STØTTET AF
Kvægafgiftsfonden

Klargøring af foderdata til estimering af genetiske parametre

Opdateret december 2020

Formålet med programmet er at klargøre foderdata til estimering af genetiske parametre. Individuelle foderindtag registreres vha. Allfeed systemet i GUDP-projektet FutureBeefCross. SAS-programmet og denne beskrivelse af programmet er foreløbige og kan blive opdaterede i takt med, at der genereres flere data.

Klargøringen af datasættet følger i høj grad de retningslinjer, der er angivet i Casey et al. (J. Anim. Sci. 2005. 83:969-982).

Dataudtræk

Data udtrækkes vha. SAS-programmet feeddataCDB.sas.

Programmet undersøger, om der er observationer, hvor besøgets sluttid er før besøgets starttid. I så fald skyldes det, at besøget strækker sig hen over midnat, og at datoen i sluttiden ikke er ændret. I disse tilfælde lægges der 1 døgn til sluttiden.

Observationer tidligere end 16. januar 2020 slettes fra datasættet.

Testmærker har ID større end eller lig med 99999900000. Testmærkerne fjernes fra programmet.

Klargøring af data

Data klargøres vha. SAS-programmet EditeringFoderdata.sas (se venligst appendiks 1).

Parvise observationer

Når en kalv eller en person læner sig op ad en foderkasse, så vil den ekstra vægt blive registreret som en opfyldning. Når kalven eller personen flytter sig igen, vil det blive registreret som et besøg. På den baggrund fjerner vi parvise observationer inden for hver foderkasse, hvor den aktuelle observation er et besøg, og den foregående observation er en opfyldning, og hvor forskellen mellem de to observationer er mindre end eller lig med 0,1 kg.

Der fjernes også parvise observationer inden for hver foderkasse, hvor den aktuelle observation er en opfyldning, og den foregående observation er et besøg, og hvor forskellen mellem de to observationer er mindre end eller lig med 0,1 kg.

Tidspunkt for besøgets afslutning

Variablen TIME er det bedste bud, vi har på besøgets varighed. Der er et tidsstempel på besøgets start men ikke på besøgets afslutning, da kalvene nogle gange holder en kort pause for så at begynde at æde igen.

Tidspunktet for besøgets afslutning var tidligere defineret på følgende måde: Ædetiden tæller så længe, der er aktivitet ved antennen. Der er indlagt en tid på 60 sekunder, som betyder, at når kalven forlader truget, afventer tidstælleren 60 sekunder, før ædetidsregistreringen stopper. Efter yderligere 120 sekunder afsluttes besøget – kommer kalven tilbage, inden tiden er løbet ud, fortsættes registreringen, og ædetiden starter igen, hvor den sidst stoppede.

Denne måde at registrere tidspunktet for besøgets afslutning forårsager i nogle tilfælde et tidsoverlap. Årsagen til tidsoverlappet af besøgene er, at dyret kan gå fra et igangværende besøg ved f.eks. foderkasse 5 og starte et nyt besøg ved f.eks. foderkasse 4. Foderkasse 5 vil i dette tilfælde stadig afvente, at kalven kommer tilbage inden for 120 sekunder, ellers afsluttes tidspunktet for ædebesøget. Tidspunktet for ædebesøget afsluttes også, hvis der registreres et andet øremærke ved foderkasse 5, før de 120 sekunder er gået.

Programmet ændrer tidspunktet for besøgets afslutning til starttidspunktet for det næste besøg af den samme kalv ved en anden foderkasse eller til starttidspunktet for det næste besøg af en anden kalv i den samme foderkasse afhængig af hvilken begivenhed, der indtræffer først. Tidspunktet for besøgets afslutning ændres ikke, hvis det oprindelige sluttidspunkt ligger før de ovenfor nævnte starttidspunkter. Dette editeringstrin er lavet for at undgå observationer, hvor den samme kalv optræder ved to forskellige foderkasser samtidig, eller at den samme foderkasse benyttes af to kalve samtidig.

Tabel 1. Eksempel på, at sluttidspunktet ændres.

ID	Station	VisitStartTime	VisitEndTime	NyVisitEndTime	TIME	Feed intake
1	1	01FEB2020:23:43:54	01FEB2020:23:59:45	01FEB20:23:47:48	841	1.17
1	2	01FEB2020:23:47:48	01FEB2020:23:49:09	01FEB20:23:49:09	140	0.15
1	3	01FEB2020:23:49:43	01FEB2020:23:57:22	01FEB20:23:55:27	409	0.40
1	1	01FEB2020:23:55:27	02FEB2020:00:01:57	02FEB20:00:01:57	412	0.50

Variablen TIME angiver, hvor lang tid kalven bruger på hvert ædebesøg. Hvis forskellen mellem tidspunktet for besøgets start og besøgets afslutning er mindre end TIME, så ændres TIME til forskellen på start- og sluttidspunktet.

Observationer for opfyldninger i besætningen der fodrer med fuldfoder

Når foderkasserne fyldes med foder, vises det som observationer, hvor ID er lig med -1, TIME er lige med 0, og variablen fyld er lig med 1. Dette gælder imidlertid kun for besætningerne, der fodrer med pelleteret foder. På den baggrund danner programmet observationer for opfyldninger i besætningen, der fodrer med fuldfoder, for at datastrukturen bliver ens for alle besætninger i datasættet.

Der fyldes foder i foderkasserne en til to gange om dagen i tidsrummet fra kl. 5 om morgenen til kl. halv to om eftermiddagen. De daglige opfyldninger defineres som de to største forskelle mellem fodermængden ved besøgets start og fodermængden ved det foregående besøgs afslutning. Hver opfyldning skal være større end eller lig med 5 kg og mindre end 100 kg.

Nye tidsvariable

Programmet danner variablene Leading time difference (LTD), der er defineret som starttidspunktet for det efterfølgende besøg minus sluttidspunktet for det nuværende besøg, og Following Time Difference (FTD), der er defineret som starttidspunktet for det nuværende besøg minus sluttidspunktet for det foregående besøg.

Observationer for de dage, hvor kalvene flyttes

Foderkasserne registrer kalvenes foderindtag pr. besøg, men vi vil gerne beregne kalvenes foderindtag pr. dag. På den baggrund er det nødvendigt at slette observationer fra de dage, hvor

kalvene kun har haft adgang til foderkasserne i en del af døgnets timer, og for de dage, hvor foderindtaget ikke er blevet registreret i alle døgnets timer. Programmet sletter derfor observationer for den første og den sidste dag, hvor kalvene får registreret foderindtag samt for de dage, hvor kalvene flyttes fra en sti til en anden. Derudover slettes observationer for den sidste dag før en periode, hvor foderindtaget ikke registreres, og for den første dag efter en periode, hvor foderindtaget ikke registreres.

Observationer for de dage, hvor foderkasserne eller overførslen af data ikke virker

Hvis foderkasserne genererer mange observationer, hvor TIME og foderindtag er lig med 0, kan det være et tegn på, at foderkasserne er i stykker. Hvis fodermængden ved besøgets start og/eller afslutning er negativ eller større end den fodermængde, der kan være i foderkasserne, kan det også være et tegn på, at foderkasserne ikke virker. Vi har fastsat tærskelværdierne til -1 kg og 60 kg for foderkasserne til pelleteret foder. Tærskelværdierne for foderkasserne til fuldfoder er fastsat til -5 kg og 120 kg.

Hvis der er flere end 20 observationer pr. dag, der skaber mistanke om, at foderkasserne inden for en sti ikke virker, sletter programmet alle observationerne inden for den pågældende sti og dag.

Hvis der er 20 eller færre observationer pr. dag, der skaber mistanke om, at foderkasserne inden for en sti ikke virker, sletter programmet alle observationer, hvor fodermængden ved besøgets start og/eller afslutning er negativ eller større end den fodermængde, der kan være i foderkasserne, inden for det pågældende dyr og dag.

Hvis der er 20 eller færre observationer pr. dag, der skaber mistanke om, at foderkasserne inden for en sti ikke virker, sletter programmet alle observationer, hvor TIME og foderindtag er lig med 0 uden at slette de øvrige observationer for det pågældende dyr og dag.

Opfyldninger og ædebesøg

Datasættet deles i opfyldninger og ædebesøg med eller uden kendt ID. Hvis ID er ukendt, er ID lig med -1.

Ædebesøg med ukendt ID

Hvis foderindtaget fra besøg med ukendt ID er større end 10 % af det totale foderindtag inden for sti og dag, slettes observationerne for den pågældende sti og dag.

Fremadrettet indeholder datasættet kun ædebesøg med kendt ID.

Nye variable

Programmet danner variablene Feed intake per visit (FIV, kg), Occupation time per visit (OTV, sekunder) og Feeding rate per visit (FRV, kg/min).

Variablene bruges til at identificere fejlbehæftede observationer.

FIV_0 er relevant for besøg med en ædetid på 0 sekunder. Tærskelværdien er 1,4 kg foder pr. besøg. FIV_hi og FIV_lo er relevante for alle besøg, og tærskelværdierne angiver hhv. den øvre og nedre grænse for foderindtag pr. besøg.

OTV_hi og OTV_lo er relevante for alle besøg, og tærskelværdierne angiver hhv. den øvre og nedre grænse for ædetid pr. besøg.

FRV_0 er relevant for besøg med en ædehastighed på 0 kg pr. minut. Tærskelværdien vedrører ædetiden og er 480 sekunder. FRV_hi og FRV_lo er relevante for alle besøg, og tærskelværdierne angiver hhv. den øvre og nedre grænse for ædehastighed pr. besøg.

LTD_lo og FTD_lo er relevante for alle besøg undtagen hhv. det sidste og det første besøg inden for foderkasse. Tærskelværdierne angiver de nedre grænser for tidsperioder mellem besøg.

Tabel 2. Tærskelværdier for 5 variable, der bruges til at identificere fejlbehæftede observationer.

Fejltype	Variabel	Relevante besøg	Tærskelværdi
FIV_0_pel	Feed Intake per Visit (FIV)	Besøg med ædetid = 0 sek.	FIV > 1,4 kg
FIV_hi_pel		Alle besøg med pelleteret foder	FIV > 2,0 kg
FIV_lo_pel		Alle besøg med pelleteret foder	FIV < 0 kg
FIV_0_tmr		Besøg med ædetid = 0 sek.	FIV > 1,4 kg
FIV_hi_tmr		Alle besøg med TMR	FIV > 3,5 kg
FIV_lo_tmr		Alle besøg med TMR	FIV < 0 kg
OTV_hi_pel	Occupation Time per Visit (OTV)	Alle besøg med pelleteret foder	OTV > 1800 sek.
OTV_lo_pel		Alle besøg med pelleteret foder	OTV < 0 sek.
OTV_hi_tmr		Alle besøg med TMR	OTV > 1800 sek.
OTV_lo_tmr		Alle besøg med TMR	OTV < 0 sek.
FRV_0_pel	Feeding rate per visit (FRV)[a]	FRV = 0 kg/min.	OTV > 480 sek.
FRV_hi_pel		Alle besøg med pelleteret foder	FRV > 2,5 kg/min.
FRV_lo_pel		Alle besøg med pelleteret foder	FRV < 0
FRV_0_tmr		FRV = 0 kg/min.	OTV > 480 sek.
FRV_hi_tmr		Alle besøg med TMR	FRV > 2,5 kg/min.
FRV_lo_tmr		Alle besøg med TMR	FRV < 0
LTD_lo	Leading Time Difference (LTD)[b]	Alle besøg undtagen det sidste besøg inden for foderkasse	LTD < 0 sek.
FTD_lo	Following Time Difference (LTD)[c]	Alle besøg undtagen det første besøg inden for foderkasse	FTD < 0 sek.

[a] FRV er ikke beregnet for besøg med OTV mindre end eller lig med 0.
[b] LTD = starttidspunktet for det efterfølgende besøg minus sluttidspunktet for det nuværende besøg.
[c] FTD = starttidspunktet for det nuværende besøg minus sluttidspunktet for det foregående besøg.

Fraværet eller tilstedeværelsen af en fejltype er kodet som hhv. 0 og 1.

Efterfølgende summeres fejltyperne inden for dyr og dag. Hvis summen af fejltyper inden for dyr og dag er større end 0, slettes den pågældende dag for det pågældende dyr. Det vil sige, fremadrettet indeholder datasættet kun dage, hvor alle variable ligger inden for de angivne tærskelværdier.

Datasæt med dagligt foderindtag
Programmet danner efterfølgende Daily Feed Intake (DFI), Daily Occupation Time (DOT), Daily Feeding Rate (DFR) og Number Of Visits (NOV) ved at summere inden for dyr og dag.

Datasættet indeholder desuden information om kalvens ID, besøgsdato, besætningsnummer, stinummer og fodertype.

Appendiks 1. SAS-programmet EditeringFoderdata.sas

```
**********;
*Libnames*;
**********;
libname ind "XXXXX";

*************;
*Import data*;
*************;
data foder;
  set ind.feeddatacdb;
run;

data foder;
  set foder;
  RENAME herd_system = HIS;
run;

proc means data = foder;
run;

proc print data = foder (obs = 10);
run;

*********************************;
*Number of animals and visitdates*;
*********************************;
proc sort data = foder;
  by ckrdyrnr visitdate;
run;

data AntalDyr (keep = ckrdyrnr visitdate)
     AntalVisitdate (keep = ckrdyrnr visitdate);
  set foder;
  by ckrdyrnr visitdate;
  if ckrdyrnr = -1 then delete;
  if first.ckrdyrnr = 1 then output AntalDyr;
  if first.ckrdyrnr = 1 or first.visitdate = 1 then output AntalVisitdate;
run;

*****************************;
*Remove variables temporarily*;
*****************************;
data foder (drop = Feedstuff);
  set foder;
run;

*********************************************;
*Create feed variable                       *;
*If the herd uses TMR then the variable pel = 0*;
*and if it uses pelleted feed then pel = 1     *;
```

```
*********************************************;
data foder;
  set foder;
  if slagt_bes = XXXXX then pel = 0;
                       else pel = 1;
run;

*********************************************************************;
*Identify first part of pairwise observations - deviations <= 100 g*;
*********************************************************************;
proc sort data = foder;
  by HIS VisitStartTime;
run;

data TjekPar;
  set foder;
  by HIS VisitStartTime;
  prevHIS = lag(HIS);
  prevCKRDYRNR = lag(ckrdyrnr);
  prevDF = lag(Delta_feed);
run;

data parvis1;
  set TjekPar;
  FeedDifference = (prevDF+Delta_feed);
  if (HIS = prevHIS) and (FeedDifference ne .) and (prevDF > 0) and (Delta_feed
< 0) and (-0.1 <= FeedDifference <= 0.1) then output parvis1;
run;

data ok parvis1;
  set parvis1;
  if (ckrdyrnr ne .) and (prevCKRDYRNR ne .) and (ckrdyrnr >= 0) and
(prevCKRDYRNR >= 0) and (ckrdyrnr ne prevCKRDYRNR) then output ok;

else output parvis1;
run;

data parvis2;
  set TjekPar;
  FeedDifference = (prevDF+Delta_feed);
  if (HIS = prevHIS) and (FeedDifference ne .) and (prevDF < 0) and (Delta_feed
> 0) and (-0.1 <= FeedDifference <= 0.1) then output parvis2;
run;

data ok parvis2;
  set parvis2;
  if (ckrdyrnr ne .) and (prevCKRDYRNR ne .) and (ckrdyrnr >= 0) and
(prevCKRDYRNR >= 0) and (ckrdyrnr ne prevCKRDYRNR) then output ok;

else output parvis2;
run;
```

```
*********************************************************************;
*Identify second part of pairwise observations - deviations <= 100 g*;
*********************************************************************;
proc sort data = foder;
  by HIS DESCENDING VisitStartTime;
run;

data TjekPar;
  set foder;
  by HIS DESCENDING VisitStartTime;
  prevHIS = lag(HIS);
  prevCKRDYRNR = lag(ckrdyrnr);
  prevDF = lag(Delta_feed);
run;

data parvis3;
  set TjekPar;
  FeedDifference = (prevDF+Delta_feed);
  if (HIS = prevHIS) and (FeedDifference ne .) and (prevDF > 0) and (Delta_feed
< 0) and (-0.1 <= FeedDifference <= 0.1) then output parvis3;
run;

data ok parvis3;
  set parvis3;
  if (ckrdyrnr ne .) and (prevCKRDYRNR ne .) and (ckrdyrnr >= 0) and
(prevCKRDYRNR >= 0) and (ckrdyrnr ne prevCKRDYRNR) then output ok;

else output parvis3;
run;

data parvis4;
  set TjekPar;
  FeedDifference = (prevDF+Delta_feed);
  if (HIS = prevHIS) and (FeedDifference ne .) and (prevDF < 0) and (Delta_feed
> 0) and (-0.1 <= FeedDifference <= 0.1) then output parvis4;
run;

data ok parvis4;
  set parvis4;
  if (ckrdyrnr ne .) and (prevCKRDYRNR ne .) and (ckrdyrnr >= 0) and
(prevCKRDYRNR >= 0) and (ckrdyrnr ne prevCKRDYRNR) then output ok;

else output parvis4;
run;

************************************************;
*Collect the data sets with pairwise observations*;
************************************************;
data ParviseObs1 (keep = slagt_bes HIS ckrdyrnr VisitStartTime VisitEndTime TIME
Delta_feed fyld);
  set parvis1 parvis4;
run;
```

```
proc sort data = ParviseObs1;
  by HIS VisitStartTime;
run;

data ParviseObs2 (keep = slagt_bes HIS ckrdyrnr VisitStartTime VisitEndTime TIME
Delta_feed fyld);
  set parvis2 parvis3;
run;

proc sort data = ParviseObs2;
  by HIS VisitStartTime;
run;

*****************************;
*Delete pairwise observations*;
*****************************;
data ParviseObs;
  set ParviseObs1 ParviseObs2;
run;

proc sort data = foder;
  by HIS ckrdyrnr VisitStartTime;
run;

proc sort data = ParviseObs NODUPRECS;
  by HIS ckrdyrnr VisitStartTime;
run;

data foder;
  merge foder (in = a)
        ParviseObs (in = b);
  by HIS ckrdyrnr VisitStartTime;
  if a = 1 and b = 0 then output foder;
run;

*********************************************************************;
*Create new time variables with next visit for the station and the calf*;
*********************************************************************;
data foder (keep = CKRDYRNR slagt_bes HIS pel pen fyld VisitStartTime VisitEnd-
Time NyVisitEndTime visitdate TIME Feed_begin Feed_end Delta_feed);
  set foder;
  NyVisitEndTime = VisitEndTime;
  format NyVisitEndTime DATETIME16.;
run;

proc sort data = foder;
  by HIS descending VisitStartTime;
run;

data foder;
  set foder;
```

```
  NextAnimal = lag1(ckrdyrnr);
  NextVST_Animal = lag1(VisitStartTime);
  format NextVST_Animal DATETIME16.;
run;

proc sort data = foder;
  by HIS VisitStartTime;
run;

data foder;
  set foder;
  by HIS;
  if last.HIS = 1 then do;
    NextAnimal = .;
    NextVST_Animal = .;
  end;
run;

proc sort data = foder;
  by ckrdyrnr descending VisitStartTime;
run;

data foder;
  set foder;
  NextHIS = lag1(HIS);
  NextVST_HIS = lag1(VisitStartTime);
  format NextVST_HIS DATETIME16.;
run;

proc sort data = foder;
  by ckrdyrnr VisitStartTime;
run;

data foder;
  set foder;
  by ckrdyrnr;
  if last.ckrdyrnr = 1 then do;
    NextHIS = .;
    NextVST_HIS = .;
  end;
run;

*****************************************************************;
*If NextVST_Animal or NextVST_HIS is earlier than VisitEndTime then*;
*VisitEndTime is changed depending on whether NextVST_Animal or    *;
*NextVST_HIS comes first                                          *;
*****************************************************************;
data foder;
  set foder;
  if (NextVST_Animal NE .) and (NextVST_Animal < NyVisitEndTime) then NyVis-
itEndTime = NextVST_Animal;
```

```
  if (NextVST_Animal NE .) and (VisitEndTime = .) then NyVisitEndTime =
NextVST_Animal;
run;

data NyVET_Animal;
  set foder;
  if VisitEndTime NE NyVisitEndTime then output NyVET_Animal;
run;

data foder;
  set foder;
  if (ckrdyrnr > 0) and (NextVST_HIS NE .) and (NextVST_HIS < NyVisitEndTime)
then NyVisitEndTime = NextVST_HIS;
run;

data NyVET_HIS;
  set foder;
  if VisitEndTime NE NyVisitEndTime then output NyVET_HIS;
run;

data foder (drop = NextAnimal NextVST_Animal NextHIS NextVST_HIS VisitEndTime);
  set foder;
run;

data foder;
  set foder;
  RENAME NyVisitEndTime = VisitEndTime;
run;

proc means data = foder;
run;

*************************************************************************;
*As a starting point TIME is more precise than (VisitEndTime-VisitStartTime) *;
*However, if (VisitEndTime-VisitStartTime) is a shorter time period than TIME*;
*then TIME is chaged to (VisitEndTime-VisitStartTime)                    *;
*************************************************************************;
data foder;
  set foder;
  if VisitStartTime NE . and VisitEndTime NE . then EatingTime = (VisitEndTime-
VisitStartTime);
                                              else EatingTime = .;
run;

data tjek;
  set foder;
  if EatingTime < TIME then output tjek;
run;

data foder (drop = EatingTime);
  set foder;
```

```
  if ckrdyrnr NE -1 and fyld = 0 and EatingTime NE . and EatingTime < TIME then
TIME = EatingTime;
  if ckrdyrnr NE -1 and fyld = 0 and EatingTime NE . and TIME = . then TIME =
EatingTime;
run;

proc means data = foder;
run;

*************************************************************;
*Number of HIS days and number of HIS in the herd that uses TMR*;
*************************************************************;
data HISdag;
  set foder;
  if pel = 0 then output HISdag;
run;

proc sort data = HISdag;
  by HIS visitdate;
run;

data HISdag (keep = HIS visitdate);
  set HISdag;
  by HIS visitdate;
  if first.HIS = 1 or first.visitdate = 1 then output HISdag;
run;

proc sort data = foder;
  by HIS VisitStartTime VisitEndTime;
run;

data HIS (keep = HIS);
  set foder;
  by HIS VisitStartTime VisitEndTime;
  if pel = 0 and first.HIS = 1 then output HIS;
run;

proc print data = HIS;
run;

*****************************************************;
*Create the variable fyld for the herd that uses TMR *;
*If feed is eaten then fyld = 0 and if feed is filled*;
*in the feed station then fyld = 1                   *;
*****************************************************;
data TjekFeed;
  set foder;
  by HIS VisitStartTime VisitEndTime;
  PrevFeed_end = lag1(Feed_end);
  PrevVET = lag1(VisitEndTime);
  TimeFromPrevVET = (VisitStartTime - PrevVET);
  format PrevVET DATETIME16.;
```

```
run;

data TjekFeed;
  set TjekFeed;
  by HIS;
  if first.HIS = 1 then do;
    PrevFeed_end = .;
    PrevVET = .;
    TimeFromPrevVET = .;
  end;
run;

data RefillFF
     TjekFeed;
  set TjekFeed;
  if pel = 0 and fyld = 0 and (PrevFeed_end - Feed_begin) < 0 then output Re-
fillFF;
                                                                  else output Tjek-
Feed;
run;

data RefillFF;
  set RefillFF;
  ckrdyrnr = -1;
  TIME = 0;
  fyld = 1;
  if PrevFeed_end = . or PrevVET = . or TimeFromPrevVET = . then delete;
run;

data RefillFF;
  set RefillFF;
  VisitEndTime = VisitStartTime;
  Feed_end = Feed_begin;
run;

data RefillFF (drop = PrevFeed_end PrevVET TimeFromPrevVET);
  set RefillFF;
  VisitStartTime = PrevVET;
  Feed_begin = PrevFeed_end;
  Delta_feed = (Feed_end - Feed_begin);
  StartTime = timepart(VisitStartTime);
  format StartTime TOD.;
run;

*********************************************************;
*Triviality limit for feed filled in feed stations on 5 kg*;
*********************************************************;
data RefillFF;
  set RefillFF;
  if ('05:00:00't <= StartTime < '13:30:00't) and (5 <= Delta_feed < 100) then
output RefillFF;
run;
```

```
proc sort data = RefillFF;
  by HIS visitdate DESCENDING Delta_feed;
run;

data RefillFirst
     Refill;
  set RefillFF;
  by HIS visitdate DESCENDING Delta_feed;
  if first.HIS = 1 or first.visitdate = 1 then output RefillFirst;
                                          else output Refill;
run;

proc sort data = Refill;
  by HIS visitdate DESCENDING Delta_feed;
run;

data RefillSecond;
  set Refill;
  by HIS visitdate DESCENDING Delta_feed;
  if first.HIS = 1 or first.visitdate = 1 then output RefillSecond;
run;

data RefillTMR;
  set RefillFirst
      RefillSecond;
run;

data foder (drop = StartTime);
  set foder
      RefillTMR;
run;

*************************************************************************;
*Create variables:                                                      *;
*Leading weight difference (LWD) = Entrance feed weight of following visit -*;
*exit feed weight of present visit                                      *;
*Following weight difference (FWD) = Entrance feed weight of present visit -*;
*exit feed weight of preceding visit                                    *;
*Leading time difference (LTD) = Entrance time of following visit -     *;
*exit time of present visit                                             *;
*Following weight difference (FTD) = Entrance time of present visit -   *;
*exit time of preceding visit                                           *;
*************************************************************************;
proc sort data = foder;
  by HIS VisitStartTime VisitEndTime;
run;

data foder;
  set foder;
  by HIS VisitStartTime VisitEndTime;
  PrevFeed_end = lag1(Feed_end);
```

```
  PrevVET = lag1(VisitEndTime);
  FWD = (Feed_begin - PrevFeed_end);
  FTD = (VisitStartTime - PrevVET);
  format PrevVET DATETIME16.;
run;

data foder;
  set foder;
  by HIS;
  if first.HIS = 1 then do;
    PrevFeed_end = .;
    PrevVET = .;
    FWD = .;
    FTD = .;
  end;
run;

proc sort data = foder;
  by HIS DESCENDING VisitStartTime DESCENDING VisitEndTime;
run;

data foder;
  set foder;
  by HIS DESCENDING VisitStartTime DESCENDING VisitEndTime;
  NextFeed_begin = lag1(Feed_begin);
  NextVST = lag1(VisitStartTime);
  LWD = (NextFeed_begin - Feed_end);
  LTD = (NextVST - VisitEndTime);
  format NextVST DATETIME16.;
run;

proc sort data = foder;
  by HIS VisitStartTime VisitEndTime;
run;

data foder (drop = PrevFeed_end PrevVET NextFeed_begin NextVST);
  set foder;
  by HIS;
  if last.HIS = 1 then do;
    NextFeed_begin = .;
    NextVST = .;
    LWD = .;
    LTD = .;
  end;
run;

proc means data = foder;
run;

proc print data = foder (obs = 10);
run;
```

```
*************************************************************************;
*Delete first and last visitdate for each animal                        *;
*due to incomplete daily feed intake                                    *;
*If an animal is moved to another pen then delete the day it is moved   *;
*If there are days without recordings then delete the last day with recordings*;
*and the first day after the recordings have been resumed               *;
*************************************************************************;
proc sort data = foder;
  by ckrdyrnr pen visitdate;
run;

data FirstDay;
  set foder;
  by ckrdyrnr pen visitdate;
  if first.ckrdyrnr = 1 or first.pen = 1 then output FirstDay;
run;

data LastDay;
  set foder;
  by ckrdyrnr pen visitdate;
  if last.ckrdyrnr = 1 or last.pen = 1 then output LastDay;
run;

proc sort data = foder;
  by ckrdyrnr visitdate;
run;

data MissingPrevDay;
  set foder;
  by ckrdyrnr visitdate;
  if first.ckrdyrnr = 1 or first.visitdate = 1 then output MissingPrevDay;
run;

proc sort data = MissingPrevDay;
  by ckrdyrnr visitdate;
run;

data MissingPrevDay;
  set MissingPrevDay;
  by ckrdyrnr visitdate;
  PrevDate = (visitdate-1);
  PrevVisitdate = lag1(visitdate);
  if first.ckrdyrnr = 1 then do;
    PrevDate = .;
    PrevVisitdate = .;
  end;
  format PrevDate PrevVisitdate date9.;
  if PrevDate ne . and PrevVisitdate ne . and PrevDate ne PrevVisitdate then
output MissingPrevDay;
run;

proc sort data = foder;
```

```
  by ckrdyrnr visitdate;
run;

data MissingNextDay;
  set foder;
  by ckrdyrnr visitdate;
  if last.ckrdyrnr = 1 or last.visitdate = 1 then output MissingNextDay;
run;

proc sort data = MissingNextDay;
  by ckrdyrnr DESCENDING visitdate;
run;

data MissingNextDay;
  set MissingNextDay;
  by ckrdyrnr DESCENDING visitdate;
  NextDate = (visitdate+1);
  NextVisitdate = lag1(visitdate);
  if first.ckrdyrnr = 1 then do;
    NextDate = .;
    NextVisitdate = .;
  end;
  format NextDate NextVisitdate date9.;
  if NextDate ne . and NextVisitdate ne . and NextDate ne NextVisitdate then
output MissingNextDay;
run;

data Incomplete (keep = ckrdyrnr visitdate);
  set FirstDay
      LastDay
      MissingPrevDay
      MissingNextDay;
  if ckrdyrnr = -1 then delete;
run;

proc sort data = Incomplete NODUP;
  by ckrdyrnr visitdate;
run;

proc sort data = foder;
  by ckrdyrnr visitdate;
run;

data IncompleteDFI
     foder;
  merge foder (in = a)
        Incomplete (in = b);
  by ckrdyrnr visitdate;
  if a = 1 and b = 1 then output IncompleteDFI;
  if a = 1 and b = 0 then output foder;
run;
```

```
*********************************;
*Number of animals and visitdates*;
*********************************;
proc sort data = foder;
  by ckrdyrnr visitdate;
run;

data AntalDyr (keep = ckrdyrnr visitdate)
     AntalVisitdate (keep = ckrdyrnr visitdate);
  set foder;
  by ckrdyrnr visitdate;
  if ckrdyrnr = -1 then delete;
  if first.ckrdyrnr = 1 then output AntalDyr;
  if first.ckrdyrnr = 1 or first.visitdate = 1 then output AntalVisitdate;
run;

***********************************************;
*Identify non-informative observations          *;
*i.e. both TIME og feed intake are equal to zero*;
***********************************************;
data TimeFeedZero (keep = slagt_bes visitdate pen ckrdyrnr);
  set foder;
  if TIME = 0 and (Feed_begin = Feed_end) then output TimeFeedZero;
run;

*************************************************************************;
*Identify observations with too many or negative kilos in the feed stations*;
*************************************************************************;
data Pel
     TMR;
  set foder;
  if pel = 1 then output Pel;
             else output TMR;
run;

data ManyKgPel
     FewKgPel;
  set Pel;
  if feed_begin > 60 or feed_end > 60 then output ManyKgPel;
  if feed_begin < -1 or feed_end < -1 then output FewKgPel;
run;

data ManyKgTMR
     FewKgTMR;
  set TMR;
  if feed_begin > 120 or feed_end > 120 then output ManyKgTMR;
  if feed_begin < -5 or feed_end < -5 then output FewKgTMR;
run;

data Kg (keep = slagt_bes visitdate pen ckrdyrnr);
  set ManyKgPel
      ManyKgTMR
```

```
      FewKgPel
      FewKgTMR;
run;

*****************************************************************;
*Collect the data sets containing non-informative observations and*;
*observations with too many or negative kilos in the feed stations*;
*****************************************************************;
data KgTimeFeed;
  set TimeFeedZero
      Kg;
run;

proc sort data = KgTimeFeed;
  by slagt_bes visitdate pen;
run;

proc freq data = KgTimeFeed order = data;
  tables visitdate * pen / out = FejlKg norow nocol nopercent;
  by slagt_bes;
  title "Number of non-informative observations and observations with too many
or negative kilos in the feed stations";
run;

data FejlKgAnimal
     FejlKgPen (keep = visitdate pen);
  set FejlKg;
  if COUNT <= 20 then output FejlKgAnimal;
                 else output FejlKgPen;
run;

*****************************************************************;
*Delete non-informative observations and observations with too many *;
*or negative kilos in the feed stations                          *;
*Observations are deleted within visitdate and pen if there are more*;
*than 20 observations per day                                    *;
*****************************************************************;
proc sort data = FejlKgPen;
  by visitdate pen;
run;

proc sort data = foder;
  by visitdate pen;
run;

data FejlPen
     foder;
  merge foder (in = a)
        FejlKgPen (in = b);
  by visitdate pen;
  if a = 1 and b = 1 then output FejlPen;
  if a = 1 and b = 0 then output foder;
```

```
run;

proc means data = foder;
run;

************************************************************************;
*Delete non-informative observations and observations with too many     *;
*or negative kilos in the feed stations                                 *;
*Observations containing too many or negative kilos in the feed stations*;
*are deleted within visitdate and animal if there are 20               *;
*or fewer observations per day                                          *;
*The non-informative observations are deleted but not all the           *;
*observations on the animal and date in question                        *;
************************************************************************;
proc sort data = Kg;
  by visitdate pen;
run;

proc sort data = FejlKgAnimal;
  by visitdate pen;
run;

data FejlKgAnimal (keep = ckrdyrnr visitdate pen);
  merge Kg (in = a)
        FejlKgAnimal (in = b);
  by visitdate pen;
  if a = 1 and b = 1 then output FejlKgAnimal;
run;

data FejlKgAnimal;
  set FejlKgAnimal;
  if ckrdyrnr = -1 then delete;
run;

proc sort data = FejlKgAnimal NODUP;
  by ckrdyrnr visitdate;
run;

proc sort data = foder;
  by ckrdyrnr visitdate;
run;

data FejlAnimal
     foder;
  merge foder (in = a)
        FejlKgAnimal (in = b);
  by ckrdyrnr visitdate;
  if a = 1 and b = 1 then output FejlAnimal;
  if a = 1 and b = 0 then output foder;
run;

data foder;
```

```
  set foder;
  if TIME = 0 and (Feed_begin = Feed_end) then delete;
run;

proc means data = foder;
run;

*********************************;
*Number of animals and visitdates*;
*********************************;
proc sort data = foder;
  by ckrdyrnr visitdate;
run;

data AntalDyr (keep = ckrdyrnr visitdate)
     AntalVisitdate (keep = ckrdyrnr visitdate);
  set foder;
  by ckrdyrnr visitdate;
  if ckrdyrnr = -1 then delete;
  if first.ckrdyrnr = 1 then output AntalDyr;
  if first.ckrdyrnr = 1 or first.visitdate = 1 then output AntalVisitdate;
run;

*****************************************************;
*Divide the dataset in observations with or without ID*;
*****************************************************;
data IkkeDyr
     Dyr;
  set foder;
  if ckrdyrnr < 0 then output IkkeDyr;
                  else output Dyr;
run;

*********************************************************************;
*Divide observations without ID in observations with or without feed*;
*filled in feed stations and with positive or negative feed intake  *;
*********************************************************************;
data IkkeDyr;
  set IkkeDyr;
  if visitdate = '16JAN2020'd then delete;
run;

data MissingID
     IkkeDyrJunk1
     IkkeDyrJunk2
     KunOpfyldning;
  set IkkeDyr;
  if fyld = 0 and Delta_feed < 0 then output MissingID;
  if fyld = 0 and Delta_feed >= 0 then output IkkeDyrJunk1;
  if fyld = 1 and Delta_feed < 0 then output IkkeDyrJunk2;
  if fyld = 1 and Delta_feed >= 0 then output KunOpfyldning;
run;
```

```
proc means data = MissingID;
  title "Missing ID";
run;

proc means data = IkkeDyrJunk1;
  title "Observations with fyld = 0 and Delta_feed >= 0";
run;

proc means data = IkkeDyrJunk2;
  title "Observations with fyld = 1 and Delta_feed < 0";
run;

proc means data = KunOpfyldning;
  title "Feed filled in feed stations";
run;

************************************************;
*Delete observations on feed filled in feed stations*;
*and Delta_feed is equal to zero                    *;
************************************************;
data KunOpfyldning;
  set KunOpfyldning;
  if Delta_feed = 0 then delete;
run;

proc means data = KunOpfyldning;
  title "Feed filled in feed stations and Delta_feed > 0";
run;

*************************************************************************;
*Divide observations with ID in observations with or without feed filled*;
*in the feed stations and with positive or negative feed intake         *;
*************************************************************************;
data KunDyr
     MereVaegt
     DyrJunk1
     DyrJunk2;
  set Dyr;
  if fyld = 0 and Delta_feed <= 0 then output KunDyr;
  if fyld = 0 and Delta_feed > 0 then output MereVaegt;
  if fyld = 1 and Delta_feed < 0 then output DyrJunk1;
  if fyld = 1 and Delta_feed >= 0 then output DyrJunk2;
run;

proc means data = KunDyr;
  title "Visits";
run;

proc means data = MereVaegt;
  title "Observations with fyld = 0 and Delta_feed > 0";
run;
```

```
proc means data = DyrJunk1;
run;

proc means data = DyrJunk2;
run;

**********************************;
*Study observations with missing ID*;
**********************************;
proc sort data = MissingID;
  by pen visitdate;
run;

proc means data = MissingID noprint;
  var Delta_feed;
  by pen visitdate;
  OUTPUT out = NoID SUM = ialtNoID N = antalNoID;
run;

data NoID (drop = _TYPE_ _FREQ_);
  set NoID;
run;

proc sort data = KunDyr;
  by pen visitdate;
run;

proc means data = KunDyr noprint;
  var Delta_feed;
  by pen visitdate;
  OUTPUT out = ID SUM = ialtID N = antalID;
run;

data ID (drop = _TYPE_ _FREQ_);
  set ID;
run;

data frekvens;
  merge NoID (in = a)
        ID (in = b);
  by pen visitdate;
  if a = 1 and b = 1 then output frekvens;
run;

data LidtSpild
     MegetSpild;
  set frekvens;
  ialt = (ialtNoID + ialtID);
  if abs(ialtNoID) <= abs(0.10*ialt) then output LidtSpild;
                                     else output MegetSpild;
run;
```

```
*********************************************************************;
*Delete observations within visitdate and pen if more than 10 percent*;
*of the feed eaten is from observations with missing ID              *;
*********************************************************************;
data MegetSpild (keep = visitdate pen);
  set MegetSpild;
run;

proc sort data = MegetSpild;
  by visitdate pen;
run;

proc sort data = KunDyr;
  by visitdate pen;
run;

data Spild
     visits;
  merge KunDyr (in = a)
        MegetSpild (in = b);
  by visitdate pen;
  if a = 1 and b = 1 then output Spild;
  if a = 1 and b = 0 then output visits;
run;

*********************************;
*Number of animals and visitdates*;
*********************************;
proc sort data = visits;
  by ckrdyrnr visitdate;
run;

data AntalDyr (keep = ckrdyrnr visitdate)
     AntalVisitdate (keep = ckrdyrnr visitdate);
  set visits;
  by ckrdyrnr visitdate;
  if ckrdyrnr = -1 then delete;
  if first.ckrdyrnr = 1 then output AntalDyr;
  if first.ckrdyrnr = 1 or first.visitdate = 1 then output AntalVisitdate;
run;

***************************;
*Thresholds for LTD and FTD*;
***************************;
data visits;
  set visits;
  LTD_lo = 0;
  FTD_lo = 0;
  if LTD < 0 then LTD_lo = 1;
  if FTD < 0 then FTD_lo = 1;
run;
```

```
************************************;
*Create variables:                  *;
*Feed intake per visit (FIV)        *;
*Occupation time per visit (OTV)    *;
*Feeding rate per visit, kg/min (FRV)*;
************************************;
data visits;
  set visits;
  FIV = (Feed_begin - Feed_end);
  OTV = TIME;
  if TIME <= 0 then FRV = .;
               else FRV = (FIV / (OTV/60));
run;

proc means data = visits;
  title "Visits";
run;

proc print data = visits (obs = 10);
run;

***********;
*Study FIV*;
***********;
data OTVzero
     FIV;
  set visits;
  if OTV = 0 then output OTVzero;
             else output FIV;
run;

data FIV_TMR
     FIV_PEL;
  set FIV;
  if FIV >= 10 then FIV = 10;
  ThresholdsFIV = round(FIV, 0.1);
  if pel = 0 then output FIV_TMR;
             else output FIV_PEL;
run;

proc freq data = FIV_TMR noprint;
  tables ThresholdsFIV / out = FREQ_FIV_TMR norow nocol nopercent;
  title "Thresholds for FIV in herds with TMR";
run;

proc print data = FREQ_FIV_TMR;
  title "Thresholds for FIV in herds with TMR";
run;

proc freq data = FIV_PEL noprint;
  tables ThresholdsFIV / out = FREQ_FIV_PEL norow nocol nopercent;
```

```
  title "Thresholds for FIV in herds with PEL";
run;

proc print data = FREQ_FIV_PEL;
  title "Thresholds for FIV in herds with PEL";
run;

proc means data = FIV_TMR;
  title "Thresholds for FIV in herds with TMR";
run;

proc means data = FIV_PEL;
  title "Thresholds for FIV in herds with PEL";
run;

***************************;
*Study FIV when OTV is zero*;
***************************;
data OTVzeroTMR
     OTVzeroPEL;
  set OTVzero;
  if FIV >= 5 then FIV = 5;
  ThresholdsFIV = round(FIV, 0.1);
  if pel = 0 then output OTVzeroTMR;
             else output OTVzeroPEL;
run;

proc freq data = OTVzeroTMR noprint;
  tables ThresholdsFIV / out = FREQ_OTVzeroTMR norow nocol nopercent;
  title "Thresholds for FIV in herds with TMR when OTV is zero";
run;

proc print data = FREQ_OTVzeroTMR;
  title "Thresholds for FIV in herds with TMR when OTV is zero";
run;

proc freq data = OTVzeroPEL noprint;
  tables ThresholdsFIV / out = FREQ_OTVzeroPEL norow nocol nopercent;
  title "Thresholds for FIV in herds with PEL when OTV is zero";
run;

proc print data = FREQ_OTVzeroPEL;
  title "Thresholds for FIV in herds with PEL when OTV is zero";
run;

proc means data = OTVzeroTMR;
  title "Thresholds for FIV in herds with TMR when OTV is zero";
run;

proc means data = OTVzeroPEL;
  title "Thresholds for FIV in herds with PEL when OTV is zero";
run;
```

```
*******************;
*Thresholds for FIV*;
*******************;
data visits;
  set visits;
  if FIV = . then FIV_0 = 1;
             else FIV_0 = 0;
  if FIV = . then FIV_hi = 1;
             else FIV_hi = 0;
  if FIV = . then FIV_lo = 1;
             else FIV_lo = 0;
  if pel = 1 and OTV = 0 and FIV > 1.400 then FIV_0 = 1;
  if pel = 1 and OTV > 0 and FIV > 2.000 then FIV_hi = 1;
  if pel = 1 and OTV > 0 and FIV < 0 then FIV_lo = 1;
  if pel = 0 and OTV = 0 and FIV > 1.400 then FIV_0 = 1;
  if pel = 0 and OTV > 0 and FIV > 3.500 then FIV_hi = 1;
  if pel = 0 and OTV > 0 and FIV < 0 then FIV_lo = 1;
run;

***********;
*Study OTV*;
***********;
data OTV_TMR
     OTV_PEL;
  set visits;
  OTVhour = OTV/3600;
  OTVmin = OTV/60;
  if OTVhour >= 10 then OTVhour = 10;
  ThresholdsOTVhour = round(OTVhour, 1);
  if OTVmin >= 90 then OTVmin = 90;
  ThresholdsOTVmin = round(OTVmin, 1);
  if pel = 0 then output OTV_TMR;
             else output OTV_PEL;
run;

proc freq data = OTV_TMR;
  tables ThresholdsOTVhour / norow nocol nopercent;
  title "Thresholds for OTV in hours in herds with TMR";
run;

proc freq data = OTV_TMR noprint;
  tables ThresholdsOTVmin / out = FREQ_OTV_TMR norow nocol nopercent;
  title "Thresholds for OTV in minutes in herds with TMR";
run;

proc print data = FREQ_OTV_TMR;
  title "Thresholds for OTV in minutes in herds with TMR";
run;

proc freq data = OTV_PEL;
  tables ThresholdsOTVhour / norow nocol nopercent;
```

```
  title "Thresholds for OTV in hours in herds with PEL";
run;

proc freq data = OTV_PEL noprint;
  tables ThresholdsOTVmin / out = FREQ_OTV_PEL norow nocol nopercent;
  title "Thresholds for OTV in minutes in herds with PEL";
run;

proc print data = FREQ_OTV_PEL;
  title "Thresholds for OTV in minutes in herds with PEL";
run;

proc means data = OTV_TMR;
  title "Thresholds for OTV in minutes in herds with TMR";
run;

proc means data = OTV_PEL;
  title "Thresholds for OTV in minutes in herds with PEL";
run;

********************;
*Thresholds for OTV*;
********************;
data visits;
  set visits;
  if OTV = . then OTV_hi = 1;
             else OTV_hi = 0;
  if OTV = . then OTV_lo = 1;
             else OTV_lo = 0;
  if pel = 1 and OTV > 1800 then OTV_hi = 1;
  if pel = 1 and OTV < 0 then OTV_lo = 1;
  if pel = 0 and OTV > 1800 then OTV_hi = 1;
  if pel = 0 and OTV < 0 then OTV_lo = 1;
run;

***********;
*Study FRV*;
***********;
data FRVzero
     FRV;
  set visits;
  if FRV = 0 then output FRVzero;
             else output FRV;
run;

data FRV_TMR
     FRV_PEL;
  set FRV;
  if FRV >= 10 then FRV = 10;
  ThresholdsFRV = round(FRV, 0.1);
  if pel = 0 then output FRV_TMR;
             else output FRV_PEL;
```

```
run;

proc freq data = FRV_TMR noprint;
  tables ThresholdsFRV / out = FREQ_FRV_TMR norow nocol nopercent;
  title "Thresholds for FRV in herds with TMR";
run;

proc print data = FREQ_FRV_TMR;
  title "Thresholds for FRV in herds with TMR";
run;

proc freq data = FRV_PEL noprint;
  tables ThresholdsFRV / out = FREQ_FRV_PEL norow nocol nopercent;
  title "Thresholds for FRV in herds with PEL";
run;

proc print data = FREQ_FRV_PEL;
  title "Thresholds for FRV in herds with PEL";
run;

proc means data = FRV_TMR;
  title "Thresholds for FRV in herds with TMR";
run;

proc means data = FRV_PEL;
  title "Thresholds for FRV in herds with PEL";
run;

***************;
*Study FRVzero*;
***************;
data FRVzero_TMR
     FRVzero_PEL;
  set FRVzero;
  OTVmin = OTV/60;
  if OTVmin >= 60 then OTVmin = 60;
  ThresholdsFRVzero = round(OTVmin, 1);
  if pel = 0 then output FRVzero_TMR;
             else output FRVzero_PEL;
run;

proc freq data = FRVzero_TMR noprint;
  tables ThresholdsFRVzero / out = FREQ_FRVzero_TMR norow nocol nopercent;
  title "Thresholds for FRVzero in herds with TMR";
run;

proc print data = FREQ_FRVzero_TMR;
  title "Thresholds for FRVzero in herds with TMR";
run;

proc freq data = FRVzero_PEL noprint;
  tables ThresholdsFRVzero / out = FREQ_FRVzero_PEL norow nocol nopercent;
```

```
  title "Thresholds for FRVzero in herds with PEL";
run;

proc print data = FREQ_FRVzero_PEL;
  title "Thresholds for FRVzero in herds with PEL";
run;

********************;
*Thresholds for FRV*;
********************;
data visits;
  set visits;
  FRV_hi = 0;
  FRV_lo = 0;
  FRV_0 = 0;
  if pel = 1 and FRV > 2.5 then FRV_hi = 1;
  if pel = 1 and FRV < 0 then FRV_lo = 1;
  if pel = 1 and FRV = 0 and OTV > 480 then FRV_0 = 1;
  if pel = 0 and FRV > 2.5 then FRV_hi = 1;
  if pel = 0 and FRV < 0 then FRV_lo = 1;
  if pel = 0 and FRV = 0 and OTV > 480 then FRV_0 = 1;
run;

***************;
*Sum of errors*;
***************;
data visits;
  set visits;
  SOE = (LTD_lo + FTD_lo + FIV_0 + FIV_hi + FIV_lo + OTV_hi + OTV_lo + FRV_0 +
FRV_hi + FRV_lo);
run;

proc means data = visits;
run;

proc sort data = visits;
  by ckrdyrnr visitdate;
run;

proc means data = visits noprint;
  var SOE;
  by ckrdyrnr visitdate;
  OUTPUT out = SOE SUM = ialt N = antal;
run;

data OKvisitdates (keep = ckrdyrnr visitdate);
  set SOE;
  if ialt = 0 then output OKvisitdates;
run;

proc sort data = visits;
  by ckrdyrnr visitdate;
```

```
run;

proc sort data = OKvisitdates;
  by ckrdyrnr visitdate;
run;

data OKvisitdates;
  merge visits (in = a)
        OKvisitdates (in = b);
  by ckrdyrnr visitdate;
  if a = 1 and b = 1 then output OKvisitdates;
run;

data DataPerDay (keep = slagt_bes ckrdyrnr visitdate pen pel);
  set OKvisitdates;
  by ckrdyrnr visitdate;
  if first.ckrdyrnr = 1 or first.visitdate = 1 then output DataPerDay;
run;

*****************************;
*Create variables:          *;
*Daily feed intake (DFI)    *;
*Daily occupation time (DOT)*;
*Daily feeding rate (DFR)   *;
*Number of visits (NOV)     *;
*****************************;
proc sort data = OKvisitdates;
  by ckrdyrnr visitdate;
run;

proc means data = OKvisitdates noprint;
  var FIV;
  by ckrdyrnr visitdate;
  OUTPUT out = DFI SUM = DFI N = NOV;
run;

data DFI (drop = _TYPE_ _FREQ_);
  set DFI;
run;

proc sort data = DFI;
  by ckrdyrnr visitdate;
run;

proc sort data = DataPerDay;
  by ckrdyrnr visitdate;
run;

data DataPerDay;
  merge DFI (in = a)
        DataPerDay (in = b);
  by ckrdyrnr visitdate;
```

```
  if a = 1 and b = 1 then output DataPerDay;
run;

proc sort data = OKvisitdates;
  by ckrdyrnr visitdate;
run;

proc means data = OKvisitdates noprint;
  var OTV;
  by ckrdyrnr visitdate;
  OUTPUT out = DOT SUM = DOT;
run;

data DOT (drop = _TYPE_ _FREQ_);
  set DOT;
run;

proc sort data = DOT;
  by ckrdyrnr visitdate;
run;

proc sort data = DataPerDay;
  by ckrdyrnr visitdate;
run;

data DataPerDay;
  merge DOT (in = a)
        DataPerDay (in = b);
  by ckrdyrnr visitdate;
  if a = 1 and b = 1 then output DataPerDay;
run;

data DataPerDay;
  set DataPerDay;
  if DOT <= 0 then DFR = .;
              else DFR = (DFI / (DOT/60));
run;

proc print data = DataPerDay (obs = 20);
  title "Daily feed intake and daily occupation time";
run;

proc means data = DataPerDay;
run;

*********************************;
*Number of animals and visitdates*;
*********************************;
proc sort data = DataPerDay;
  by ckrdyrnr visitdate;
run;
```

```
data AntalDyr (keep = ckrdyrnr visitdate)
     AntalVisitdate (keep = ckrdyrnr visitdate);
  set DataPerDay;
  by ckrdyrnr visitdate;
  if ckrdyrnr = -1 then delete;
  if first.ckrdyrnr = 1 then output AntalDyr;
  if first.ckrdyrnr = 1 or first.visitdate = 1 then output AntalVisitdate;
run;
```