Data model for supplementary registrations in the PigIT project

Anders Ringgaard Kristensen

PigIT Report No. 2 • April 2014

Data model for supplementary registrations in the PigIT project

Anders Ringgaard Kristensen

PigIT report No. 2 • April 2014

This note is also available on www at URL:
http://www.pigit.net/publications/PigIT-Report2.pdf

Centre for Herd-oriented Education, Research and Development
Department of Large Animal Sciences
University of Copenhagen

Contents

Chapter 1

Introduction

In addition to the automatic sensor registrations, a number of supplementary (typically manually recorded) registrations are available from the farms. Those data include farmer observations of events in the experimental pens, insertion and removal of pigs and information about the nutritional contents of feedstuffs. In some herds there will furthermore be specific observations like results from manual weighing of pigs.

The objective of this note is to define a data model for these supplementary registrations and describe how to integrate them in the PigIT database.

The procedures for collecting and transferring from individual farms will be described in separate notes (see Kristensen, 2014, for an example).

Chapter 2

Overview of the data

Basically, three kinds of supplementary data are collected in each farm (in addition to the automatic registrations):

- Daily pen level observations of tail biting, diarrhea and fouling.
- Data on pig movements at pen level
  - Insertion of pigs
  - Dead pigs
  - Removal of pigs
  - Slaughtered pigs
- Data on feedstuffs

Furthermore, some farms will have additional registrations as for instance individual weighing of pigs in selected pens.

In the data model suggested in this note, emphasis will be put on integration with the other parts of the PigIT database and on avoiding redundant information. A diagram illustrating the full data model for the supplementary data is shown in Figure A.1.

The R code necessary to create the tables of the data model is shown in Appendix B.

Chapter 3

Tail biting, diarrhea and fouling

3.1 Main table

The events "tail biting", "diarrhea" and "fouling" are observed daily at pen level by the staff in the herd. The main table for these observations is shown in Table 3.1. The field `DatabaseId` is included because it is analogous to similar fields in tables for sensor data. Since, however, all registrations are linked to a specific pen and each pen to a specific herd, it is actually redundant and could be omitted. The link to the pen in question is given through the foreign key `PenId` which refers to entries in the `PigIt.PigPen` table.

The chosen table structure is very general and can easily handle other kinds of observations at pen level in case it is decided at a later stage of the project to include other kinds of observations.

3.2 Supplementary tables

In addition to the main table, three other tables have been defined. The observer (staff member) referred by the foreign key `ObserverId` has to be pre-defined in

Table 3.1: Fields of the table `PigIt.FarmerObservations`

Field	Type	Explanation
Id	uniqueidentifier	Primary key
DatabaseId	int	Farm number
PenId	uniqueidentifier	Foreign key, identification of pen
Timestamp	datetime	Time of registration (date)
ObserverId	uniqueidentifier	Foreign key, identification of observer
EventId	uniqueidentifier	Foreign key, identification of event
EventPropertyValue	real	An associated value observed
EventTreatmentId	uniqueidentifier	Foreign key, identification of treatment

Table 3.2: Fields of the table `PigIt.Observer`

Field	Type	Explanation
Id	uniqueidentifier	Table key
FarmId	uniqueidentifier	Foreign key, identification of pig farm
Initials	nvarchar(32)	Initials of the observer
Name	nvarchar(100)	Optional name of the observer

Table 3.3: Fields of the table `PigIt.ObservedEvents`

Field	Type	Explanation
Id	uniqueidentifier	Primary key
EventName	nvarchar(100)	Name of the event

the table `PigIt.Observer` shown as Table 3.2. The foreign key `FarmId` refers to entries in the `PigIt.PigFarm` table.

Only pre-defined events can be observed. For each event, an optional numerical value can be observed and a pre-defined event specific treatment can be specified. The `PigIt.FarmerObservations` table is supposed to have an entry for each pen every day. If nothing is observed, a specially defined "Nothing observed" event is entered into the table. This principle has been chosen in order to distinguish "negative" observations from lacking observations. The list of events currently defined for entries of the `PigIt.FarmerObservations` table is (with reference to the `EventName` field):

- Nothing to report,
- Tailbite,
- Diarrhea,
- Fouling.

The table `PigIt.ObservedEvents` is shown in Table 3.3.

In order to provide a textual interpretation of the `EventPropertyValue` field of the Table 3.1, the table `PigIt.ObservedEventProperties` is created. It is shown in Table 3.4. Based on the identity of an observed event, a string describing the property is given in the field `PropertyName`. Currently, only the events "Diarrhea" and "Fouling" have numerical properties:

Diarrhea: Number of spots,

Fouling: Percent of lying area.

The percentage of the lying area observed for "Fouling" is observed as 25, 50 or 100 percent.

Table 3.4: Fields of the table `PigIt.ObservedEventProperties`

Field	Type	Explanation
Id	uniqueidentifier	Primary key
EventId	uniqueidentifier	Foreign key, event in question
PropertyName	nvarchar(100)	Name of the property

Table 3.5: Fields of the table `PigIt.ObservedEventTreatment`

Field	Type	Explanation
Id	uniqueidentifier	Primary key
EventId	uniqueidentifier	Foreign key, event in question
TreatmentName	nvarchar(100)	Name of the treatment

Finally, a table `PigIt.ObservedEventTreatment` defines for each event type a list of possible treatments. The table is shown in Table 3.5. Currently the following treatments are defined:

Nothing to report: Only one dummy treatment is defined:

- No treatment.

Tailbite: Four alternative treatments can be chosen:

- No treatment,
- Pigs removed,
- Sinner removed,
- Rooting material provided.

Diarrhea: Four alternative treatments can be chosen:

- No treatment,
- Pig treated,
- Pen treated,
- Section treated.

Fouling: Three alternative treatments can be chosen:

- No treatment,
- Temperature or sprinkler adjusted,
- Medication.

3.3 Diagram

A database diagram showing the `PigIt.FarmerObservation` table and its relations to other tables is shown in Figure 3.1.

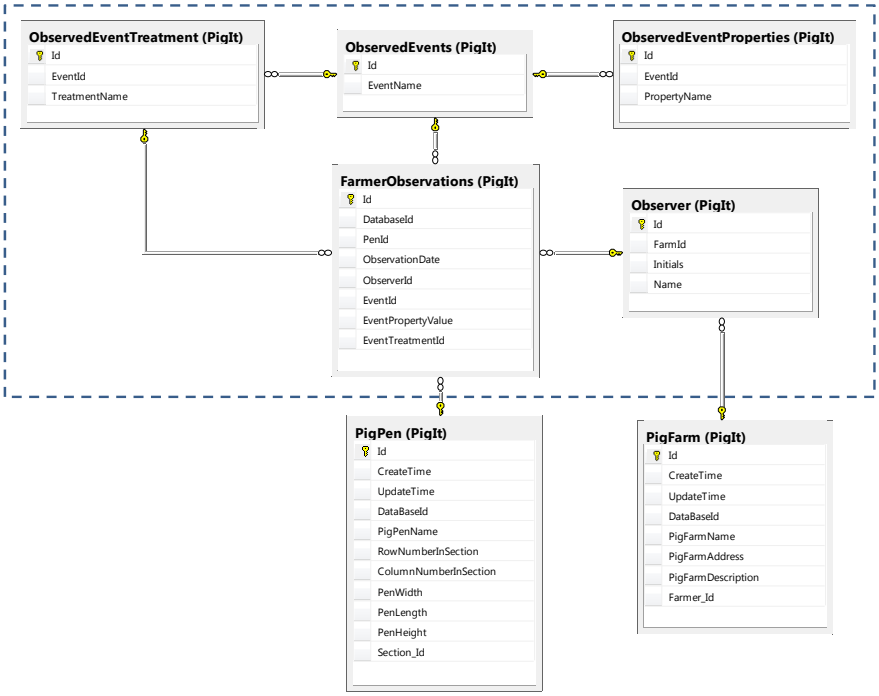

Figure 3.1: Data model for the farmer observations at pen level. The table `PigIt.FarmerObservation` is shown in the middle surrounded by tables it refers to. Tables outside the dashed rectangle are part of the original PigIT database.

Chapter 4

Pig movements

4.1 Main table

The main purpose of the pig movement registrations is to make sure that it is always well-defined how many pigs there are in a given pen on a given day. In addition some supplementary observations can be done.

Basically, each time one or more pigs are inserted or removed from a pen, a registration is made. The main table `PigIt.PigMovements` is shown as Table 4.1. Just like the farmer observations table (Table 3.1) the field `DatabaseId` is redundant and could be skipped. For each observation, references to the pen, the observer (farm staff) and the event type are given. The movement as such is specified by the date, the number of pigs and the total weight of the pigs. The type of movement is specified through the `EventId` field referencing an entry in the `PigIt.ObservedEvents` table (Table 3.3).

Table 4.1: Fields of the table `PigIt.PigMovements`

Field	Type	Explanation
Id	uniqueidentifier	Primary key
DatabaseId	int	Farm number
PenId	uniqueidentifier	Foreign key, identification of pen
EventDate	date	Date of event
ObserverId	uniqueidentifier	Foreign key, identification of observer
EventId	uniqueidentifier	Foreign key, identification of event
NumberOfPigs	int	Number of pigs
TotalPigWeight	real	Total weight of pigs
UpdateTime	datetime	Time of last update

4.2 Supplementary tables

The two previously defined tables `PigIt.Observer` (see Table 3.2) for observer identity and `PigIt.ObservedEvents` (see Table 3.3) for event type are also used with pig movements. Special entries for movements are inserted into the `PigIt.ObservedEvents` table:

- Pigs inserted,
- Pigs removed,
- Pigs died,
- Pigs slaughtered.

4.3 Diagram

A database diagram showing the `PigIt.PigMovements` table and its relations to other tables is shown in Figure 4.1.

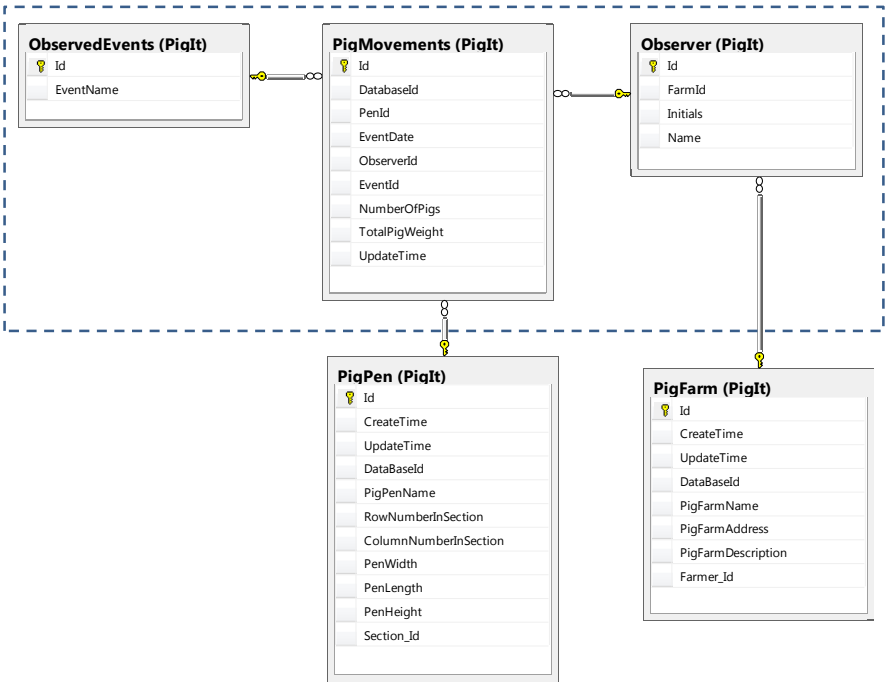

Figure 4.1: Data model for the pig movements at pen level. The table `PigIt.PigMovements` is shown in the middle surrounded by tables it refers to. Tables outside the dashed rectangle are part of the original PigIT database.

Chapter 5

Feedstuffs

5.1 Main table

In the sensor registration part of the PigIT database, a table with feedstuffs already exists. It is based on readings from the feeding computer in the herds of the project. It only contains information about name and energy content of the feedstuff (read from the feeding computer).

In order to have a better description of the feedstuffs, an additional table, `PigIt.FeedAnalysis`, is created with the fields shown in Table 5.1. The field `FeedId` links to entries of the existing table `PigIt.Feed`.

5.2 Diagram

A database diagram showing the `PigIt.FeedAnalysis` table and its relation to other tables is shown in Figure 5.1.

Table 5.1: Fields of the table `PigIt.FeedAnalysis`

Field	Type	Explanation
Id	uniqueidentifier	Primary key
FeedId	uniqueidentifier	Foreign key, identification of feedstuff
DryMatter	real	Dry matter percentage
CrudeProtein	real	Crude protein, percent of feed
EnergyFEsv	real	Energy, feed units per kg. dry matter
Lysine	real	Lysine, percent of crude protein
Methionine	real	Methionine, percent of crude protein
Cysteine	real	Cysteine, percent of crude protein
Threonine	real	Threonine, percent of crude protein
Tryptophan	real	Tryptophan, percent of crude protein
Isoleucine	real	Isoleucine, percent of crude protein
Leucine	real	Leucine, percent of crude protein
Histidine	real	Histidine, percent of crude protein
Tyrosine	real	Tyrosine, percent of crude protein
Valine	real	Valine, percent of crude protein

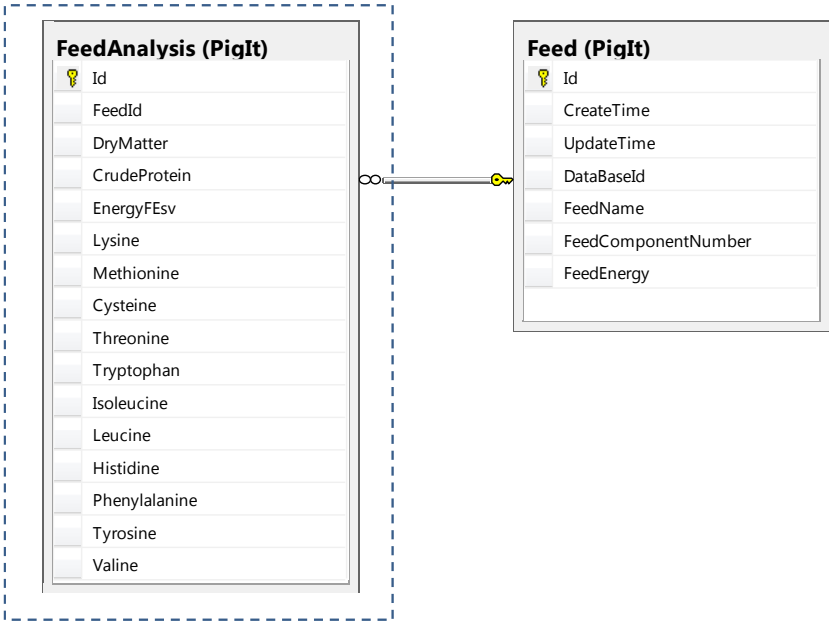

Figure 5.1: Data model for feedstuffs. The table `PigIt.FeedAnalysis` is shown to the left of the table it refers to. The table outside the dashed rectangle is part of the original PigIT database.

Chapter 6

Other registrations

6.1 Manual weighings of individual pigs

6.1.1 Main table

In some herds pigs in selected pens are individually identified by RFID ear tags and weighed regularly (see for instance Kristensen, 2014). In those cases, the recordings are inserted into the table `PigIt.ManualWeighings` in the PigIT database. The fields of the table are shown in Table 6.1.

6.1.2 Supplementary tables

In addition to the main table, a table listing individual pigs is created. It is shown as Table 6.2.

Again, the field `DatabaseId` is redundant and could be skipped.

6.1.3 Diagram

A database diagram showing the `PigIt.ManualWeighings` table and its relation to other tables is shown in Figure 6.1.

Table 6.1: Fields of the table `PigIt.ManualWeighings`

Field	Type	Explanation
Id	uniqueidentifier	Primary key
PigId	uniqueidentifier	Foreign key, identification of pig
PigWeight	real	Pig weight (kg)
ObservationTime	date	Date of weighing

Table 6.2: Fields of the table `PigIt.IndividualPigs`

Field	Type	Explanation
Id	uniqueidentifier	Primary key
DatabaseId	int	Farm number
PenId	uniqueidentifier	Foreign key, identification of pen
InsertionDate	date	Date inserted

Figure 6.1: Data model for manual weighings. The table `PigIt.ManualWeighings` is shown to the left of the table it refers to. The table outside the dashed rectangle is part of the original PigIT database.

Table 6.3: Fields of the table `PigIt.FeedConsumptionPeriod`

Field	Type	Explanation
Id	uniqueidentifier	Primary key
AnalysisId	uniqueidentifier	Foreign key, identification of feedstuff
FirstDay	date	First day of period
LastDay	date	Last day of period
DispenserId	uniqueidentifier	Foreign key, identification of dispenser
ConsumedAmount	real	The amount consumed (kg)

6.2 Feed consumption at dispenser level

6.2.1 Main table

In some herds feed consumption is registered at dispenser level at regular intervals (see for instance Kristensen, 2014). In those cases, the recordings are inserted into the table `PigIt.FeedConsumptionPeriod` in the PigIT database. The fields of the table are shown in Table 6.3. The field `AnalysisId` refers to Table 5.1 and the field `DispenserId` refers to the table listing the dispensers.

6.2.2 Diagram

A database diagram showing the `PigIt.FeedConsumptionPeriod` table and its relation to other tables is shown in Figure 6.2.

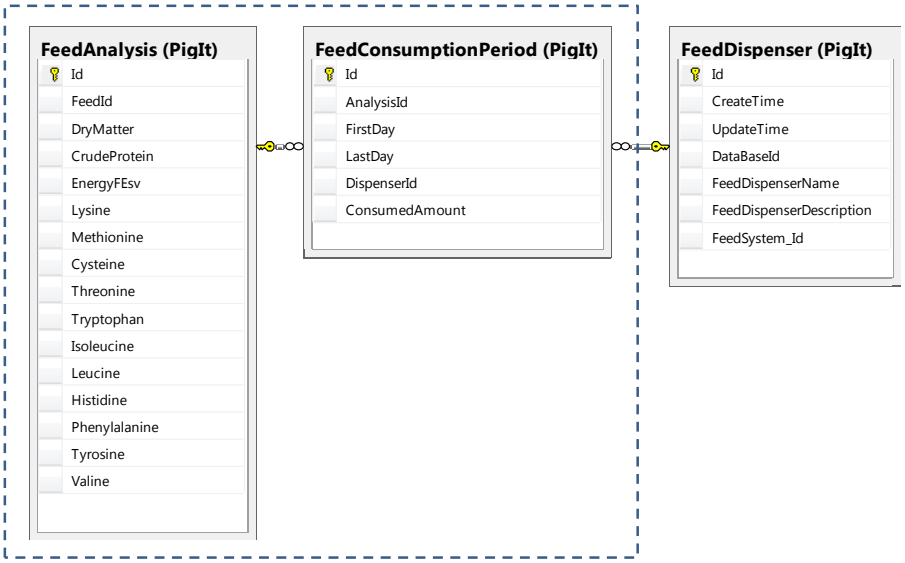

Figure 6.2: Data model for feed consumption at dispenser level. The table `PigIt.FeedConsumptionPeriod` is shown to the left of the table it refers to. The table outside the dashed rectangle is part of the original PigIT database.

Chapter 7

Tables for communication with the DLBR database

7.1 Introduction

Some herds use the commercial DLBR SvineIT Management Information System for data recording (see for instance Kristensen, 2014). For easier communication with the DLBR system two special tables have been created. They are not part of the PigIt schema and in order to distinguish them from other tables they are named with the prefix "DLBR_". Currently only two such tables are defined:

- The `DLBR_PenLink` table defining a translation table for pen identities in the DLBR database and pen identities in the PigIT database.
- The `DLBR_Feedstuff` table containing the feedstuff definitions from the DLBR database and linking them to entries of the feed analysis table.

7.2 The tables

The `DLBR_PenLink` table defining a translation table for pen identities in the DLBR database and pen identities in the PigIT database has a very simple structure as seen in Table 7.1. The field `PigIT_PenId` links to an entry in the `PigIt.PigPen` table.

Table 7.1: Fields of the table `DLBR_PenLink`

Field	Type	Explanation
DLBR_PenId	uniqueidentifier	Primary key
PigIT_PenId	uniqueidentifier	Primary key, Foreign key, identification of pen in PigIt

Table 7.2: Fields of the table `DLBR_Feedstuff`

Field	Type	Explanation
DLBR_FeedId	uniqueidentifier	Primary key, reused from DLBR
AnalysisId	uniqueidentifier	Foreign key, identification of feedstuff
UpdateTime	datetime	Time of last update
UpdateResponsible	uniqueidentifier	Foreign key, identification of person
FeedName	nvarchar(32)	Name of feedstuff
CodeLot	int	The feed component number
NutritionalContents	text	Nutritional contents

The `DLBR_Feedstuff` has a more complicated structure as shown in Table 7.2. The field `NutritionalContents` is a text field defing the nutritional contents of the feedstuff in a format described in Kristensen (2014, Section 4.1.6).

7.3 Diagram

A database diagram showing the `DLBR_Feedstuffs` table and its relation to other tables is shown in Figure 7.1.

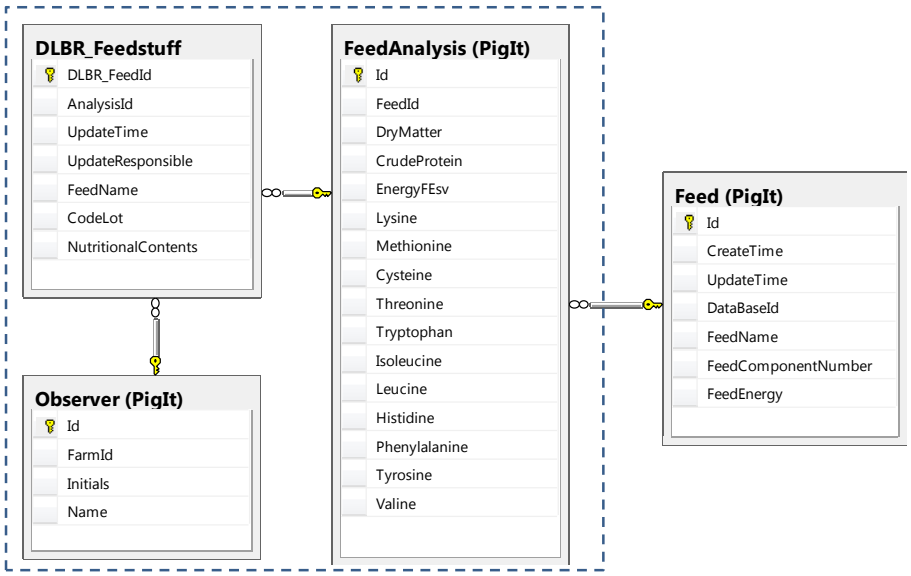

Figure 7.1: Data model for feedstuffs from the DLBR database. The table `PigIt.DLBR_Feedstuff` is shown to the left of the table it refers to. The table outside the dashed rectangle is part of the original PigIT database.

Bibliography

Kristensen, A. R., 2014. Logbook and supplementary registrations at the Kappel farm. PigIT Report 3, The PigIT project, Department of Large Animal Sciences, University of Copenhagen.

Appendix A

Full data model for supplementary data

A full data model for the supplementary registrations described in this note is shown in Figure A.1.

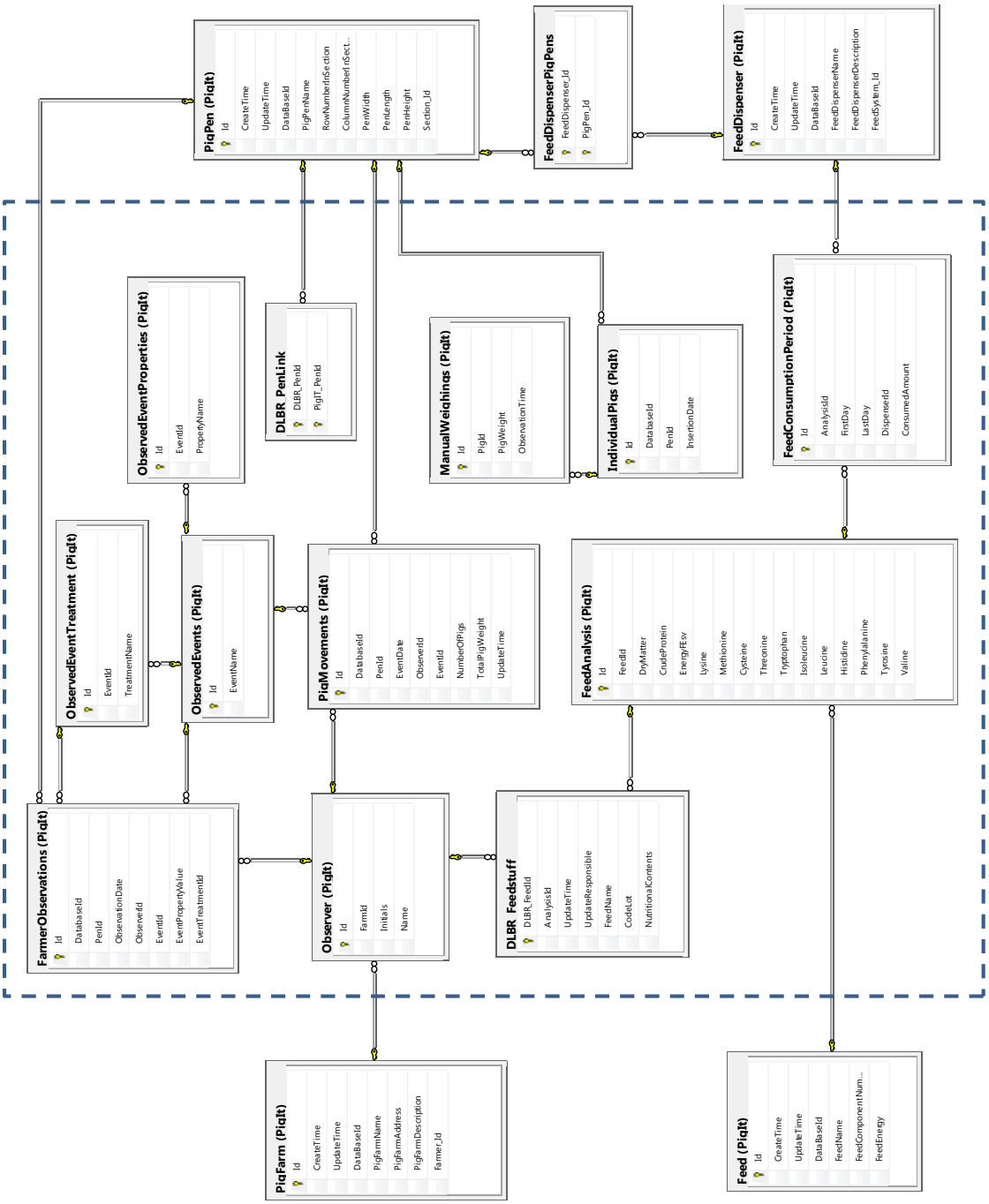

Figure A.1: Data model for the supplementary registrations. The tables within the dashed rectangle are tables created to store the supplementary registrations. Tables outside the rectangle are part of the original PigIT database.

Appendix B

R source code for creation of tables

```
# The RODBC package is used for communication with the database
library(RODBC)

################################################################################
# Establish connection to the database
################################################################################

print("Connecting to the PigIT database")
pigIt = odbcConnect("PigIT")
pigIt

################################################################################
# Create tables
################################################################################

# Create a table linking pen IDs in the DLBR database to pen IDs in PigIt
tabMap = sqlQuery(pigIt, "CREATE TABLE DLBR_PenLink
                 (DLBR_PenId uniqueidentifier NOT NULL,
                  PigIT_PenId uniqueidentifier NOT NULL,
                  PRIMARY KEY(DLBR_PenId, PigIT_PenId),
                  FOREIGN KEY(PigIT_PenId) references PigIt.PigPen(Id))")
tabMap


# Create a table with observed events and enter the events
newTab = sqlQuery(pigIt, "CREATE TABLE PigIt.ObservedEvents
                 (Id uniqueidentifier NOT NULL,
                  EventName nvarchar(100) NOT NULL,
                  PRIMARY KEY(Id)
                  )")

# Insert currently defined events
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEvents (Id, EventName)
               VALUES (NEWID(), 'Diarrhea')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEvents (Id, EventName)
               VALUES (NEWID(), 'Tailbite')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEvents (Id, EventName)
               VALUES (NEWID(), 'Fouling')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEvents (Id, EventName)
               VALUES (NEWID(), 'Nothing to report')")
```

```
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEvents (Id, EventName)
                VALUES (NEWID(), 'Pigs inserted')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEvents (Id, EventName)
                VALUES (NEWID(), 'Pigs died')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEvents (Id, EventName)
                VALUES (NEWID(), 'Pigs removed')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEvents (Id, EventName)
                VALUES (NEWID(), 'Pigs slaughtered')")

# Create a table with observed properties of events and enter the possible properties
newTab = sqlQuery(pigIt, "CREATE TABLE PigIt.ObservedEventProperties
                 (Id uniqueidentifier NOT NULL,
                  EventId uniqueidentifier NOT NULL,
                  PropertyName nvarchar(100) NOT NULL,
                  PRIMARY KEY(Id),
                  FOREIGN KEY(EventId) references PigIt.ObservedEvents(Id)
                  )")

# Insert currently defined properties
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventProperties
              (Id, EventId, PropertyName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Diarrhea'),
              'Number of spots')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventProperties
              (Id, EventId, PropertyName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Fouling'),
              'Percent of lying area')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventProperties
              (Id, EventId, PropertyName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Nothing to report'),
              'Nothing to observe')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventProperties
              (Id, EventId, PropertyName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Tailbite'),
              'Nothing to observe')")

# create a table with possible treatments
newTab = sqlQuery(pigIt, "CREATE TABLE PigIt.ObservedEventTreatment
                 (Id uniqueidentifier NOT NULL,
                  EventId uniqueidentifier NOT NULL,
                  TreatmentName nvarchar(100) NOT NULL,
                  PRIMARY KEY(Id),
                  FOREIGN KEY(EventId) references PigIt.ObservedEvents(Id))")

# Insert diarrhea treatments
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventTreatment
              (Id, EventId, TreatmentName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Diarrhea'),
              'No treatment')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventTreatment
              (Id, EventId, TreatmentName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Diarrhea'),
              'Pig treated')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventTreatment
              (Id, EventId, TreatmentName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Diarrhea'),
              'Pen treated')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventTreatment
              (Id, EventId, TreatmentName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Diarrhea'),
              'Section treated')")
```

```
# Insert tail biting treatments
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventTreatment
              (Id, EventId, TreatmentName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Tailbite'),
              'No treatment')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventTreatment
              (Id, EventId, TreatmentName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Tailbite'),
              'Pigs removed')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventTreatment
              (Id, EventId, TreatmentName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Tailbite'), '
              Sinner removed')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventTreatment
              (Id, EventId, TreatmentName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Tailbite'),
              'Rooting material provided')")

# Insert fouling treatments
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventTreatment
              (Id, EventId, TreatmentName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Fouling'),
              'No treatment')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventTreatment
              (Id, EventId, TreatmentName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Fouling'),
              'Medication')")
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventTreatment
              (Id, EventId, TreatmentName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Fouling'),
              'Temperature or sprinkler adjusted')")

# Insert dummy treatment when nothing is observed
ins = sqlQuery(pigIt, "INSERT INTO PigIt.ObservedEventTreatment
              (Id, EventId, TreatmentName) VALUES (NEWID(),
              (SELECT Id FROM PigIt.ObservedEvents WHERE EventName = 'Nothing to report'),
              'No treatment')")


# Create a table with people making observations
newTab = sqlQuery(pigIt, "CREATE TABLE PigIt.Observer
                 (Id uniqueidentifier NOT NULL,
                  FarmId uniqueidentifier NOT NULL,
                  Initials nvarchar(32) NOT NULL,
                  Name nvarchar(100),
                  PRIMARY KEY(Id),
                  FOREIGN KEY(FarmId) references PigIt.PigFarm(Id))")


# Create a table with farmer observations
newTab = sqlQuery(pigIt, "CREATE TABLE PigIt.FarmerObservations
                 (Id uniqueidentifier NOT NULL,
                  DatabaseId int NOT NULL,
                  PenId uniqueidentifier NOT NULL,
                  ObservationDate datetime NOT NULL,
                  ObserverId uniqueidentifier NOT NULL,
                  EventId uniqueidentifier NOT NULL,
                  EventPropertyValue real,
                  EventTreatmentId uniqueidentifier NOT NULL,
                  PRIMARY KEY(Id),
                  FOREIGN KEY(PenId) references PigIt.PigPen(Id),
                  FOREIGN KEY(ObserverId) references PigIt.Observer(Id),
```

```
                 FOREIGN KEY(EventId) references PigIt.ObservedEvents(Id),
                 FOREIGN KEY(EventTreatmentId) references PigIt.ObservedEventTreatment(Id)
                 )")

# Create a table with pig movements
newTab = sqlQuery(pigIt, "CREATE TABLE PigIt.PigMovements
                (Id uniqueidentifier NOT NULL,
                 DatabaseId int NOT NULL,
                 PenId uniqueidentifier NOT NULL,
                 EventDate date NOT NULL,
                 ObserverId uniqueidentifier NOT NULL,
                 EventId uniqueidentifier NOT NULL,
                 NumberOfPigs int NOT NULL,
                 TotalPigWeight real,
                 UpdateTime datetime NOT NULL,
                 PRIMARY KEY(Id),
                 FOREIGN KEY(PenId) references PigIt.PigPen(Id),
                 FOREIGN KEY(ObserverId) references PigIt.Observer(Id),
                 FOREIGN KEY(EventId) references PigIt.ObservedEvents(Id))")
newTab

# Create a table with individual pigs (with RFID tags)
newTab = sqlQuery(pigIt, "CREATE TABLE PigIt.IndividualPigs
                (Id uniqueidentifier NOT NULL,
                 DatabaseId int NOT NULL,
                 PenId uniqueidentifier NOT NULL,
                 InsertionDate date NOT NULL,
                 PRIMARY KEY(Id),
                 FOREIGN KEY(PenId) references PigIt.PigPen(Id))")
newTab

# Create a table with manual individual weighings
newTab = sqlQuery(pigIt, "CREATE TABLE PigIt.ManualWeighings
                (Id uniqueidentifier NOT NULL,
                 PigId uniqueidentifier NOT NULL,
                 PigWeight real NOT NULL,
                 ObservationTime date NOT NULL,
                 PRIMARY KEY(Id),
                 FOREIGN KEY(PigId) references PigIt.IndividualPigs(Id))")
newTab

# Create a table with nutritional contents of feedstuffs
newTab = sqlQuery(pigIt, "CREATE TABLE PigIt.FeedAnalysis
                 (Id uniqueidentifier NOT NULL,
                 FeedId uniqueidentifier,
                 DryMatter real NOT NULL,
                 CrudeProtein real NOT NULL,
                 EnergyFEsv real NOT NULL,
                 Lysine real NOT NULL,
                 Methionine real NOT NULL,
                 Cysteine real NOT NULL,
                 Threonine real NOT NULL,
                 Tryptophan real NOT NULL,
                 Isoleucine real NOT NULL,
                 Leucine real NOT NULL,
                 Histidine real NOT NULL,
                 Phenylalanine real NOT NULL,
                 Tyrosine real NOT NULL,
                 Valine real NOT NULL,
                 PRIMARY KEY(Id),
                 FOREIGN KEY(FeedId) references PigIt.Feed(Id))")
newTab
```

```
# Create a table with feedstuffs from the DLBR database
newTab = sqlQuery(pigIt, "CREATE TABLE DLBR_Feedstuff
                  (DLBR_FeedId uniqueidentifier NOT NULL,
                  AnalysisId uniqueidentifier NOT NULL,
                  UpdateTime datetime NOT NULL,
                  UpdateResponsible uniqueidentifier NOT NULL,
                  FeedName nvarchar(32),
                  CodeLot int,
                  NutritionalContents text NOT NULL,
                  PRIMARY KEY(DLBR_FeedId),
                  FOREIGN KEY(AnalysisId) references PigIt.FeedAnalysis(Id),
                  FOREIGN KEY(UpdateResponsible) references PigIt.Observer(Id))")
newTab

# Create a table with feed consumption at dispenser level
newTab = sqlQuery(pigIt, "CREATE TABLE PigIt.FeedConsumptionPeriod
                 (Id uniqueidentifier NOT NULL,
                  AnalysisId uniqueidentifier NOT NULL,
                  FirstDay date NOT NULL,
                  LastDay date NOT NULL,
                  DispenserId uniqueidentifier NOT NULL,
                  ConsumedAmount real NOT NULL,
                  PRIMARY KEY(Id),
                  FOREIGN KEY(AnalysisId) references PigIt.FeedAnalysis(Id),
                  FOREIGN KEY(DispenserId) references PigIt.FeedDispenser(Id))")
newTab
```

PigIT Report No. 2 • April 2014
http://www.pigit.net/publications/PigIT-Report2.pdf

Centre for Herd-oriented Education, Research and Development
Department of Large Animal Sciences
University of Copenhagen