

# PPS data management recipe

## Version 3.0

Instructions for data management for a new project

Authors:

Joost van Heerwaarden

Gerrie W.J. van de Ven

Marcel Lubbers

Antonius G.T. Schut

<b>Introduction .....</b>	<b>2</b>
<b>Data .....</b>	<b>3</b>
Directory structure .....	3
Data formatting requirements .....	4
Metadata .....	5
Data collection for field surveys .....	5
Database data .....	5
Long term storage of research data .....	5
Digital Object Identifier (DOI) .....	6
<b>Data privacy issues .....</b>	<b>6</b>
Privacy legislation .....	6
Privacy officer .....	6
<b>Websites .....</b>	<b>6</b>
PhD website .....	6
BSc-MSc-Internship website .....	7
Full access websites .....	7
<b>Coding &amp; Scripting .....</b>	<b>7</b>
Code versioning, GitLab .....	7
Improving, changing existing models .....	7
<b>Backup .....</b>	<b>7</b>
<b>Set-up script .....</b>	<b>8</b>

## Introduction

A good data management plan with a data storage and processing structure is an essential part of research. This document describes the recipe that is developed for research at PPS. This recipe helps to ensure that data, analyses and results are stored and documented in a systematic way and facilitates good scientific practices with re-usable data, transparent methods and repeatable results.

Proper data management contributes to making research more transparent and accessible and helps ensuring that ethical and quality requirements related to collection, storage and pre- and post-processing data are met. Further, it is an essential part of open science where data can safely be shared with other users and needs full documentation describing legal restrictions on use and sharing, data collection protocols and description of the data with associated metadata.

The recipe aims to help you organise your data and set up data collection protocols in a structured manner which will ensure transparent use of methods and facilitates processing of data in a later stage. The investment in the implementation of the PPS-recipe will pay off in later stages of your research and allows supervisors and colleagues better to provide support.

The PPS-recipe allows flexibility, yet requires that data processing is done with scripts and not within spreadsheets. We strongly recommend to work with ASCII formatted files that only contain text, such as comma or tab delimited text files (.csv; .txt). Processing of data in spreadsheets needs to be avoided because this not transparent and can often not be reproduced. Our recipe works with .csv files that are used for processing of data and automatically produces a Microsoft Excel (.xlsx) file as last step in the process, containing a single dataset with a sheet including a description of the data, a sheet with processed data and a sheet with metadata providing a unit (e.g. kg/ha) and description of what each value in the data columns represent (e.g. standardized maize grain yield with a 12.5% moisture content).

At PPS, we have developed a set of guidelines and structures that aim to ensure data is handled and stored uniformly. It is based around the following set of principles:

1. Integrity of raw data (stored in ASCII formatted files when possible)
2. Full documentation of the nature and content of all data files by providing meta-data and variable definitions.
3. Reproducible pathway from (raw) data to results by commented scripts

In principle, each project such as a PhD thesis chapter, MSc thesis, or research paper should be associated with a self-contained directory structure containing all data, scripts, results and written output that can be archived and made available for storage or sharing.

Below is a detailed description of what this directory structure looks like, how data needs to be formatted and documented and how to set up scripts to handle and analyse data transparently. This distribution comes with a helper R script that can be sourced to make an initial project folder structure, and that includes some example data and scripts that demonstrate the entire path from setting up a project directory, to processing raw data to a distributable file with a metadata page to simple analysis that produces and stores results. **A brief description of the implementation of this script can be found at the end of this document.**

## Data

### Directory structure

The hierarchical directory structure associated with a project is presented in Figure 1. A central project directory with an informative name contains four main directories: *data*, *results*, *scripts*, and *writing*. The *writing* directory is meant to contain the project output document, i.e. a thesis chapter or research paper.

The *data* directory contains the sub-directories *definitions\_protocols*, *processed* and *raw*.

The *raw* directory is the most important one, since it is here that files with all raw data should be stored. All raw data needed for the generation of project results should be stored in this directory, even if they are already stored elsewhere. The important thing to note about raw data is that has to be as close to initial data entry format as possible and should not be edited or handled by the user after it has been stored in the *raw* directory. There are however a set of guidelines that raw data is expected to adhere to. These will be outlined further below.

The *scripts* directory contains all computer code needed for data processing, cleaning and analysis. Ideally, they form a numbered set of scripts (e.g. “1. Raw data cleaning.r”) that, when run successively, reproduces all results referred to in the project output document (e.g. report, paper or chapter) from the raw data. The scripts should contain sufficient commentary to be human readable. Results generated by scripts are written to the *results* directory and include the number of the script (e.g. “1. Cleaned data.csv” or “1. Histograms of raw values.pdf”) for easy traceability. The analysis scripts write the tables and figures referred to in the project output document to the corresponding *results/tables* and *results/figures* directories. As a by-product of running the scripts, a set of intermediary or processed data files may be produced. These are written to the *data/processed* directory. This directory also contains the distributable copies of the data, including complete meta-data. Finally, the *data/definitions\_protocols* directory may contain documents describing the specifics of data collection such as protocols and detailed descriptions of how variables in the data are defined. Ideally, variable definitions that will be distributed with the data refer to these documents when needed. Statements related to ethical and privacy guidelines followed can also be stored here.

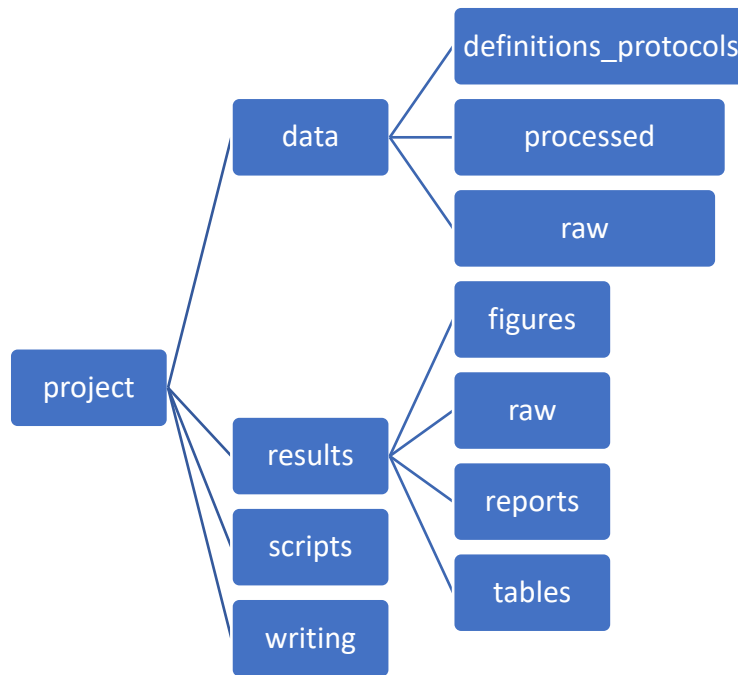


Figure 1 Hierarchical directory structure for storing all data, scripts, results and written output for a research project.

#### Data formatting requirements

Raw data files need to adhere to a couple of basic formatting rules to ensure that they can be handled efficiently and uniformly.

First of all, raw data in a single ASCII file has columns representing one single variable, factor or identifier. **Use underscores** instead of spaces **in column names**.

**Repeated observations** on the same object or experimental unit (e.g. yields measured over different seasons) are recorded:

- in case an ASCII file or MS-Excel is used, on separate lines by adding a row for each observation, repeating all information related to the object and adding a time column to indicate which date- and when required a timepoint the information in a row refers to.
- In case a Database (e.g. MySQL) is used, as separate records of a Database table storing all information mentioned above.

**Dates** are formatted as **YYYYMMDD** to support sorting and overcome scripting problems in programming languages.

**Experimental data** must include a location, name of the experiment, treatment, plot, replicate, latitude and longitude, date of observation (etc).

Second, numeric variables have the **unit added to the variable name, separated by a "\$"**, e.g. Dry\_grain\_yield\$kg\_ha. Use standard (SI) abbreviations for units but avoid special symbols that may cause problems when handling and storing data. Use clear variable names and be consistent, e.g. Fresh\_grain\_yield\$kg\_ha, Dry\_grain\_yield\$kg\_ha or GrainYield\$kgDMha.

**Please see the example data downloaded by the helper script to see what a data file should look like.**

## Metadata

Each raw data file is linked to a file containing extensive metadata (or in case a Database is used, each raw data table is linked to a corresponding table containing the extensive metadata). Metadata is meant to provide all information needed to make data usable for others. It includes a data identifier, information on the project and project funding, author and contact information, data type, species or subjects the data refers to, geographic areas and dates covered.

## Data collection for field surveys

Whenever you're going to conduct field surveys, we strongly recommend to use the Open Data Kit (ODK) software. This ODK has been used by many others in our field and is best for ease of data handling and requires the least amount of your time. There are many standard questions and forms available that you may use for your project. You are strongly advised to use those forms and follow generic procedures to allow re-use of interview data. Please, contact Marcel Lubbers ([marcel.lubbers@wur.nl](mailto:marcel.lubbers@wur.nl)) to get your ODK credentials and consult Joost van Heerwaarden ([joost.vanheerwaarden@wur.nl](mailto:joost.vanheerwaarden@wur.nl)) for advice before creating your own ODK survey forms.

## Database data

Whenever working with a data, it is recommended to create a backup on a daily base. For databases that work with the Structured Query Language (SQL), you can create a so-called SQL dump. These SQL databases contain tables with data but also a table structure that allows users to link the separate tables. This SQL dump is in fact a backup of your database, storing all database data as well as the table structure as one ASCII file. It is advised to always create a dump before making any changes to your data, so you can restore your database easily if things go wrong! The creation of a SQL dump is supported by all database management systems, e.g. MySQL, MariaDB, SQLite.

The process to make a dump is relatively easy. For example, if you use MySQL software as a database management system, you can use the command below to create a SQL dump file:

```
mysqldump -u username -p password database_name > database_name.sql
```

Where *database\_name* is the name of the database for which you want to create the SQL dump file, use ".sql" as the extension of the file (this way you can easily recognize a SQL dump file). Restoring such a SQL dump is also straight-forward, for dumps created with MySQL the following command can be used:

```
mysql -u username -p password database_name < database_name.sql
```

## Long term storage of research data

4TU (an initiative of four technical universities, namely Eindhoven, Delft, Twente and Wageningen) and DANS-Easy (Data Archiving and Networked Services) offer researchers a trusted long-term archive for storing and reusing research data.

For the work done at PPS the use of 4TU is more appropriate than DANS-Easy, as this archive is focusing on the technical sciences. Please ensure to comply with the privacy

legislation when sharing data! Whenever possible, it is best to archive your data as an ASCII file. For databases, besides an ASCII file for all data stored in tables, include a SQL dump (see *database data* section for clarification).

If data formats can't be stored as ASCII file (e.g. geographical data or geodatabases including gridded files (geotiffs, etc.) or vector files (ESRI's shapefiles, etc.), store these as they are.

### Digital Object Identifier (DOI)

When sharing datasets to the public, we advise to associate it with a so-called **digital object identifier**, to identify it permanently. Once the DOI is allocated, it never changes for that dataset. That means items allocated with a DOI can be more easily found and anyone looking for a specific dataset can be sure they have found the correct material. DOIs are an ISO standard (ISO 26324-2012) and are unique worldwide. You can contact the WUR library to get one for your dataset!

## Data privacy issues

### Privacy legislation

Private data, photo or movies cannot be shared publicly without consent of the person(s) involved. Please respect the [privacy legislation](#), also to prevent fines. This often means that two datasets need to be created, one complete dataset and an anonymized dataset (excluding all traces to private information, e.g. names, phone numbers, e-mail addresses, GPS coordinates, etc.). Only completely the anonymized dataset may be shared with a wider public, sharing of the complete dataset is (often) illegal without a formal agreement that requires appropriate legal procedures and approval. Note that strict regulations apply and follow the appropriate procedures.

### Privacy officer

Many (internationally/EU) funded research projects that involve surveys, pilots or other human participants deal with personal data. As it is quite complicated to fill in all the required deliverables and statements on data protection on time, We recommend contacting our local privacy officer, Joan Schrijvers ([joan.schrijvers@wur.nl](mailto:joan.schrijvers@wur.nl)). She can guide you through the process and help with filling in the required forms and deliverables. You can also always ask her for help when creating consent forms for surveys, since these can be complicated as well.

## Websites

### PhD website

Especially for PhD students at PPS, a website was developed to store your final version of each thesis chapter, including the corresponding datasets, as well as your Data Management Plan. The DMP needs to be updated each time you use new data sets, being sets owned by yourself or from other institutes.

The Data Management Plan uses a template, so it takes only a few minutes to create one. Before creating your own plan, please have a look at the example plan:

<https://phd.pps.wur.nl/example-management-plan-imagine-world-no-phosphorus>

Also on the site, script examples and documentation for R, Python and Matlab are provided.

The URL for this website is: <https://phd.pps.wur.nl>

#### BSc-MSc-Internship website

Final versions of the thesis and datasets for MSc, BSc and Internship students are stored and made available to the public. The URL for this website is: <https://bscmssc.pps.wur.nl>. On this website, student reports and associated data and scripts can be found.

#### Full access websites

To access all information stored on these sites or add content yourself, you need to login first with your login credentials. Please, contact Marcel Lubbers ([marcel.lubbers@wur.nl](mailto:marcel.lubbers@wur.nl)) to get your login credentials. It is not allowed to publish or distribute any document or data set stored on these websites owned by other authors or institutes without their consent.

## Coding & Scripting

#### Code versioning, GitLab

Whenever coding or scripting is involved, the well-known GIT code versioning system should be used (unless it's agreed by your supervisor it's not possible), the GitLab of WUR.

Using GitLab also prevents data loss. Commit changes as soon as possible.

Ask Mink Zijlstra ([mink.zijlstra@wur.nl](mailto:mink.zijlstra@wur.nl)) to setup your GIT.

#### Improving, changing existing models

All changes need be well documented, including links to literature used.

## Backing up

When connected to the WUR network if you work on your "M" drive, backups are made automatically and older versions of files can be restored when needed. However, when working offline or on a local drive, you need to ensure that your work is backed up. You can back up your work on a daily base by using backup services offered by WUR (e.g. GitLab, MS-OneDrive). So, if your laptop gets stolen or is broken, you can restore your files and your work will not be lost! This is especially important when working in the field with unique data from for example experiments that cannot be repeated.

If you're abroad and your network is functioning well, use a virtual private network (VPN) connection to the WUR network as if you're stationed at WUR in Wageningen. How this can be done is detailed on the IT page of WUR. As an alternative for some operations MyworkSpace (<http://myworkspace.wur.nl>) can also be used.

If it's not possible to backup your daily work by using services offered by WUR (e.g. GitLab, MS-OneDrive), you can use an external hard disk or memory stick. Just remember to keep them apart from your laptop, in case of theft or loss. Upon your return to Wageningen, ensure to backup all your work using GitLab or any other backup service offered by WUR.

## Set-up script

A standard *R* script is available that when run from the main project directory (with this directory set as “home”). It can be sourced from GitHub as follows (if you haven’t done so already):

1. Make an empty folder
2. Set it to working director in R
3. Paste the following in the terminal:

```
source("https://git.wur.nl/langu001/PPS_data_management/-  
/raw/master/Master_Script.R")
```

This automatically downloads scripts and files that:

1. create the data management structures described above. Briefly, it creates the required directory structure, read any raw data files (.csv or .xlsx) stored in the project directory and place them in the *data/raw* directory.  
The script also provides an example for the final processing step and processes the raw data into a corresponding processed .xlsx file, containing worksheets with 1. meta-data with all required fields and 2. A list of all variables in the data with appropriate units and extra columns for variable definitions and privacy sensitivity. Please check the “example\_data.csv” and “Meststof proef WUR.csv” data files in the *data/raw* directory and the associated “example\_data\_metadata.xlsx” and “Meststof proef WUR\_metadata.xlsx” files in the *data/processed* directory.
2. It places a set of R scripts in the *scripts* directory (numbered 0-2, and a functions file) that provide examples of reading, cleaning and analysing data. They are run automatically (by sourcing the *0\_build\_project.R* file) and output is written to the *results/figures* and the *results/raw* directories. There is also a separate *RawDataConvertScript.R* that when sourced, converts raw files in the working directory or *data/raw* directory into an .xlsx file with metadata as described under point 1. Please have a look at what the scripts do to get an idea of the logic behind them.
3. It places the current Document and Readme in *writing* directory.