

# Package: expDB (via r-universe)

September 9, 2024

**Type** Package

**Title** Database for Experiment Dataset

**Version** 0.1.0

**Maintainer** Bangyou Zheng <bangyou.zheng@csiro.au>

**Description** A 'SQLite' database is designed to store all information of experiment-based data including metadata, experiment design, managements, phenotypic values and climate records. The dataset can be imported from an 'Excel' file.

**License** MIT + file LICENSE

**URL** <https://expdb.bangyou.me/>, <https://github.com/byzheng/expdb>

**BugReports** <https://github.com/byzheng/expdb/issues>

**Encoding** UTF-8

**LazyData** true

**Imports** DBI, RSQLite, utils, readxl, png, grid, tibble, lubridate, dplyr, methods, weaana (>= 0.1.1), tidyr, tidyverse, magrittr, stats, reshape2, stringr, rlang

**RoxygenNote** 7.1.2

**Suggests** testthat

**Repository** <https://byzheng.r-universe.dev>

**RemoteUrl** <https://github.com/byzheng/expdb>

**RemoteRef** HEAD

**RemoteSha** 331dd2e7f06fac8970aedd1dfe7d2d00353d5140

## Contents

dbAddDesigns . . . . .	3
dbAddFertilization . . . . .	3
dbAddGene . . . . .	4
dbAddGeneAllele . . . . .	4
dbAddGenotype . . . . .	5

dbAddIrrigatons . . . . .	5
dbAddLog . . . . .	6
dbAddMethods . . . . .	6
dbAddMets . . . . .	7
dbAddNodes . . . . .	7
dbAddPhenotype . . . . .	8
dbAddResearcher . . . . .	8
dbAddSites . . . . .	9
dbAddSource . . . . .	9
dbAddTraits . . . . .	10
dbAddTrials . . . . .	10
dbAddTrialSoil . . . . .	11
dbAddWeather . . . . .	11
dbAppendTable . . . . .	12
dbExportMet . . . . .	12
dbGenotypeCheckName . . . . .	13
dbGetDryWeightPerStem . . . . .	13
dbGetFertilization . . . . .	14
dbGetFieldMaturity . . . . .	14
dbGetFieldPopulation . . . . .	15
dbGetFieldStemNumber . . . . .	15
dbGetFieldTillerNumber . . . . .	16
dbGetGene . . . . .	16
dbGetGenotype . . . . .	17
dbGetIrrigation . . . . .	17
dbGetLog . . . . .	18
dbGetMetInfo . . . . .	18
dbGetOrganFinalLeafNumber . . . . .	19
dbGetOrganHaunIndex . . . . .	19
dbGetPhenotype . . . . .	20
dbGetPlantFlowering . . . . .	20
dbGetPlantHeading . . . . .	21
dbGetPlantStemElongation . . . . .	21
dbGetPlantStemNumber . . . . .	22
dbGetPlantTillerNumber . . . . .	22
dbGetSites . . . . .	23
dbGetSource . . . . .	23
dbGetTraits . . . . .	24
dbGetTrials . . . . .	24
dbGetWeather . . . . .	25
dbGetZadoksStage . . . . .	25
dbImportXLSX . . . . .	26
dbInsertUpdateByRow . . . . .	26
dbListTrials . . . . .	27
expdbConnect . . . . .	27
expdbCreateDB . . . . .	28
expdbDisconnect . . . . .	28
getIdByUniqueIndex . . . . .	29

<i>dbAddDesigns</i>	3
<i>harvestQuadratDetail</i> . . . . .	29
<b>Index</b>	<b>31</b>

---

<i>dbAddDesigns</i>	<i>Add design for a trial</i>
---------------------	-------------------------------

---

**Description**

Add design for a trial

**Usage**

`dbAddDesigns(con, data, extra_design = NULL)`

**Arguments**

<code>con</code>	a connection object as produced by <code>dbConnect</code>
<code>data</code>	Trial design
<code>extra_design</code>	The extra columns for design

**Value**

no return values

---

<i>dbAddFertilization</i>	<i>Insert or Update fertilization into expDB</i>
---------------------------	--

---

**Description**

Insert or Update fertilization into expDB

**Usage**

`dbAddFertilization(con, data)`

**Arguments**

<code>con</code>	a connection object as produced by <code>dbConnect</code>
<code>data</code>	A data frame includes all columns

**Value**

no return values

---

`dbAddGene`*Add gene information into database*

---

**Description**

Add gene information into database

**Usage**

```
dbAddGene(con, genes)
```

**Arguments**

<code>con</code>	a connection object as produced by <code>dbConnect</code>
<code>genes</code>	A <code>data.frame</code> of genes

**Value**

No return values

---

`dbAddGeneAllele`*Add gene allele information into database*

---

**Description**

Add gene allele information into database

**Usage**

```
dbAddGeneAllele(con, genes)
```

**Arguments**

<code>con</code>	a connection object as produced by <code>dbConnect</code>
<code>genes</code>	A <code>data.frame</code> of genes

**Value**

No return values

---

dbAddGenotype	<i>Add genotypes into expDB</i>
---------------	---------------------------------

---

**Description**

Add genotypes into expDB

**Usage**

```
dbAddGenotype(con, genotypes)
```

**Arguments**

con	a connection object as produced by dbConnect
genotypes	A string vector of genotypes

**Value**

No return values

---

dbAddIrrigatons	<i>Insert or Update irrigation into expDB</i>
-----------------	---

---

**Description**

Insert or Update irrigation into expDB

**Usage**

```
dbAddIrrigatons(con, data)
```

**Arguments**

con	a connection object as produced by dbConnect
data	A data frame includes all columns

**Value**

no return values

---

dbAddLog	<i>Add log from expDB</i>
----------	---------------------------

---

**Description**

Add log from expDB

**Usage**

```
dbAddLog(con, msg, date = format(Sys.time(), format = "%Y-%m-%d"))
```

**Arguments**

con	a connection object as produced by dbConnect
msg	Add message into expdb
date	Create time of message

**Value**

No return values

---

dbAddMethods	<i>Insert or Update methods into expDB</i>
--------------	--

---

**Description**

Insert or Update methods into expDB

**Usage**

```
dbAddMethods(con, data)
```

**Arguments**

con	a connection object as produced by dbConnect
data	A data frame includes all columns

**Value**

no return values

---

dbAddMets	<i>Insert and update met into expDB</i>
-----------	---

---

**Description**

Insert and update met into expDB

**Usage**

```
dbAddMets(con, data)
```

**Arguments**

con	a connection object as produced by dbConnect
data	Met design

**Value**

no return values

---

dbAddNodes	<i>Add nodes into expDB</i>
------------	-----------------------------

---

**Description**

Add nodes into expDB

**Usage**

```
dbAddNodes(con, data)
```

**Arguments**

con	a connection object as produced by dbConnect
data	phenotype value

**Value**

no return values

---

dbAddPhenotype	<i>Add design for a trial</i>
----------------	-------------------------------

---

**Description**

Add design for a trial

**Usage**

```
dbAddPhenotype(con, data)
```

**Arguments**

con	a connection object as produced by dbConnect
data	phenotype value

**Value**

no return values

---

dbAddResearcher	<i>Insert and update researcher into expDB</i>
-----------------	--

---

**Description**

Insert and update researcher into expDB

**Usage**

```
dbAddResearcher(con, data)
```

**Arguments**

con	A connection object as produced by dbConnect
data	A data frame includes all columns

**Value**

no return values



---

dbAddSites	<i>Insert or update site into expDB</i>
------------	---

---

**Description**

Insert or update site into expDB

**Usage**

```
dbAddSites(con, data)
```

**Arguments**

con	a connection object as produced by dbConnect
data	A data frame includes all columns

**Value**

no return values

---

dbAddSource	<i>Insert and update source into expDB</i>
-------------	--

---

**Description**

Insert and update source into expDB

**Usage**

```
dbAddSource(con, data)
```

**Arguments**

con	A connection object as produced by dbConnect
data	A data frame includes all columns

**Value**

no return values

---

dbAddTraits	<i>Insert or update trait into expDB</i>
-------------	--

---

**Description**

Insert or update trait into expDB

**Usage**

```
dbAddTraits(con, data)
```

**Arguments**

con	a connection object as produced by dbConnect
data	A data frame includes all columns

**Value**

no return values

---

dbAddTrials	<i>Insert or Update trial into expDB</i>
-------------	--

---

**Description**

Insert or Update trial into expDB

**Usage**

```
dbAddTrials(con, data)
```

**Arguments**

con	a connection object as produced by dbConnect
data	A data frame includes all columns

**Value**

no return values

---

dbAddTrialSoil      *Add soil for a trial*

---

**Description**

Add soil for a trial

**Usage**

```
dbAddTrialSoil(  
  con,  
  data,  
  units = list(thickness = "cm", no3 = "kg/ha", nh4 = "kg/ha")  
)
```

**Arguments**

con	a connection object as produced by dbConnect
data	Soil profiles for trials
units	a list for the unit

**Value**

no return values

---

dbAddWeather      *Add weather records into expDB*

---

**Description**

Add weather records into expDB

**Usage**

```
dbAddWeather(con, data, name = NULL)
```

**Arguments**

con	a connection object as produced by dbConnect
data	A string character for the path to met file, a WeaAna object, or a data frame.
name	The met name in the database if data is a data frame.

**Value**

no return values

---

dbAppendTable	<i>Append a table into db and check the column name</i>
---------------	---

---

**Description**

Append a table into db and check the column name

**Usage**

```
dbAppendTable(con, table, data)
```

**Arguments**

con	A connection object as produced by dbConnect
table	The target table name
data	A data frame to write into table

---

dbExportMet	<i>Export trials weather records to met file</i>
-------------	--

---

**Description**

Export trials weather records to met file

**Usage**

```
dbExportMet(con, output, na = NA, ...)
```

**Arguments**

con	a connection object as produced by dbConnect
output	The folder of output files
na	The character for missing value with default NA
...	All other arguments to define range of export trials. All trials will be export if there are not arguments. Supported arguments include trial (or trialcode)

**Value**

Write weather records into files. No return values.

---

dbGenotypeCheckName *Check genotype names*

---

**Description**

Check genotype names

**Usage**

```
dbGenotypeCheckName(con, genotype)
```

**Arguments**

con                    a connection object as produced by dbConnect  
genotype              The genotype name will be checked

**Value**

A vector with check genotype names

---

dbGetDryWeightPerStem *Get the dry weight per stem*

---

**Description**

Get the dry weight per stem

**Usage**

```
dbGetDryWeightPerStem(con, trials = NULL, ...)
```

**Arguments**

con                    a connection object as produced by dbConnect  
trials                 A data.frame to specify trials. If not NULL, other arguments will be ignored.  
...                    Arguments pass to dbGetTrials

**Value**

A data.frame for selected dry weight per stem

---

dbGetFertilization     *Get fertilization from database*

---

**Description**

Get fertilization from database

**Usage**

```
dbGetFertilization(con, ...)
```

**Arguments**

con                    a connection object as produced by dbConnect  
...                    Other arguments to specify meta data

**Value**

a data.frame for fertilization information

---

dbGetFieldMaturity     *Estimation of maturity*

---

**Description**

Estimation of maturity

**Usage**

```
dbGetFieldMaturity(con, trials = NULL, ...)
```

**Arguments**

con                    a connection object as produced by dbConnect  
trials                 A data.frame to specify trials. If not NULL, other arguments will be ignored.  
...                    Arguments pass to dbGetTrials

**Value**

A data.frame for selected maturity time

---

dbGetFieldPopulation *Estimation of plant populations*

---

**Description**

Estimation of plant populations

**Usage**

```
dbGetFieldPopulation(con, trials = NULL, ...)
```

**Arguments**

con                    a connection object as produced by dbConnect  
trials                A data.frame to specify trials. If not NULL, other arguments will be ignored.  
...                   Arguments pass to dbGetTrials

**Value**

A data.frame for selected field population

---

dbGetFieldStemNumber *Estimation of stem number per unit area*

---

**Description**

Estimation of stem number per unit area

**Usage**

```
dbGetFieldStemNumber(con, trials = NULL, ...)
```

**Arguments**

con                    a connection object as produced by dbConnect  
trials                A data.frame to specify trials. If not NULL, other arguments will be ignored.  
...                   Arguments pass to dbGetTrials

**Value**

A data.frame for selected stem number

---

`dbGetFieldTillerNumber`*Estimation of tiller number per unit area*

---

**Description**

Estimation of tiller number per unit area

**Usage**

```
dbGetFieldTillerNumber(con, trials = NULL, ...)
```

**Arguments**

<code>con</code>	a connection object as produced by <code>dbConnect</code>
<code>trials</code>	A <code>data.frame</code> to specify trials. If not <code>NULL</code> , other arguments will be ignored.
<code>...</code>	Arguments pass to <code>dbGetTrials</code>

**Value**

A `data.frame` for selected tiller number

---

`dbGetGene`*Get the gene information*

---

**Description**

Get the gene information

**Usage**

```
dbGetGene(con)
```

**Arguments**

<code>con</code>	a connection object as produced by <code>dbConnect</code>
------------------	---

**Value**

a `data.frame` with all gene information



---

dbGetGenotype	<i>Get the genotype information</i>
---------------	-------------------------------------

---

**Description**

Get the genotype information

**Usage**

```
dbGetGenotype(con, name_only = FALSE)
```

**Arguments**

con	a connection object as produced by dbConnect
name_only	Only return the name of genotypes

**Value**

data.frame with genotype information or a vector with genotype name if name\_only = TRUE.

---

dbGetIrrigation	<i>Get irrigation from database</i>
-----------------	-------------------------------------

---

**Description**

Get irrigation from database

**Usage**

```
dbGetIrrigation(con, ...)
```

**Arguments**

con	a connection object as produced by dbConnect
...	Other arguments to specify meta data

**Value**

a data frame for irrigation information

---

dbGetLog	<i>Get log from expDB</i>
----------	---------------------------

---

**Description**

Get log from expDB

**Usage**

```
dbGetLog(con)
```

**Arguments**

con                    a connection object as produced by dbConnect

**Value**

A data.frame with all logs

---

dbGetMetInfo	<i>Get met information</i>
--------------	----------------------------

---

**Description**

Get met information

**Usage**

```
dbGetMetInfo(con, name)
```

**Arguments**

con                    a connection object as produced by dbConnect

name                   The met name

**Value**

a data.frame for met information

---

dbGetOrganFinalLeafNumber  
*Get the final leaf number*

---

**Description**

The final leaf number is first retrieved from trait "O\_FinalLeafNumber", then calculated from trait "O\_HaunIndex" if "O\_FinalLeafNumber" is not observed. Final leaf number equals the maximum value of "O\_HaunIndex", which should be an integer.

**Usage**

```
dbGetOrganFinalLeafNumber(con, trials = NULL, ...)
```

**Arguments**

con	a connection object as produced by dbConnect
trials	A data.frame to specify trials. If not NULL, other arguments
...	Arguments to specific trials.

**Value**

A data.frame for selected final leaf number

---

dbGetOrganHaunIndex *Get Haun Index*

---

**Description**

The Haun Index is retrieved from trait "O\_HaunIndex", extending the final observation

**Usage**

```
dbGetOrganHaunIndex(con, trials = NULL, avg = TRUE, ...)
```

**Arguments**

con	a connection object as produced by dbConnect
trials	A data.frame to specify trials. If not NULL, other arguments
avg	Whether to calculate the average value
...	Arguments to specific trials.

**Value**

A data.frame for selected Haun Index

---

dbGetPhenotype	<i>Get phenotype values through a group of conditions</i>
----------------	---

---

**Description**

Get phenotype values through a group of conditions

**Usage**

```
dbGetPhenotype(
  con,
  traits = NULL,
  direction = "long",
  tt = FALSE,
  gene = FALSE,
  ...
)
```

**Arguments**

con	a connection object as produced by dbConnect
traits	A list of traits. All traits will be returned if NULL
direction	One of 'long' or 'wide' for reshape function
tt	Whether to calculate thermal time
gene	Whether to get gene information
...	All other arguments to define range of export trials

**Value**

a data.frame for selected phenotypic values

---

dbGetPlantFlowering	<i>Estimation of flowering time</i>
---------------------	-------------------------------------

---

**Description**

Estimation of flowering time

**Usage**

```
dbGetPlantFlowering(con, trials = NULL, ...)
```

**Arguments**

con            a connection object as produced by dbConnect  
trials        A data.frame to specify trials. If not NULL, other arguments will be ignored.  
...           Arguments pass to dbGetTrials

**Value**

A data.frame for selected flowering time

---

dbGetPlantHeading        *Estimation of heading time*

---

**Description**

Estimation of heading time

**Usage**

```
dbGetPlantHeading(con, trials = NULL, ...)
```

**Arguments**

con            a connection object as produced by dbConnect  
trials        A data.frame to specify trials. If not NULL, other arguments will be ignored.  
...           Arguments pass to dbGetTrials

**Value**

A data.frame for selected heading time

---

dbGetPlantStemElongation  
*Estimation of stem elongation*

---

**Description**

Estimation of stem elongation

**Usage**

```
dbGetPlantStemElongation(con, trials = NULL, ...)
```

**Arguments**

con            a connection object as produced by dbConnect  
 trials        A data.frame to specify trials. If not NULL, other arguments will be ignored.  
 ...           Arguments pass to dbGetTrials

**Value**

A data.frame for selected stem elongation stage

---

dbGetPlantStemNumber    *Estimation of stem number per plant*

---

**Description**

Estimation of stem number per plant

**Usage**

```
dbGetPlantStemNumber(con, trials = NULL, ...)
```

**Arguments**

con            a connection object as produced by dbConnect  
 trials        A data.frame to specify trials. If not NULL, other arguments will be ignored.  
 ...           Arguments pass to dbGetTrials

**Value**

A data.frame for selected stem number for individual plant

---

dbGetPlantTillerNumber  
                                  *Estimation of tiller number per plant*

---

**Description**

Estimation of tiller number per plant

**Usage**

```
dbGetPlantTillerNumber(con, trials = NULL, ...)
```

**Arguments**

con            a connection object as produced by dbConnect  
 trials        A data.frame to specify trials. If not NULL, other arguments will be ignored.  
 ...           Arguments pass to dbGetTrials

**Value**

A data.frame for selected tiller number for individual plant

---

dbGetSites	<i>Get site into expDB</i>
------------	----------------------------

---

**Description**

Get site into expDB

**Usage**

dbGetSites(con)

**Arguments**

con            a connection object as produced by dbConnect

**Value**

a data.frame for all sites in the data base

---

dbGetSource	<i>Get source from expDB</i>
-------------	------------------------------

---

**Description**

Get source from expDB

**Usage**

dbGetSource(con)

**Arguments**

con            a connection object as produced by dbConnect

**Value**

A data.frame for all source in the data base

---

 dbGetTraits

*Get trait list*


---

**Description**

Get trait list

**Usage**

```
dbGetTraits(con)
```

**Arguments**

con                    a connection object as produced by dbConnect

**Value**

a data.frame for all traits in the data base

---

dbGetTrials

*Get trials by a groups of conditions.*


---

**Description**

Get trials by a groups of conditions.

**Usage**

```
dbGetTrials(con, design = TRUE, ...)
```

**Arguments**

con                    a connection object as produced by dbConnect

design                  Whether include design

...                    All other arguments to define range of export trials. All trials will be export if there are not arguments. Supported arguments include trial (or trialcode)

**Value**

A data.frame for selected trials



---

dbGetWeather	<i>Get weather records from expDB</i>
--------------	---------------------------------------

---

**Description**

Get weather records from expDB

**Usage**

```
dbGetWeather(con, name, format = "data_frame", na = NA_character_, tz = "UTC")
```

**Arguments**

con	a connection object as produced by dbConnect
name	The met name
format	The format of export dataset.
na	The character for missing value with default NA
tz	Time zone applied for hourly temperature

**Value**

a data.frame for all weather records

---

dbGetZadoksStage	<i>Obtain the key phenology stage</i>
------------------	---------------------------------------

---

**Description**

Obtain the key phenology stage

**Usage**

```
dbGetZadoksStage(con, trials, key_stage)
```

**Arguments**

con	a connection object as produced by dbConnect
trials	A data.frame to specify trials. If not NULL, other arguments will be ignored.
key_stage	The key zadoks stage

**Value**

A data.frame for selected Zadoks stage

---

dbImportXLSX	<i>Import data from excel file</i>
--------------	------------------------------------

---

**Description**

Import data from excel file

**Usage**

```
dbImportXLSX(con, xlsx, ignore_genotype = TRUE, ignore_trait = TRUE, ...)
```

**Arguments**

con	a connection object as produced by dbConnect
xlsx	The path to excel file
ignore_genotype	Ignore genotype tables when importing
ignore_trait	Ignore trait table when importing
...	Other arguments. Supported arguments include <ul style="list-style-type: none"> <li>• extra_design: Extra columns in the experiment design.</li> <li>• tz: The time zone for the hourly climates.</li> </ul>

**Value**

No return values

---

dbInsertUpdateByRow	<i>Insert new rows or update existing rows to a specific table according to a specific column (unique) by each row</i>
---------------------	--

---

**Description**

Insert new rows or update existing rows to a specific table according to a specific column (unique) by each row

**Usage**

```
dbInsertUpdateByRow(con, table, data, unique_col = "name")
```

**Arguments**

con	A connection object as produced by dbConnect
table	The target table name
data	A data frame to write into table
unique_col	A character vector to indentify each row in the table

---

dbListTrials	<i>List all trials</i>
--------------	------------------------

---

**Description**

List all trials

**Usage**

```
dbListTrials(con)
```

**Arguments**

con                    a connection object as produced by dbConnect

**Value**

A data.frame for all trials in the data base

---

expdbConnect	<i>Connect to expDB</i>
--------------	-------------------------

---

**Description**

Connect to expDB

**Usage**

```
expdbConnect(filename)
```

**Arguments**

filename              The filename of SQLite

**Value**

a connection object as produced by dbConnect

**Examples**

```
## Not run:  
con <- connect('filename')  
  
## End(Not run)
```

expdbCreateDB            *create to expDB*

---

**Description**

create to expDB

**Usage**

```
expdbCreateDB(filename, system_traits = TRUE)
```

**Arguments**

filename            The filename of new expDB  
system\_traits      Whether to import system traits

**Value**

a connection object as produced by dbConnect

---

expdbDisconnect        *Didconnect to expDB*

---

**Description**

Didconnect to expDB

**Usage**

```
expdbDisconnect(con)
```

**Arguments**

con                a connection object as produced by dbConnect.

**Value**

no return values

**Examples**

```
## Not run:  
con <- connect('filename')  
disconnect(con)  
  
## End(Not run)
```

---

getIdByUniqueIndex     *Get index id by unique\_columns.*

---

### Description

Get index id by unique\_columns.

### Usage

```
getIdByUniqueIndex(
  con,
  table,
  data,
  unique_col = "name",
  data_col = unique_col,
  ignore_case = FALSE
)
```

### Arguments

con	a connection object as produced by dbConnect
table	the table name
data	A data frame to write into table
unique_col	A character vector to identify each row in the table
data_col	The column names in the data.frame
ignore_case	Whether ignore_case

### Value

id of unique\_col

---

harvestQuadratDetail     *Process quadrat (detail) harvestQuadratDetail*

---

### Description

Process quadrat (detail) harvestQuadratDetail

### Usage

```
harvestQuadratDetail(con, records)
```

**Arguments**

con                    a connection object as produced by dbConnect  
records                Phenotype records

**Value**

no return values

# Index

dbAddDesigns, [3](#)  
dbAddFertilization, [3](#)  
dbAddGene, [4](#)  
dbAddGeneAllele, [4](#)  
dbAddGenotype, [5](#)  
dbAddIrrigations, [5](#)  
dbAddLog, [6](#)  
dbAddMethods, [6](#)  
dbAddMets, [7](#)  
dbAddNodes, [7](#)  
dbAddPhenotype, [8](#)  
dbAddResearcher, [8](#)  
dbAddSites, [9](#)  
dbAddSource, [9](#)  
dbAddTraits, [10](#)  
dbAddTrials, [10](#)  
dbAddTrialSoil, [11](#)  
dbAddWeather, [11](#)  
dbAppendTable, [12](#)  
dbExportMet, [12](#)  
dbGenotypeCheckName, [13](#)  
dbGetDryWeightPerStem, [13](#)  
dbGetFertilization, [14](#)  
dbGetFieldMaturity, [14](#)  
dbGetFieldPopulation, [15](#)  
dbGetFieldStemNumber, [15](#)  
dbGetFieldTillerNumber, [16](#)  
dbGetGene, [16](#)  
dbGetGenotype, [17](#)  
dbGetIrrigation, [17](#)  
dbGetLog, [18](#)  
dbGetMetInfo, [18](#)  
dbGetOrganFinalLeafNumber, [19](#)  
dbGetOrganHaunIndex, [19](#)  
dbGetPhenotype, [20](#)  
dbGetPlantFlowering, [20](#)  
dbGetPlantHeading, [21](#)  
dbGetPlantStemElongation, [21](#)  
dbGetPlantStemNumber, [22](#)  
dbGetPlantTillerNumber, [22](#)  
dbGetSites, [23](#)  
dbGetSource, [23](#)  
dbGetTraits, [24](#)  
dbGetTrials, [24](#)  
dbGetWeather, [25](#)  
dbGetZadoksStage, [25](#)  
dbImportXLSX, [26](#)  
dbInsertUpdateByRow, [26](#)  
dbListTrials, [27](#)  
  
expdbConnect, [27](#)  
expdbCreateDB, [28](#)  
expdbDisconnect, [28](#)  
  
getIdByUniqueIndex, [29](#)  
  
harvestQuadratDetail, [29](#)