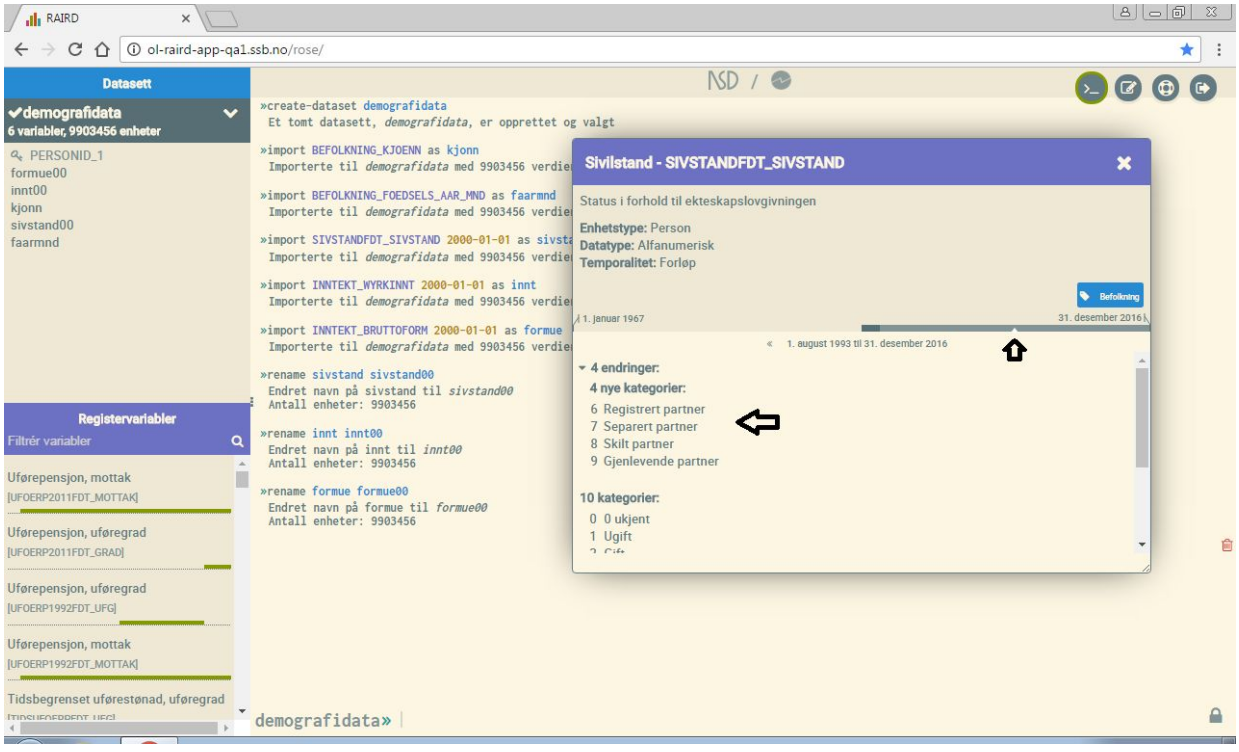
I eksempelet nedenfor blir dette eksemplifisert for variabelen "Sivilstand". Variabelen presenteres da i et eget vindu som en kan flytte på og justere. Det gir detaljert informasjon om variabelen:

- Nøkkelinformasjon: Variabelnavn, variabel-label, variabelbeskrivelse, variabeltype

- Detaljert interaktiv tidslinje som gir mulighet til å studere endringer i kodingen over tid: Endringer i kodingen vises gjennom ulike farger som illustrerer hvilke tidsperioder de gjelder for. Klikker en på de ulike feltene i tidslinjen får en frem en liste over de kodene som var gyldige i den aktuelle perioden. I eksempelet er det markert i feltet som gjelder 1. august 1993 - 31. desember 2016, og det dukker da opp en liste på 10 kategorier

- Informasjon om endringer: I eksempelet vises det "4 endringer". Dette er antallet endringer i forhold til forrige tidsperiode. En kan klikke på "4 endringer", og få opp en liste over kodene som er nye


RAIRD
ol-raird-app-qa1.ssb.no/rose/#
Datasett
demografidata
6 variabler, 9903456 enheter
PERSONID_1
formue00
innt00
kjonn
sivstand00
faarmnd
Registervariabler
Filtrér variabler
Økonomisk sosialhjelp, mottak
[SOSHJLPFDT_MOTTAK]
Sivilstand
[SIVSTANDFDT_SIVSTAND]
Rehabiliteringspenger, mottak
[REHABFDT_MOTTAK]
Rehabiliteringspenger, innvilgelsesgrad
[REHABFDT_INNV_GRAD]
Rehabiliteringspenger, dagsats
»create-dataset demografidata
Et tomt datasett, demografidata, er opprettet og valgt
Sivilstand - SIVSTANDFDT_SIVSTAND
Status i forhold til ekteskapslovgivningen
Enhetstype: Person
Datatype: Alfanumerisk
Temporalitet: Forløp
Befolkning
1. januar 1967
31. desember 2016
1. august 1993 til 31. desember 2016
4 endringer:
10 kategorier:
0 0 ukjent
1 Ugift
2 Gift
3 Enke/enkemann
4 Skilt
5 Separert
6 Registrert partner
7 Separert partner
8 Skilt partner
9 Gjenlevende partner
demografidata»

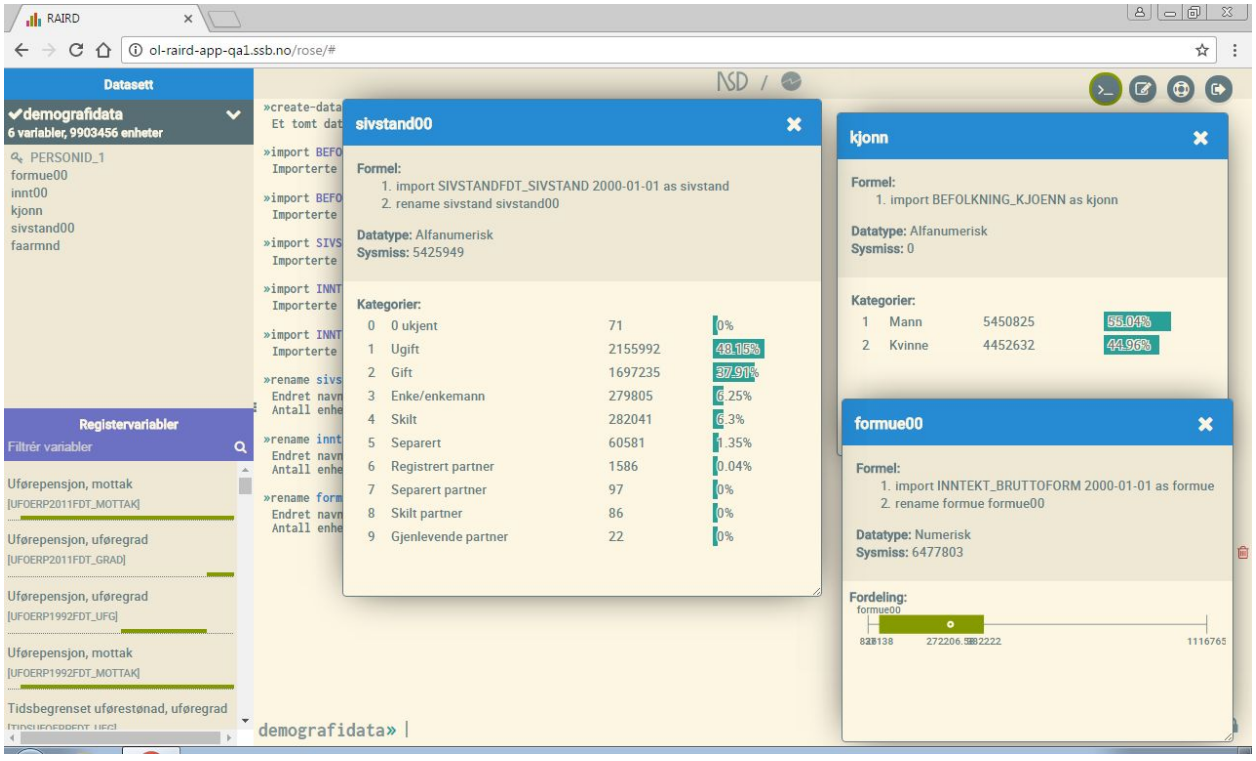

RAIRD
ol-raird-app-qa1.ssb.no/rose/
Datasett
demografidata
6 variabler, 9903456 enheter
PERSONID_1
formue00
innt00
kjonn
sivstand00
faarmnd
Registervariabler
Filtrér variabler
Uførepensjon, mottak
[UFOERP2011FDT_MOTTAK]
Uførepensjon, uføregrad
[UFOERP2011FDT_GRAD]
Uførepensjon, uføregrad
[UFOERP1992FDT_UFG]
Uførepensjon, mottak
[UFOERP1992FDT_MOTTAK]
Tidsbegrenset uførestønad, uføregrad
»create-dataset demografidata
Et tomt datasett, demografidata, er opprettet og valgt
»import BEFOLKNING_KJOENN as kjonn
Importerte til demografidata med 9903456 verdier
»import BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
Importerte til demografidata med 9903456 verdier
»import SIVSTANDFDT_SIVSTAND 2000-01-01 as sivstand
Importerte til demografidata med 9903456 verdier
»import INNTEKT_WYRKINNT 2000-01-01 as innt
Importerte til demografidata med 9903456 verdier
»import INNTEKT_BRUTTOFORM 2000-01-01 as formue
Importerte til demografidata med 9903456 verdier
»rename sivstand sivstand00
Endret navn på sivstand til sivstand00
Antall enheter: 9903456
»rename innt innt00
Endret navn på innt til innt00
Antall enheter: 9903456
»rename formue formue00
Endret navn på formue til formue00
Antall enheter: 9903456
Sivilstand - SIVSTANDFDT_SIVSTAND
Status i forhold til ekteskapslovgivningen
Enhetstype: Person
Datatype: Alfanumerisk
Temporalitet: Forløp
Befolkning
1. januar 1967
31. desember 2016
1. august 1993 til 31. desember 2016
4 endringer:
4 nye kategorier:
6 Registrert partner
7 Separert partner
8 Skilt partner
9 Gjenlevende partner
10 kategorier:
0 0 ukjent
1 Ugift
demografidata»

For variabler som er importert til brukerens datasett ("demografidata") vises det en litt annen type informasjon som kan være nyttig når en arbeider med mange ulike variabler. Denne informasjonen justerer seg fortløpende dersom en endrer på variablene, og vises i separate popup-vinduer når en klikker på variabler i listen tilhørende det aktuelle datasettet ditt:

- Formel: Øverst i vinduet finner en "skapelseshistorikken" til den aktuelle variabelen. Dette brukes til å slå opp hvordan en variabel har blitt laget eller omkodet

- Nøkkelinformasjon: Variabeltype og antall enheter som har verdi for manglende data (sysmiss)

- Frekvensfordeling og enkel statistikk: For kategoriske variabler vises frekvensfordelingen, mens det for kontinuerlige variabler vises en standard boksplot med boks for de to midterste kvartilene, gjennomsnitt og minimums- og maksimumsverdi (såkalte whiskers). Om verdier blir uleselige pga. overlapp, så kan dette popup-vinduet utvides.



1.3 Kommandolinjen

Kommandolinjen er den sentrale delen av brukergrensesnittet, og det er der en skriver inn kommandoer for hva en ønsker å gjøre. Dette omfatter import/innhenting av variabler til arbeidsområdet, lage deskriptiv statistikk for enkeltvariabler, kommandoer for bearbeiding og omkoding av variabler, administrative hjelpekommandoer (`help`, `history`, `clear` etc) eller analysekommandoer. Se vedlegg A for fullstendig liste over tilgjengelige kommandoer.

Analysesystemet microdata.no har en innebygget selvutfyllingsløsning som foreslår relevante kommandoer ut fra hva en starter å taste inn. Ofte er det nok å taste inn noen få tegn før systemet foreslår den ønskede kommando. Ved å trykke på <tab>-tasten på tastaturet, legges kommandoen korrekt inn.

De fleste kommandoer forutsetter at det legges inn tilleggsinformasjon, og selvutfylling fungerer også i slike sammenhenger. Kommandoen `import` trenger informasjon om variabelnavn og måletidspunkt for å kunne utføres. Ønsker en å importere variabelen *kjønn*, så kan en taste inn bokstaven “k”. Det vil være nok til å få opp en liste over alle variabler med bokstaven “k” i navnet - i praksis blir det svært mange. Fortsetter en med “j” vil systemet liste alle variabler med “kj” i navnet. Etter å ha tastet inn “ø” vil bokstavkombinasjonen være såpass unik at systemet har redusert antall alternativer til et fåtall, og brukeren kan velge den korrekte variabelen med piltastene på tastaturet og deretter bruke *<tab>*-tasten. Når variabel er valgt må man også angi et tidspunkt gitt at ikke det er snakk om en konstant opplysning slik som “kjønn” (da trenger en ikke angi tidspunkt). Default-verdi for systemet er den sist brukte datoen. Om dette er ok, brukes *<tab>*-tasten. Hvis ikke legger en inn aktuell ny dato i stedet. Systemet vil også foreslå aktuelle opsjoner til slutt. For `import` kan en bruke opsjonen `as`. Den brukes til å lage et alias til den aktuelle variabelen. Registervariablene i databasen har ofte litt upraktiske og lange navn som med fordel kan døpes om gjennom `as`- opsjonen for bl.a. å skape mer leselige analyse- og statistikkutskrifter.

1.4 Nyttige kommandoer

Om en lurer på hva slags kommandoer en kan bruke, kan en starte med `help`. Da listes alle tilgjengelige kommandoer. For dem som kjenner til analyseprogrammet Stata, vil de fleste kommandoene virke velkjente. Analysesystemet microdata.no benytter en Stata-liknende syntax, det innebærer også at kommandoer skrives på engelsk.

Ønsker en full informasjon om enkeltkommandoer, taster en inn `help` og deretter navn på kommando, f.eks. `help import`. Da vises en forklaring med eksempler på bruk. En komplett oversikt over kommandoer og funksjoner vises i vedlegg A og B.

En annen nyttig kommando er `history`. Den lister opp alle kommandoer som har vært brukt i en sesjon. Denne listen kan kopieres og limes inn i systemets skripteditor, eller i et privat dokument. Dette er en enkel måte å ta kopi av arbeidet sitt på.

Kommandoen `clear` kan brukes til å slette alt innhold i arbeidsområdet dersom en ønsker å begynne helt på nytt. Denne bør derfor brukes med varsomhet!

Et alternativ til `clear` kan være å bruke den kjente tastekombinasjonen <*Ctrl*> + <*Z*>. Dette er en mye brukt angrefunksjon, der en gjør om det siste en gjorde (går altså ett steg tilbake). Dette kan gjøres så mange ganger en ønsker, helt til en står igjen med et tomt arbeidsområde. Tastekombinasjonen <*Ctrl*> + <*Y*> kan brukes for å gjøre om den siste "angringen". Brukere av Apple-PC'er bruker kombinasjonene <*Cmd*>+<*Z*> og <*Cmd*>+<*Y*>.

Et nyttig hjelpemiddel når en jobber i kommandolinjen er å bruke <*piltast opp*> på tastaturet. Da kan en scrolle gjennom alle tidligere kommandoer en har benyttet i kronologisk rekkefølge og velge den en ønsker å bruke på nytt. Da slipper en å legge inn manuelt samme kommandolinje flere ganger. Ofte ønsker en å kjøre ulike varianter av samme kommando (f.eks. med litt andre variabler eller parametre). Da kan en lete opp en allerede brukt kommando, og så endre litt på den før en kjører den på nytt.

1.5 Skripteditor

Skripteditoren lar brukere legge inn sekvenser av kommandoer som kan kjøres samlet som et skript. En har to muligheter når en er klar for å kjøre skriptet: Nederst til høyre er det to knapper, der den ene ("Kjør") kjører gjennom skriptet på høyre side av skriptområdet og foretar en validering (kontrollerer om alt er korrekt). Den andre knappen ("Send til kommandolinjen") sender alle kommandoene til arbeidsområdet og utfører dem der.

Alle kommandolinjer i et skript eksekveres sekvensielt og viser resultatet fortløpende mens det kjøres. Ved eventuelle feil i syntaxen, stoppes kjøringen der hvor feilen befinner seg. En kan i slike tilfelle rette opp feilen og kjøre skriptet på nytt.

Om en foretar justeringer i et skript og kjører på nytt, vil arbeidsområdet erstattes av resultatet fra den nye skriptkjøringen (brukes knappen "Kjør", ønsker en bare å teste skriptet og det vil da ikke bli gjort noen endringer i arbeidsområdet). Pass på å ikke overskrive arbeidet ditt om dette ikke er tatt vare på i form av skript.

Merk at systemet "husker" tidligere kjørte kommandosekvenser. Så om en kjører akkurat samme kommandoskript på nytt uten endringer, vil det bare ta noen sekunder å hente frem

resultatet av kjøringen. Dette prinsippet gjelder også dersom innledende deler av et skript er kjørt tidligere, der en kun har endret på siste delen. Da vil systemet "huske" det som tidligere er blitt kjørt, og kun bruke ressurser på å jobbe seg gjennom den delen av skriptet som er endret.

1.5.1 Kjøre deler av et skript

Det er mulig å kjøre deler av et skript. Dette kan gjøres på to måter:

A) Markere ut enkeltlinjer ved å definere dem som hjelpetekst

- Legg inn tegnene "//" foran de aktuelle linjene en ønsker å holde utenfor. Systemet tolker alt som kommer bak "//" som hjelpetekst, og det vil derfor ikke kjøres.

- I de neste trinn tar en så bort denne hjelpetekst-markeringen for flere og flere kommandolinjer helt til hele skriptet er ferdig kjørt. Husk at de eksekverte kommandolinjene ligger i minnet, så systemet kjører i praksis ikke alle linjene på nytt hver gang, kun de linjene der en fjerner "//"-tegnene

Eksempel: Bruk av kommentarlinjer

Eksempel: Lineære regresjonsanalyser #6

```
1  textblock
2  Lineær regresjonsanalyse
3  ------------------------
4
5  Lineære regresjonsanalyser (OLS) brukes til å estimere marginaleffekter/
6  koeffisientverdier for et sett med forklaringsvariabler, der
7  utfalls-/responsvariabelen er metrisk. Gjennom opsjoner kan en tilpasse outputen
8  (ikke vise fastleddet, endre på signifikansnivået m.m.).
9  endblock
10
11 // Starter med å hente variablene en trenger
12 create-dataset demografidata
13 import BEFOLKNING_KJOENN as kjonn
14 import BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
15 import SIVSTANDFDT_SIVSTAND 2000-01-01 as sivstand
16 import INNTEKT_BRUTTOFORM 2000-01-01 as formue
17 import INNTEKT_WYRKINNT 2005-01-01 as innt05
18
```

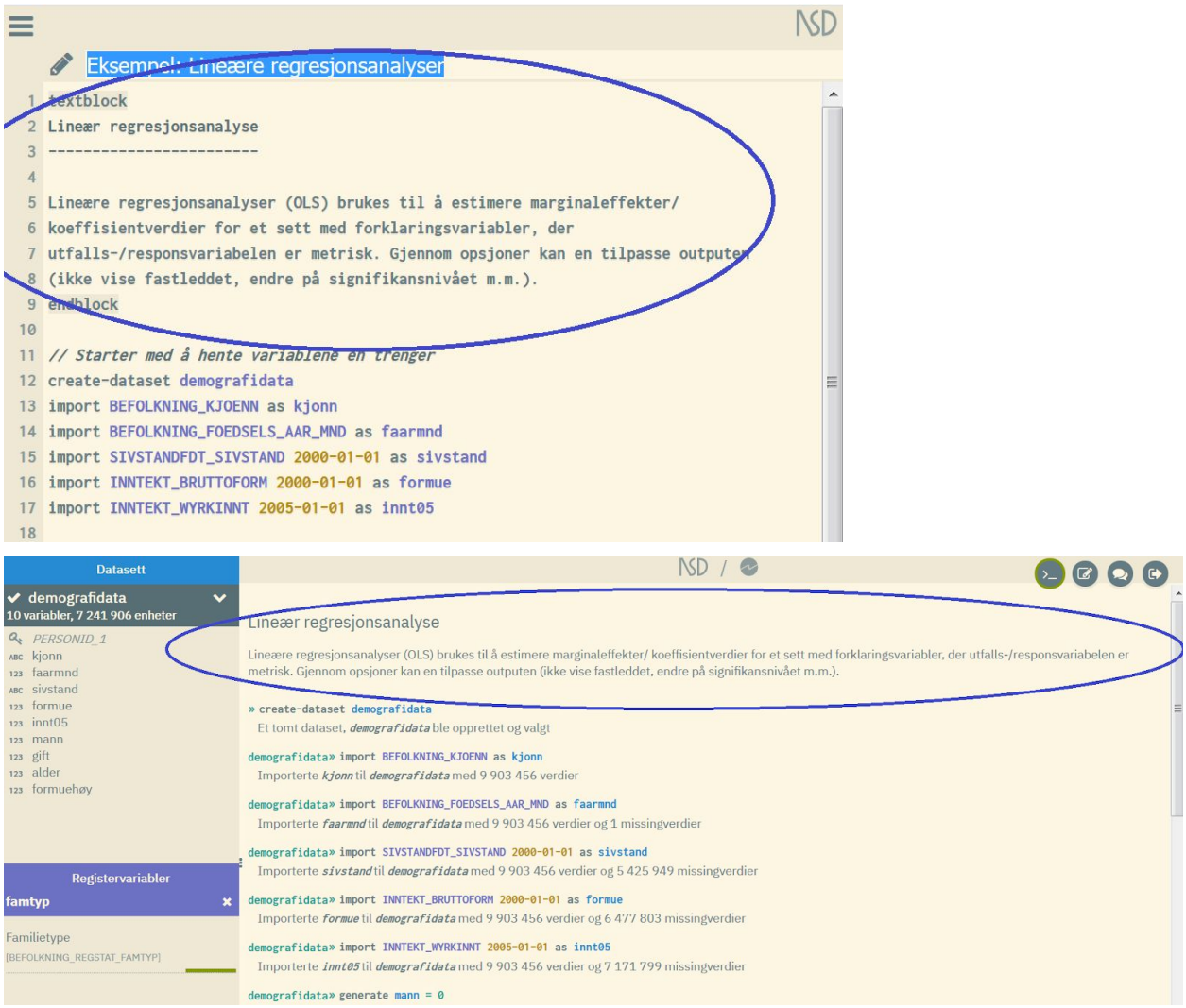
B) Markere ut større deler av et skript ved å definere blokker som hjelpetekst

- Legg inn hhv. `textblock` og `endblock` før/etter en blokk med kommandolinjer. Alt i mellom vil da bli holdt utenom kjøringen. Merk at meningen med `textblock` og `endblock` er å legge inn analysetekst/kommentarer til analyser utført i analyseområdet. Analyseområdet kan derfor fylles opp med de kommandolinjene som ble "markert ut", og det kan se litt rotete ut. Men etterhvert som færre og færre linjer blir holdt utenfor, vil analyseområdet inneholde gradvis mindre av dette.

- Fordelen med `textblock` og `endblock` er at det er mindre tidkrevende dersom antallet linjer som skal markeres ut er omfattende

- På samme måte som ved bruk av tegnene "//", vil systemet "huske" det som er kjørt fra før og vil hoppe rett ned til den delen av skriptet der en har tatt bort "textblock-markeringen". Merk at dette ikke fungerer dersom det er begynnelsen av skriptet som først blir markert ut, for så å bli kjørt etterpå

Eksempel: Bruk av textblock i skript



1.5.2 Fordeler ved bruk av skripteditor

Det er mange fordeler ved å aktivt bruke skripteditoren til å eksekvere sekvenser av kommandoer:

- Fungerer som sikring av arbeid
    - Når en navngir et skript, lagres det i systemet og kan hentes frem igjen når en måtte ønske
    - En kan lagre skript i tekstformat ved å kopiere over til et tekstdokument. Dette fungerer som ekstra sikkerhet. Om en av ulike grunner mister arbeidet, kan en hente det frem, lime det inn i editoren og kjøre på nytt. Da gjenskapes alt

analysearbeidet slik det opprinnelig var. NB! Ekstern lagring av skript bør gjøres i rent tekstformat av typen ".txt" via programmer som Notisblokk o.l. Mer avanserte tekstbehandlingsverktøyer som Word og Google Doc foretar tekstformateringer som gjør at enkelte tegn kan bli forandret, bl.a. enkeltfnutter. Når dette så limes inn i microdata.no igjen, risikerer en at systemet ikke gjenkjenner tegnene og låser seg.

- Skript er en måte å systematisere og huske arbeidet på. En kan endre på rekkefølgen i kommandosekvenser eller gjøre andre justeringer, og dessuten legge til hjelpetekst (se kapittel 1.5.1) som gjør det lettere både for seg selv og andre å skjønne hva hensikten med de ulike operasjonene er

- Skript fungerer som en logg over arbeid (kan legges til analyserapporter for å dokumentere arbeidet)

- Det blir enkelt å gjøre justeringer i en analyse. Om en ønsker å gjøre ting litt annerledes, kan en endre skriptet og kjøre på nytt. Endrede skript kan lagres med nye navn. Det gjør det enklere å dokumentere og sammenlikne resultater

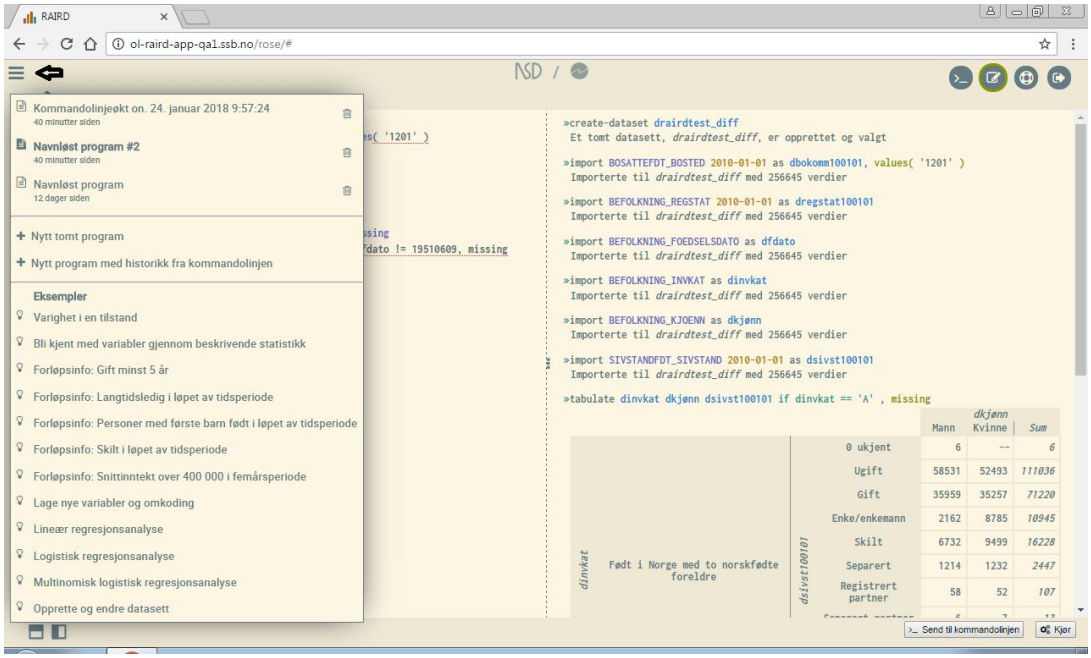
- Å bruke skriptmuligheten aktivt gjør det enklere å samarbeide med andre. En kan sende skript i tekstformat til andre kollegaer, f.eks. via mail

- En kan jobbe med skript på samme måte som i Google Doc eller andre tekstbehandlingsprogrammer som f.eks. Word: Det er mulig å redigere ved å klippe og lime tekst, markere tekstbolker og flytte rundt på dem etter behov

- Systemet "husker" tidligere kjørte skript gitt at de er uendret. Dermed vil det bare ta noen sekunder å gjenskape et tidligere kjørt resultat. Merk at dersom en endrer enkelte ting i et skript og kjører på nytt, så vil systemet behandle dette som et helt nytt sett med kommandoer, og det vil ta betraktelig mye lenger tid å kjøre gjennom det. Om en derimot har behov for å kun justere på kommandolinjer på slutten av et skript, vil systemet hoppe rett til den aktuelle delen og kun bruke ressurser på å prosessere dette. Det øvrige blir hentet frem fra minnet.

1.5.3 Organisering av kommandoskript

Bildet under viser de ulike mulighetene en har for å organisere skript (programmer). Menyknappen øverst til venstre i skriptvinduet gir mulighet til å opprette et nytt tomt program (om en vil starte fra scratch), eller å hente inn alle kommandoer en har brukt i det aktive arbeidsvinduet ("Nytt program med historikk fra kommandolinjen").



Innholdet i aktive skript lagres fortløpende med et defaultnavn etterhvert som en arbeider (slik som i Google Doc). Ved å legge inn en selvvalgt tittel øverst i skriptet, der hvor det står "Navnløst program", vil defaultnavn erstattes med dette. Alt arbeid en gjør på skriptet vil da automatisk lagres med dette navnet.

En kan lagre så mange skript en ønsker gjennom å navngi dem med nye navn. Systemet vil også foreta en automatisk lagring av gjeldende skript med jevne mellomrom (sikkerhetskopi).

Nederst i program-menyen finnes det eksempler på kommandosekvenser for ulike typer oppgaver. De kan brukes som aktive skript som kan kjøres direkte. De kan også redigeres og lagres med nye navn (kan brukes som en mal).

1.5.4 Problemløsninger ved bruk av skript

Det er nesten ikke til å unngå at systemet finner feil i kommandosyntaxen når skript kjøres. Dette kan være feilstavinger eller en logiske feil som ikke lar seg eksekvere. Systemet vil da stoppe kjøringen av skriptet der feilen befinner seg, og markere den aktuelle linjen. En passende feilmelding vil også bli gitt.

Løsning:

i) Sjekk hva som kan ha gått galt. Se spesielt på den linjen som er markert med feil. Kjør den delen av skriptet som ikke inneholder feil, altså frem til den linjen som viser feil (se kapittel 1.5.1 for hvordan en kjører deler av et skript)

ii) Dobbeltsjekk om syntaxen er riktig, at variabelnavnet er riktig skrevet, at datoen for import er gyldig (det kan tenkes at en variabel ikke har data for det aktuelle måletidspunktet)

iii) Bruk statistiske hjelpemidler som kommandoene `tabulate` eller `summarize`. Se om det er noe feil i måten de aktuelle variabler er kodet på. Sjekk også om verdiformatet er korrekt (numerisk eller alfanumerisk)

iv) Dummyvariabler eller kategoriske variabler der minst én av kategoriene har få observasjoner kan føre til uønskede analyseresultater:

- Ved bruk av regresjonsanalyser vil kun de enhetene med gyldige verdier for samtlige variabler som inngår bli analysert
- Enkelte dummyvariabler kan i utgangspunktet ha tilstrekkelig antall observasjoner for begge verdiene 0 og 1, men kan etter at enheter holdes utenfor regresjonsanalysen nå stå med observasjoner kun for én av verdiene
- Analysen vil da bli stoppet og en feilmelding bli gitt. En løsning kan da være å kode om de aktuelle variablene slik at en får flere enheter i kategoriene med minst observasjoner i. Eventuelt kan en droppe de problematiske variablene fra analysen

2. Oppretting og endring av datasett

Dataanalyser i microdata.no krever at en først knytter seg opp mot en databank, deretter oppretter et tomt datasett, og så importerer de ønskede variabler inn i datasettet. Sistnevnte gjøres gjennom bruk av import-kommandoer (se kapittel 2.3.1, 2.3.2 og 2.4).

Om en ønsker å justere på utvalget eller fjerne variabler en ikke ønsker å jobbe videre med, brukes kommandoene `drop` eller `keep` (se kapittel 2.6 og 2.7). En kan spesifisere hvilken variabel en ønsker å slette. Dersom ingen variabel spesifiseres, slettes hele records i kombinasjon med en IF-betingelse.

2.1 Koble til databank

Når en logger seg inn i analyseområdet for første gang, eller oppretter en helt ny arbeidsøkt, vil alle vinduer være tomme. For å kunne lage datasett er det nødvendig å først koble seg opp mot tilgjengelig databanker. Som regel er det tilstrekkelig å koble seg opp mot databanken som inneholder en omfattende samling med registerdata fra SSB. Microdata.no vil etterhvert tilby databanker med data også fra andre datakilder enn SSB.

En kan selv bestemme hvilken versjon av databanken en vil koble seg mot. Siste versjon vil alltid inneholde de nyeste variabler og årganger, og anbefales derfor å bruke. Det er imidlertid mulig å koble seg mot tidligere versjoner også, om en f.eks. ønsker å avdekke eventuelle effekter som kan knyttes til forskjeller i versjonene. Når data oppdateres med nye årganger, vil dette kunne påvirke hendelser tilbake i tid for de ulike variablene. Derfor opprettes det alltid nye databank-versjoner i slike tilfeller.

I variabeloversikten vil en finne en oversikt over alle tilgjengelige databanker, versjoner av disse, og hvilke variabler de inneholder. Kapittel 1.2 beskriver dette nærmere.

Kommando for kobling mot databanker:

```
require <databank:versjon> as <alias>
```

Eksempel:

```
require no.ssb.fdb:1 as fdb1
```

2.2 Opprette et datasett

For å kunne jobbe med data og analyser i microdata.no, trenger en å opprette et datasett som en kan fylle med variabler. Dette gjøres ved å skrive følgende kommando i kommandolinjen:

```
create-dataset <datasett>
```

Eksempel:

```
create-dataset mittdatasett
```

Det er tilstrekkelig å opprette ett datasett, men en kan i prinsippet opprette så mange en ønsker. Ett eksempel der dette er aktuelt er når en vil arbeide med variabler organisert ulikt (har forskjellige enhetsnivåer). I tillegg til et datasett på individnivå (én record per individ) kan en bruker ønske å analysere andre datasett organisert med enkelthendelser som enhet.

En kan i prinsippet opprette et datasett *før* en kobler seg mot en databank, men det vil ikke være mulig å hente variabler før oppkoblingen mot databanken er foretatt.

2.3 Hente variabler inn i et datasett

Neste trinn er å fylle datasettet med ønskede variabler.

Alle underliggende variabler i microdata.no er i utgangspunktet organisert på samme måte; på hendelsesnivå:

individnummer x verdi x startdato x stoppdato

I microdata.no skiller en videre mellom 4 typer variabler:

1) Hendelsesvariabler med sekvenser av hendelser (hver observasjon representerer en tilstandsendring, dvs. at variabelen endrer verdi, og en har variable start- og stoppdatoer)

2) Konstante variabler med kun én observasjon per enhet (faste opplysninger som f.eks. kjønn, fødselsdato, fødeland)

3) Statistiske statusvariabler målt på faste tidspunkt (startdato=stoppdato, og en kjenner kun verdien på det aktuelle tidspunktet)

4) Variabler med kumulative årlige aggregat som gjelder fra 1/1 til 31/12 for hvert år (årsinntekt o.l.)

En bygger opp datasett gjennom å benytte kommandoen `import`, der en vanligvis spesifiserer en uttrekksdato/måledato (unntaket er variabler med konstante verdier som f.eks. kjønn).

I kapittel 2.3.1 vises det i detalj hvordan man bruker kommandoen `import` til å importere variabler inn i et datasett med person som enhetsnivå. Kapittel 2.3.2 viser alternativ import av variabler med hendelsesnivå som enhet (personer er representert ved flere observasjoner over tid, avhengig av antallet hendelser som har skjedd). Til dette brukes kommandoen `import-event`.

2.3.1 Datasett med tverrsnittsopplysninger

Kommandoen `import` brukes til import av følgende opplysninger:

- Konstante opplysninger
- Tverrsnittsopplysninger på valgfrie datoer (gjør i praksis et valgfritt uttrekk fra hendelsesvariabler)
- Statusopplysninger på forhåndsgitte måledatoer
- Årlige aggregerte opplysninger

En spesifiserer navnet på variabelen en ønsker å hente til sitt arbeidsdatasett, samt datering for når opplysningen skal måles. Systemet foreslår relevante variabler og dateringer gjennom en selvutfyllingsfunksjon som minimerer sjansen for å skrive feil.

For hver gang `import` kjøres, vil en ny variabel legges til arbeidsdatasettet (kobles på automatisk). Det resulterende datasettet vil bestå av én observasjon per enhet (individ), og med et valgfritt antall variabler.

Ved import av konstante variabler trenger ikke måledatering oppgis:

```
import <variabel> as <alias>
```

For øvrige variabler må en oppgi en ønsket måledato formatet *YYYY-MM-DD:*

```
import <variabel> <måledato> as <alias>
```

For statusvariabler må imidlertid aktuell statistikkdato benyttes siden verdiene i prinsippet kun vil gjelde bare for denne aktuelle datoen (en kjenner ikke til faktiske endringsdatoer for slike variabler). Analysesystemet microdata.no forslår i slike tilfeller de aktuelle statistikkdatoer gjennom selvutfyllingsfunksjonen slik at dette blir lett å oppgi. Ved import av tverrsnittsdata foreslås i stedet den sist brukte dateringen. For variabler med årlige kumulative mål er det den årlige verdien for det aktuelle året en importerer, så det har ikke noe å si konkret hvilken dato en oppgir så lenge en angir riktig år.

Eksempel: Datamatrise ved bruk av import (4 variabler)

ID	**Variabel 1**	**Variabel 2**	**Variabel 3**	**Variabel 4**
123456	1	200000	0301	1
135791	1	410000	0301	1
147036	2	515000	1201	sysmiss
159371	2	309011	1101	sysmiss
160505	2	357000	1101	1
173951	2	399000	0301	3

Viktig:

Kommandoen `import` gjør i praksis to operasjoner:

a) Henter verdier for en gitt variabel
b) Kobler variabel på eksisterende datasett gjennom en såkalt “left-join” kobling

At variabler kobles på datasettet gjennom “left-join” innebærer at en kun importerer verdier for enheter (individer) i det eksisterende datasettet. Derfor er det viktig å starte med å importere en variabel der færrest mulig i en tenkt populasjon har manglende verdier, som f.eks. kjønn, landbakgrunn eller fødselsdato. Starter en derimot med å importere en variabel som angir sykefravær på en gitt dato, er det kun personene med sykefravær på denne datoen som en

arbeider videre med ved etterfølgende import-steg. Det vil da ikke være mulig å hente opplysninger om andre personer i senere trinn. Det første import-steget vil med andre ord definere utvalget en jobber videre med.

Merk at en har anledning til å "trimme" ned utvalgspopulasjonen underveis i prosessen med å bygge opp et datasett, gjennom kommandoene `drop` og `keep`, jfr. kapittel 2.6.

Personer i et eksisterende datasett som har manglende verdi for en importert variabel vil fortsatt være med i utvalget, men vil få en såkalt sysmiss-verdi (se kapittel 2.6).

Om en har en klar idé om hvilke enheter (individer) som skal inngå i en analysepopulasjon, kan det være lurt å "trimme" ned utvalget en jobber med så tidlig som mulig. Dette vil kunne gi betydelige forbedringer i hvor raskt systemet jobber.

Om første variabel som importeres er av universell karakter, f.eks. "Kjønn", vil datasettet ditt bestå av flest mulig individer fra den totale databasen, inkludert personer som er døde, emigrert eller ikke født på det aktuelle tidspunktet en er interessert i. Dette kan løses ved å importere variabelen "BEFOLKNING_REGSTAT" målt ved det aktuelle tidspunktet, og deretter beholde alle individer som har verdien '1' (= bosatt i Norge). Eksempel:

```
require no.ssb.fdb:1 as fdb1
create-dataset demografi
import fdb1/BEFOLKNING_KJOENN as kjønn
import fdb1/BEFOLKNING_REGSTAT 2005-01-01 as regstat05
keep if regstat05 == '1'
```

2.3.2 Datasett med hendelsesopplysninger

I tillegg til å hente ut opplysninger på valgte eller gitte datoer, kan en foreta beregninger basert på hendelser over tid. F.eks. kan en være interessert i å finne individer som giftet seg i løpet av en lengre tidsperiode, som ble arbeidsledig over et gitt tidsspenn, eller som var arbeidsledige i over 6 måneder i en gitt periode. Til dette benyttes kommandoen `import-event` som importerer *alle* records (= hendelser) per enhet (= individ) over et angitt tidsspenn. I tillegg til variabelnavnet angis 2 tidspunkter for datauttrekkets hhv. start- og stopp-tidspunkt. Da vil alle hendelser som har skjedd mellom de to tidspunktene hentes til ditt datasett. Datasettet vil inneholde et varierende antall records for hver enhet (= individ), avhengig av hvor mange endringshendelser som har skjedd for hver enkelt.

Merk at en bare kan importere én hendelsesorganisert variabel til et gitt datasett, og at et slikt datasett ikke kan inneholde andre variabler. En må altså opprette ett arbeidsdatasett for hver hendelsesorganiserte variabel en trenger å arbeide med. Importen gjøres på følgende måte:

```
create-dataset <dataset>
import-event <variabel> <startdato> to <stoppdato> as <alias>
```

Eksempel: Datamatrise ved bruk av import-event (tidsintervall: 2000-01-01 - 2003-01-01)[1]

ID	Start	Stopp	Variabel
123456	2000-01-01	2000-05-30	1
123456	2000-05-31	2001-12-31	4
123456	2002-01-01	2003-08-15	2
135791	2000-04-10	2002-03-03	2
135791	2002-03-04	2002-11-11	3
147036	2002-02-28	2004-07-16	1

[1] Merk at alle hendelser som overlapper med perioden 2000-01-01 - 2003-01-01 tas med ved import

2.3.3 Tverrsnitts- vs. hendelsesorganiserte datasett

Til forskjell fra tverrsnittsorganiserte datasett som bygges opp gjennom kommandoen `import`, kan en ikke bygge opp datasett gjennom `import-event`. Grunnen til dette er at datauttrekk på hendelsesnivå alltid vil ha ulikt antall records per individ, og det vil gi liten mening å koble slike uttrekk sammen til et felles datasett. Derfor må en opprette et nytt datasett for hvert hendelsesbaserte datauttrekk (se kapittel 2.3.2).

Hensikten med hendelsesorganiserte datauttrekk er som nevnt å foreta beregninger basert på hendelser over tid, gjennom kommandoen `collapse`. Dette vil transformere det hendelsesbaserte datasettet til et datasett på enhetsnivå (én record per individ), med det aggregerte målet som variabelverdi (målt over det angitte tidsspenn), noe som gjør det mulig å koble variabelen sammen med tverrsnittsorganiserte datasett for videre analyser.

I kapittel 2.8 beskrives metoden for å koble sammen datasett.

2.4 Datasett med regelmessige målinger over tid (paneldata)

For å kunne foreta avanserte regresjonsanalyser i form av paneldataanalyse, må data organiseres på en annen måte enn ved vanlige regresjonsanalyser. Paneldata er datasett der hver enhet har oppgitt verdier for samtlige variabler målt over et gitt antall måletidspunkt. Dette har den fordelen at en kan ta med tidskomponenten i analyser, og at en får mye større datagrunnlag og gjerne analyser av en bedre kvalitet.

Det finnes et stort batteri av teknikker for paneldataanalyse, skillet går på hvilke antakelser som gjøres om variablenes variasjon over tid. Vanlige varianter som brukes er “fixed effect”- og “random effect”-analyser. Denne analyseformen vil bli gjennomgått i kapittel 5.6.

Data som skal brukes i paneldataanalyse må importeres på følgende måte:

```
create-dataset <dataset>
import-panel <variabelliste> <måledatoliste> as <alias>
```

Eksempel: Datamatrise ved bruk av import-panel (3 variabler, 3 måletidspunkt)

ID	**Tid**	**Variabel 1**	**Variabel 2**	**Variabel 3**
123456	2000-01-01	1	200000	0301
123456	2001-01-01	1	210000	0301
123456	2002-01-01	2	215000	1201
135791	2000-01-01	2	305011	1101
135791	2001-01-01	2	301000	1101
135791	2002-01-01	3	299000	0301
147036	2000-01-01	1	150000	2030
147036	2001-01-01	1	159000	2030
147036	2002-01-01	3	199000	0301

Merk:

- Paneldatasett blir fort veldig store ettersom alle enheter/individer i datasettet måles T ganger, der T står for antall målinger. Dette gjelder særlig om en importerer mange variabler i tillegg
- En god praksis ved opprettelse av paneldatasett er å først lage en populasjon av passende størrelse, så duplisere denne og til slutt importere paneldata inn i det tomme datasettet med den dupliserte populasjonen

Eksempel: Lage populasjon, duplisere enheter inn i nytt datasett, og til slutt importere paneldata for den gitte populasjonen (= bosatte i Oslo per 1/1 2010 i alderen 18-39 år)[1]

```
require no.ssb.fdb:1 as fdb1

create-dataset populasjon
import fdb1/BOSATTEFDT_BOSTED 2010-01-01 as bosted
import fdb1/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
generate alder = 2010 - int(faarmnd/100)
keep if alder >= 18 & alder < 40 & bosted == '0301'

clone-units populasjon paneldata

use paneldata
import-panel fdb1/INNTEKT_WLONN fdb1/SIVSTANDFDT_SIVSTAND
fdb1/BOSATTEFDT_BOSTED 2011-12-31 2012-12-31 2013-12-31 2014-12-31
2015-12-31
```

[1] Paneldatasett lages ved hjelp av en enkelt import-panel-kommando. En kan ikke importere i flere omganger til det samme paneldatasettet. En kan heller ikke mikse vanlige tverrsnittsdata og/eller forløpsdata med paneldata.

2.5 Forflytte seg mellom datasett

For å flytte seg mellom datasett brukes følgende kommando:

```
use <dataset>
```

Unntaket er når en oppretter et nytt datasett gjennom `create-dataset`. Da flyttes en automatisk til det nye datasettet.

2.6 Filtrering av datasettutvalg

Man kan benytte en filteropsjon til å spesifisere hvilke verdier som skal inngå i datasettet, f.eks. om en bare vil importere kvinner:

```
import BEFOLKNING_KJOENN as kjønn, values ( '2' )
```

Alternativt kan en først importere variabler på vanlig måte og så trimme ned utvalget etterpå gjennom kommandoene drop eller keep:

```
import BEFOLKNING_KJOENN as kjønn
drop if kjønn == '1'
```

IF-betingelser kan brukes i mange sammenhenger i microdata.no, også i forbindelse med `drop` og `keep`, og kan bygges opp med de vanlige logiske operatorene:

- Større enn >
- Mindre enn <
- Er lik ==
- Større enn eller lik >=
- Mindre enn eller lik <=
- Er ulik !=
- Eller |
- Og &

For å fjerne personer under 18 år fra utvalget, kan en skrive følgende:

```
keep if alder >= 18
```

Verdi for manglende data (“missingverdier”) kan angis på følgende måte:

```
sysmiss(<variabel>)
```

For å fjerne alle individer uten oppgitt lønnsinntekt, kan en da skrive:

```
drop if sysmiss( lonn )
```

Det er også mulig å trekke et tilfeldig utvalg av en datapopulasjon. Dette gjøres med kommandoen `sample`. For mer om syntax og eksempler, bruk kommandoen `help sample`.

2.7 Fjerning av variabler fra datasett

I en analysesituasjon ser en ofte underveis at en ikke trenger alle variabler en først har importert. En del variabler brukes gjerne bare som grunnlag for å avlede verdier for nye variabler, og da trenger en ikke dra med seg alle de underliggende.

Å rydde et datasett gjøres gjennom kommandoen `drop`, der en legger til navnet på den variabelen en vil fjerne:

```
drop <variabel>
```

Som vi har sett, kan denne kommandoen brukes både til å fjerne enheter (= rekker i datamatrisen), jfr kapittel 2.6, **og** variabler (= kolonner i datamatrisen).

2.8 Kobling av datasett

Datasett bygges vanligvis opp gjennom kommandoen `import,` der en legger til én og én variabel målt ved gitte tidspunkter. Slike variabler har samme enhetstype, dvs. person gitt ved identifikatoren `PERSONID_1`. Sammenkoblingen gjøres automatisk av analysesystemet, og en trenger bare å forholde seg til `import`-kommandoen der en spesifiserer variabelnavn, uttrekksdato og eventuelt alias.

Analysesystemet gjør det imidlertid mulig å analyse data med andre enhetstyper. Det kan være på hendelsesnivå, kommunenivå, familienivå, kursnivå, etc. Slike data med andre enhetstyper enn person kan ikke uten videre importeres rett inn i et persondatasett. De må først bearbeides til passende enhetsnivå gitt ved variabelen en ønsker å bruke som koblingsnøkkel i måldatasettet, og deretter kobles på datasettet ved bruk av kommandoen `merge`.

Data på et lavere enhetsnivå enn person, f.eks. hendelses- eller kursnivå, må aggregeres til personnivå ved bruk av kommandoen `collapse()` før bruk av `merge`.

Data på et høyere enhetsnivå enn person, f.eks. kommune- eller familienivå, kan imidlertid kobles på persondatasettet ved bruk av aktuell variabel på samme nivå i måldatasettet (brukes som koblingsnøkkel).

Se kapittel 2.10 for eksempler på hvordan koble på opplysninger om hhv. foreldre, familier og kurs. Sistnevnte illustrerer sammenkoblinger av data på lavere nivå enn person (kursdata er opplysninger om pågående utdanning gitt ved aktuelt kurs/fag som tas på et gitt tidspunkt, der personer kan ta flere kurs samtidig). Data om foreldre og familier illustrerer sammenkoblinger av data på *høyere* nivå enn person.

2.9 Eksempler: Oppretting og justering av et datasett

```
textblock
Henter først variablene en trenger

Først kobler en seg mot databanken, deretter opprettes et datasett som kalles `demografidata` ved
hjelp av `create-dataset <dataset>`. Da vil all import og all bearbeiding foregå mot dette med mindre
en aktivt skifter datasett med kommandoen `use <dataset>`

Tilhørende start- og stoppdatoer følger også med ved import
endblock

require no.ssb.fdb:1 as fdb1

create-dataset demografidata
import fdb1/BEFOLKNING_KJOENN as kjonn
import fdb1/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
import fdb1/SIVSTANDFDT_SIVSTAND 2000-01-01 as sivstand
import fdb1/INNTEKT_BRUTTOFORM 2000-01-01 as formue

// Endrer navn på variabler ved å legge til årsangivelse
rename sivstand sivstand00
rename formue formue00

// Sletter variabelen kjonn fra datasettet
drop kjonn

// Beholder kun gifte personer i datasettet
keep if sivstand00 == '2'
```

2.10 Eksempler: Sammenkoblinger av data på andre nivå enn person

Eksempel: Sammenkobling av foreldreopplysninger

```
textblock
Hvordan benytte foreldreopplysninger i analyser
------------------------------------------------------------

I databasen finnes det variabler for fars og mors fødselsnumre, noe som gjør at en kan koble
opplysninger om foreldre på et persondatasett.

Gjennom kommandoen `merge` kan en koble sammen datasett. Identifikatorvariabel i kildedatasettet
brukes som standard. Dette kan imidlertid overstyres gjennom en "on-opsjon".

I eksempelet lages det egne datasett for fedre og mødre, som kobles sammen med et persondatasett
via koblingsnøklene `fnr_far` og `fnr_mor`.
endblock

//Kobler til databank
require no.ssb.fdb:1 as fdb1

// Lager et persondatasett med lenker til far og mor
create-dataset persondata
import fdb1/INNTEKT_WYRKINNT 2010-01-01 as inntekt
import fdb1/BEFOLKNING_KJOENN as kjønn
import fdb1/NUDB_BU 2010-01-01 as utd
import fdb1/BEFOLKNING_FAR_FNR as fnr_far
import fdb1/BEFOLKNING_MOR_FNR as fnr_mor

// Henter opplysninger om far og kobler på persondatasett
create-dataset fardata
import fdb1/INNTEKT_WYRKINNT 2010-01-01 as inntekt_far
import fdb1/NUDB_BU 2010-01-01 as utd_far
merge inntekt_far utd_far into persondata on fnr_far

// Henter opplysninger om mor og kobler på persondatasett
create-dataset mordata
import fdb1/INNTEKT_WYRKINNT 2010-01-01 as inntekt_mor
import fdb1/NUDB_BU 2010-01-01 as utd_mor
```

```
merge inntekt_mor utd_mor into persondata on fnr_mor

// Kjører en enkel lineær regresjon for å teste sammenheng mellom egen og foreldres inntekt
use persondata
generate mann = 0
replace mann = 1 if kjønn == '1'

destring utd, force
generate høyutd = 0
replace høyutd = 1 if utd >= 700000 & utd < 999999
replace høyutd = utd if sysmiss(utd)

destring utd_far, force
generate høyutd_far = 0
replace høyutd_far = 1 if utd_far >= 700000 & utd_far < 999999
replace høyutd_far = utd_far if sysmiss(utd_far)

destring utd_mor, force
generate høyutd_mor = 0
replace høyutd_mor = 1 if utd_mor >= 700000 & utd_mor < 999999
replace høyutd_mor = utd_mor if sysmiss(utd_mor)

summarize inntekt inntekt_far inntekt_mor
histogram inntekt_far, percent
histogram inntekt_mor, percent
correlate inntekt_far inntekt_mor
tabulate høyutd_far høyutd_mor

regress inntekt mann inntekt_far inntekt_mor høyutd høyutd_far høyutd_mor
```

Eksempel: Sammenkobling av data på familienivå

```
textblock
Aggregere opplysninger til familienivå
----------------------------------------

Individer kan knyttes opp mot et familienummer som kan brukes til å aggregere opplysninger på
familienivå. Individer tilhørende samme familie vil være registrert med det samme familienummeret
som består av person-id'en til den eldste personen i familien.

I eksempelet opprettes først et persondatasett der en filtrerer ned på person i familier bestående av
ektepar med små barn (kode 2.1.1). Deretter kobles det på demografiske opplysninger.

Familieinntekt er en opplysning på familienivå, dvs. familie = enhet. Derfor må en opprette et nytt
datasett for dette formålet (datasett kan ikke bestå av variabler med ulike enhetstyper). En
importerer da yrkesinntekt på personnivå, og bruker så kommandoen `collapse (sum)` til å summere
inntektene på familienivå (`by(famnr)`). Resultatet blir et datasett med familie som enhet.

Til slutt kobles familieinntekt på persondatasettet vha. kommandoen `merge`.
endblock

//Kobler til databank
require no.ssb.fdb:1 as fdb1

// Oppretter først et persondatasett for personer i familier bestående av ektepar med små barn
create-dataset persondata
import fdb1/BEFOLKNING_REGSTAT_FAMTYP 2010-01-01 as famtype
tabulate famtype
keep if famtype == '2.1.1'

// Legger til diverse demografiske opplysninger
import fdb1/BEFOLKNING_KJOENN as kjønn
import fdb1/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
generate alder = 2010 - int(faarmnd/100)
import fdb1/BOSATTEFDT_BOSTED 2010-01-01 as bosted
generate fylke = substr(bosted,1,2)
import fdb1/BEFOLKNING_BARN_I_HUSH 2010-01-01 as antbarn

// Oppretter datasett for generering av total yrkesinntekt per familie => enhet = familie
create-dataset familiedata
```

```
import fdb1/BEFOLKNING_REGSTAT_FAMNR 2010-01-01 as famnr
import fdb1/INNTEKT_WYRKINNT 2010-01-01 as yrkesinnt
collapse (sum) yrkesinnt, by(famnr)
rename yrkesinnt familieinnt

// Kobler familieinntekt på persondatasettet (enhet = personer)
merge familieinnt into persondata on PERSONID_1

// Lager familiestatistikk. Familienummeret består av person-id til eldste person i familien, så når en
fjerner individer med manglende yrkesinntekt (=familieinntekt) sitter en igjen med et datasett med
familie som enhet. Alle personopplysninger vil da gjelde for eldste person i familien
use persondata
drop if sysmiss(familieinnt)

rename alder alder_eldst
rename kjønn kjønn_eldst

define-labels fylketekst '01' 'Østfold' '02' 'Akershus' '03' 'Oslo' '04' 'Hedmark' '05' 'Oppland' '06'
'Buskerud' '07' 'Vestfold' '08' 'Telemark' '09' 'Aust-Agder' '10' 'Vest-Agder' '11' 'Rogaland' '12'
'Hordaland' '14' 'Sogn og Fjordane' '15' 'Møre og Romsdal' '16' 'Sør-Trøndelag' '17' 'Nord-Trøndelag'
'18' 'Nordland' '19' 'Troms' '20' 'Finnmark' '99' 'Uoppgitt'

assign-labels fylke fylketekst

tabulate fylke

histogram alder_eldst, discrete
histogram antbarn, percent

tabulate antbarn
tabulate antbarn, cellpct
tabulate antbarn kjønn_eldst

summarize familieinnt
barchart (mean) familieinnt, by(fylke)
barchart (mean) familieinnt, by(antbarn)
histogram familieinnt, freq
histogram familieinnt, by(antbarn) percent
```

Eksempel: Sammenkobling/uthenting av data på kursnivå

```
textblock
Hente ut informasjon om pågående utdanning (kurs)
------------------------------------------------

Opplysninger om pågående utdanning (såkalte kursdata) foreligger med kurs som enhetsnivå (med tilhørende kurs-identifikator). Kurs er gitt ved kombinasjonen person x kurstype, der hvert individ i praksis kan være representert med flere kurstyper til samme tid.

Siden kursdata ikke har person som enhet, kan ikke disse importeres inn i persondatasett på vanlig måte, men må kobles på vha. kommandoen `merge`.

Først må en legge til en lenke mellom kurs-id og person-id på kursdataene. Deretter må en aggregere opp til personnivå vha. kommandoen `collapse` før en tilslutt kobler på persondatasettet.

I eksempelet opprettes det først et persondatasett bestående av personer bosatt i Norge (regstatus == '1') per 2010-01-01. Deretter hentes det ut historikk over pågående utdanning for perioden 2010-2012, der en tar vare på utdanning på høyere nivå (master eller høyere, nivå 7 og 8). Kommandoen `collapse (count)` brukes til å telle opp antall observasjoner med pågående utdanning per individ over perioden 2010-2012, og resultatet kobles så på persondatasettet for videre analyse.

NB! Merk at variabelen `kurstype` etter bruk av `collapse` vil bestå av verdier for den aktuelle statistikken som blir kjørt, i dette tilfellet antall observasjoner (`count`).
endblock

//Kobler til databank
require no.ssb.fdb:1 as fdb1

// Oppretter persondatasett for bosatte per 2010-01-01 med variabel kjønn
create-dataset persondata
import fdb1/BEFOLKNING_KJOENN as kjønn
import fdb1/BEFOLKNING_REGSTAT 2010-01-01 as regstatus
keep if regstatus == '1'

// Henter personer som tar høyere utdanning i perioden 2010-2012
create-dataset kursdata
import-event fdb1/NUDB_KURS_NUS 2010-01-01 to 2012-01-01 as kurstype
destring kurstype, force
keep if kurstype >= 700000 & kurstype < 999999
```

```
// Kobler på lenke mellom kurs-id og fødselsnumre
create-dataset lenke_kurs_person
import fdb1/NUDB_KURS_FNR as fnr
merge fnr into kursdata

// Lager statistikk (collapser) over antall hendelser med høy utdanning per individ, og kobler dette på persondatasettet
use kursdata
collapse (count) kurstype, by(fnr)
rename kurstype ant_kurs
merge ant_kurs into persondata

// Lager tabell over antall personer som tok høy utdanning i 2010-2012
use persondata
generate utdanning_høy = 0
replace utdanning_høy = 1 if ant_kurs >= 1
tabulate utdanning_høy kjønn
```

3. Tilrettelegging av variabler

De fleste variabler som importeres til et datasett må kodes om før videre analyse. Dessuten har en som regel behov for å lage egne variabler basert på de importerte opplysningene. Dette gjøres gjennom kommandoene `generate`, `replace` og `recode`.

3.1 Opprettelse av nye variabler og omkoding: generate/replace

For å lage en ny variabel brukes kommandoen `generate`, der en spesifiserer navnet og hvilken verdi den skal ha. Dette kan være en konkret verdi eller en verdi basert på en likning/formel. IF-betingelser brukes til å angi hvilke tilfeller/enheter som skal få en verdi.

Merk at kommandoen `generate` bare kan brukes til å angi én verdi. Vil en legge til flere koder, kan `replace`-kommandoen benyttes i videre steg for å fullføre prosessen.

En kan også bruke `generate` til å kopiere andre variabler: `generate` *<nyvar>* = *<gammelvar>*. Også dette kan kombineres med IF-betingelser.

Eksempel på koding av dummyen “mann”:

```
import BEFOLKNING_KJOENN as kjønn
generate mann = 1
replace mann = 0 if kjønn != '1'
```

Det er mange mulige måter å lage logiske betingelser på, som alle vil gi samme resultat. En kunne alternativt kodet på følgende måte:

```
import BEFOLKNING_KJOENN as kjønn
generate mann = 0
replace mann = 1 if kjønn == '1'
```

Merk følgende når en koder om eller lager nye variabler:

- "=" brukes når verdier settes gjennom generate eller replace. "==" brukes ved logiske IF-betingelser.

- Verdier for alfanumeriske variabler må angis med "enkeltfnutter" (`'1', '2', …` etc), mens for numeriske variabler angis verdiene som rene tall uten "fnutter" (`1, 2, ….` etc).
    - Variabelformatet finner en ved å se på den aktuelle variabelen øverst til venstre (datasettvinduet) eller nederst til venstre (registervariabelvinduet).

- Kode for manglende data ("missingverdi") angis på følgende måte:
  `sysmiss( <variabel> )`
    - Eksempel (en filtrerer bort records/enheter med manglende verdier for kjønn):

```
import BEFOLKNING_KJOENN as kjønn
generate mann = 1
replace mann = 0 if kjønn != '1'
drop if sysmiss( kjønn )
```

- Følgende logiske operatorer kan brukes i forbindelse med IF-betingelser:

    - Større enn >
    - Mindre enn <
    - Er lik ==
    - Større enn eller lik >=
    - Mindre enn eller lik <=
    - Er ulik !=
    - Eller |
    - Og &

- Dummyvariabler MÅ av metodiske grunner være numeriske og bestå av verdiene 1 og 0. En kan altså ikke ha en dummyvariabel med kun verdien 1. Dette vil generere uønskede resultat, eller feilmelding når en kjører regresjonsanalyser. I praksis må en derfor passe på å kode alle enheter som ikke har "suksess"-verdien med verdien 0 (se eksempel øverst på forrige side).

- Ved bruk av dummyvariabler i IF-betingelser, trenger en ikke angi verdien 1.
    - Eksempel: I stedet for `tabulate sivstand if mann == 1`, kan en skrive `tabulate sivstand if mann`

- Om hensikten med tilretteleggingen av variablene er å benytte regresjonsanalyser, bør kategoriske verdier kodes på numerisk form. Hvis ikke risikerer en at systemet ikke

godtar variabelinputen, og at en får feilmelding når en kjører kommandoer som `regress, logit,` etc.

- Av metodiske grunner bør kategoriske variabler vanligvis tilrettelegges som dummyvariabler slik som i eksempelet med variabelen “mann” ovenfor. Dette gjelder også flerkategorivariabler (mer enn to kategorier) som f.eks. “Utdanningsnivå”. I slike tilfeller lager en et sett med dummyvariabler som i kombinasjon tilsvarer flerkategorivariabelen (i praksis vil hver kategori minus referanse-/basis-kategorien representeres ved separate dummyvariabler, der en tolkningsmessig måler effekten av de enkelte kategorier sammenliknet med referansekategorien). Prosessen med å lage sett av dummyvariabler kan automatiseres gjennom å bruke prefikset “i.” foran variabelnavnet i de ulike regresjons-uttrykkene. Da benyttes automatisk den laveste verdi som referanseverdi.

- Missingverdier: Vær obs på at alle enheter der minst én av variablene har en missingverdi blir ekskludert fra regresjonskjøringer. Variabler med mange missingverdier som ikke blir kodet om vil da kunne føre til at regresjonsanalysen blir utført på et mye mindre datasett enn planlagt. Dette er noe en bør være klar over under tilretteleggingen. I eksempelet med kjønn vil det typisk være få enheter/individer med missingverdi, men det kan være andre variabler som angir f.eks. trygdeytelser som “Uføregrad”. Et flertall vil her ha missingverdi, og bare dem som er uføre vil ha en gyldig verdi. En bør da kode på følgende måte:

```
import PENSJONER_UFOERGRAD 2010-01-01 as uførgrad
generate ufør = 1
replace ufør = 0 if sysmiss( uførgrad )
```

- Missingverdier for inntektsvariabler: Dette vil typisk gjelde alle personer med inntekt = 0. Hvis også disse bør være med i analysen, må de kodes om til 0-verdier:

```
replace inntekt = 0 if sysmiss(inntekt)
```

3.2 Omkoding av variabler: recode

Alternativt til kommandoen `replace,` kan `recode` benyttes til å kode om variabler. For variabler med mange verdier som skal omkodes, er denne kommandoen et nyttig verktøy. Omkodingen kan da i mange tilfeller fullføres i en enkelt kommandolinje, noe som bidrar til kortere tidsbruk for prosesseringen. En kan dessuten bruke kommandoen til å kode om flere variabler om gangen.

Eksempel på koding av variabelen "mann" ved hjelp av kommandoen `recode`:

```
import BEFOLKNING_KJOENN as mann
destring mann, force
recode mann (2 = 0)
```

Merk at kommandoen `recode` kan kun benyttes på numeriske variabler. Alfanumeriske variabler (stringvariabler) må først konverteres til numerisk verdiformat gjennom å benytte kommandoen `destring,` slik som vist i eksempelet over. Opsjonen `force` anbefales i tillegg, siden denne sørger for at konverteringen gjennomføres også i de tilfeller hvor det finnes enkeltverdier som inneholder ikke-numeriske tegn (disse settes til missing-verdi). Se kapittel 3.7 for mer informasjon om `destring`.

Eksempler på måter å kode om grupper av tall for variablene *var1* og *var2*:

- `recode var1 var2 (1 2 3 = 0)` *(verdiene 1-3 kodes til 0)*
- `recode var1 var2 (1/7 = 0)` *(verdiene 1-7 kodes til 0)*
- `recode var1 var2 (1/7 = 0) (nonmissing = 1) (missing = 99)` *(øvrige gyldige verdier kodes til 1, missingverdier kodes til 99)*
- `recode var1 var2 (1/7 = 0) (* = 99)` (alle andre verdier kodes til 99)

3.3 Bruk av funksjoner

I tillegg til de vanlige matematiske operatorene

`=, +, -, /, *, ( , )`

har en gjennom microdata.no tilgang på et stort antall funksjoner som vil være til hjelp når en skal generere variabler. Et konkret eksempel er når en skal angi bosted på fylkesnivå. Siden bostedsinformasjon i utgangspunktet angis som kommunenummer (= tosifret fylkesnummer + tosifret tilleggsnummer som angir kommune) gjennom en firesifret alfanumerisk variabel, må en benytte funksjonen `substr()` for å hente ut de to første sifrene som angir fylke:

```
generate fylke = substr(bosted,1,2)
```

Sifrene 1 og 2 inni funksjonen angir hhv. startposisjon for verdien som skal leses inn og antall posisjoner som skal leses. Kommunen Bergen har verdien '1201'. Å trekke ut de 2 første sifrene vil generere fylkesverdien for Hordaland som er '12'.

Et annet typisk bruksområde for funksjonen `substr()` er når en skal hente ut utdanningsnivå på et grovere aggregeringsnivå enn den totale 6-sifrede kodingen . Det er vanlig å bruke en inndeling på 1-sifret eller 2-sifret nivå. Da er denne funksjonen et nyttig hjelpemiddel.

Andre sentrale funksjoner er `round()` og `int()` evt. `floor()`. Disse er nyttige om en skal gjøre om fra desimaltall til heltall eller trekke ut delverdier fra en større tallverdi. `round()` avrunder desimaltall på vanlig måte, mens `int()` og `floor()` avrunder nedover. Om en f.eks. ønsker å hente ut fødselsåret fra den numeriske variabelen `faarmnd` (år og måned på formen YYYYMM), kan dette gjøres på følgende måte:

```
generate faar = int( faarmnd / 100)
```

I denne operasjonen dividerer vi på 100 og beholder heltallet (samme som å avrunde nedover). Dette vil i praksis trekke ut de fire første sifrene fra en numerisk 6-sifret verdi. For verdien 201006 vil en altså hente ut verdien 2010. Dette kan brukes til å generere alder ut i fra fødselsdato:

```
generate alder = 2013 - int( faarmnd / 100)
```

Resultatet vil gi alder for samtlige individer i datasett målt per 2013. Om fødselsdato er oppgitt med 8 siffer (*YYYYMMDD*), må en justere formelen for å få samme resultat:

```
generate alder = 2013 - int( faarmnd / 10000)
```

I vedlegg B presenteres en fullstendig liste over tilgjengelige funksjoner. Merk at disse stiller krav til hvilke typer variabler de brukes på. F.eks. kan funksjonen `substr()` kun brukes på alfanumeriske variabler.

3.4 Generere aggregerte verdier over tid - collapse

Kommandoen `collapse` benyttes om en ønsker å beregne verdier målt over et angitt tidsspenn. Eksempler kan være beregninger av varighet i en tilstand målt over et gitt tidsintervall, uthenting av tilstand/status i et gitt tidsintervall, uthenting av antall forekomster i gitte tilstander/statuser i et gitt tidsintervall, eller summering av verdier over et gitt tidsintervall.

Dette gjøres på hendelsesorganiserte datasett (se kapittel 2.3.2) gjennom følgende kommando:

```
collapse ( <aggregeringstype> ) <datasett>, by(<enhetsid>)
```

En angir altså hva slags type aggregering en vil foreta i parentesen bak `collapse,` og deretter navnet på et hendelsesorganisert datasett. Aggregeringstype kan være følgende:

- Max
- Min
- Mean
- Count
- Sum

Opsjonen `by(<enhetsid>)` brukes til å angi enhetstypen en skal aggregere over. Dette vil som regel være individ, gitt ved enhetsid `PERSONID_1`.

Eksempel 1: Beregne antall ganger individene har skiftet sivilstand i løpet av 2000-2005

```
require no.ssb.fdb:1 as fdb1
create-dataset sivstforløp
import-event fdb1/SIVSTANDFDT_SIVSTAND 2000-01-01 to 2005-01-01 as
     sivstperiode
collapse ( count ) sivstperiode, by(PERSONID_1)
rename sivstperiode antsivstand
replace antsivstand = antsivstand - 1
tabulate antsivstand
```

Eksempel 2: Beregne antall ganger individene har skilt seg i løpet av 2000-2005

```
require no.ssb.fdb:1 as fdb1
create-dataset sivstforløp
import-event fdb1/SIVSTANDFDT_SIVSTAND 2000-01-01 to 2005-01-01 as
     sivstperiode
keep if sivstperiode == '4'
collapse ( count ) sivstperiode, by(PERSONID_1)
rename sivstperiode antgangerskilt
tabulate antgangerskilt
```

Merk at variabelen `sivstperiode` i utgangspunktet angir sivilstand (hver nye record representerer en endring i sivilstand). Gjennom trinnene i eksemplene blir imidlertid variabelen transformert fra å inneholde sivilstand på hendelsesnivå til etterpå å inneholde `count`-verdien målt over 2000-2005 på enhetsnivå (= individ). Etter at `collapse` er kjørt inneholder variabelen `sivstperiode` altså en verdi som angir hvor mange sivilstatuser hvert individ har hatt over den angitte perioden (eksempel 1) eller hvor mange ganger en har hatt sivilstatusen "skilt" (= antall ganger skilt) (eksempel 2).

NB! For å kunne jobbe videre med den aggregerte verdien generert gjennom `collapse`, må en koble datasettet sammen med de øvrige variabler som ligger i hoveddatasettet bygget opp gjennom `import`-prosedyren (se kapittel 2.3.1). Se kapittel 2.8 for hvordan dette gjøres.

3.5 Endre navn på variabler

Variabelnavn kan endres fritt. Det er hensiktsmessig at variabler har forståelige og intuitive navn. Dette gjøres enkelt gjennom følgende kommando:

```
rename <variabelnavn_gml> <variabelnavn_ny>
```

Siden variabler i microdata.no er sterkt knyttet til tid, kan det være en god regel å inkludere tidspunkt (årstall) i navnet. Eksempel:

```
rename sivstand sivstand00
```

3.6 Lage labler

Tabellkjøringer og annen statistisk output blir mer forståelige om en knytter tekst til de ulike kategoriske verdiene for en variabel. I microdata.no kan man definere et sett med "value labels" som kan knyttes til alle variabler med samme type verdikoding:

```
define-labels <labelsettnavn> <verdi1> <label1> <verdi2>
<label2> …. <verdin> <labeln>

assign-labels <variabel> <labelsettnavn>
```

Først angis altså labler for de ulike verdiene, og i neste trinn knyttes settet med labler til nærmere spesifiserte variabler.

Eksempel på en kategorisk (alfanumerisk) bostedsvariabel på fylkesnivå (variabelen `fylke`). Settet med labler (navngitt som "fylkerstring") kan gjennom kommandoen `assign-labels` knyttes til så mange variabler en ønsker, gitt at de har samme type verdisett (en slipper altså å opprette samme settet med labler flere ganger):

```
define-labels fylkerstring '01' 'Østfold' '02' 'Akershus' '03'
'Oslo' '04' 'Hedmark' '05' 'Oppland' '06' 'Buskerud' '07'
'Vestfold' '08' 'Telemark' '09' 'Aust-Agder' '10' 'Vest-Agder'
'11' 'Rogaland' '12' 'Hordaland' '14' 'Sogn og Fjordane' '15'
'Møre og Romsdal' '16' 'Sør-Trøndelag' '17' 'Nord-Trøndelag'
'18' 'Nordland' '19' 'Troms' '20' 'Finnmark' '99' 'Uoppgitt'

assign-labels fylke fylkerstring
```

Om en variabel har numeriske verdier, droppes fnutter rundt verdiene i define-labels-kommandoen. Lablene som lenkes mot verdiene kan med fordel ha fnutter rundt seg uansett. Da står en fritt til å velge tegn og symboler, inkludert mellomrom.

NB! Kommategn må ikke brukes inni labler! Dette vil gi feilmelding ved kjøring (kommategnet er reservert til å brukes kun i forbindelse med opsjoner).

3.7 Endre verdiformat fra alfanumerisk (tekst) til numerisk

Mange av variablene tilgjengelig i microdata.no har alfanumerisk verdiformat, dvs. at de betraktes som tekst. Gjennom kommandoen `destring` kan en konventere slike om til numeriske variabler. Som standard vil variabelen overskrives med det nye formatet:

```
destring <variabel/variabelliste> [, <opsjoner>]
```

Om enkelte verdier består av ikke-numeriske karakterer, f.eks. ",", ".", "-", "kr", vil konverteringen ikke gjennomføres med mindre en bruker opsjonene `force` eller `ignore()`.

Opsjonen `force` tvinger systemet til å konvertere til numerisk uansett, men verdier bestående av ikke-numeriske karakterer vil få manglende verdi: `sysmiss`

Gjennom opsjonen `ignore()` kan en definere hvilke karakterer/tegn som skal ignoreres under konverteringen. Dette kan være aktuelt dersom verdier inneholder bindestreker, kommategn, tusenskilletegn eller annet. Eksempelet under vil ignorere punktum, komma og bindestrek i verdier der dette finnes for variabelen *var1*:

```
destring var1, ignore('.,-')
```

Alfanumeriske verdier som benytter komma som desimalskilletegn ('2,1', '10000,00' etc) kan konverteres direkte til numeriske verdier der en beholder desimalene. Den konverterte verdien vil da få punktum som desimaltegn. Dette gjøres gjennom opsjonen `dpcomma`.

3.8 Eksempel

```
// Kobler til databank
require no.ssb.fdb:1 as fdb1

// Henter variablene en trenger
create-dataset demografidata
import fdb1/BEFOLKNING_KJOENN as kjonn
import fdb1/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
import fdb1/INNTEKT_BRUTTOFORM 2000-01-01 as formue
import fdb1/BOSATTEFDT_BOSTED 2000-01-01 as bosted

// Beregner alder i 2000 ut i fra fødselsår
generate alder = 2000 - int( faarmnd / 100 )

// Lager en dummyvariabel som angir mann ut i fra kjønn
generate mann = 0
replace mann = 1 if kjonn == '1'

// Gruppere formue i 4 intervaller
generate formueint = 1
replace formueint = 2 if formue > 150000
replace formueint = 3 if formue > 250000
replace formueint = 4 if formue > 400000

// Angi formue i 1000 kr
generate formue1000 = formue / 1000

// Koder om fra kommune- til fylkesnivå
generate fylke = substr(bosted,1,2)

// Legger til verdilabler for å navngi fylker med navn (=> penere deskriptiv output)

define-labels fylkerstring '01' 'Østfold' '02' 'Akershus' '03' 'Oslo' '04' 'Hedmark' '05' 'Oppland' '06'
'Buskerud' '07' 'Vestfold' '08' 'Telemark' '09' 'Aust-Agder' '10' 'Vest-Agder' '11' 'Rogaland' '12'
```

```
'Hordaland' '14' 'Sogn og Fjordane' '15' 'Møre og Romsdal' '16' 'Sør-Trøndelag' '17' 'Nord-Trøndelag' '18' 'Nordland' '19' 'Troms' '20' 'Finnmark' '99' 'Uoppgitt'

assign-labels fylke fylkerstring

tabulate fylke
```

4. Hvordan gjøre seg kjent med variabler

I microdata.no kan en bruke ulike teknikker for å utforske variabler og datasett. Den enkleste er bruk av tabeller (enveis- eller krysstabeller) eller oppsummeringsstatistikk (for metriske variabler). En kan også visualisere gjennom histogrammer, søylediagrammer, kakediagrammer og anonymiserte plotdiagrammer (hexbinplot) på en oversiktlig måte.

Analysesystemet microdata.no har foreløpig følgende kommandoer tilgjengelig for produksjon av deskriptiv statistikk:

- Tabulate
- Summarize
- Boxplot
- Hexbin
- Piechart
- Histogram
- Barchart
- Sankey

Gjennom options kan en vise alternative fremstillinger av de samme fordelingene, og en kan utelate enheter fra tabellene/figurene gjennom IF-betingelser.

4.1 Tabulate - frekvenstabeller

Kommandoen `tabulate` brukes til å lage frekvenstabeller, og er den vanligste statistikk-kommandoen når en skal gjøre seg kjent med variabelinnholdet, samt når en skal lage deskriptiv statistikk.

Kommandoen kan brukes på alle kategoriske variabler. Disse vil ofte være alfanumeriske, men det er fullt mulig å lage frekvenstabeller for numeriske variabler også, gitt at antallet verdier ikke blir for omfattende.

Standardfremvisningen for tabeller generert gjennom `tabulate` er frekvenstall (antall enheter), og disse kan være enveis, toveis og flerdimensjonale. Som standard vises rekke- og kolonneverdienes verdi-labler for variabler som har dette, og missingverdier utelates fra tabellgrunnlaget.

Gjennom bruk av opsjoner kan en kontrollere presentasjonen og overstyre standardfremvisningen:

- Vise prosentandeler i stedet for frekvenstall
- Vise kolonne- og rekkeverdier uten verdi-labler
- Vise tabeller der missingkategorier tas med i tabellgrunnlaget
- Lage såkalte volumtabeller som viser oppsummerende verdier (gjennomsnitt, sum m.m.) for valgfrie variabler innen hver celle
- Gjennomføre en chi-test (tester for avvik fra en helt tilfeldig bivariat fordeling) gjennom en `chi2`-opsjon

Som for de fleste kommandoer i microdata.no kan en i forbindelse med `tabulate` benytte filtre i form av IF-betingelser for å styre hvilke enheter som skal inngå i den aktuelle tabellen. En trenger altså ikke nødvendigvis trimme utvalget i datasettet i forkant av statistiske kjøringer.

Syntax:

```
tabulate <variabel/variabelliste> [, <opsjoner>]
```

Tips:

Tabeller generert gjennom kommandoen `tabulate` kan eksporteres over i andre programmer som Excel, Word, Google Sheets m.m. Dette gjøres gjennom å klikke på et “copy”-ikon som dukker opp når en holder musepekeren over tabellen. Deretter bruker en tastekombinasjonen <*Ctrl*> + <*C*> og limer så inn i ønsket dokument. Dette kan også gjøres på annen type output, som f.eks. `regress`.

4.1.1 Enveis frekvenstabeller

Eksempel på frekvenstabell for variablene “bostedsfylke per 2000-01-01” og kjønn:

```
»tabulate fylke00
```

fylke00	
Østfold	144599
Akershus	292087
Oslo	310657
Hedmark	108743
Oppland	109887
Buskerud	143930
Vestfold	124566
Telemark	95748
Aust-Agder	59546
Vest-Agder	90565
Rogaland	223820
Hordaland	258569
Sogn og Fjordane	66239
Møre og Romsdal	146564
Sør-Trøndelag	156169
Nord-Trøndelag	74196
Nordland	140258
Troms	91962
Finnmark	45540
Uoppgitt	6
Sum	*2683675*

```
»tabulate kjonn
```

kjonn	
Mann	1424545
Kvinne	1274038
Sum	*2698574*

4.1.2 Flerdimensjonale frekvenstabeller

Flerdimensjonale frekvenstabeller er tabeller med fordelinger over 2 eller flere variabler. Disse inneholder såkalte celleverdier, rekke- og kolonnesummer, og totalverdi (nederst til høyre).

Eksempel på frekvenstabell med fordeling over kjønn og registerstatus:

```
»tabulate kjonn regstat
```

		regstat				
		bosatt	utvandret	død	uregistrert person	*Sum*
kjonn	Mann	1414045	3157	7	17	*1417207*
	Kvinne	1266295	2538	8	6	*1268840*
	Sum	*2680342*	*5695*	*11*	*15*	*2686050*

Eksempel på tredimensjonal frekvenstabell med fordeling over kjønn, registerstatus og bostedsfylke (merk at hele tabellen ikke vises i illustrasjonen under):

```
»tabulate kjonn regstat fylke00
```

			regstat				
			bosatt	utvandret	uregistrert person	død	*Sum*
kjonn	Mann	fylke00 Østfold	76494	19	--	--	*76517*
		Akershus	151246	59	--	--	*151300*
		Oslo	159035	209	11	--	*159249*
		Hedmark	57504	7	--	--	*57517*
		Oppland	58036	6	--	--	*58050*
		Buskerud	75605	25	--	--	*75625*
		Vestfold	65492	16	--	--	*65513*
		Telemark	50891	6	--	--	*50890*
		Aust-Agder	31863	14	--	--	*31874*
		Vest-Agder	48504	15	--	--	*48514*
		Rogaland	119190	25	--	--	*119216*
		Hordaland	136460	42	--	--	*136512*
		Sogn og Fjordane	35231	10	--	--	*35231*
		Møre og Romsdal	78937	18	--	--	*78947*
		Sør-Trøndelag	82371	16	--	--	*82394*
		Nord-Trøndelag	39680	7	--	--	*39689*
		Nordland	74471	19	--	--	*74488*
		Troms	48745	20	--	--	*48769*
		Finnmark	24157	11	--	--	*24173*
		Østfold	67909	16	--	--	*67927*
		Akershus	140466	35	--	--	*140494*

4.1.3 Frekvenstabeller og prosentuering

Kommandoen `tabulate` kan, i tillegg til å vise frekvenser (antall enheter) for kombinasjoner av verdier for de aktuelle variabler, brukes til å vise prosentandeler. Dette styres gjennom opsjoner:

- `rowpct` rekkeprosent (andel av rekketotalen)

- `colpct` kolonneprosent (andel av kolonnetotalen)
- `cellpct` celleprosent (andel av totalen)
- `freq` frekvensverdi (standardvisning, brukes kun i kombinasjon med prosentuering)

Flere opsjoner kan kombineres i samme kommando, og en kan f.eks. vise både frekvenser og rekkeprosenter i en og samme tabell (se eksempel nr 4 nedenfor).

Eksempler:

```
»tabulate sivstand00 sivstand05, rowpct
```

sivstand00 \ sivstand05	Ugift	Gift	Skilt	Separert	0 ukjent	Enke/enkemann	Registrert partner	Separert partner	Skilt partner	Gjenlevende partner	Sum
0 ukjent	64.71	29.41	9.8	9.8	--	--	--	--	--	--	100
Ugift	91.72	7.73	0.17	0.29	0	0.02	0.06	0	0	0	100
Gift	0	89.95	3.38	2.49	0	4.17	0	0	--	--	100
Enke/enkemann	--	0.93	0.02	0.03	0	99.03	--	--	--	--	100
Skilt	--	10.8	88.49	0.56	0	0.11	0.05	0	--	--	100
Separert	0.01	16.36	52.87	28.52	0.01	2.18	0.04	0.01	--	--	100
Registrert partner	--	0.53	--	--	--	--	76.66	7.96	13.53	1.92	100
Separert partner	--	--	--	--	--	--	18.18	20.45	56.82	--	100
Skilt partner	--	5.68	--	--	--	--	13.64	7.95	64.77	--	100
Gjenlevende partner	--	--	--	--	--	--	--	--	--	123.08	100
Sum	*45.61*	*38.91*	*7.6*	*1.52*	*0*	*6.29*	*0.06*	*0.01*	*0.01*	*0*	--

```
»tabulate sivstand00 sivstand05, colpct
```

sivstand00 \ sivstand05	Ugift	Gift	Skilt	Separert	0 ukjent	Enke/enkemann	Registrert partner	Separert partner	Skilt partner	Gjenlevende partner	Sum
0 ukjent	0	0	0	0.01	--	--	--	--	--	--	0
Ugift	100	9.88	1.13	9.59	72.73	0.16	46.35	32.79	14.55	10	49.73
Gift	0	87.7	16.88	62.31	31.82	25.17	1.22	2.05	--	--	37.93
Enke/enkemann	--	0.11	0.01	0.08	22.73	74.09	--	--	--	--	4.7
Skilt	--	1.73	72.46	2.29	22.73	0.11	4.73	4.1	--	--	6.22
Separert	0	0.58	9.52	25.71	22.73	0.47	0.91	2.46	--	--	1.37
Registrert partner	--	0	--	--	--	--	45.6	49.18	53.97	58	0.04
Separert partner	--	--	--	--	--	--	0.63	7.38	13.23	--	0
Skilt partner	--	0	--	--	--	--	0.47	2.87	15.08	--	0
Gjenlevende partner	--	--	--	--	--	--	--	--	--	32	0
Sum	*100*	*100*	*100*	*100*	*100*	*100*	*100*	*100*	*100*	*100*	--

```
»tabulate sivstand00 sivstand05, cellpct
```

sivstand00 \ sivstand05	Ugift	Gift	Skilt	Separert	0 ukjent	Enke/enkemann	Registrert partner	Separert partner	Skilt partner	Gjenlevende partner	*Sum*
0 ukjent	0	0	0	0	--	--	--	--	--	--	*0*
Ugift	45.61	3.84	0.09	0.15	0	0.01	0.03	0	0	0	*49.73*
Gift	0	34.12	1.28	0.95	0	1.58	0	0	--	--	*37.93*
Enke/enkemann	--	0.04	0	0	0	4.66	--	--	--	--	*4.7*
Skilt	--	0.67	5.51	0.03	0	0.01	0	0	--	--	*6.22*
Separert	0	0.22	0.72	0.39	0	0.03	0	0	--	--	*1.37*
Registrert partner	--	0	--	--	--	--	0.03	0	0	0	*0.04*
Separert partner	--	--	--	--	--	--	0	0	0	--	*0*
Skilt partner	--	0	--	--	--	--	0	0	0	--	*0*
Gjenlevende partner	--	--	--	--	--	--	--	--	--	0	*0*
Sum	*45.61*	*38.91*	*7.6*	*1.52*	*0*	*6.29*	*0.06*	*0.01*	*0.01*	*0*	--

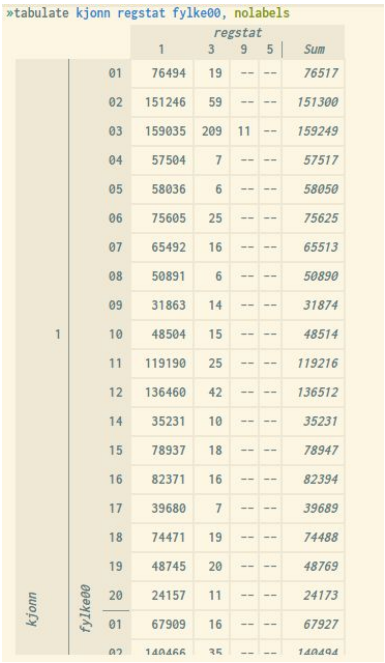
```
»tabulate sivstand00 sivstand05, rowpct freq
```

sivstand00 \ sivstand05	Ugift	Gift	Skilt	Separert	0 ukjent	Enke/enkemann	Registrert partner	Separert partner	Skilt partner	Gjenlevende partner	*Sum*
0 ukjent	33 64.71	15 29.41	5 9.8	5 9.8	--	--	--	--	--	--	*51* *100*
Ugift	1915459 91.72	161418 7.73	3612 0.17	6112 0.29	16 0	432 0.02	1175 0.06	80 0	55 0	5 0	*2088366* *100*
Gift	41 0	1432952 89.95	53856 3.38	39697 2.49	7 0	66463 4.17	31 0	5 0	--	--	*1593035* *100*
Enke/enkemann	--	1830 0.93	34 0.02	54 0.03	5 0	195650 99.03	--	--	--	--	*197576* *100*
Skilt	--	28222 10.8	231185 88.49	1457 0.56	5 0	280 0.11	120 0.05	10 0	--	--	*261270* *100*
Separert	5 0.01	9396 16.36	30361 52.87	16379 28.52	5 0.01	1250 2.18	23 0.04	6 0.01	--	--	*57425* *100*
Registrert partner	--	8 0.53	--	--	--	--	1156 76.66	120 7.96	204 13.53	29 1.92	*1508* *100*
Separert partner	--	--	--	--	--	--	16 18.18	18 20.45	50 56.82	--	*88* *100*
Skilt partner	--	5 5.68	--	--	--	--	12 13.64	7 7.95	57 64.77	--	*88* *100*
Gjenlevende partner	--	--	--	--	--	--	--	--	--	16 123.08	*13* *100*
Sum	*1915545* *45.61*	*1633855* *38.91*	*319053* *7.6*	*63707* *1.52*	*22* *0*	*264057* *6.29*	*2535* *0.06*	*244* *0.01*	*378* *0.01*	*50* *0*	*4199434* --

4.1.4 Frekvenstabeller og kategori-labler

Om en kun ønsker å vise kolonne- og rekkeverdiene, og ikke verdi-lablene, kan opsjonen `nolabels` benyttes.

Eksempel:



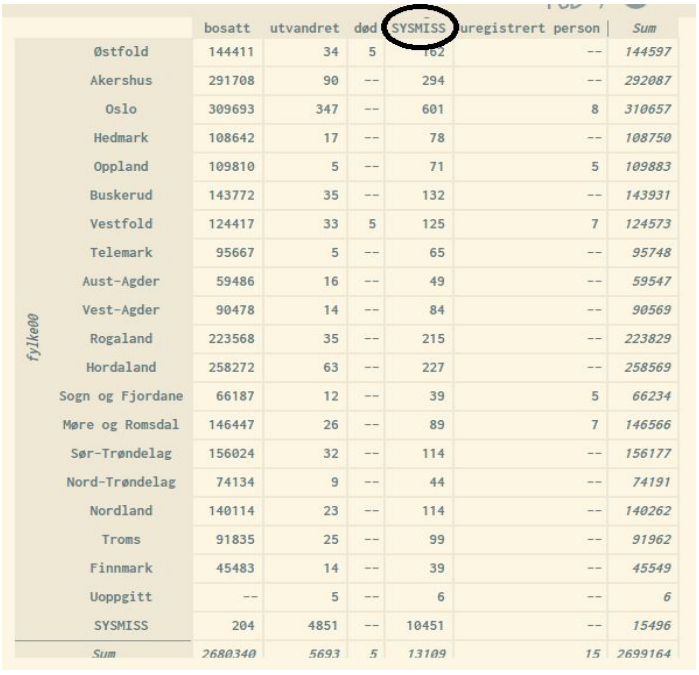
```
»tabulate kjonn regstat fylke00, nolabels
```

kjonn	fylke00	regstat 1	3	9	5	Sum
1	01	76494	19	--	--	76517
	02	151246	59	--	--	151300
	03	159035	209	11	--	159249
	04	57504	7	--	--	57517
	05	58036	6	--	--	58050
	06	75605	25	--	--	75625
	07	65492	16	--	--	65513
	08	50891	6	--	--	50890
	09	31863	14	--	--	31874
	10	48504	15	--	--	48514
	11	119190	25	--	--	119216
	12	136460	42	--	--	136512
	14	35231	10	--	--	35231
	15	78937	18	--	--	78947
	16	82371	16	--	--	82394
	17	39680	7	--	--	39689
	18	74471	19	--	--	74488
	19	48745	20	--	--	48769
	20	24157	11	--	--	24173
	01	67909	16	--	--	67927
	02	140466	35	--	--	140494

4.1.5 Frekvenstabeller og missingverdier

Som standard brukes ikke missingverdier i tabellgrunnlaget når kommandoen `tabulate` kjøres. Disse holdes altså utenfor tallberegningene med mindre en benytter opsjonen `missing`.

Eksempel:



fylke00	bosatt	utvandret	død	SYSMISS	uregistrert person	Sum
Østfold	144411	34	5	162	--	144597
Akershus	291708	90	--	294	--	292087
Oslo	309693	347	--	601	8	310657
Hedmark	108642	17	--	78	--	108750
Oppland	109810	5	--	71	5	109883
Buskerud	143772	35	--	132	--	143931
Vestfold	124417	33	5	125	7	124573
Telemark	95667	5	--	65	--	95748
Aust-Agder	59486	16	--	49	--	59547
Vest-Agder	90478	14	--	84	--	90569
Rogaland	223568	35	--	215	--	223829
Hordaland	258272	63	--	227	--	258569
Sogn og Fjordane	66187	12	--	39	5	66234
Møre og Romsdal	146447	26	--	89	7	146566
Sør-Trøndelag	156024	32	--	114	--	156177
Nord-Trøndelag	74134	9	--	44	--	74191
Nordland	140114	23	--	114	--	140262
Troms	91835	25	--	99	--	91962
Finnmark	45483	14	--	39	--	45549
Uoppgitt	--	5	--	6	--	6
SYSMISS	204	4851	--	10451	--	15496
Sum	2680340	5693	5	13109	15	2699164

4.1.6 Frekvenstabeller og filtrering

Om en ønsker å lage tabeller for delpopulasjoner, kan en filtrere gjennom bruk av IF-betingelser. En trenger altså ikke justere på datasettet i forkant.

Eksempler:

```
»tabulate fylke00 regstat if regstat == '1'
```

		regstat	
		bosatt	*Sum*
fylke00	Østfold	144408	*144408*
	Akershus	291704	*291704*
	Oslo	309690	*309690*
	Hedmark	108645	*108645*
	Oppland	109805	*109805*
	Buskerud	143766	*143766*
	Vestfold	124413	*124413*
	Telemark	95670	*95670*
	Aust-Agder	59486	*59486*
	Vest-Agder	90473	*90473*
	Rogaland	223572	*223572*
	Hordaland	258279	*258279*
	Sogn og Fjordane	66183	*66183*
	Møre og Romsdal	146440	*146440*
	Sør-Trøndelag	156028	*156028*
	Nord-Trøndelag	74128	*74128*
	Nordland	140113	*140113*
	Troms	91842	*91842*
	Finnmark	45483	*45483*
	Sum	*2680141*	*2680141*

```
»tabulate fylke00 regstat if alder > 30
```

		regstat				
		bosatt	utvandret	død	uregistrert person	*Sum*
fylke00	Østfold	100971	11	5	--	*100993*
	Akershus	209788	67	--	--	*209858*
	Oslo	211918	218	--	13	*212144*
	Hedmark	78091	6	--	--	*78100*
	Oppland	78183	8	--	--	*78188*
	Buskerud	101441	16	--	--	*101464*
	Vestfold	87173	16	--	5	*87185*
	Telemark	66740	--	--	--	*66741*
	Aust-Agder	40247	7	--	--	*40258*
	Vest-Agder	60803	12	--	--	*60815*
	Rogaland	149250	32	--	--	*149279*
	Hordaland	177392	41	--	--	*177436*
	Sogn og Fjordane	45405	--	--	--	*45404*
	Møre og Romsdal	100535	18	--	7	*100551*
	Sør-Trøndelag	109421	26	--	--	*109448*
	Nord-Trøndelag	52323	5	--	--	*52327*
	Nordland	97681	18	--	--	*97691*
	Troms	63562	18	--	--	*63582*
	Finnmark	31089	11	--	--	*31095*
	Sum	*1862018*	*550*	*9*	*12*	*1862583*

4.1.7 Volumtabeller

Kommandoen `tabulate` kan i tillegg til å vise frekvenser og prosentandeler brukes til å lage såkalte volumtabeller. I hver celle vil det da i stedet vises oppsummerende statistikk for en valgfri variabel. En kan fritt velge hvilken variabel som skal oppsummeres gjennom følgende opsjon:

```
summarize( <variabel> )
```

Som standard vises gjennomsnittsverdi som statistikk, men en kan overstyre dette ved å benytte følgende tilleggsopsjoner i `tabulate`-kommandoen:

- `mean` Gjennomsnitt (standard)
- `max` Maxverdi
- `min` Minimumsverdi
- `sum` Sum
- `std` Standardavvik
- `p25` 25%-kvartil
- `p50` 50%-kvartil
- `p75` 75%-kvartil

Eksempel på volumtabell der en viser gjennomsnittsformue fordelt på bostedsfylke og kjønn:

```
»tabulate fylke00 kjonn, summarize(formue)
```

		kjonn		
		Mann	Kvinne	*Sum*
fylke00	Østfold	420698.87	192280.33	*313535.11*
	Akershus	482632.84	258851.53	*374659.58*
	Oslo	407740.27	253787.41	*332438.7*
	Hedmark	448040.46	207501.65	*335125.52*
	Oppland	461985.03	203399.93	*340732.13*
	Buskerud	451406.11	221951.55	*342941.33*
	Vestfold	440535.63	210095.86	*331709.72*
	Telemark	390275.19	181083.63	*292646.24*
	Aust-Agder	396850.1	183568.96	*298338.32*
	Vest-Agder	432110.14	192591.27	*321157.44*
	Rogaland	423818.48	185711.51	*312884.07*
	Hordaland	383759.34	189964.32	*292278.38*
	Sogn og Fjordane	486762.24	203607.58	*354812.34*
	Møre og Romsdal	435580.27	189451.31	*322286.72*
	Sør-Trøndelag	370140.51	180124.65	*280522.87*
	Nord-Trøndelag	399271.57	162695.47	*289981.45*
	Nordland	350952.27	162837.57	*262965.02*
	Troms	335615.35	170122.5	*257822.14*
	Finnmark	295861.55	165703.72	*234849.97*
	Sum	*416819.53*	*205087.38*	*316849.45*

4.2 Summarize og boxplot - statistikk for metriske variabler

Kommandoene `summarize` og `boxplot` brukes til å vise oppsummerende statistikk for metriske/kontinuerlige variabler. I likhet med andre statistikker i microdata.no, kan en lage statistikk også for delpopulasjoner via IF-betingelser (trenger ikke justere på datasettet i forkant).

Nedenfor vises eksempler for variablene inntekt og formue målt per 2000-01-01. Kommandoen `boxplot` viser en grafisk fremstilling gjennom et standard boxplot med boks for de to midterste kvartilene, gjennomsnitt samt minimums- og maksimumsverdi:

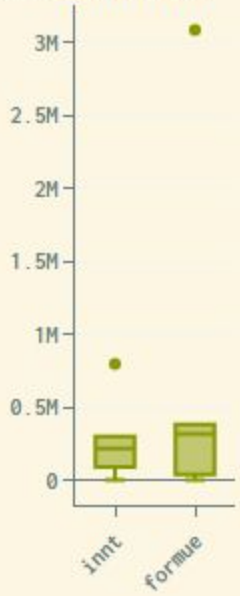
```
»summarize innt formue
```

Variabel	mean	std	count	min	max	25%	50%	75%
innt	216734.7462	157690.3886	2699162	1265	794619	89175	216523	299527
formue	315841.0182	476449.5303	2591219	33	3084837	36844.5	169486	380881

```
»summarize formue if alder > 50
```

Variabel	mean	std	count	min	max	25%	50%	75%
formue	525248.1495	611764.7615	671330	305	3084837	149883.75	341365.5	641339.75

```
»boxplot innt formue
```

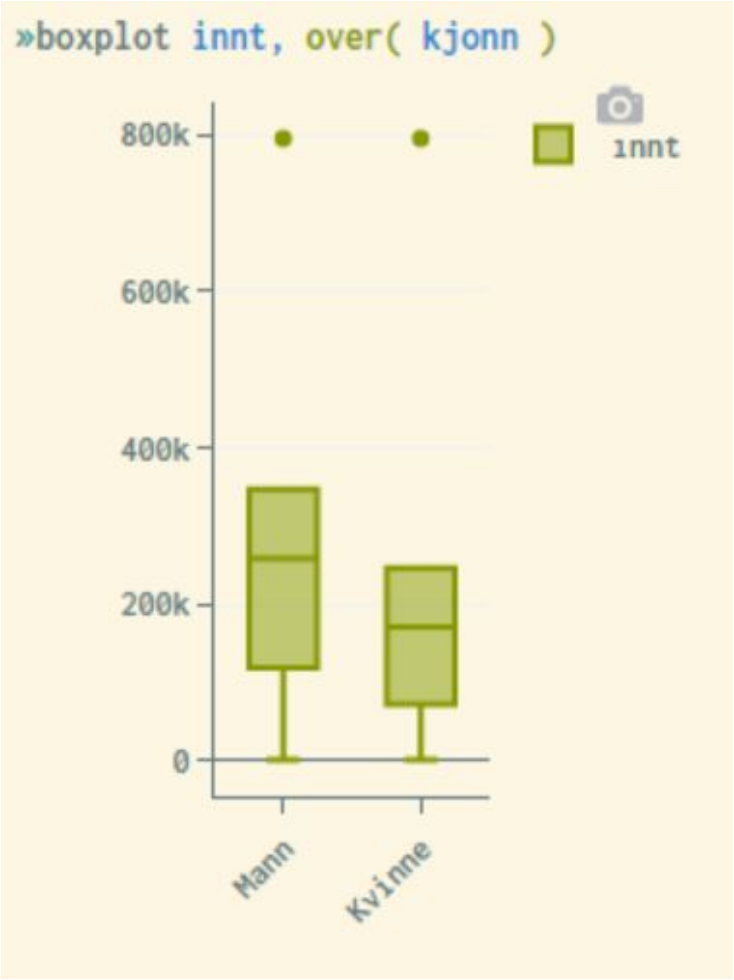


Om en holder musepekeren over de ulike områdene i boxplot-figuren, vil en kunne se hvilke verdier de ulike punktene representerer.

Kommandoen `boxplot` gir mulighet til å vise separate tall for gitte kategorier representert ved en annen kategorisk variabel:

```
boxplot <variabel1>, over ( variabel2 )
```

Eksempel på boxplot for inntekt per 2000-01-01 fordelt på kjønn:

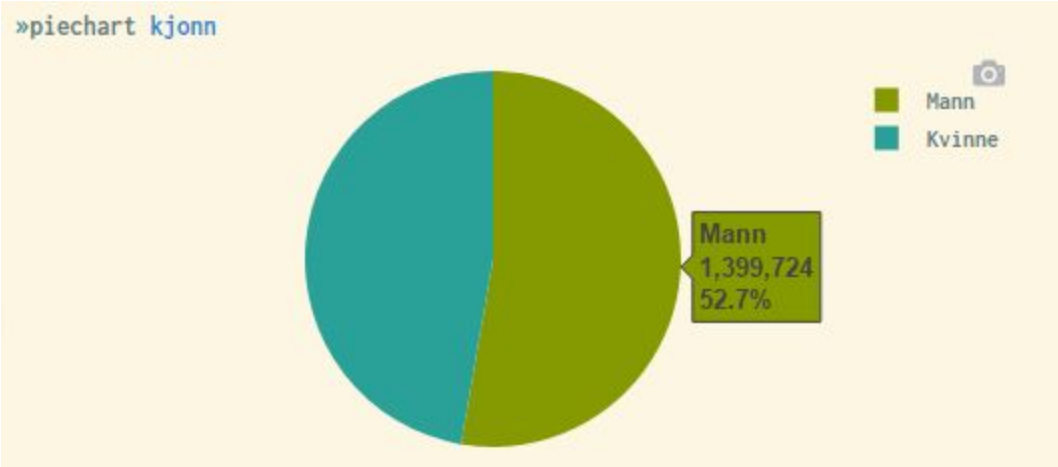


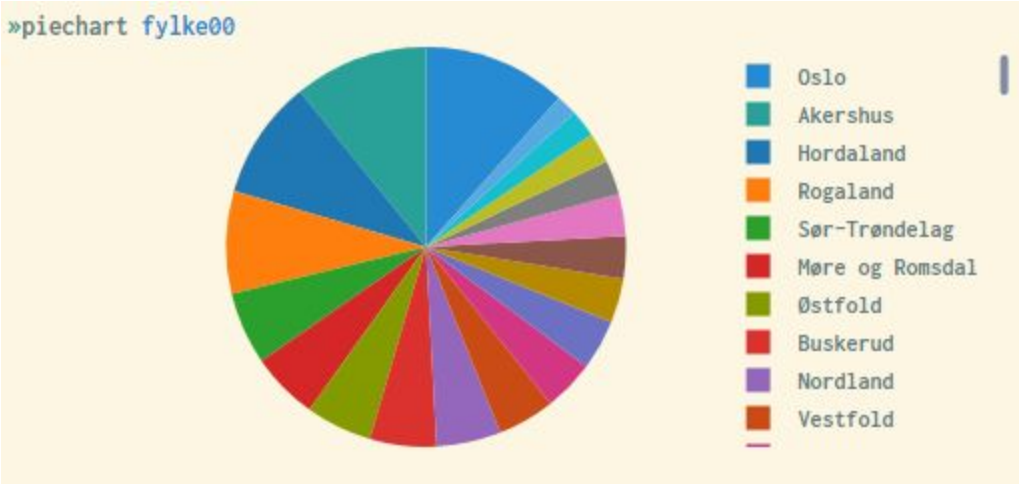
4.3 Piechart - kakediagram

Gjennom følgende kommando kan en lage kakediagram for kategoriske variabler:

```
piechart <variabel>
```

Om en holder musepekeren over de ulike feltene i figuren, vil en kunne se tilhørende verdier.





4.4 Histogram - grafisk fremstilling av frekvensfordelinger

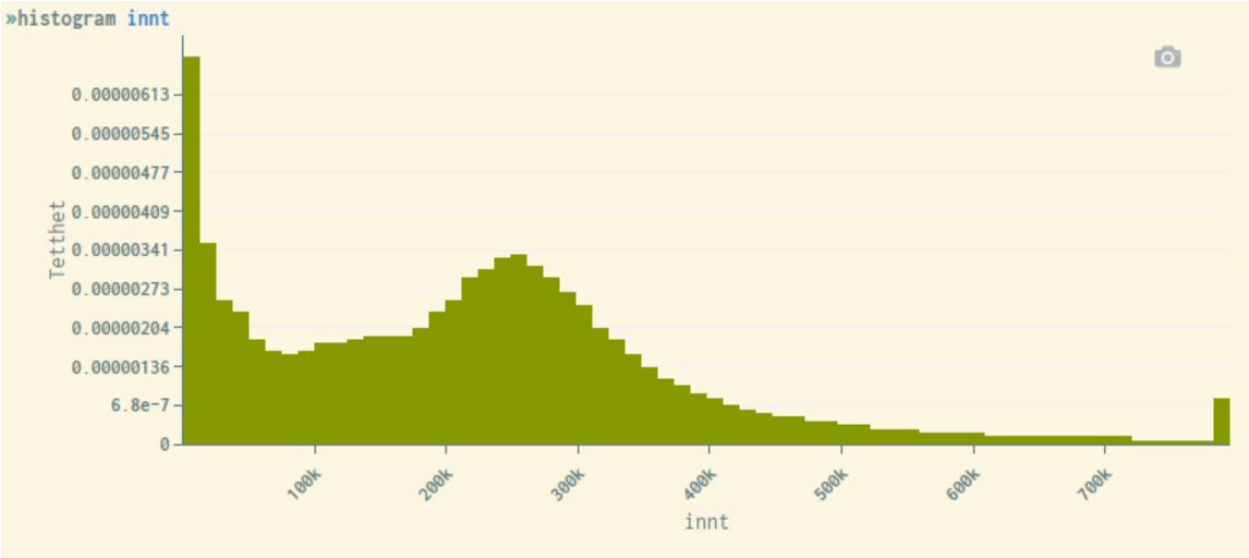
Histogrammer er grafiske fremstillinger av univariate fordelinger for kontinuerlige variabler (f.eks. inntekt). Hver søyle representerer frekvensverdien for et gitt forhåndsbestemt intervall for den aktuell variabelen. Gjennom opsjonene `bin()` og `width()` kan en overstyre dette og selv bestemme hhv. antall søyler og søyleintervallenes bredde. Nedenfor illustreres dette gjennom eksempler.

Standardfremvisningen viser tetthet som frekvensverdi. Også dette kan overstyres gjennom opsjoner, slik at måleenheten på y-aksen i stedet viser faktisk frekvens (antall), andel eller prosentandel. Følgende opsjoner kan benyttes til dette: `freq, fraction, percent`

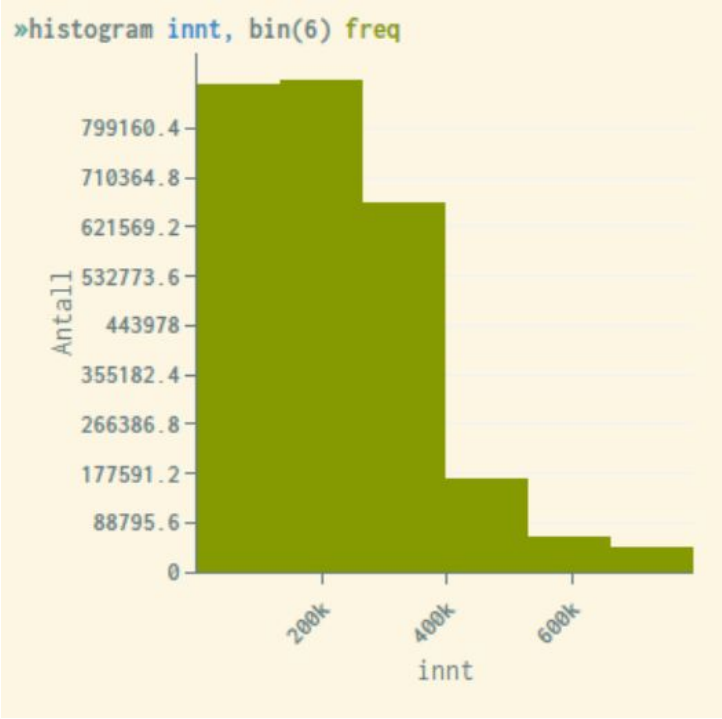
Personer med svært høye eller svært lave inntekter kan lett identifiseres om verdi-intervallet blir for smalt, noe som er problematisk med tanke på personvernet. Derfor foretar systemet en topp-/bunnkoding der de 1% høyeste og 1% laveste verdiene erstattes av grenseverdien til hhv. den siste og første prosentilen. Derfor vil alltid første og siste søyle være mye høyere enn nabosøylene, som illustrert i eksemplene nedenfor. Denne topp-/bunnkodingen omtales i detalj i Vedlegg C.

Om en holder musepekeren over de ulike søylene i figuren, vil en få opp intervallet samt frekvensverdi for den aktuelle søylen.

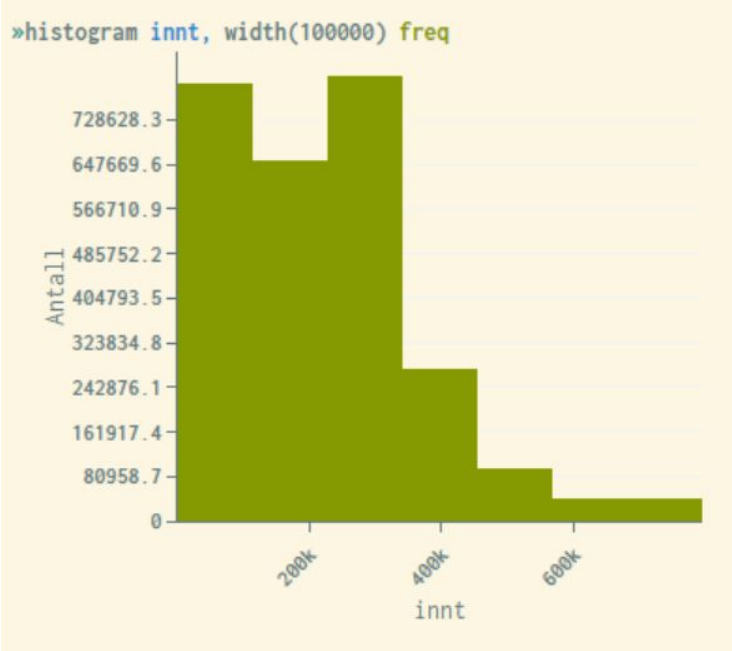
Eksempel:



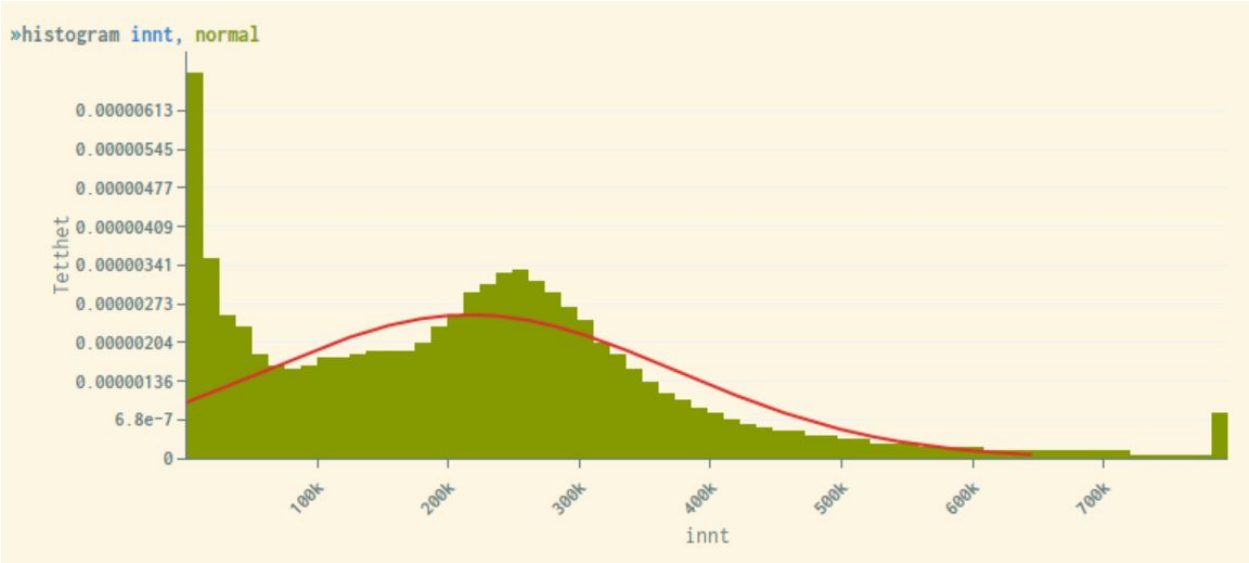
Histogram over inntekt fordelt på 6 søyler og frekvenstall på y-aksen (hver søyle har samme intervallbredde for inntekt):



Histogram over inntekt der hver søyle får intervallbredde for inntekt lik 100’000:

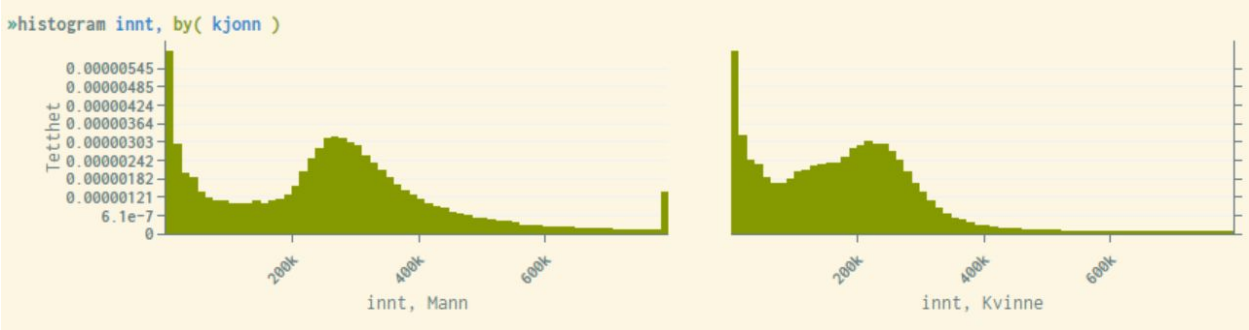


Gjennom opsjonen `normal`, kan en legge en normalfordelt kurve over søylene i figuren. Dette er til hjelp for å se på grad av avvik fra en normalfordeling:



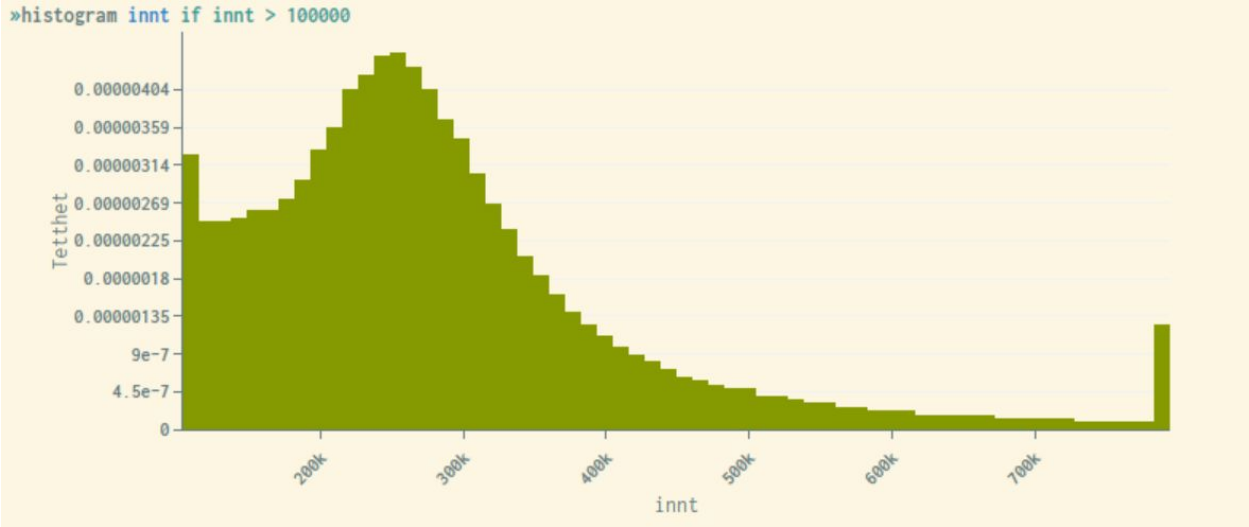
Histogrammer kan vises over fordelinger for en annen variabel som må være kategorisk, f.eks. Kjønn. Dette gjøres gjennom opsjonen `by( <variabel> )`.

Eksempel:



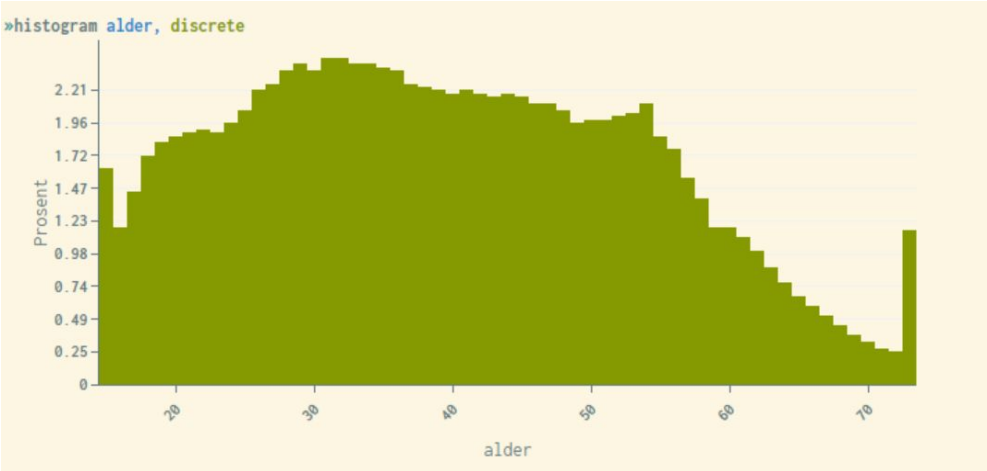
Som for andre statistiske fremstillinger i microdata.no, kan en gjøre en filtrering gjennom IF-betingelser, der en kun viser histogram for en delpopulasjon.

Eksempel der en viser histogram kun for personene med inntekt over 100 000:



Som nevnt vil histogram som standard dele inn i et forhåndsbestemt antall søyler/intervaller. Gjennom opsjonen `discrete` kan en overstyre dette og vise en søyle for hver enkelt verdi. Dette er ikke hensiktsmessig for metriske variabler av økonomisk art (blir svært mange søyler), men for kontinuerlige variabler med et begrenset antall verdier kan denne fremstillingen være nyttig. Eksempler på variabler kan være “alder”, prosentandeler, eller beløp som er ferdig avrundet (til nærmeste 10 000 eller 100 000).

Eksempel på bruk av opsjonen `discrete` for variabelen “alder” (en ser også her at systemet sørger for at første og siste søyle er mye høyere enn nabosøyler pga. topp-/bunn-kodingen, siden personer med svært høy/lav alder er relativt lette å identifisere):



4.5 Barchart - søylediagram

Kommandoen `barchart` brukes til å lage vanlige søyledigrammer. En angir aktuell variabel/variabler samt statistikken som søylene skal måle. Gjennom opsjonen `over()`, kan en fordele søylene over en eller flere andre variabler som må være kategoriske (f.eks. kjønn).

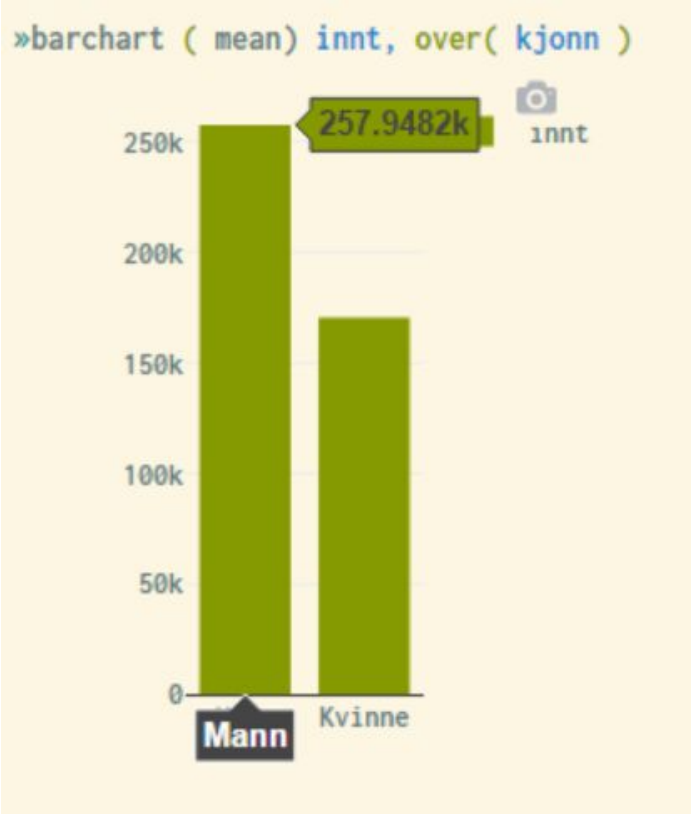
Som for andre grafiske fremstillinger i microdata.no, kan en gjennom å holde musepekeren over de ulike felt i figuren få opp tilhørende verdier.

Syntax:

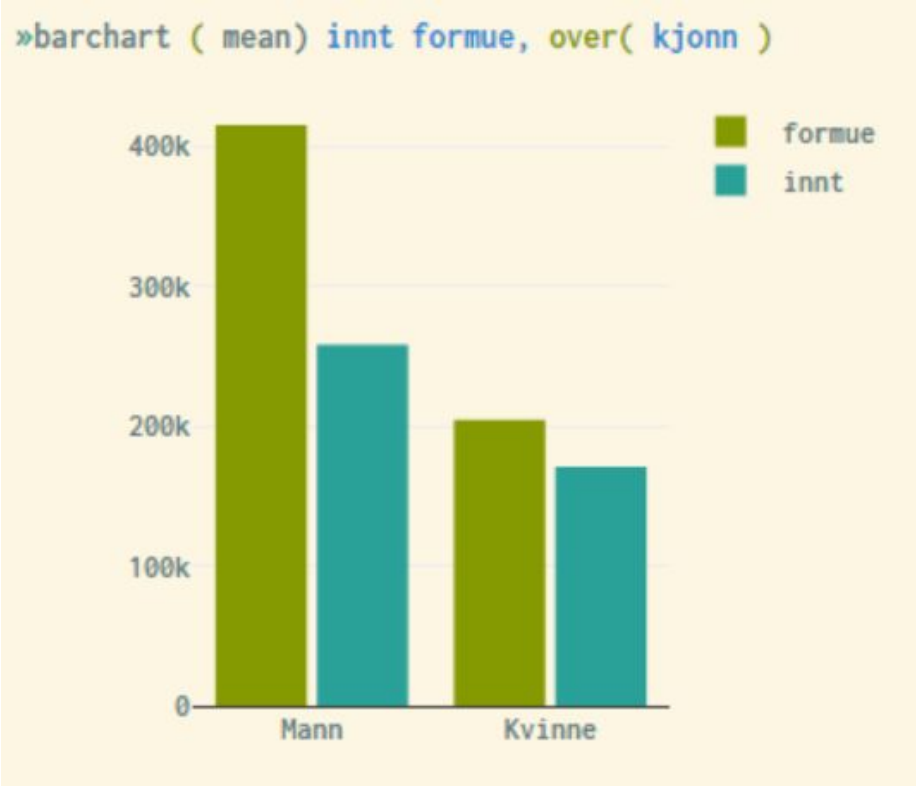
```
barchart ( <statistikkmål>) <variabelliste>[, over( <variabelliste> )]
```

For å lage stablede søylediagrammer, kan en bruke opsjonen `stack`.

Eksempel på søylediagram over gjennomsnittsinntekt fordelt på kjønn:



Eksempel på søylediagram over gjennomsnittsinntekt og gjennomsnittsformue fordelt på kjønn:



4.6 Hexbin - anonymiserende plotdiagram

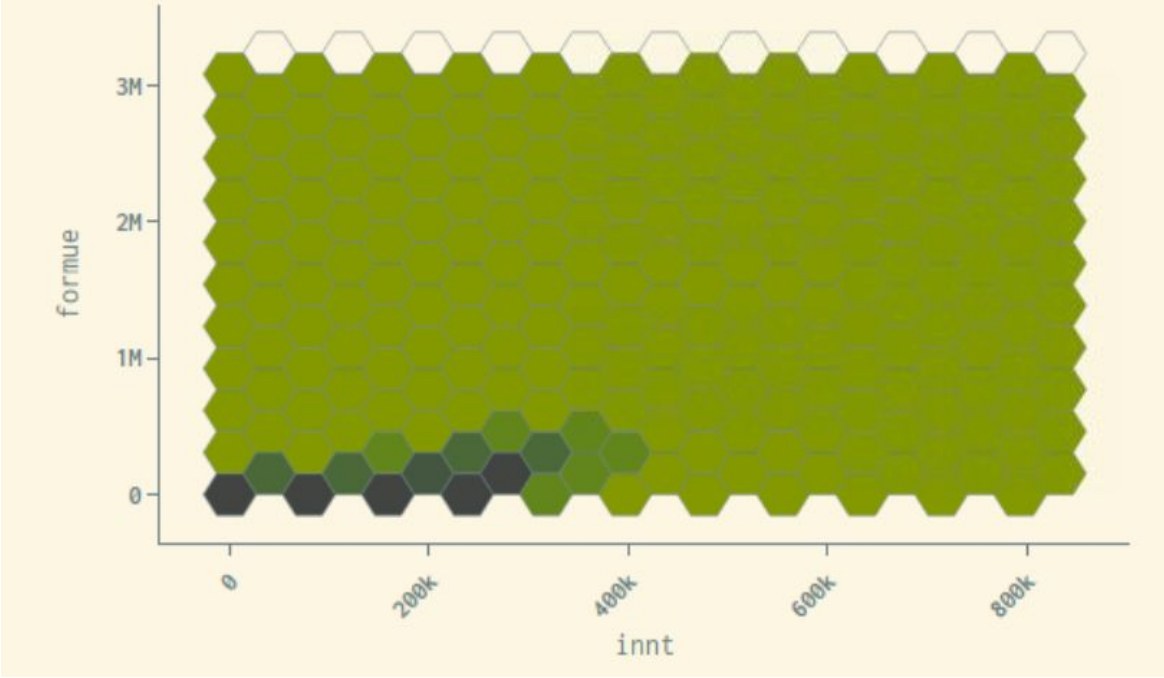
Hexbin-diagram er i praksis et anonymisert plotdigram der det to-dimensjonale arealet er delt inn i et gitt antall hexagonfelt. Fargen på disse hexagonene representerer tettheten av observasjoner/punkter i det aktuelle arealet. Jo mørkere farge, dess flere punkter befinner seg der. En får da en grafisk oversikt over fordelingen av enheter (individer) mellom to kontinuerlige variabler. Hexbindiagrammer egner seg ikke for kategoriske variabler.

En kan også her holde musepekeren over de ulike felt i figuren - da får en opp de aktuelle underliggende verdier.

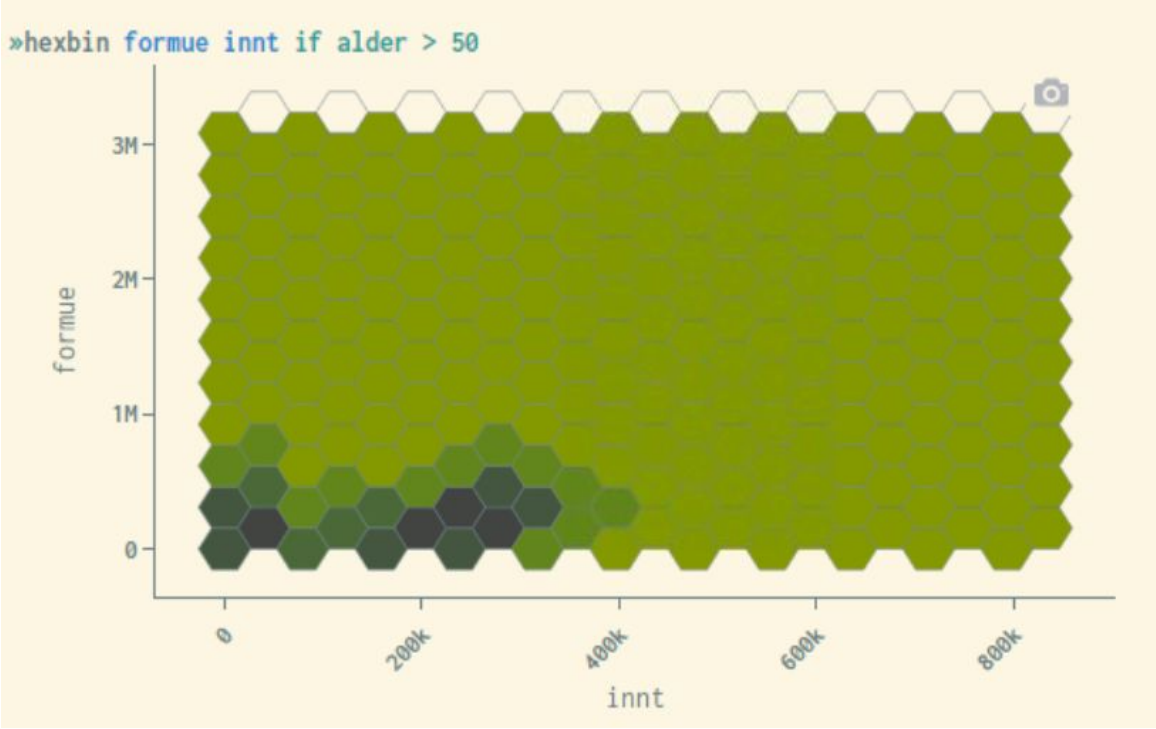
Også hexbin-diagram kan filtreres gjennom IF-opsjoner for å vise figur for delpopulasjoner. En kan dessuten justere på antallet hexagoner samt antallet intervaller gjennom opsjoner.

Eksempler:

```
»hexbin formue innt
```




```
»hexbin formue innt if alder > 50
```



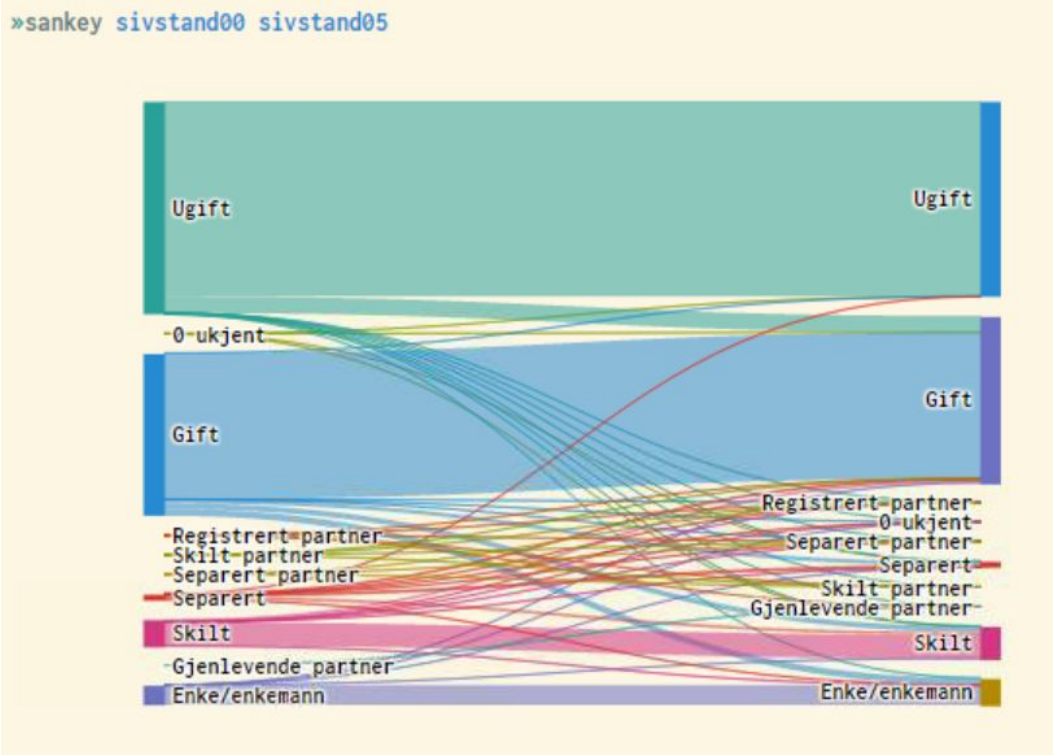
4.7 Sankey - overgangsdiagram

Sankey-diagram er en måte å visualisere overganger mellom statuser. I microdata.no kan dette brukes til å få oversikt over enheters (individers) bevegelser mellom to tidspunkter, enten for samme variabel eller for ulike variabler. En kan se på bevegelser mellom ulike typer tilstander (f.eks. arbeidssøkerstatus -> jobbstatus), eller endringer i fordelinger for samme variabel over tid (f.eks. bosted00 -> bosted05 eller sivilstand00 -> sivilstand05).

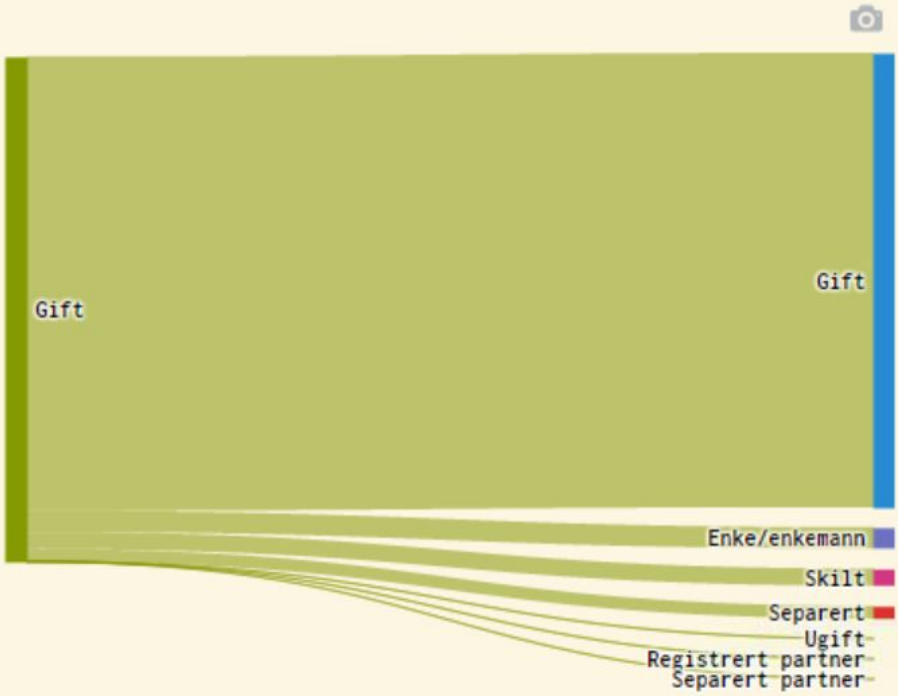
Overgangsvisualiseringen forutsetter at en benytter to kategoriske tverrsnittsvariabler. Antallet kategorier bør ikke være for stort, da diagrammet fort kan bli uleselig. Dette kan løses gjennom å kode om til færre kategorier eller ved å bruke et IF-filter som styrer hvilke overganger en vil se på.

Ved å holde musepekeren over et overgangsfelt vil en få frem antallet enheter som omfattes.

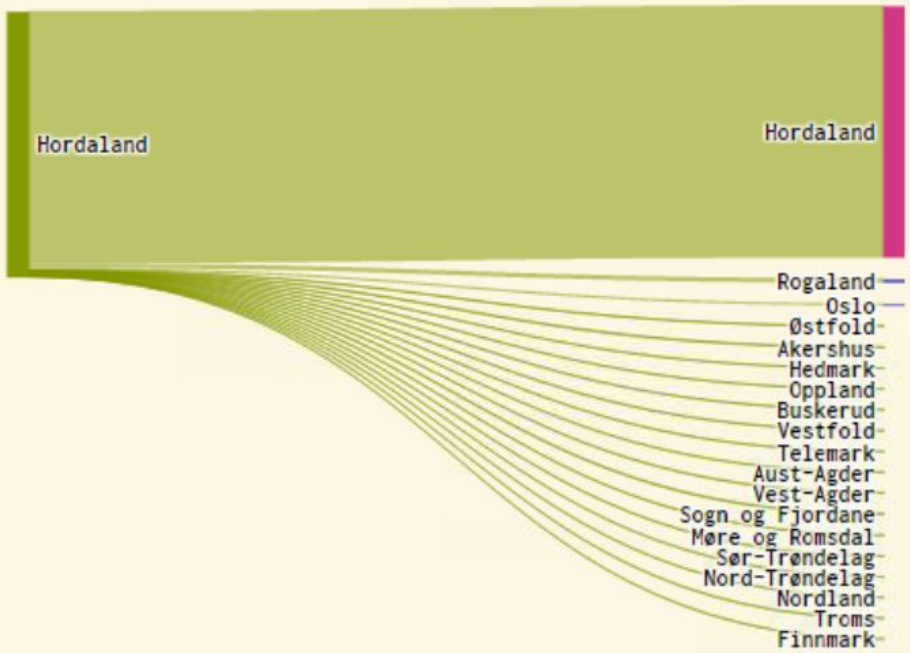
Eksempler:



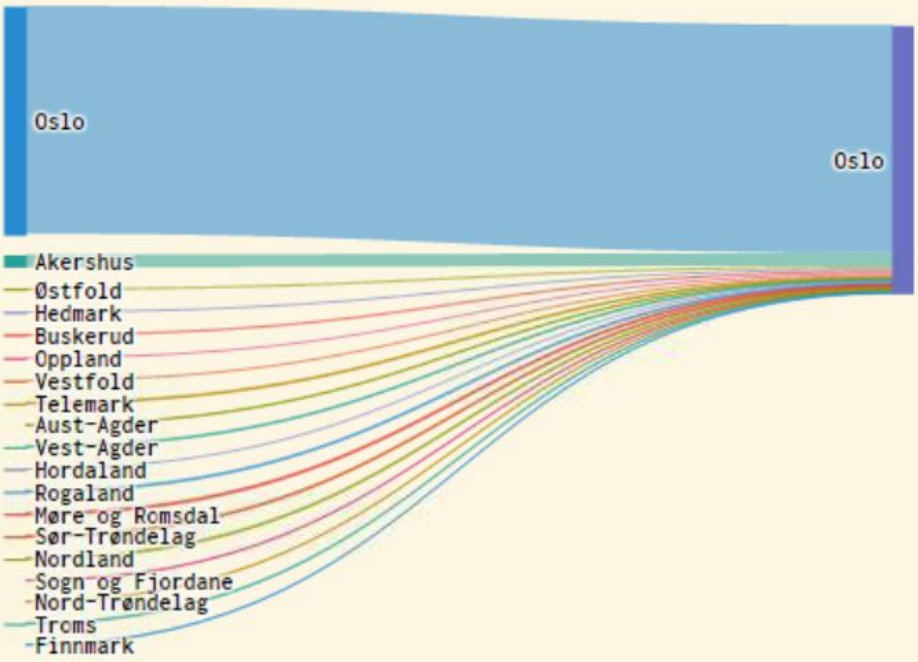
```
»sankey sivstand00 sivstand05 if sivstand00 == '2'
```



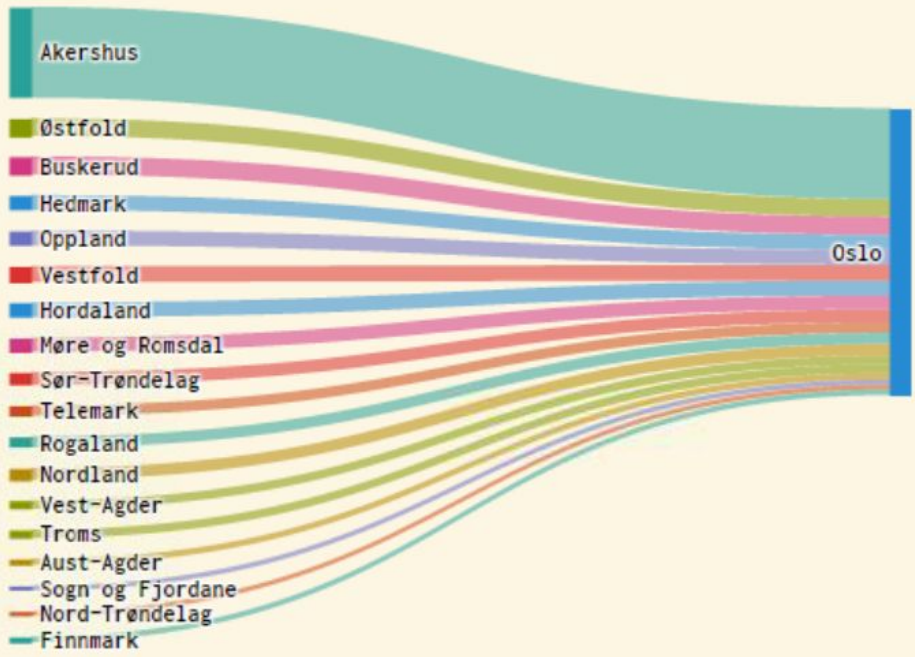

```
»sankey fylke00 fylke05 if fylke00 == '12'
```



```
»sankey fylke00 fylke05 if fylke05 == '03'
```




```
»sankey fylke00 fylke05 if fylke05 == '03' & fylke00 != '03'
```



4.8 Eksempler

Syntaxene nedenfor kan brukes til å gjenskape eksemplene på deskriptiv statistikk som presenteres i kapittel 4. Disse vil være tilgjengelig som ferdige kjørbare skript i microdata.no.

4.8.1 Tabulate

```
require no.ssb.fdb:1 as fdb1

create-dataset demografidata
Import fdb1/ INNTEKT_WYRKINNT 2000-01-01 as innt
import fdb1/INNTEKT_BRUTTOFORM 2000-01-01 as formue
import fdb1/BEFOLKNING_KJOENN as kjonn
import fdb1/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
import fdb1/SIVSTANDFDT_SIVSTAND 2000-01-01 as sivstand00
import fdb1/SIVSTANDFDT_SIVSTAND 2005-01-01 as sivstand05
import fdb1/BOSATTEFDT_BOSTED 2000-01-01 as bosted00
import fdb1/BEFOLKNING_REGSTAT 2000-01-01 as regstat

// Koder om fra kommune- til fylkesnivå
generate fylke00 = substr(bosted00 ,1, 2)


// Genererer deskriptiv statistikk

// Frekvenstabeller - enveis og toveis
// Den beste måten å gjøre seg kjent med diskrete variabler er gjennom frekvenstabeller. Disse viser antallet enheter for hver kategori, og gir dessuten overblikk over hvilke kategorier som benyttes for den aktuelle variabelen. En kan se på enkeltvariabler gjennom enveistabeller, men også kombinere to eller flere i en og samme tabell (krysstabeller). Dette gir innblikk i hvordan frekvensene fordeler seg, kontrollert for verdier på andre variabler

define-labels fylkerstring '01' 'Østfold' '02' 'Akershus' '03' 'Oslo' '04' 'Hedmark' '05' 'Oppland' '06' 'Buskerud' '07' 'Vestfold' '08' 'Telemark' '09' 'Aust-Agder' '10' 'Vest-Agder' '11' 'Rogaland' '12' 'Hordaland' '14' 'Sogn og Fjordane' '15' 'Møre og Romsdal' '16' 'Sør-Trøndelag' '17' 'Nord-Trøndelag' '18' 'Nordland' '19' 'Troms' '20' 'Finnmark' '99' 'Uoppgitt'

assign-labels fylke00 fylkerstring

tabulate fylke00
tabulate kjonn
```

```
tabulate kjonn regstat
tabulate kjonn regstat fylke00

// Krysstabell med kun kategoriverdier (ikke labler)
tabulate kjonn regstat fylke00, nolabels

// Krysstabell der missingverdier tas med
tabulate fylke00 regstat, missing

// Krysstabell der vi bare ser på dem som er bosatt per 1.1.2000
tabulate fylke00 regstat if regstat == '1'

// Krysstabell der vi bare viser resultatet for personer over 30 år
generate alder = 2000 - int(faarmnd/100)
tabulate fylke00 regstat if alder > 30

// Prosenttabeller
tabulate sivstand00 sivstand05, rowpct
tabulate sivstand00 sivstand05, colpct
tabulate sivstand00 sivstand05, cellpct
tabulate sivstand00 sivstand05, rowpct freq

// tabulate-kommandoen kan også brukes til å lage volumtabeller vha. summarize-option. Dette viser statistikk som gjennomsnitt m.m. for en gitt variabel fordelt på kategorier spesifisert gjennom de valgte tabulate-variable

tabulate fylke00 kjonn, summarize(formue)
```

4.8.2 Summarize og boxplot

```
// Nøkkelstatistikk for metriske/kontinuerlige variabler
// Kommandoen summarize brukes for å vise nøkkelinformasjon om metriske/kontinuerlige variabler. Verdier som vises er gjennomsnitt, kvartiler m.m. Kommandoen boxplot viser de samme tallene grafisk gjennom standard boxplotfremstilling

require no.ssb.fdb:1 as fdb1

create-dataset demografidata
import fdb1/INNTEKT_WYRKINNT 2000-01-01 as innt
import fdb1/INNTEKT_BRUTTOFORM 2000-01-01 as formue
import fdb1/BEFOLKNING_KJOENN as kjonn
```

```
import fdb1/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
import fdb1/BOSATTEFDT_BOSTED 2000-01-01 as bosted00

// Koder om fra kommune- til fylkesnivå
generate fylke00 = substr(bosted00,1,2)

// Lager alder per 2000
generate alder = 2000 - int(faarmnd/100)

summarize innt formue
summarize formue if alder > 50
summarize formue if bosted00 == '0301'

boxplot innt formue
boxplot innt, over( kjonn )
```

4.8.3 Histogram og barchart

```
// Histogram og barchart

require no.ssb.fdb:1 as fdb1

create-dataset demografidata
import fdb1/INNTEKT_WYRKINNT 2000-01-01 as innt
import fdb1/INNTEKT_BRUTTOFORM 2000-01-01 as formue
import fdb1/BEFOLKNING_KJOENN as kjonn
import fdb1/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd

// Lager alder per 2000
generate alder = 2000 - int(faarmnd/100)

// Histogram (frekvensfordelinger)
// Dette er en måte å vise frekvensfordelinger for metriske/kontinuerlige variabler på en grafisk måte der verdiene grupperes i passende intervaller og tilegnes søyler som viser graden av forekomst. Søylearealene i diagrammet summerer seg til 1 som default, men en kan overstyre dette gjennom options. Gjennom options kan en dessuten selv velge inndelingen av av verdier (hvor mange søyler en ønsker), legge på en normalfordelingskurve som referanse m.m.

histogram innt
histogram innt, freq
histogram innt, fraction
histogram innt, percent
```

```
histogram innt, normal
histogram innt, bin(6) freq
histogram innt, width(100000) freq

histogram innt, by( kjonn )
histogram innt if innt > 100000


// Ved bruk av discrete-option kan en også lage histogrammer for diskrete variabler. Da vil hver kategori representeres av respektive søyler

histogram alder, discrete


// Søylediagram
// Slike diagrammer er fine til å fremstille statistikk for kontinuerlige/metriske variabler på en oversiktlig måte. En kan kombinere flere variabler og bryte ned tallene på kategoriske egenskaper (kjønn, utdanningsnivå etc)

barchart ( mean) innt, over( kjonn )
barchart ( mean) innt formue, over( kjonn )
```

4.8.4 Piechart og hexbin-plot

```
require no.ssb.fdb:1 as fdb1

create-dataset demografidata
import fdb1/INNTEKT_WYRKINNT 2000-01-01 as innt
import fdb1/INNTEKT_BRUTTOFORM 2000-01-01 as formue
import fdb1/BEFOLKNING_KJOENN as kjonn
import fdb1/BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
import fdb1/BOSATTEFDT_BOSTED 2000-01-01 as bosted00

// Koder om fra kommune- til fylkesnivå
generate fylke00 = substr(bosted00,1,2)

define-labels fylkerstring '01' 'Østfold' '02' 'Akershus' '03' 'Oslo' '04' 'Hedmark' '05' 'Oppland' '06' 'Buskerud' '07' 'Vestfold' '08' 'Telemark' '09' 'Aust-Agder' '10' 'Vest-Agder' '11' 'Rogaland' '12' 'Hordaland' '14' 'Sogn og Fjordane' '15' 'Møre og Romsdal' '16' 'Sør-Trøndelag' '17' 'Nord-Trøndelag' '18' 'Nordland' '19' 'Troms' '20' 'Finnmark' '99' 'Uoppgitt'
```

```
assign-labels fylke00 fylkerstring


// Genererer alder per 2000
generate alder = 2000 - int(faarmnd/100)

// Kakediagram
// Dette er en fin måte å vise den prosentvise fordelingen for diskrete variabler på en grafisk måte

drop if alder < 16

piechart kjonn
piechart fylke00


// Hexbinplot
// Dette er en anonymiserende måte å fremstille plotdiagrammer (egner seg best for kontinuerlige/metriske variabler), der tettheten i plottene fargelegges på en systematisk måte for å avdekke mønstre i fordelingen mellom to variabler

hexbin formue innt
hexbin formue innt if alder > 50
```

4.8.5 Sankey-diagram

```
// Overgangsdiagram (Sankey)

require no.ssb.fdb:1 as fdb1

create-dataset demografidata
import fdb1/SIVSTANDFDT_SIVSTAND 2000-01-01 as sivstand00
import fdb1/SIVSTANDFDT_SIVSTAND 2005-01-01 as sivstand05
import fdb1/BOSATTEFDT_BOSTED 2000-01-01 as bosted00
import fdb1/BOSATTEFDT_BOSTED 2005-01-01 as bosted05

// Koder om fra kommune- til fylkesnivå
generate fylke00 = substr(bosted00,1,2)
generate fylke05 = substr(bosted05,1,2)

define-labels fylkerstring '01' 'Østfold' '02' 'Akershus' '03' 'Oslo' '04' 'Hedmark' '05' 'Oppland' '06'
```

```
'Buskerud' '07' 'Vestfold' '08' 'Telemark' '09' 'Aust-Agder' '10' 'Vest-Agder' '11' 'Rogaland' '12'
'Hordaland' '14' 'Sogn og Fjordane' '15' 'Møre og Romsdal' '16' 'Sør-Trøndelag' '17' 'Nord-Trøndelag'
'18' 'Nordland' '19' 'Troms' '20' 'Finnmark' '99' 'Uoppgitt'

assign-labels fylke00 fylkerstring
assign-labels fylke05 fylkerstring

sankey fylke00 fylke05 if fylke00 == '12'
sankey fylke00 fylke05 if fylke05 == '03'
sankey fylke00 fylke05 if fylke05 == '03' & fylke00 != '03'

sankey sivstand00 sivstand05
sankey sivstand00 sivstand05 if sivstand00 == '2'
```

5. Avansert analyse

I tillegg til de deskriptive mulighetene som finnes i microdata.no, kan en også foreta avanserte analyser i form av regresjonsanalyser m.m. Foreløpig er følgende analysemetoder tilgjengelige i microdata.no:

- correlate
- anova
- regress
- logit / probit
- mlogit
- regress-panel

Mer analysefunksjonalitet vil bli lagt til systemet fortløpende, i tråd med tilbakemeldinger fra brukere. I prinsippet kan alt av analysefunksjonalitet som finnes i STATA også implementeres i microdata.no.

5.1 Correlate - korrelasjon

Kommandoen `correlate` brukes til å analysere statistisk sammenheng mellom variabler, altså korrelasjon. Det rapporteres verdier mellom -1 og 1 i en korrelasjonsmatrise for de spesifiserte variablene, der minus- og pluss-verdier impliserer hhv. negativ og positiv sammenheng. Verdien 0 indikerer ingen sammenheng. Jo nærmere +/- 1 en kommer, jo sterkere korrelasjon.

Syntax:

```
correlate <variabelliste> [if] [, options]
```

Dersom en ikke oppgir variabelliste, vises en korrelasjonsmatrise for alle variablene i datasettet.

En kan gjennom opsjoner vise alternative mål:

- covariance Viser kovarians i stedet for korrelasjonskoeffisient
- pairwise Parvis fremvisning
- obs Viser antall observasjoner som ligger bak hver korrelasjonskoeffisient
- sig Viser signifikansverdien for hver korrelasjonskoeffisient

Eksempler:

```
»correlate alder formue
```

	formue
alder	0.3291

```
»correlate alder formue, obs
```

		formue
alder	corr	0.3291
	obs	3172121

```
»correlate alder formue, sig
```

		formue
alder	corr	0.3291
	sig	0

5.2 Anova

Anova-tester kan ses på som en forenklet regresjonsanalyse, der en undersøker om gjennomsnittsverdien til en avhengig kontinuerlig variabel er forskjellig i to eller flere uavhengige grupper gitt ved en annen kategorisk variabel. Ett eksempel er å teste om gjennomsnittslønnen er forskjellig for personer med hhv. lav, middels og høy utdannelse (benytter en uavhengig variabel der utdanningsnivå deles inn i tre grupper).

En Anova-test kan sjekke om det eksisterer signifikante forskjeller mellom minst to av gruppene (gitt ved den uavhengige variabelen), men gir ikke svar på hvilke(n) gruppe(r) dette gjelder. Til dette må en foreta en regresjonsanalyse (se kapittel 5.3).

Syntax:

```
anova <variabel> <variabelliste> [if] [, options]
```

Om en tester for kun 2 variabler, dvs. én avhengig og én uavhengig variabel, foretar man en enveis Anova-test. En kan også teste en avhengig variabel opp mot to andre kategoriske uavhengige variabler. Dette kalles toveis Anova-test.

Eksempel:

```
»anova innt05 mann
    Antall Obs: 2489943          Root MSE:
           R²: 0.0746       1.905535e+5
                              Justert R²: 0.0746
```

Kilde	Del. SS	Frihetsgrad	MS	F	Sanns > F
Residual	9.04113e+16	2.489941e+6	3.631062e+10	-	-
mann	7.291912e+15	1	7.291912e+15	2.0082e+5	0

5.3 Regress - ordinær lineær regresjonsanalyse

Kommandoen `regress` brukes til å utføre en ordinær lineær regresjonsanalyse (OLS) der den avhengige variabelen er en kontinuerlig/metrisk variabel. Et typisk eksempel er inntekt.

Kort fortalt går modellen ut på å estimere (mulige) marginaleffekter av et sett med uavhengige variabler (forklaringsvariabler) på den avhengige variabelen (responsvariabelen). Marginaleffekt er et mål på hvor mye den avhengige variabelen estimeres til å øke i verdi dersom en uavhengig variabel øker med én måleenhet.

Det viktigste å se på når en skal tolke resultatet av en regresjonskjøring er forklaringskraften til:

a) Modellen som helhet
b) Hver enkelt variabel

Dette gjøres ved å studere de respektive signifikansverdiene "*Justert R^2*" og "*$P > |t|$*".

Nedenfor vises et eksempel på resultatet av en regresjonsskjøring i microdata.no. Tallene i den nederste delen knyttes til de ulike variablene, mens tallene øverst viser til analysemodellen som helhet.

```
»regress innt05 mann gift alder formuehøy
```

Kilde	SS	df	MS
Modell	1.074715e+16	4	2.686788e+15
Residual	8.695606e+16	2.489938e+6	3.492298e+10
Total	9.770321e+16	2.489942e+6	3.923915e+10

```
Antall Obs: 2489943
F(4, 2489938): 7693
          R²: 0.1099
Justert R²: 0.1099
  Root MSE: 1.868769e+5
```

innt05	Coef.	Std. Avvik	t	P>|t|	[% Konf.	Intervall]
alder	-1157.7	10.741	-107.78	0	-1178.8	-1136.7
formuehøy	88753.	387.74	228.89	0	87993.	89513.
gift	55131.	276.38	199.47	0	54590.	55673.
mann	99052.	242.13	409.07	0	98577.	99526.
Konst	2.455021e+5	393.99	623.1	0	2.447299e+5	2.462744e+5

Justert R^2 gir et samlemål for hvor mye av den observerte variansen i den avhengige variabelen som kan forklares av summen av de uavhengige variablene, angitt i prosent. Skalaen går fra 0 til 1, der nærmest mulig 1 er det ideelle. I praksis vil en aldri nå verdien 1 ved analyser av sosioøkonomiske individdata pga. tilfeldig støy og uobserverte årsakssammenhenger, og typiske verdier vil gjerne ligge i intervallet 0 - 0.5.

R^2 vil alltid øke i verdi for hver ekstra uavhengige variabel som legges til. Dette betyr ikke nødvendigvis at modellen blir bedre, spesielt om variablene en legger til *ikke* er signifikante. *Justert* R^2 tar hensyn til dette og vil kun øke i verdi dersom de ekstra variablene er signifikante. Derfor anbefales det å se på dette måltallet.

Dersom *justert* R^2 får *lavere* verdi ved å tilføye en ekstra uavhengig variabel, er dette en indikasjon på den valgte variabelen kan ha en relativt høy grad av korrelasjon med noen av de andre uavhengige variablene, dvs. multikollinearitet. Dette er absolutt noe en bør unngå.

P > |*t*| eller *p-verdiene* (i kolonne 4 i den nedre hovedtabellen) angir sannsynligheten for at *t*-verdien fremkommer som et resultat av ren tilfeldighet. For å kunne si at en variabel er signifikant, må den tilhørende *p*-verdien være lavere enn 0.05 ved et 5%-signifikansnivå. Verdier nærmest mulig 0 er ideelt.

Verdien *t* (kolonne 3) er kort fortalt et standardisert mål for koeffisientverdien (=marginaleffekten), jfr. verdiene i *Coef.*-kolonnen (kolonne 1), der en ved et 5%-signifikansnivå får grenseverdier på +/- 1.96. Verdier som overstiger 1.96 med positivt eller negativt fortegn vil altså regnes som signifikante på et 5%-nivå (5%-nivå er den grenseverdien som er vanlig å operere med).

En kan også se på 95%-konfidensintervallet presentert i de to kolonnene lengst til høyre i hovedtabellen. Dersom intervallet inkluderer verdien 0, kan en utelukke at den aktuelle koeffisienten viser en signifikant sammenheng mellom den tilhørende uavhengige variabelen og responsvariabelen.

Koeffisientverdiene i kolonne 1 er kun relevante for signifikante variabler, og viser marginaleffekten på responsvariabelen av en enhets økning i verdien på den tilhørende uavhengige variabelen.

I eksempelet ovenfor ser en at alle variablene er signifikante med god margin (høye t-verdier). Alder har negativ effekt på inntekt, mens de øvrige variabler har positiv effekt. *"Konst"* peker på konstantleddet, dvs. startverdien på responsvariabelen når alle uavhengige variabler har verdien 0, og har ikke noen stor betydning rent tolkningsmessig.

En *Justert* R^2 på 0.1099 er dessuten ikke så verst med tanke på at modellen i eksempelet kun har 4 forklaringsvariabler.

Om *Justert* R^2 har svært lav verdi, kan en enten føye til flere relevante uavhengige variabler, eller vurdere om en har benyttet riktig modelldesign. Går verdien *ned* ved tilføying av en ekstra variabel (i stedet for opp), kan dette tyde på multikollinearitet (innbyrdes sammenheng mellom uavhengige variabler). Det er derfor viktig å tenke nøye gjennom hvilke uavhengige variabler en benytter i en forklaringsmodell. Det en må passe ekstra godt på er at det er minst mulig grad av innbyrdes korrelasjon mellom dem. Om en er i tvil, kan korrelasjonsmatriser fint benyttes for å avdekke slikt. Analysesystemet microdata.no har en kommando til dette formålet:

```
corr <variabelliste>
```

I kapittel 5.1 gjennomgås kommandoen `corr`.

5.4 Logit og probit - logistisk regresjonsanalyse

Logistisk regresjonsanalyse brukes til å beregne sannsynligheten for «suksess» (én tilstand foran en annen) eller for å havne i en av flere mulige tilstander.

Kommandoene `logit` og `probit` brukes til å utføre en logistisk analyse der den avhengige variabelen er en kategorisk variabel med 2 mulige utfall (dummyvariabel). Eksempler kan være jobb/ikke jobb, alderpensjonist/ikke pensjonert etc. Logit-modeller antar at sannsynligheten for «suksess» følger en logaritmisk (log) fordeling, mens probit-varianten antar en normalfordeling. De to fordelingene er tilnærmet like, og resultatene vil derfor bli tilnærmet like. Logit er imidlertid den mest brukte modellen, og det er den vi fokuserer på i eksemplene nedenfor.

Resultatet av kommandoen `logit` gir en tabell med standardverdier som koeffisienter, standardavvik, z-verdier, p-verdier og konfidensintervall. Tallene inni hovedtabellen knyttes til de ulike variablene, mens tallene øverst viser til analysemodellen som helhet (gir pekepinn på modellens kvalitet/forklaringskraft).

Eksempel:

```
»logit høyinnt mann gift alder formuehøy
   Antall iter: 8              LR chi2(5): 7.6908e+5
      Log sans:               Prob > chi2: 0
 -1.532013e+6                   Pseudo R2: 0.2006
    Antall obs: 7241907
```

høyinnt	Coef.	Std. Avvik	z	P>|z|	[% Konf.	Intervall]
alder	-0.0454	0.0001	-418.84	0	-0.0456	-0.0452
formuehøy	1.3824	0.0043	315.11	0	1.3738	1.391
gift	1.8289	0.0036	506.76	0	1.8219	1.836
mann	1.2097	0.0036	329.65	0	1.2025	1.2169
Konst	-2.1892	0.0048	-449.09	0	-2.1988	-2.1796

I eksempelet over er den avhengige variabelen “høyinnt” kodet på følgende måte:

```
generate høyinnt = 0
replace høyinnt = 1 if innt05 > 400000
```

I likhet med ordinær lineær regresjonsanalyse (se kapittel 5.3) er det noen tall som er viktigere å se på enn andre. P-verdien *Prob > chi2* angir hvor god modellen er, dvs. den angir hvor stor forklaringskraft summen av alle variablene i modellen har. Jo nærmere 0 dess bedre, og verdier bør være under 0.05.

Pseudo R2 er en variant av *justert* R^2 (rapporteres ved ordinære lineære regresjonsanalyser) som angir hvor stor andel av variansen i responsvariabelen som blir forklart av de uavhengige variablene (skala fra 0 til 1 der en søker høyest mulig verdi). Dette samlemålet må imidlertid tolkes med stor grad av varsomhet ettersom den i mange tilfeller angir en verdi som enten er kunstig høy eller lav. *Prob > chi2* er derfor å anbefale for logistiske regresjonsmodeller.

Variablenes p-verdi $P > |z|$ tilsvarer $P > |t|$ i ordinær lineær regresjonsanalyse. Grenseverdien er også her 0.05 om en opererer med et signifikansnivå på 5% (noe de fleste bruker). Rapporterte verdier under dette gjør at en kan konkludere med at tilhørende variabel er signifikant på et 5% nivå.

Det er det samme om en ser på *z*-verdi eller tilhørende p-verdi. Verdien *z* er en standardisert versjon av coeffisientverdien, som har forventning lik 0 og der verdier som overstiger +/- 1.96 impliserer at den aktuelle variabelen som koeffisienten tilhører har en signifikant påvirkning på sannsynligheten for «suksess». Positive verdier angir positiv effekt, og vice versa.

Konfidensintervallet gitt ved de to kolonnene lengst til høyre kan tolkes på samme måte som for ordinær lineær regresjonsanalyse, dvs. om det inkluderer verdien 0 tyder dette på null signifikans.

En ser av eksempelet ovenfor at alle forklaringsvariablene er signifikante med god margin (har høye z-verdier). “Alder” har en negativ effekt på sannsynligheten for å havne i en høyinntektsgruppe, mens de andre variablene har en tilsvarende positiv effekt. Videre ser en at modellens P-verdi er lik 0, noe som viser at vi har en god forklaringsmodell.

5.5 Mlogit - multinomisk logistisk regresjonsanalyse

En kan analysere logistiske regresjonsmodeller der den avhengige variabelen har flere enn 2 utfall. Til dette brukes en multinomisk modell.

En kan benytte logistiske modeller med flere enn 2 mulige utfall for responsvariabelen, dvs. multinomiske logit-modeller, gjennom følgende kommando:

```
mlogit <responsvariabel> <uavhengige variabler> [, <opsjoner>]
```

I det rapporterte resultatet vil hovedtabellen bli mer omfattende enn for vanlige (binomiske) logistiske modeller. En får et sett med koeffisienter, standardfeil, z-verdier etc for hvert mulige utfall minus referanseutfallet. Ved f.eks. 3 utfall får en altså 2 sett med verdier. Ved tolkninger av disse så sammenlikner en med sannsynligheten for å havne i referanseutfallet. Betydningen av de ulike rapporterte målene i tabellen gjennomgås i kapittel 5.4.

Eksempel:

```
»mlogit inntgr mann gift alder formuehøy
   Antall iter: 8            LR chi2(10): 2.027e+6
      Log sans:              Prob > chi2: 0
-3.9626159e+6                  Pseudo R2: 0.2036
   Antall obs: 7241907
```

	inntgr	Coef.	Std. Avvik	z	P>|z|	[% Konf.	Intervall]
2	alder	-0.0561	0	-712.67	0	-0.056	-0.056
	formuehøy	0.7157	0.005	156.58	0	0.707	0.725
	gift	1.8572	0.003	674.68	0	1.852	1.863
	mann	-0.2324	0.002	-101.07	0	-0.237	-0.228
	Konst	0.4212	0.003	134.66	0	0.415	0.427
3	alder	-0.0586	0	-495.28	0	-0.059	-0.058
	formuehøy	1.5778	0.005	334.31	0	1.569	1.587
	gift	2.3359	0.004	601.74	0	2.328	2.343
	mann	1.1127	0.004	298.24	0	1.105	1.12
	Konst	-1.4676	0.005	-290.81	0	-1.478	-1.458

I eksempelet over er den avhengige variabelen “inntgr” kodet på følgende måte:

```
generate inntgr = 1
replace inntgr = 2 if innt05 > 200000
replace inntgr = 3 if innt05 > 400000
```

5.6 Regress-panel - paneldataanalyse

Paneldata er datasett der hver enhet har oppgitt verdier for samtlige variabler målt over et gitt antall måletidspunkt. Dette har den fordelen at en kan ta med tidskomponenten i analyser, og at en får mye større datagrunnlag og gjerne analyser av en bedre kvalitet.

Se kapittel 2.4 for hvordan en oppretter datasett for paneldata-analyse. Der finner en også et skript-eksempel.

Det finnes et stort batteri av paneldataanalyser som kan tas i bruk, avhengig av hvilke antakelser som gjøres om de ulike variablenes variasjon over tid. Vanlige varianter som brukes er “fixed effect”- og “random effect”-analyser.

I eksempelet nedenfor brukes årslønn (årlig lønnsinntekt) som avhengig variabel, og dummyvariabler for hhv. sivilstatus=gift og bosted=oslo brukes som forklaringsvariabler. I tillegg er 5 måletidspunkter benyttet: 31/12 i årene 2011-2015. Populasjon = alle personer som fullførte et masterstudium i løpet av høstsemesteret 2010.

Eksempel 1: Panel-regresjon med fixed effects

```
paneldata2» regress-panel årslønn gift oslo, fe
```

Antall Obs:	20225	R^2 i:	0.03406
Antall grupper:	4247	R^2 mellom:	-0.00656
Min obs/grp:	1	R^2 total:	0.00487
Snitt obs/grp:	4.76218	Corr(u_i, Xb):	-0.11256
Maks obs/grp:	5		
F(2,15976):	281.69067574	Sigma u:	206600.87339816123
Prob > F:	1.11022e-16	Sigma e:	126287.16304855184
		Rho:	0.72799

årslønn	Coef.	Std.feil	t	P>|t|	[95% Konf.	intervall]
gift	1.0535662e+5	4609.16	22.858	0	1.0506759e+5	1.0564565e+5
oslo	36284.5	4911.99	7.38693	1.57651e-13	35976.5	36592.5
Konst	4.6886852e+5	2598.49	180.438	0	4.6870557e+5	4.6903147e+5

Eksempel 2: Panel-regresjon med random effects (samme datasett som eksempel 1)

```
paneldata2» regress-panel årslønn gift oslo, re
```

Antall Obs:	20225	R^2 i:	0.0333
Antall grupper:	4247	R^2 mellom:	0.00315
Min obs/grp:	1	R^2 total:	0.00896
Snitt obs/grp:	4.76218		
Maks obs/grp:	5	Sigma u:	196036.29557757667
F(2,20222):	96.71945325	Sigma e:	126287.16304855184
Prob > F:	1.11022e-16	Rho:	0.70671

årslønn	Coef.	Std.feil	t	P>|t|	[95% Konf.	intervall]
gift	91013.1	3913.31	23.2573	0	90767.7	91258.5
oslo	27222.5	4026.04	6.76161	1.40187e-11	26970	27475
Konst	4.6809574e+5	3756.31	124.615	0	4.6786019e+5	4.6833129e+5

I tillegg til regresjons-analyser, er det mulig å gjøre seg kjent med paneldatasett gjennom ulike deskriptive verktøy:

- `tabulate-panel` tilsvarer kommandoen `tabulate` for vanlige datasett, jfr. kapittel 4.1, men viser verdier for alle måletidspunkt. Opsjoner for prosentuering kan brukes i likhet med `tabulate`. Spesifiseres flere variabler, vises det flerdimensjonale krysstabeller for de aktuelle variabler

- `summarize-panel` tilsvarer kommandoen `summarize` for vanlige datasett, jfr. kapittel 4.2, men viser verdier for alle måletidspunkt. Verdier vises vertikalt og ikke horisontalt, og en må holde musepeker over tallene for å vise deres betydning

- `transitions-panel` viser to-veis frekvens/sannsynlighet for overganger mellom alle kombinasjoner av kategoriske verdier over tid (overgangssannsynligheter), for en gitt variabel. Forspalten representerer utgangsverdiene, mens tabellhodet representerer overgangsverdien. Spesifiseres flere variabler, vises toveis overgangstabeller for hver variabel i respektive tabeller. Overganger representeres som standard gjennom frekvenser og prosenter (rekkevis). Overganger enten fra eller til manglende verdi (sysmiss) holdes utenfor tabuleringen.

Eksempel 3: Tabulate-panel for hhv. gift og oslo (samme datasett som eksempel 1 og 2)

```
paneldata2» tabulate-panel gift
```

		date@panel					
		2011-12-31	2012-12-31	2013-12-31	2014-12-31	2015-12-31	*Total*
gift	0	3517	3398	3237	3047	2908	*16110*
	1	1083	1208	1374	1556	1691	*6900*
Total		*4601*	*4604*	*4606*	*4598*	*4600*	*23005*

```
paneldata2» tabulate-panel oslo
```

		date@panel					
		2011-12-31	2012-12-31	2013-12-31	2014-12-31	2015-12-31	*Total*
oslo	0	3052	2993	2977	3010	3053	*15073*
	1	1551	1611	1623	1597	1550	*7936*
Total		*4601*	*4604*	*4606*	*4598*	*4600*	*23005*

Eksempel 4: Summarize-panel for den avhengige variabelen årslønn (samme datasett)

```
paneldata2» summarize-panel årslønn
```

date@panel		
	2011-12-31	408852.72 196058.35 4047 6066 1103085 303401 413573 497759.75
	2012-12-31	484347.39 198684.67 4065 13348 1202311 400655 474860 569977.25
	2013-12-31	527803.61 213332.71 4041 22857 1304797 431857.25 513947 617650.5
	2014-12-31	567728.92 235573.16 4043 21912 1462289 452964.25 546906 665830.75
	2015-12-31	596611.39 247937.55 4026 15000 1572028 474567.5 571551 701034.5
	Total	*516957.13* *228922.05* *20227* *6066* *1572028* *406669* *501847* *620650*

Eksempel 5: Transitions-panel (overgangsrater mellom kombinasjoner av kategoriske verdier) for variablene oslo og gift (samme datasett)

```
paneldata2» transitions-panel oslo gift
```

oslo

	0	1	*Total*
0	11567 96.21	455 3.784	*12022* *100*
1	460 7.203	5926 92.79	*6386* *100*
Total	*12027* *65.33*	*6381* *34.66*	*18408* *100*

gift

	0	1	*Total*
0	12474 94.48	728 5.514	*13202* *100*
1	120 2.305	5086 97.69	*5206* *100*
Total	*12594* *68.41*	*5814* *31.58*	*18408* *100*

Kommentar til tabell i eksempel 5:
I 96.21% av tilfellene vil personer som ikke er bosatt i Oslo ha samme tilstand året etter (neste måling). Resten, 3.78%, vil flytte til Oslo. Blant dem i populasjonen som bor i Oslo i et gitt tidspunkt, vil 7.2% flytte ut av Oslo mens 92.8% blir værende året etter (neste måling).

Samme prinsipp gjelder for variabelen *gift*: Her ser vi at 5.5% endrer status fra ikke-gift til gift fra ett år til et annet (neste måling) over den totale måleperioden. 2.3% endrer status fra gift til ikke-gift.

Vedlegg A: Oversikt over kommandoer

Kommando	**Formål**	**Type funksjonalitet**
`clear`	Tøm kommandoøkten	Støttekommando
`help`	Hjelp	Støttekommando
`help-function`	Hjelp - funksjoner	Støttekommando
`history`	Kommandohistorikk	Støttekommando
`load`	Laste inn et skript som en kommandolinjeøkt	Støttekommando
`save`	Lagre en kommandoøkt som et skript	Støttekommando
`variables`	Vise metadata for registervariable	Støttekommando
`clone-dataset`	Duplisere et datasett	Datasett-kommando
`clone-units`	Duplisere en datapopulasjon	Datasett-kommando
`create-dataset`	Lage nytt, tomt, navngitt datasett	Datasett-kommando
`delete-dataset`	Slette datasett	Datasett-kommando
`rename-dataset`	Endring av datasettnavn	Datasett-kommando
`require`	Koble til databank	Datasett-kommando
`use`	Bruk et navngitt datasett	Datasett-kommando
`assign-labels`	Sette labler på variabler og kategorier	Tilretteleggingskommando
`clone-variables`	Duplisere variabler	Tilretteleggingskommando
`collapse`	Aggregering av datasett	Tilretteleggingskommando
`define-labels`	Definere lister av value/labels	Tilretteleggingskommando
`destring`	Konvertere fra alfanumeriske til numeriske verdier	Tilretteleggingskommando
`drop`	Slette variabler eller records (drop if)	Tilretteleggingskommando
`drop-labels`	Fjerne kodelister	Tilretteleggingskommando
`generate`	Lage ny variabel på basis av uttrykk	Tilretteleggingskommando
`import`	Import av variabel til datasett	Tilretteleggingskommando
`import-event`	Import av forløpsvariabel (tidsrom) til datasett (må være tomt)	Tilretteleggingskommando
`import-panel`	Import av paneldata til datasett (må være tomt)	Tilretteleggingskommando
`keep`	Behold variabler eller records (slett resten)	Tilretteleggingskommando

list-labels	List ut egendefinerte value/label-lister	Tilretteleggingskommando
merge	Koble variabler fra et datasett inn i et annet. Koblingsnøkkel er enhetsidentifikator i kildedatasettet. Dette kan overstyres vha. en "on"-opsjon der en selv bestemmer koblingsnøkkel.	Tilretteleggingskommando
recode	Omkoding av numeriske variabler	Tilretteleggingskommando
rename	Omdøping av variabel	Tilretteleggingskommando
replace	Endre innhold i eksisterende variabler	Tilretteleggingskommando
sample	Lage et tilfeldig utvalg av totalpopulasjon	Tilretteleggingskommando
split	Dele opp strengvariabler i deler	Tilretteleggingskommando
anova	Anova/ancova variansanalyse	Analysekommando
barchart	Søylediagram for kategoriske variabler	Analysekommando
boxplot	Boksplott for numeriske variabler	Analysekommando
ci	Konfidensintervall og standardfeil	Analysekommando
correlate	Korrelasjonsmatrise	Analysekommando
hexbin	Anonymisert scatterplot	Analysekommando
histogram	Histogram	Analysekommando
logit	Logistisk regresjonsanalyse: Logit	Analysekommando
mlogit	Multinomisk logistisk regresjonanalyse	Analysekommando
normaltest	Et utvalg normalfordelingstester for angitte variabler	Analysekommando
piechart	Kakediagram	Analysekommando
probit	Logistisk regresjonsanalyse: Probit	Analysekommando
regress	Lineær regresjon	Analysekommando
regress-panel	Lineær regresjon for paneldata	Analysekommando
sankey	Sankey-diagram (overgangsdiagram)	Analysekommando
summarize	Oppsummerende statistikk (numeriske variabler)	Analysekommando
summarize-panel	Oppsummerende statistikk for paneldata	Analysekommando
tabulate	Frekvens- og volumtabeller (kategoriske variabler)	Analysekommando
tabulate-panel	Frekvenstabeller for paneldata	Analysekommando
transitions-panel	Overgangssannsynligheter for paneldata	Analysekommando

Vedlegg B: Oversikt over funksjoner

Matematiske funksjoner

- ln(arg1)
  - Beskrivelse: Den naturlig logaritmen av arg1 (den inverse av exp(arg1))
  - arg1: Verdier mellom 1e–323 og 8e+307
  - Output: Verdier mellom -744 og 709
  - Eksempler:

- log10(arg1)
  - Beskrivelse: Base 10-logaritmen av arg1
  - arg1: Verdier mellom 1e–323 og 8e+307
  - Output: Verdier mellom −323 og 308
  - Eksempler:

- exp(arg1)
  - Beskrivelse: Eksponentialfunksjonen e^{arg1} (den inverse av ln(arg1))
  - arg1: Verdier mellom −8e+307 og 709
  - Output: Verdier mellom 0 og 8e+307
  - Eksempler:

- sqrt(arg1)
  - Beskrivelse: Kvadratroten av arg1
  - arg1: Verdier mellom 0 og 8e+307
  - Output: Verdier mellom 0 og 1e+154
  - Eksempler:

- abs(arg1)
  - Beskrivelse: Absoluttverdien av arg1 (dvs. fjerner negative fortegn)
  - arg1: Verdier mellom −8e+307 og 8e+307
  - Output: Verdier mellom 0 og 8e+307
  - Eksempler:

- sin(arg1)
  - Beskrivelse: Sinusverdien av arg1
  - arg1: Verdier mellom −1e+18 og 1e+18
  - Output: Verdier mellom -1 og 1
  - Eksempler:

- cos(arg1)
  - Beskrivelse: Returnerer cosinusverdien av arg1
  - arg1: Verdier mellom −1e+18 og 1e+18
  - Output: Verdier mellom -1 og 1
  - Eksempler:

- tan(arg1)
  - Beskrivelse: Returnerer tangensverdien av arg1
  - arg1: Verdier mellom −1e+18 og 1e+18
  - Output: Verdier mellom −1e+17 og 1e+17 eller missing
  - Eksempler:

- asin(arg1)
  - Beskrivelse: Returnerer radianverdien av arcsinusen til arg1
  - arg1: Verdier mellom -1 og 1
  - Output: Verdier mellom −π/2 og π/2
  - Eksempler:

- acos(arg1)
  - Beskrivelse: Returnerer radianverdien av arccosinusen til arg1
  - arg1: Verdier mellom -1 og 1
  - Output: Verdier mellom 0 og π
  - Eksempler:

- atan(arg1)
  - Beskrivelse: Returnerer radianverdien av arctangens til arg1
  - arg1: Verdier mellom −8e+307 og 8e+307
  - Output: Verdier mellom −π/2 to π/2
  - Eksempler:

- ceil(arg1)
  - Beskrivelse: Heltallsavrunding oppover
  - arg1: Verdier mellom −8e+307 og 8e+307
  - Output: Heltall mellom −8e+307 og 8e+307

- Eksempler: `ceil(5.2) = 6`
  `ceil(-5.2) = -6`

- floor(arg1)
  - Beskrivelse: Heltallsavrunding nedover. Tilsvarer funksjonen int(arg1)
  - arg1: Verdier mellom −8e+307 og 8e+307
  - Output: Heltall mellom −8e+307 og 8e+307
  - Eksempler: `floor(5.8) = 5`
    `floor(-5.8) = -5`

- int(arg1)
  - Beskrivelse: Heltallsverdien av arg1 (dvs. dropper desimaltall). Tilsvarer funksjonen floor(arg1)
  - arg1: Verdier mellom −8e+307 og 8e+307
  - Output: Heltall mellom −8e+307 og 8e+307
  - Eksempler: `int(5.8) = 5`
    `int(-5.8) = -5`

- logit(arg1)
  - Beskrivelse: Logverdien av oddsratioen til arg1 (= ln {arg1/(1-arg1)})
  - arg1: Verdier mellom 0 og 1 (ikke inkludert)
  - Output: Verdier mellom −8e+307 og 8e+307 eller missing
  - Eksempler:

- lnfactorial(arg1)
  - Beskrivelse: Den naturlige logaritmen av n-faktor (= ln(n!))
  - n: Heltallsverdier mellom 0 og 1e+305
  - Output: Verdier mellom 0 og 8e+307
  - Eksempler:

- comb(n,k)
  - Beskrivelse: Kombinatorisk funksjonsverdi (= n!/{k!(n-k)!})
  - n: Heltallsverdier mellom 1 og 1e+305
  - k: Heltallsverdier mellom 0 og n
  - Output: Verdier mellom 0 og 8e+307 eller missing
  - Eksempler:

- round(arg1,arg2)
  - Beskrivelse: Avrunder til nærmeste heltall dersom arg2 utelates eller settes lik 1. arg2 bestemmer hvilket nivå en skal avrunde på
  - arg1: Verdier mellom −8e+307 og 8e+307
  - arg2: Verdier mellom −8e+307 og 8e+307 (default = 1 dersom arg2 droppes)
  - Output: Verdier mellom −8e+307 og 8e+307
  - Eksempler:
```
round(5.2) = 5
round(5.8) = 6
round(5.8,1) = 6
round(5.8,5) = 5
round(5.8,10) = 10
round(5.8621,0.01) = 5.86
```

- max(arg1, arg2, ….., argn)
  - Beskrivelse: Henter ut maxverdien av variablene arg1, arg2, ….., argn for den gitte enhet. Missingverdier ignoreres (regnes som nullverdier) gitt at ikke alle verdier = missing
  - arg1, arg2, ….., argn: Variabler med tallverdier mellom −8e+307 og 8e+307 eller missing
  - Output: Verdier mellom −8e+307 og 8e+307 eller missing
  - Eksempler:

- min(arg1, arg2, ….., argn)
  - Beskrivelse: Henter ut minimumsverdien av variablene arg1, arg2, ….., argn for den gitte enhet. Missingverdier ignoreres (regnes som nullverdier) gitt at ikke alle verdier = missing
  - arg1, arg2, ….., argn: Variabler med tallverdier mellom −8e+307 og 8e+307 eller missing
  - Output: Verdier mellom −8e+307 og 8e+307 eller missing
  - Eksempler:

Strengefunksjoner

- string(arg1)
  - Beskrivelse: Konverterer verdien arg1 til string-format
  - arg1: Verdier mellom −8e+307 og 8e+307 eller missing
  - Output: Verdien arg1 konvertert til string-format
  - Eksempler: `string(1234567) = '1234567'`

- upper(arg1)
  - Beskrivelse: Konverterer tekst/string til “uppercase” (ASCII) (unicode-karakterer utenfor ASCII-spekteret ignoreres)
  - arg1: String-verdier
  - Output: String-verdier konvertert til “uppercase”
  - Eksempler: `upper('abcde') = 'ABCDE'`
    `upper('abcdé') = 'ABCDé'`

- lower(arg1)
  - Beskrivelse: Konverterer tekst/string til “lowercase” (ASCII) (unicode-karakterer utenfor ASCII-spekteret ignoreres)
  - arg1: String-verdier
  - Output: String-verdier konvertert til “lowercase”
  - Eksempler: `lower('ABCDE') = 'abcde'`
    `lower('ABCDÉ') = 'abcdÉ'`

- ltrim(arg1)
  - Beskrivelse: Fjerner ledende blanke karakterer (mellomrom) fra tekst
  - arg1: String-verdier
  - Output: String-verdier der ledende blanke karakterer er fjernet
  - Eksempler: `trim('    this') = 'this'`

- rtrim(arg1)
  - Beskrivelse: Fjerner blanke karakterer (mellomrom) fra slutten tekst
  - arg1: String-verdier
  - Output: String-verdier der alle blanke karakterer er fjernet fra slutten av tekstverdi
  - Eksempler: `trim('this    ') = 'this'`

- trim(arg1)
  - Beskrivelse: Fjerne blanke karakterer (mellomrom) både fra start og slutt på tekstverdi
  - arg1: String-verdier
  - Output: String-verdier der blanke karakterer er fjernet både fra start og slutt av tekstverdi
  - Eksempler: `trim('    this    ') = 'this'`

- length(arg1)
  - Beskrivelse: Angir antall karakterer i en tekstverdi (ASCII) (merk at for unicode-karakterer utenfor ASCII-spekteret angis verdien i antall bytes i stedet)
  - arg1: String-verdier
  - Output: Heltall >= 0

- Eksempler: `length('ab') = 2`

- substr(arg1,arg2,arg3)
  - Beskrivelse: Henter ut delteksten som starter ved posisjon arg2 og med lengde arg3
  - arg1: String-verdier
  - arg2: Heltall >=1 og <= -1 (ved negative verdier regnes posisjon i forhold til posisjonen til siste karakter)
  - arg3: Heltall >= 1
  - Output: Deltekst av arg1
  - Eksempler:
    ```
    substr('y32ssx',2,3) = '32s'
    substr('y32ssx',-3,2) = 'ss'
    substr('y32ssx',1,1) = 'y'
    ```

Sysmiss

- sysmiss(arg1)
  - Beskrivelse: Logisk funksjon som settes til "true" dersom variabelen arg1 har verdien "missing" (= ingen observasjoner i underliggende datasett)
  - arg1: Variabel (alle typer)
  - Output: "true" eller "false"
  - Eksempler:
    ```
    generate variabel1 = 0 if sysmiss(variabel2)
    ```

Tetthetsfunksjoner

- ibeta(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den kumulative beta-fordelingen med formparametrene arg1 og arg2, også kalt den regulariserte ufullstendige betafunksjonen eller den ufullstendige betafunksjonsratioen (ibeta() = 0 dersom arg3 < 0, ibeta() = 1 dersom arg3 > 1)
  - arg1: Verdier mellom 1e-10 og 1e+17
  - arg2: Verdier mellom 1e-10 og 1e+17
  - arg3: Verdier mellom -8e+307 og 8e+307 (relevante verdier: 0 <= arg3 <= 1)
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- betaden(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra sannsynlighetstettheten til beta-fordelingen med formparametrene arg1 og arg2 (betaden() = 0 dersom arg3 < 0 eller arg3 > 1)
  - arg1: Verdier mellom 1e-323 og 8e+307
  - arg2: Verdier mellom 1e-323 og 8e+307
  - arg3: Verdier mellom -8e+307 og 8e+307 (relevante verdier: 0 <= arg3 <= 1)
  - Output: Verdier mellom 0 og 8e+307
  - Eksempler:

- ibetatail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den omvendte kumulative beta-fordelingen med formparametrene arg1 og arg2, også kalt den komplementære ufullstendige betafunksjonen (ibetatail() = 1 dersom arg3 < 0, ibetatail() = 0 dersom arg3 > 1)
  - arg1: Verdier mellom 1e-10 og 1e+17
  - arg2: Verdier mellom 1e-10 og 1e+17
  - arg3: Verdier mellom -8e+307 og 8e+307 (relevante verdier: 0 <= arg3 <= 1)
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- invibeta(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den inverse kumulative beta-fordelingen med formparametrene arg1 og arg2
  - arg1: Verdier mellom 1e-10 og 1e+17
  - arg2: Verdier mellom 1e-10 og 1e+17
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- invibetatail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den inverse omvendte kumulative beta-fordelingen med formparametrene arg1 og arg2 (ibetatail(a,b,x) = p => invibetatail(a,b,p) = x)
  - arg1: Verdier mellom 1e-10 og 1e+17
  - arg2: Verdier mellom 1e-10 og 1e+17
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- binomial(arg1,arg2,arg3)
  - Beskrivelse: Returnerer sannsynligheten for å observere floor(arg2) eller færre suksesser i floor(arg1) forsøk der sannsynligheten for suksess i ett forsøk er arg3. binomial() = 0 dersom arg2 < 0. binomial() = 1 dersom arg2 > arg1
  - arg1: Verdier mellom 0 og 1e+17
  - arg2: Verdier mellom −8e+307 og 8e+307 (relevante verdier: 0 <= arg2 < arg1)
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- binomialp(arg1,arg2,arg3)
  - Beskrivelse: Returnerer sannsynligheten for å observere floor(arg2) suksesser i floor(arg1) forsøk der sannsynligheten for suksess i ett forsøk er arg3
  - arg1: Verdier mellom 1 og 1e+6
  - arg2: Verdier mellom 0 og arg1
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- binomialtail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer sannsynligheten for å observere floor(arg2) eller flere suksesser i floor(arg1) forsøk der sannsynligheten for suksess i ett forsøk er arg3. binomialtail() = 1 dersom arg2 < 0. binomialtail() = 0 dersom arg2 > arg1
  - arg1: Verdier mellom 0 og 1e+17
  - arg2: Verdier mellom −8e+307 og 8e+307 (relevante verdier: 0 <= arg2 < arg1)
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- chi2(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den kumulative kjikvadratfordelingen med arg1 frihetsgrader (chi2() = 0 dersom arg2 < 0)
  - arg1: Verdier mellom 2e–10 og 2e+17
  - arg2: Verdier mellom −8e+307 og 8e+307 (relevante verdier: arg2 >= 0)
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- chi2den(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra sannsynlighetstettheten til kjikvadratfordelingen med arg1 frihetsgrader (chi2den() = 0 dersom arg2 < 0)
  - arg1: Verdier mellom 2e–10 og 2e+17
  - arg2: Verdier mellom −8e+307 og 8e+307 (relevante verdier: arg2 >= 0)
  - Output: Verdier mellom 0 og 8e+307
  - Eksempler:

- chi2tail(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den omvendte kumulative kjikvadratfordelingen med arg1 frihetsgrader (chi2tail() = 1 dersom arg2 < 0). chi2tail() = 1 − chi2()
  - arg1: Verdier mellom 2e–10 og 2e+17
  - arg2: Verdier mellom −8e+307 og 8e+307 (relevante verdier: arg2 >= 0)
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- invchi2(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den inverse av den kumulative kjikvadratfordelingen med arg1 frihetsgrader (chi2(arg1,arg2) = p => invchi2(arg1,p) = arg2)
  - arg1: Verdier mellom 2e–10 og 2e+17
  - arg2: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 8e+307
  - Eksempler:

- invchi2tail(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den inverse av den omvendte kumulative kjikvadratfordelingen med arg1 frihetsgrader (chi2tail(arg1,arg2) = p => invchi2tail(arg1,p) = arg2)
  - arg1: Verdier mellom 2e–10 og 2e+17
  - arg2: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 8e+307
  - Eksempler:

- nchi2(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den kumulative ikke-sentrerte kjikvadratfordelingen med arg1 frihetsgrader og sentreringsparameter arg2 (noncentral parameter), der arg3 er kjikvadratverdien (nchi2() = 0 dersom arg3 < 0)
  - arg1: Verdier mellom 2e–10 og 1e+6
  - arg2: Verdier mellom 0 og 10000
  - arg3: Verdier mellom −8e+307 og 8e+307 (relevante verdier: arg3 >= 0)
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- nchi2den(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra sannsynlighetstettheten til den ikke-sentrerte kjikvadratfordelingen med arg1 frihetsgrader og sentreringsparameter arg2 (noncentral parameter), der arg3 er kjikvadratverdien (nchi2den() = 0 dersom arg3 < 0)
  - arg1: Verdier mellom 2e–10 og 1e+6
  - arg2: Verdier mellom 0 og 10000
  - arg3: Verdier mellom −8e+307 og 8e+307 (relevante verdier: arg3 >= 0)
  - Output: Verdier mellom 0 og 8e+307
  - Eksempler:

- nchi2tail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den omvendte kumulative ikke-sentrerte kjikvadratfordelingen med arg1 frihetsgrader og sentreringsparameter arg2 (noncentral parameter), der arg3 er kjikvadratverdien (nchi2tail() = 1 dersom arg3 < 0)
  - arg1: Verdier mellom 2e–10 og 1e+6
  - arg2: Verdier mellom 0 og 10000
  - arg3: Verdier mellom −8e+307 og 8e+307 (relevante verdier: arg3 >= 0)
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- t(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den kumulative Student's t-fordelingen med arg1 frihetsgrader
  - arg1: Verdier mellom 2e-10 og 2e+17
  - arg2: Verdier mellom −8e+307 og 8e+307
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- tden(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra sannsynlighetstettheten til Student's t-fordelingen med arg1 frihetsgrader
  - arg1: Verdier mellom 1e–323 og 8e+307
  - arg2: Verdier mellom −8e+307 og 8e+307
  - Output: Verdier mellom 0 og 0.39894 . . .
  - Eksempler:

- ttail(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den omvendte kumulative Student's t-fordelingen med arg1 frihetsgrader
  - arg1: Verdier mellom 2e–10 og 2e+17
  - arg2: Verdier mellom −8e+307 og 8e+307
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- invt(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den inverse kumulative Student's t-fordelingen med arg1 frihetsgrader (t(arg1,arg2) = p => invt(arg1,p) = arg2)
  - arg1: Verdier mellom 2e–10 og 2e+17
  - arg2: Verdier mellom 0 og 1
  - Output: Verdier mellom −8e+307 og 8e+307
  - Eksempler:

- invttail(arg1,arg2)
  - Beskrivelse: Returnerer en verdi fra den inverse omvendte kumulative Student's t-fordelingen med arg1 frihetsgrader (ttail(arg1,arg2) = p => invttail(arg1,p) = arg2)
  - arg1: Verdier mellom 2e–10 og 2e+17
  - arg2: Verdier mellom 0 og 1
  - Output: Verdier mellom −8e+307 og 8e+307
  - Eksempler:

- nt(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den kumulative ikke-sentrerte Student's t-fordelingen med arg1 frihetsgrader og sentreringsparameter arg2 (nt(arg1,0,arg3) = t(arg1,arg3))
  - arg1: Verdier mellom 1e–100 og 1e+10
  - arg2: Verdier mellom -1000 og 1000
  - arg3: Verdier mellom −8e+307 og 8e+307
  - Output: Verdier mellom 0 og 1

- Eksempler:

- ntden(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra sannsynlighetstettheten til den ikke-sentrerte Student's t-fordelingen med arg1 frihetsgrader og sentreringsparameter arg2
  - arg1: Verdier mellom 1e–100 og 1e+10
  - arg2: Verdier mellom -1000 og 1000
  - arg3: Verdier mellom −8e+307 og 8e+307
  - Output: Verdier mellom 0 og 0.39894 . . .
  - Eksempler:

- nttail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den omvente kumulative ikke-sentrerte Student's t-fordelingen med arg1 frihetsgrader og sentreringsparameter arg2
  - arg1: Verdier mellom 1e–100 og 1e+10
  - arg2: Verdier mellom -1000 og 1000
  - arg3: Verdier mellom −8e+307 og 8e+307
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- invnttail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den inverse omvendte kumulative ikke-sentrerte Student's t-fordelingen med arg1 frihetsgrader og sentreringsparameter arg2 (nttail(arg1,arg2,arg3) = p => invnttail(arg1,arg2,p) = arg3)
  - arg1: Verdier mellom 1 og 1e+6
  - arg2: Verdier mellom -1000 og 1000
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier mellom -8e+10 og 8e+10
  - Eksempler:

- F(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den kumulative F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner (F() = 0 dersom arg3 < 0)
  - arg1: Verdier mellom 2e–10 og 2e+17
  - arg2: Verdier mellom 2e–10 og 2e+17
  - arg3: Verdier mellom −8e+307 og 8e+307 (relevante verdier: arg3 >= 0)
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- Fden(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra sannsynlighetstettheten til F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner (Fden() = 0 dersom arg3 < 0)
  - arg1: Verdier mellom 1e–323 og 8e+307
  - arg2: Verdier mellom 1e–323 og 8e+307
  - arg3: Verdier mellom -8e+307 og 8e+307 (relevante verdier: arg3 >= 0)
  - Output: Verdier mellom 0 og 8e+307
  - Eksempler:

- Ftail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den omvendte kumulative F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner (Ftail() = 1 - F(), Ftail() = 1 dersom arg3 < 0)
  - arg1: Verdier mellom 2e–10 og 2e+17
  - arg2: Verdier mellom 2e–10 og 2e+17
  - arg3: Verdier mellom −8e+307 og 8e+307 (relevante verdier: arg3 >= 0)
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- invF(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den inverse kumulative F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner (F(arg1,arg2,arg3) = p => invF(arg1,arg2,p) = arg3)
  - arg1: Verdier mellom 2e–10 og 2e+17
  - arg2: Verdier mellom 2e–10 og 2e+17
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 8e+307
  - Eksempler:

- invFtail(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra den inverse omvendte kumulative F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner (Ftail(arg1,arg2,arg3) = p => invFtail(arg1,arg2,p) = arg3)
  - arg1: Verdier mellom 2e–10 og 2e+17
  - arg2: Verdier mellom 2e–10 og 2e+17
  - arg3: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 8e+307
  - Eksempler:

- nF(arg1,arg2,arg3,arg4)
  - Beskrivelse: Returnerer en verdi fra den kumulative ikke-sentrerte F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner, og sentreringsparameter arg3 (nF(arg1,arg2,0,arg4) = F(arg1,arg2,arg4), nF() = 0 dersom arg4 < 0)
  - arg1: Verdier mellom 2e–10 og 1e+8
  - arg2: Verdier mellom 2e–10 og 1e+8
  - arg3: Verdier mellom 0 og 10000
  - arg4: Verdier mellom -8e+307 og 8e+307 (relevante verdier: arg4 >= 0)
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- nFden(arg1,arg2,arg3,arg4)
  - Beskrivelse: Returnerer en verdi fra sannsynlighetstettheten til den ikke-sentrerte F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner, og sentreringsparameter arg3 (nFden(arg1,arg2,0,arg4) = Fden(arg1,arg2,arg4), nFden() = 0 dersom arg4 < 0)
  - arg1: Verdier mellom 1e–323 og 8e+307
  - arg2: Verdier mellom 1e–323 og 8e+307
  - arg3: Verdier mellom 0 og 1000
  - arg4: Verdier mellom -8e+307 og 8e+307 (relevante verdier: arg4 >= 0)
  - Output: Verdier mellom 0 og 8e+307
  - Eksempler:

- nFtail(arg1,arg2,arg3,arg4)
  - Beskrivelse: Returnerer en verdi fra den omvendte kumulative ikke-sentrerte F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner, og sentreringsparameter arg3 (nFtail() = 1 dersom arg4 < 0)
  - arg1: Verdier mellom 1e–323 og 8e+307
  - arg2: Verdier mellom 1e–323 og 8e+307
  - arg3: Verdier mellom 0 og 1000
  - arg4: Verdier mellom -8e+307 og 8e+307 (relevante verdier: arg4 >= 0)
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- invnFtail(arg1,arg2,arg3,arg4)
  - Beskrivelse: Returnerer en verdi fra den inverse omvendte kumulative ikke-sentrerte F-fordelingen med arg1 og arg2 frihetsgrader i henholdsvis teller og nevner, og sentreringsparameter arg3 (nFtail(arg1,arg2,arg3,arg4) = p => invnFtail(arg1,arg2,arg3,p) = arg4)
  - arg1: Verdier mellom 1e–323 og 8e+307
  - arg2: Verdier mellom 1e–323 og 8e+307
  - arg3: Verdier mellom 0 og 1000
  - arg4: Verdier mellom 0 og 1
  - Output: Verdier mellom 0 og 8e+307
  - Eksempler:

- normal(arg1)
  - Beskrivelse: Returnerer en verdi fra den kumulative standardiserte normalfordelingen
  - arg1: Verdier mellom -8e+307 og 8e+307
  - Output: Verdier mellom 0 og 1
  - Eksempler:

- normalden(arg1,arg2,arg3)
  - Beskrivelse: Returnerer en verdi fra normalfordelingen med snittverdi arg2 og standardavvik arg3
  - arg1: Verdier mellom -8e+307 og 8e+307
  - arg2: Verdier mellom -8e+307 og 8e+307
  - arg3: Verdier mellom 1e-308 og 8e+307
  - Output: Verdier mellom 0 og 8e+307
  - Eksempler:

Datofunksjoner

Dateringer i microdata.no benytter et datoformat som angir antall dager målt fra 01.01.1970. Fordelen med dette er at en enkelt kan gjøre beregninger på antall dager mellom to tidspunkter, og for eksempel beregne varighet i en tilstand (varighet = stoppdato - startdato).

Datofunksjonene listet opp nedenfor kan brukes til å gjøre om fra innebygd datoformat til mer intuitive verdier, som for eksempel årstall, måned, dag i uken osv.

- date(arg1, arg2, arg3)
  - Beskrivelse: Gjør om fra angitt dato til innebygd datoformat (antall dager fra 01.01.1970)
  - arg1: Årstall (4-sifret)
  - arg2: Måned (1-12)
  - arg3: Dag (1-31)
  - Output: Innebygd datoformat (antall dager fra 01.01.1970)
  - Eksempler:
    - date(2015,12,31) = 16800
    - date(1970,1,1) = 0
    - date(1967,5,27) = -950

- year(arg1)
  - Beskrivelse: Henter ut årstall fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Årstall tilsvarende datoverdi
  - Eksempler:
    - year(16800) = 2015
    - year(0) = 1970
    - year(-950) = 1967

- month(arg1)
  - Beskrivelse: Henter ut månedsverdi fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Månedsverdi tilsvarende datoverdi (1-12)
  - Eksempler:
    - month(16800) = 12
    - month(0) = 1
    - month(-950) = 5

- day(arg1)
  - Beskrivelse: Henter ut verdi for dag i måneden fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Dag i måneden tilsvarende datoverdi (1-31)
  - Eksempler:
    - day(16800) = 31
    - day(0) = 1
    - day(-950) = 27

- dow(arg1)
  - Beskrivelse: Henter ut verdi for dag i uken fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Dag i uken tilsvarende datoverdi (1-7) (1 = mandag, 2 = tirsdag etc)
  - Eksempler:
    - dow(16800) = 4
    - dow(0) = 4
    - dow(-950) = 6

- doy(arg1)
  - Beskrivelse: Henter ut verdi for dag i året fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Dag i året tilsvarende datoverdi (1-366)
  - Eksempler:
    - doy(16800) = 365
    - doy(0) = 1
    - doy(-950) = 147

- week(arg1)
  - Beskrivelse: Henter ut ukenummer fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
  - arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
  - Output: Ukenummer tilsvarende datoverdi (1-53)
  - Eksempler:
    - week(16800) = 53

        - ❏ week(0) = 1
        - ❏ week(-950) = 21

- ❏ quarter(arg1)
    - ❏ Beskrivelse: Henter ut kvartalstall fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
    - ❏ arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
    - ❏ Output: Kvartalstall tilsvarende datoverdi (1-4)
    - ❏ Eksempler:
        - ❏ quarter(16800) = 4
        - ❏ quarter(0) = 1
        - ❏ quarter(-950) = 2

- ❏ halfyear(arg1)
    - ❏ Beskrivelse: Henter ut halvårstall fra datoverdi. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
    - ❏ arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
    - ❏ Output: Halvårstall tilsvarende datoverdi (1-2)
    - ❏ Eksempler:
        - ❏ halfyear(16800) = 2
        - ❏ halfyear(0) = 1
        - ❏ halfyear(-950) = 1

- ❏ isoformatdate(arg1)
    - ❏ Beskrivelse: Konverterer fra datoverdi til formatet YYYY-MM-DD. Kan brukes på start- og stoppvariabler for å konvertere fra innebygd datoverdi-format (1970-01-01 = 0)
    - ❏ arg1: Dateringsvariabel med datoverdier (START@<*variabelnavn*> eller STOP@<*variabelnavn*>)
    - ❏ Output: Datering på formatet YYYY-MM-DD (string-verdier)
    - ❏ Eksempler:
        - ❏ isoformatdate(16800) = '2015-12-31'
        - ❏ isoformatdate(0) = '1970-01-01'
        - ❏ isoformatdate(-950) = '1967-05-27'

Vedlegg C: Konfidensialitet i microdata.no

Bakgrunn

Lov om offisiell statistikk og Statistisk sentralbyrå (LOV-1989-06-16-54) § 2-5 (bruk av opplysninger) ledd (1) sier at «Dersom opplysninger utleveres, skal taushetsplikt etter § 2-4 også gjelde mottageren av opplysningene». Slike data kan kun utleveres til forskere i godkjente forskningsinstitusjoner eller forskere som har finansiering fra Norges forskningsråd. Det stilles derfor strenge krav ved utlevering av data til forskning og søknad om tilgang til mikrodata for forskning er en lang prosess. Kriteriene for å søke om tilgang til registerdata for forskning kan du finne på SSBs sider om data til forskning.

Analysesystemet microdata.no er utviklet for å gjøre det mulig å få tilgang til mikrodata fra registre uten å måtte gå gjennom den omstendelige søknadsprosessen det er for å få data utlevert. Men det er en betingelse for en slik forenkling at sikkerheten og konfidensialiteten til mikrodataene er like godt ivaretatt som ved utlevering, helst bedre. Det har derfor fra begynnelsen vært et eksplisitt krav at brukerne ikke skal kunne se mikrodata eller på annen måte være i stand til å avsløre informasjon om enkeltpersoner. Når SSB publiserer offisiell statistikk er det aggregerte data. Likevel må SSB passe på ved ulike typer tiltak at det ikke er mulig å føre informasjon tilbake til enkeltpersoner eller andre typer statistiske enheter som statistikken dreier seg om.

De resultatene som microdata.no produserer for sine brukere, tabeller eller analyser, er i likhet med SSBs statistikk aggregerte data. Men uten begrensninger vil en bruker av microdata.no lett kunne produsere tabeller og andre typer statistiske resultater som SSB ikke ville kunne publisere. For å forhindre at det skjer er det innført flere typer tiltak som skal begrense mulighetene for å kunne avsløre informasjon som skal være konfidensiell.

Dette vedlegget vil beskrive de tiltakene som er implementert for å ivareta konfidensialitet i microdata.no. Tiltakene er basert på scenarier for hvordan konfidensialiteten i microdata.no kan angripes eller utilsiktet komme i fare. Disse scenariene vil ikke bli beskrevet. Det vil bli lagt vekt på det som er nødvendig for at brukeren skal kunne forstå tiltakene og forholde seg til de statistiske resultatene på riktig måte.

De tiltakene som er beskrevet nedenfor er de som er implementert så langt. Det vil kunne komme flere tiltak etter hvert eller også justeringer av de tiltakene som er beskrevet nedenfor. Dette vedlegget vil bli oppdatert når det kommer endringer.

Tiltak 1: Minste populasjonsstørrelse

Det er ikke tillatt å definere undersøkelsespopulasjoner med færre enn 1000 personer. Forsøk på å definere slike populasjoner vil bli møtt med en melding av typen

```
Problem på linje 3:
Stoppet av avsløringskontroll. For få enheter i måldatasettet
```

Tiltak 2: Winsorisering

Winsorisering er en teknikk som ofte brukes i analyser for å hindre at ekstreme observasjoner skal få for stor innvirkning på analyseresultatene. Teknikken anvendes på alle numeriske variabler og består i å kutte av fordelingen i begge ender ved bestemte prosentiler. Vi benytter 2% winsorisering som betyr at de 1% høyeste verdiene settes til 99-prosentilen og de 1% laveste verdiene settes til 1-prosentilen. Dette skjer ved import av variabler og i forhold til fordelingen i den populasjonen som brukeren har filtrert ut som sin studiepopulasjon.

Siden fordelingene til mange numeriske statistiske variabler er skjevfordelte, typisk med lange haler i øvre ende (eks. inntekter og formuer) vil winsorisering påvirke gjennomsnitts- og standardavvik. Begge typer statistikker vil typisk bli estimert for lave. På den andre siden vil medianer, kvartiler og andre prosentiler ikke bli påvirket.

Winsoriseringen vil automatisk finne sted ved kommandoene

```
import

keep if

drop if
```

og vil alltid være relatert til fordelingen i den populasjonen som er definert ved disse kommandoene uansett i hvilken rekkefølge de utføres.

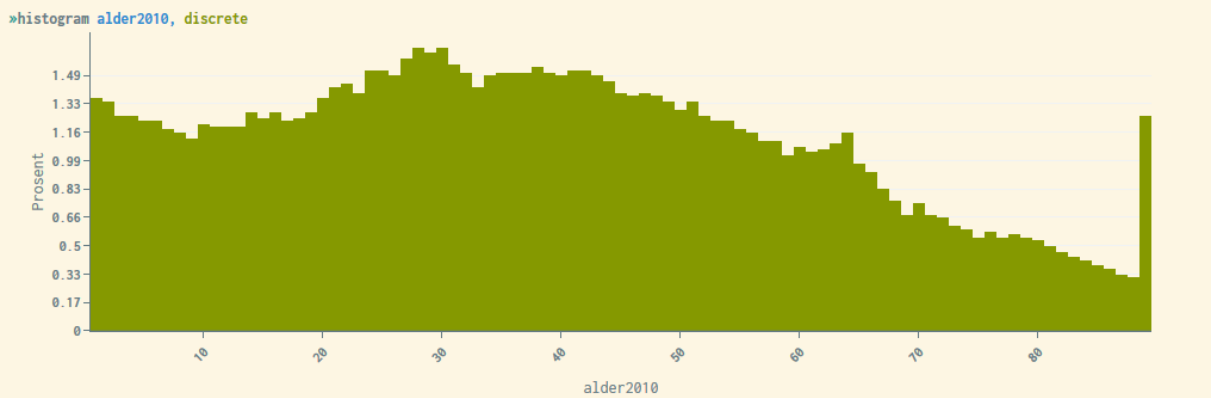
Eksempel: Betrakt følgende skript hvor målet er å lage deskriptive statistikker og histogram for aldersfordelingen i befolkningen per 2010.

```
import BEFOLKNING_FOEDSELS_AAR_MND as faarmnd
generate faar = floor(faarmnd/100)
drop faarmnd
generate alder2010 = 2010 - faar
summarize alder2010
histogram alder2010, discrete
```

Resultatet vil se slik ut:

summarize alder2010

Variabel	mean	std	count	min	max	25%	50%	75%
alder2010	38.7467	22.681	255743	1	89	21	37	55



Alle som er eldre enn 89 år vil bli satt til 89 år og 0-åringer vil bli satt til 1 år. Dette er årsaken til den store søylen til høyre i histogrammet. Vi er oppmerksomme på at winsoriseringen kan skape problemer ved studier av de aller eldste. Det samme gjelder studier av andre grupper som er definert ved å tilhøre halen i fordelingen til en numerisk variabel.

Winsorisering påvirker alle statistikker, analyser og grafiske plott og hindrer at de mest ekstreme verdiene blir synlige eller influerer på analysene.

Tiltak 3: Støylegging

Alle opptellinger av antall enheter i en populasjon eller delpopulasjon ved kommandoen **tabulate** eller i **summarize** er støylagte. Summer av numeriske statistikkvariabler knyttet til enhetene i en tabellcelle, for eksempel inntekter, vil bli justert proporsjonalt med støyleggingen slik at gjennomsnittstall er upåvirket. Der hvor støyleggingen resulterer i at antallet enheter bak summen blir 0, settes summen til 0 og gjennomsnittet, som da blir 0/0 blir satt til NaN.

La n være det originale antallet uten støylegging (for eksempel i en tabellcelle), og X er støyen (stokastisk, helt tall). Da vil microdata.no vise det støylagte tallet

$$Y = X + n$$

Støyleggingen er definert ut fra statistiske fordelinger med følgende krav:

1. Minste positive tallet som kan vises i opptellinger, Y, skal være 5. Tallene 1-4 skal ikke kunne vises i tabellene. Men $Y = 0$ vil kunne forekomme
2. Ingen opptellinger (antall) skal støylegges med mer enn ±5, dvs. $-5 \leq X \leq 5$.
3. Det skal ikke være mulig å gjenta samme opptelling flere ganger innen rammen av samme populasjon og få ulike resultater. Støyleggingen skal i den forstand være *konstant*.
4. Det må ikke være mulig å skille mellom virkelige nuller og nuller som er et resultat av støylegging.
5. Støyen er stokastisk med forventning 0, $E(X) = 0$.
6. Under betingelsen 1-3 og 5 skal støyfordelingen som genererer X være en maksimum entropifordeling, det vil si at hvis $p(x) = P(X = x), -5 \leq x \leq 5$, så skal $p(x)$ maksimere

$$\mathcal{E}(p|n) = -\sum_{x=-5}^{5} p(x|n)\log\big(p(x|n)\big) = -E(\log p(X|n))$$

Maksimum entropifordelingen er i en viss forstand den støyfordeling som fjerner mest informasjon om den originale verdien n under de gitte betingelsene.

For å støtte brukerens tolkning av den usikkerheten som støyleggingen medfører, presenterer vi de eksakte støyfordelingene som betingelsene 1-3 og 5-6 genererer for ulike verdier av n i tabell C1. På grunnlag av tabell C1 utleder vi *konfidensfordelinger* for n gitt Y i tabellene C2 og C3. En konfidensfordeling er selv en stokastisk størrelse som avhenger av den observerte verdien av Y og ikke en sannsynlighetsfordeling for n. (Og her har *konfidens* ingenting å gjøre med konfidensialitet.)

$p(x)$	$n=0$	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$	$n=8$	$n=9$	$n>=10$
$x=-5$	0,0000	0,0000	0,0000	0,0000	0,0000	0,2987	0,0000	0,0000	0,0000	0,0000	0,0909
$x=-4$	0,0000	0,0000	0,0000	0,0000	0,3988	0,0000	0,0000	0,0000	0,0000	0,1296	0,0909
$x=-3$	0,0000	0,0000	0,0000	0,5175	0,0000	0,0000	0,0000	0,0000	0,1908	0,1219	0,0909
$x=-2$	0,0000	0,0000	0,6555	0,0000	0,0000	0,0000	0,0000	0,2940	0,1634	0,1147	0,0909
$x=-1$	0,0000	0,8149	0,0000	0,0000	0,0000	0,0000	0,4880	0,2145	0,1400	0,1079	0,0909
$x=0$	1,0000	0,0000	0,0000	0,0000	0,0000	0,1573	0,2522	0,1565	0,1199	0,1015	0,0909
$x=1$	0,0000	0,0000	0,0000	0,0000	0,1657	0,1383	0,1303	0,1142	0,1027	0,0955	0,0909
$x=2$	0,0000	0,0000	0,0000	0,1646	0,1390	0,1217	0,0674	0,0833	0,0880	0,0899	0,0909
$x=3$	0,0000	0,0000	0,1499	0,1309	0,1166	0,1070	0,0348	0,0608	0,0754	0,0846	0,0909
$x=4$	0,0000	0,1108	0,1116	0,1041	0,0978	0,0941	0,0180	0,0444	0,0645	0,0796	0,0909
$x=5$	0,0000	0,0743	0,0830	0,0828	0,0821	0,0828	0,0093	0,0324	0,0553	0,0748	0,0909
$\mathrm{Sum}(p(x))$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
$E(X\|n)$	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$Var(X\|n)$	0,0000	4,4460	7,8320	10,2306	11,7688	12,6325	1,7215	3,9038	6,0585	8,0973	10,0000
$\mathcal{E}(p\|n)$	0,0000	0,6038	1,0126	1,3459	1,6219	1,8497	1,3775	1,8547	2,1210	2,2874	2,3979

Tabell C1 Sannsynlighetsfordelinger for støy for ulike verdier av n.

Kombinasjoner der $Y = n + X < 0$.

Kombibasjoner der $1 \leq Y = n + X \leq 5$

Ved å summere de tallene i tabell C1 som gir samme verdi for y= $x + n$, og dividere med summen (standardisere til sum lik 1) kan en utlede tabell C2. Tabell C2 angir det vi kan kalle *konfidensgrader* for hver verdi av n gitt den verdi y som microdata.no har returnert for Y. Siden vi her ser på n som et fast tall og ikke en stokastisk variabel, er konfidensgradene $f(n|y)$ *ikke* sannsynligheter i vanlig forstand, selv om de summerer seg til 1.

For $n > 10$ vil støyfordelingen være flat med $p(x) = \frac{1}{11} \approx 0{,}0909$ for alle $x\epsilon\{-5, \dots, 5\}$ som for $n = 10$.

Vær oppmerksom på at i tabeller støylegges indre celler og marginalceller uavhengig av hverandre. ***Støylagte tabeller blir derfor ikke additive****. Støyvariansen på marginalceller blir den samme som på indre celler, og mindre enn for summen av de indre cellene som de uten støylegging skal være en sum av.*

$cf(n\|Y=y)$	$Y=0$	$Y=5$	$Y=6$	$Y=7$	$Y=8$	$Y=9$	$Y=10$	$Y=11$	$Y=12$	$Y=13$	$Y=14$	$Y=15$
$n=0$	**0,2713**	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=1$	**0,2211**	**0,0571**	**0,0486**	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=2$	**0,1779**	**0,0772**	**0,0730**	**0,0670**	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=3$	**0,1404**	**0,0848**	**0,0857**	**0,0840**	**0,0781**	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=4$	**0,1082**	**0,0853**	**0,0910**	**0,0941**	**0,0922**	**0,0861**	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=5$	**0,0810**	**0,0810**	**0,0905**	**0,0982**	**0,1009**	**0,0988**	**0,0930**	0,0000	0,0000	0,0000	0,0000	0,0000
$n=6$	0,0000	**0,2513**	**0,1650**	**0,1051**	**0,0635**	**0,0365**	**0,0202**	**0,0109**	0,0000	0,0000	0,0000	0,0000
$n=7$	0,0000	**0,1514**	**0,1404**	**0,1262**	**0,1077**	**0,0874**	**0,0683**	**0,0519**	**0,0356**	0,0000	0,0000	0,0000
$n=8$	0,0000	**0,0983**	**0,1070**	**0,1129**	**0,1130**	**0,1078**	**0,0988**	**0,0881**	**0,0710**	**0,0580**	0,0000	0,0000
$n=9$	0,0000	**0,0667**	**0,0798**	**0,0925**	**0,1017**	**0,1065**	**0,1073**	**0,1051**	**0,0930**	**0,0835**	**0,0761**	0,0000
$n=10$	0,0000	**0,0468**	**0,0595**	**0,0733**	**0,0857**	**0,0954**	**0,1021**	**0,1063**	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=11$	0,0000	0,0000	**0,0595**	**0,0733**	**0,0857**	**0,0954**	**0,1021**	**0,1063**	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=12$	0,0000	0,0000	0,0000	**0,0733**	**0,0857**	**0,0954**	**0,1021**	**0,1063**	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=13$	0,0000	0,0000	0,0000	0,0000	**0,0857**	**0,0954**	**0,1021**	**0,1063**	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=14$	0,0000	0,0000	0,0000	0,0000	0,0000	**0,0954**	**0,1021**	**0,1063**	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=15$	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	**0,1021**	**0,1063**	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=16$	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	**0,1063**	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=17$	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	**0,1000**	**0,0954**	**0,0924**	**0,0909**
$n=18$	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	**0,0954**	**0,0924**	**0,0909**
$n=19$	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	**0,0924**	**0,0909**
$n=20$	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	**0,0909**
Sum	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000

Tabell C2 Konfidensfordeling for n for ulike verdier av y. Positive konfidensgrader er gitt fet skrift.

$CD(n|y)$ i tabell C3 er kumulative aggregeringer av konfidensgradene i tabell C2 definert ved

$$CD(n|y) = \sum_{j=0}^{n} cd(j|y)$$

$CD(n\|y)$	$Y=0$	$Y=5$	$Y=6$	$Y=7$	$Y=8$	$Y=9$	$Y=10$	$Y=11$	$Y=12$	$Y=13$	$Y=14$	$Y=15$
$n=0$	0,2713	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=1$	0,4924	0,0571	0,0486	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=2$	0,6703	0,1343	0,1217	0,0670	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=3$	0,8107	0,2190	0,2073	0,1510	0,0781	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=4$	0,9190	0,3044	0,2983	0,2450	0,1703	0,0861	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
$n=5$	1,0000	0,3854	0,3888	0,3432	0,2712	0,1849	0,0930	0,0000	0,0000	0,0000	0,0000	0,0000
$n=6$	1,0000	0,6367	0,5539	0,4483	0,3347	0,2214	0,1132	0,0109	0,0000	0,0000	0,0000	0,0000
$n=7$	1,0000	0,7882	0,6943	0,5746	0,4424	0,3088	0,1815	0,0627	0,0356	0,0000	0,0000	0,0000
$n=8$	1,0000	0,8864	0,8012	0,6875	0,5554	0,4166	0,2802	0,1508	0,1066	0,0580	0,0000	0,0000
$n=9$	1,0000	0,9532	0,8810	0,7800	0,6572	0,5231	0,3875	0,2559	0,1997	0,1415	0,0761	0,0000
$n=10$	1,0000	1,0000	0,9405	0,8533	0,7429	0,6185	0,4896	0,3622	0,2997	0,2369	0,1685	0,0909
$n=11$	1,0000	1,0000	1,0000	0,9267	0,8286	0,7139	0,5917	0,4685	0,3998	0,3323	0,2609	0,1818
$n=12$	1,0000	1,0000	1,0000	1,0000	0,9143	0,8092	0,6938	0,5748	0,4998	0,4277	0,3532	0,2727
$n=13$	1,0000	1,0000	1,0000	1,0000	1,0000	0,9046	0,7958	0,6811	0,5998	0,5230	0,4456	0,3636
$n=14$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,8979	0,7874	0,6999	0,6184	0,5380	0,4545
$n=15$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,8937	0,7999	0,7138	0,6304	0,5455
$n=16$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9000	0,8092	0,7228	0,6364
$n=17$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9046	0,8152	0,7273
$n=18$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9076	0,8182
$n=19$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	0,9091
$n=20$	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000

Tabell C3 Kumulativ konfidensfordeling for ulike verdier av y.

La $\{5-9\}$ betegne verdimengden $\{5,6,7,8,9\}$. Konfidensgraden til verdimengden $\{5-9\}$ hvis vi observerer for eksempel $Y=7$, blir da

$$cd(\{5-9\}|Y=7) = CD(9|Y=7) - CD(4|Y=7) = 0{,}7800 - 0{,}2450 = 0{,}5350$$

Dette tilsvarer et konfidensintervall. Merk igjen at det ikke er det samme som sannsynligheter siden n ikke er stokastisk.

Hvis $Y > 15$ vil konfidensfordelingen $cd(y|n)$ i tabell C2 være flat lik $\frac{1}{11} \approx 0{,}0909$ for alle verdier av n fra $Y - 5$ til $Y + 5$ som for $Y = 15$. La a og b være hele tall med $b > a$. Anta at $Y = y \geq 15$. Da blir

$$cd(\{a - b\}|y) = CD(b|y) - CD(a|y) = (\min(b, y + 5) - \max(a, y - 5))/11$$

Eksempel: La $a = 37, b = 44$ og $y = 39$. Da blir

$$cd(\{37 - 44\}) = \frac{\min(44,44) - \max(37,34)}{11} = \frac{44 - 37}{11} = \frac{7}{11} \approx 0{,}6364.$$

Ved aggregering av numeriske størrelser, for eksempel i tabellceller, vil summer bli justert i forhold til støyleggingen.

La Z_i være verdien av en numerisk variabel (for eksempel en inntektsvariabel) på person nr. i og T_c den originale summen av denne variabelen i en celle c med n_c personer, altså

$$T_c = \sum_{i \epsilon c} Z_i,$$

Anta n_c er støylagt til Y_c. Da vil T_c bli justert til

$$T_c^* = \frac{Y_c}{n_c} T_c$$

Denne justeringen kan være dramatisk for celler med få observasjoner men vil ha mindre betydning i celler med mange observasjoner. Det er også meningen. Merk også at

$$\bar{Z}_c = \frac{T_c^*}{Y_c} = \frac{T_c}{n_c}$$

Slik at gjennomsnittet ikke blir berørt, bortsett fra hvis $Y_c = 0$. Da settes $\bar{Z}_c = NaN$.

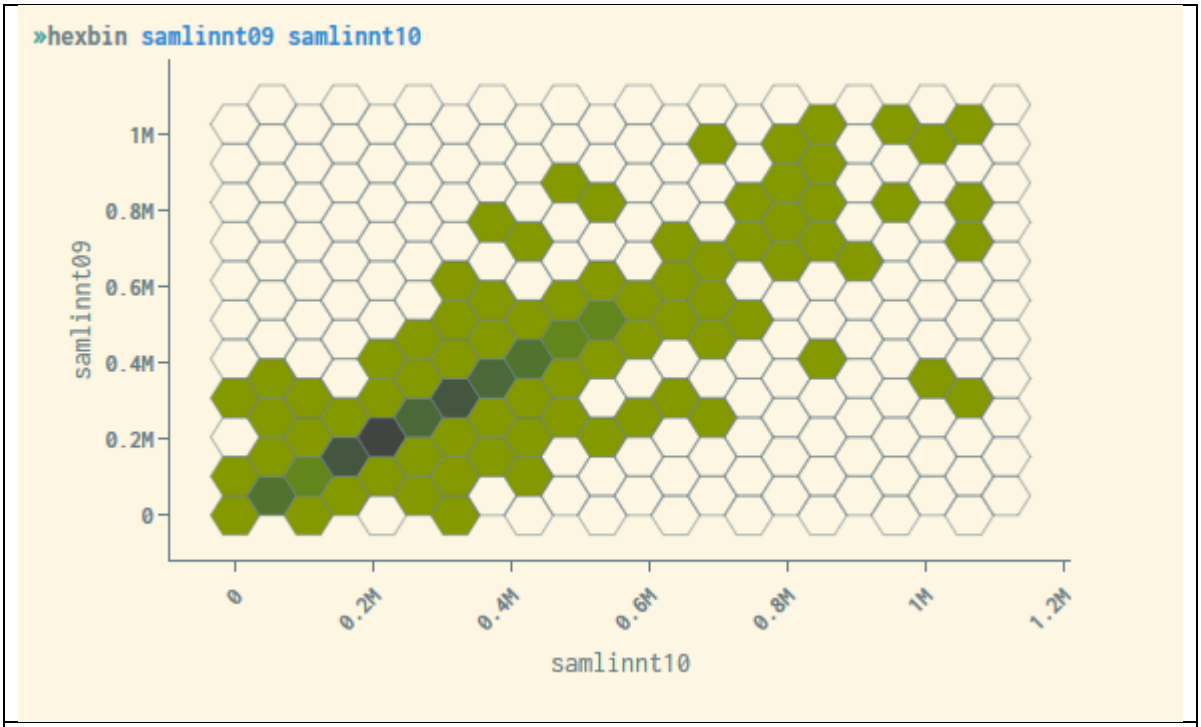
Hvis $Y_c \leq 9$ vil også standardavvik, median og kvartiler, som kan bestilles ved **summarize** opsjonen i **tabulate** settes til NaN.

Tiltak 4: Grafiske plott. Hexbinplott

Det er vanlig å bruke spredningsplott for å etablere et visuelt bilde av data eller også vise sammenheng mellom numeriske variabler. Slike plott kan være veldig avslørende, spesielt hvis det er få observasjoner i forhold til det grafiske området og for de punktene i plottet som ligger utenfor hovedmassen av punkter. Hvis vi for en enhet/person i populasjonen kjenner verdien på en av de variablene som spenner ut plottet kan vi ofte lese av verdien til den andre variabelen med for stor nøyaktighet.

For å hindre at dette kan skje har vi i microdata.no valgt å *glatte* slike plott med en glatteteknikk. For dette formålet har vi forsøksvis valgt å fokusere på en teknikk som kalles *hexbin-plott*. I et hexbin-plott deles det grafiske området inn i regulære sekskanter.

Eksempel på hexbin-plott laget i microdata.no:



Figur C1. Hexbinplott over samlet inntekt i 2009 mot 2010 for en liten populasjon.

I et hexbin-plott skaleres det grafiske området på grunnlag av de største og minste verdiene som forekommer for de variablene som plottes. De største og minste verdiene er påvirket av winsoriseringen omtalt i tiltak 2. Sekskantene gis en farge eller fargetone som angir et intervall for hvor mange enheter det er i dem, for eksempel 30-59, 60-89 osv. Intervallene for antall enheter/personer som hver fargetone representerer er like lange, og tilpasses automatisk basert på fordelingen av data.

Hexbin plot er under utprøving. I den versjonen som foreløpig er lagt ut er alle sekskanter hvor antall personer er færre enn 20% av det mest befolkede hexagonet blanket. Dette kriteriet vil bli justert så snart det er mulig å gi det prioritet. Vær oppmerksom på at winsoriseringen av de numeriske variablene som spenner ut plottet vil påvirke plottet.