

Example SQL Commands

Updated October 2025

BRAHMS © Copyright, University of Oxford, 2025. All Rights Reserved

For licensing enquiries, contact brahms@innovation.ox.ac.uk

For technical enquiries, contact brahms@biology.ox.ac.uk

http://herbaria.plants.ox.ac.uk/bol/brahms/software/v8

Contents

Introduction to SQL commands 3
Clear all values from a field 3
Set a field to upper/lower case - also with a where clause 3
Copy data from a standard field to a custom field for the same table 5
Copy data from a custom field to a standard field for the same table 5
Move data from fieldx to fieldy in a related table 5
Transfer a value from one table to another based on where clauses 5
Append to a value from a custom field 6
Restricting an edit to tagged records using 'common table expression' (CTE) 6
SQL command to set custom field to null (or to any value) for tagged records 7
Update collection event gazetteer ID for tagged records 7
Set all tagged species taxstatus to 'zZz' for a given known as user name 7
Set all tagged species taxstatus to 'zZz' for a given knownas user name where taxstatus is currently xxx 7
Set all species taxstatus to 'zZz' for a given knownas user name here the tag is '#' 7
Transfer data from one or more standard fields to custom fields in common names 8
Update collectionevent image totals based on sum of all specimen images 8
Transfer data from living collection custom field to living accession custom field 8
Transfer data from a collection event custom field to a specimen custom field 9
Copy link field in RDE to another field 9

Introduction to SQL commands

The following SQL commands can be executed from within BRAHMS using the SQL tab on the main Query Tool. Users need access/permission to use this tool. Commands should be executed with care as they work directly on BRAHMS data, **by-passing standard BRAHMS data rules**.

SQL commands can **only be used** by those with permission to access the SQL command tab on the main BRAHMS Query tool.

- Commands may differ depending on the file store you are using. The commands listed here, where possible, are structured to work in MSSQL Server, PostgreSQL and SQLite databases. RDE files are always stored in SQLite. If different command formats are required for separate data stores, this will be indicated in these examples.
- Where commands are restricted to tagged records, where possible, we use a special construct known as Common Table Expressions (CTE) and thus the 'With mytags' syntax. Bear in mind that tags are user specific – hence the username must be included in these commands. Also note that there is no ; after the 'With mytags …' expression.
- SQL commands always end with a ";"
- You can run multiple commands at the same time.
- SQL commands can be run from any table. For example, if running a command that edits species records, you do not have to run the command from the species table. The SQL command itself specifies the table/field(s) to be edited.
- SQL commands are not case sensitive. Text string matches using 'like' are case sensitive.
- You can include comments prefixed by -- . These lines are not executed.
- Take care that single apostrophes are simple apostrophes rather than Word apostrophes. 'abc' is correct while 'abc' will fail.
- Commands are usually written with line breaks and indents to make more readable. White space and blank lines are ignored.

Clear all values from a field

Without a 'where' clause, this command edits all records in the species table.

```
update species set taxstatus = null;
```

Commands can be restricted using a where clause, in this case where the calculated full name starts 'Pinus a'. Note the use of the wildcard %.

```
update species set taxstatus = null where calcfullname like 'Pinus a%'
```

Set a field to upper/lower case - also with a where clause

Note: if the 'where' clause is omitted, the command would apply to all records.

```
update species set taxstatus = upper(taxstatus) where taxstatus is not null and calcfullname like 'pinus a%';
update species set taxstatus = lower(taxstatus) where taxstatus is not null and calcfullname like 'pinus a%';
```

Copy data from a standard field to a custom field for the same table

The tables and fields used in the following example can be replaced by those you wish to use. The linkeddata table stores your custom field data. 'str06' refers to the field of that name and type (here a text field). When using any command that refers custom fields, make sure you check the linkeddata table field name is correct for the data you want to edit. Check using the Column Manager custom columns Editor tab.

-- first, make sure a linked data record is added for all species records and update the field linkeddataid in the species table. His ID field is always the same as the table's primary key, in this case speciesid.
-- copy taxstatus to a custom field

```
insert into linkeddata (linkeddataid, databaseprojectid)
select speciesid, databaseprojectid from species
where linkeddataid is NULL;
update species set linkeddataid = speciesid;

update linkeddata set  linkeddata.str06 = sp.taxstatus
from species as sp
where sp.linkeddataid = linkeddata.linkeddataid;
```

Copy data from a custom field to a standard field for the same table

```
update species set taxstatus = lnk.str06
from linkeddata as lnk
where species.linkeddataid = lnk.linkeddataid;
```

Move data from fieldx to fieldy in a related table

```
update propagationrecords set requesteddeliverycontainer = q1.requesteddeliverycontainer from
(select propagationrecordid, requesteddeliverycontainer from  gardenprojectrsos) as q1
where  propagationrecords.propagationrecordid = q1.propagationrecordid;


update propagationrecords set prodstartdate = q1.dat10
from (select dat10, propagationrecordid from propagationrecordsgridview) as q1
where propagationrecords. propagationrecordid = q1. propagationrecordid
```

Transfer a value from one table to another based on where clauses

In this example, the species ID is copied from the living accessions table to the related collection event table (= wild origin data), restricting this to a particular species.

```
UPDATE collectionevents set speciesid = livingaccessions.speciesid
from livingaccessions where collectionevents.collectioneventid = livingaccessions.collectioneventid
and collectionevents.speciesid = '2ce235b1-c4fa-4b4b-988a-64271cbc3989';
```

NB. If you wanted to this for several species, you can use a comma separated list as follows:

and collectionevents.speciesid in ('2ce235b1-c4fa-4b4b-988a-64271cbc3989', 'nextguid', 'nextguid', …);

Append to a value from a custom field

This command appends a custom field value to an existing to a standard field. Note, if either of the strings is null, the result of the command is null and hence, nothing is copied (see below). If using PG or SQlite, replace the + with || .

```
update species
set plantspread = plantspread + ' - ' + sg.str36
from speciesgridview as sg
where sg.speciesid = species.speciesid and sg.str36 is not null;
```

To also copy the value where plantspread is null, the following can be used. This will only add the ' – ' when needed.

```
update species
set plantspread = case when sg.plantspread is null then sg.str36 else sg.plantspread + ' - ' + sg.str36 end
from speciesgridview as sg
where sg.speciesid = species.speciesid and sg.str36 is not null;
```

Restricting an edit to tagged records using 'common table expression' (CTE)

The following example updates the botanicgardenid field in the living accession table for tagged records only.

```
With mytags(recordid) as (select recordid from tags where tags.brahmsuserid = (select top 1 brahmsuserid from brahmsusers where knownas = 'your known as username'))

UPDATE livingaccessions
set botanicgardenid = 'dafae258-2446-470a-a451-0fb46629356b'
from mytags where mytags.recordid = livingaccessionid;
```

Another example updating the botanicgardenlocalityid field in the living collections table for tagged records only.

```
With mytags(recordid) as (select recordid from tags where tags.brahmsuserid = (select top 1 brahmsuserid from brahmsusers where knownas = 'denis'))

UPDATE livingcollections
set botanicgardenlocalityid = '618f4397-c16b-4528-a4a7-006429228cdd'
from mytags where mytags.recordid = livingcollectionid;
```

SQL command to set custom field to null (or to any value) for tagged records

```
With mytags(recordid) as (select recordid from tags where tags.brahmsuserid = (select top 1 brahmsuserid from brahmsusers where knownas = 'DESKTOP-5OS1ASA\denis' and dataviewname = 'CollectionEventDataView')
UPDATE linkeddata
set str41 = null       --or = 'my text'
from mytags where mytags.recordid = linkeddata.linkeddataid;
```

Update collection event gazetteer ID for tagged records

```
With mytags(recordid) as (select recordid from tags where tags.brahmsuserid = (select top 1 brahmsuserid from brahmsusers where knownas = 'DESKTOP-5OS1ASA\denis')
UPDATE collectionevents
set gazetteerid = '8d44cbb1-8080-4ffb-b4e9-7618f74de325'
from mytags
where mytags.recordid = collectionevents.collectioneventid;
```

Set all tagged species taxstatus to 'zZz' for a given known as user name

```
With mytags(recordid) as (select recordid from tags where tags.brahmsuserid = (select top 1 brahmsuserid from brahmsusers where knownas = 'DESKTOP-5OS1ASA\denis')

Update species
set taxstatus = 'zZz'
from mytags
where mytags.recordid = species.speciesid;
```

Set all tagged species taxstatus to 'zZz' for a given knownas user name where taxstatus is currently xxx

```
With mytags(recordid) as (select recordid from tags where tags.brahmsuserid = (select top 1 brahmsuserid from brahmsusers where knownas = 'DESKTOP-5OS1ASA\denis')
Update species
set taxstatus = 'zZz'
from mytags
where mytags.recordid = species.speciesid and taxstatus = 'xxx';
```

Set all species taxstatus to 'zZz' for a given knownas user name here the tag is '#'

```
With mytags(recordid) as (select recordid from tags where tags.brahmsuserid = (select top 1 brahmsuserid from brahmsusers where knownas = 'DESKTOP-5OS1ASA\denis' and tagcharacter = '#')
Update species
set taxstatus = 'zZz'
from mytags
where mytags.recordid = species.speciesid;
```

Transfer data from one or more standard fields to custom fields in common names

The same syntax can be used for other tables that have custom fields. Note that in this example, more than one field in transferred in the same command. Also in this example, the field language is wrapped in quotes as in some cases it is a reserved programming word. Note that multiple set commands are separated by “,”

```
 -- first, general command to ensure linked data record exists

insert into linkeddata (linkeddataid, databaseprojectid)

select commonnameid, databaseprojectid from commonnames
where linkeddataid is NULL;

 -- ensure the field linkeddataid in common names is updated (it’s actually a copy of the primary key)
 update commonnames set linkeddataid = commonnameid where linkeddataid is null;


-- finally do the transfer for 1 to many fields at one time...
update linkeddata set  str01 = cn."language", str02 = cn.vernacularname
from commonnames as cn
where cn.linkeddataid=linkeddata.linkeddataid;
```

Update collectionevent image totals based on sum of all specimen images

In the Collection Event table, this command updates the calc field #images with the specimen image total, overwriting the default Collection Events images total. Note that the next revision of BRAHMS will include a separate field in Collection Events for specimen image total.

```
UPDATE  collectionevents
set calcimages = q1.totimages from
(select collectioneventid, sum(calcimages) as totimages
from specimensgridview group by collectioneventid) as q1
where collectionevents. collectioneventid = q1.collectioneventid;
```

Transfer data from living collection custom field to living accession custom field

This commands illustrates how you could transfer one or more values from custom fields in one table to custom fields in a related table. Note that it uses the gridview rather than the table itself as this avoids the use of joins. The custom fields are already in the gridview and can be accessed directly there.

```
-- First check custom columns exist, create if needed.

insert into linkeddata (linkeddataid, databaseprojectid) select livingaccessionid, databaseprojectid from livingaccessions

where linkeddataid is NULL;
```

```
-- Ensure the field linkeddataid in living accessions is updated (it's a copy of the primary key)
update livingaccessions set linkeddataid = livingaccessionid where linkeddataid is null;
-- Finally transfer 1 to many fields at one time. In this example, 2 fields are processed.
-- Not that the custom fields must be created first. Check you have the correct names.
update linkeddata set date01 = lc.date01, str13 = lc.str11
from livingcollectionsgridview as lc
where lc.livingaccessionid = linkeddata.linkeddataid;
```

Transfer data from a collection event custom field to a specimen custom field

In this example, similar to the above, custom data are moved in different tables. However, as the collection event table does not have a specimen ID, it's necessary to do a join.

```
-- First check custom columns exist, create if needed.
insert into linkeddata (linkeddataid, databaseprojectid) select specimenid, databaseprojectid from specimens
where linkeddataid is NULL;
-- Ensure the field linkeddataid is updated (it's a copy of the primary key)
update specimens set linkeddataid = specimenid where linkeddataid is null;
-- Finally transfer 1 to many fields at one time. In this example, 1 field is processed.
update linkeddata set str17 = ce.str05
from collectioneventsgridview as ce inner join specimens on ce.collectioneventid = specimens. collectioneventid
where specimens.specimenid = linkeddata.linkeddataid;
```

Copy link field in RDE to another field

This command copies the content of a custom field in RDE (seed) to a standard field limiting to 30 characters.

```
update rdeseedaccessions  set minoradminname = case when length(str01) >30 then
substr(str01,1,30) else str01 end   from rdelinkeddata
where rdeseedaccessions.rdelinkeddataid = rdelinkeddata.rdelinkeddataid;
```