**Session 11: Modelling with CANDY – advanced run-setup & result analysis**

**Aims:**

1. Understand batch-file setup including
   1. Pre-processing
   2. Post-processing
2. Basic result analysis possibilities
   1. Statistical evaluation
   2. plotting
3. Along the way: understand upcoming errors and mistakes

**Important to note:**

While it is easier to process data via the graphical user interfaces, this will eventually backfire when you have to process many simulation-runs and tons of data! This is why we strongly recommend using R software as soon as possible. If you are not familiar with R, we are here to help and teach you, plus there is an easy to understand pre-made script for all scripting tasks that you can request! And now we wish you lots of fun and enjoy modelling!

**Step up your CANDY-game with batch files**

**BASIC THEORIE ABOUT CANDY BATCH FILES**

Based on our previous simulation run in Session 10, we can further explore on how to improve our CANDY simulation setup, just before we jump into basic result analysis.

For starter, executing each simulation run via the CANDY GUI is, again, not very efficient in the long run. It is great for a single simulation run, but oh boy, will you click yourself to oblivion if you have to do thousands and thousands of runs (trust me, I’ve been there) … And as we all know, modellers are lazy!

So fret not! There is a beautiful solution coming in the form of batch files, allowing for an automized setup of simulation runs with the help of pre-and postprocessing scripts to update our database.

The basic idea of the batch files is to provide initial run conditions via the master batch file which then calls…

… a ‘slave’ batch file, containing extended simulation run information(s) including

* Model switches
* Start and enddate
* Database to use
* …

for each individual simulation run.

The slave batch calls for each individual simulation run always the

* pre-processing script
* CANDY exe itself
* postprocessing script

until all runs are successfully simulated. The great thing is, that you can define multiple simulation run parameters in the master batch which are used within the respective slave batch!

**Info:** The basic outline for the batch file setup and further information about the switches is given in the manual **CANDY22\_manual**!

The great advantage of batch files encompassing pre-and postprocessing, is the great possibilities including the ability to store results from multiple runs in new tables, already filter them and much more. Your own imagination (and SQL syntax knowledge) is the limit here 😉.

With that out of the way, let us start to create our own batch file setup. As example, we will first, recreate the simulation run we previously set up in Session 10 with batch files.

**SETUP A SIMULATION RUN WITH BATCH FILES**

To execute the ‘base’ simulation run of Session 10, we have to create the four files

* master\_batch.bat
* slave\_batch.bat
* preprocessing.sql
* postprocessing.sql

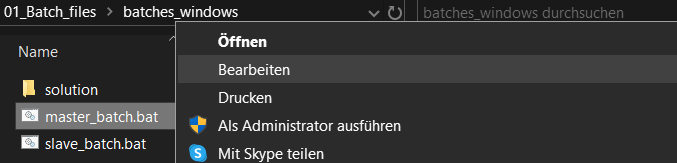
We will create them step by step together!

**MASTER BATCH**

There is an already prepared, partially filled out master batch which you can find in folder ***01\_Batch\_files/batches\_windows***

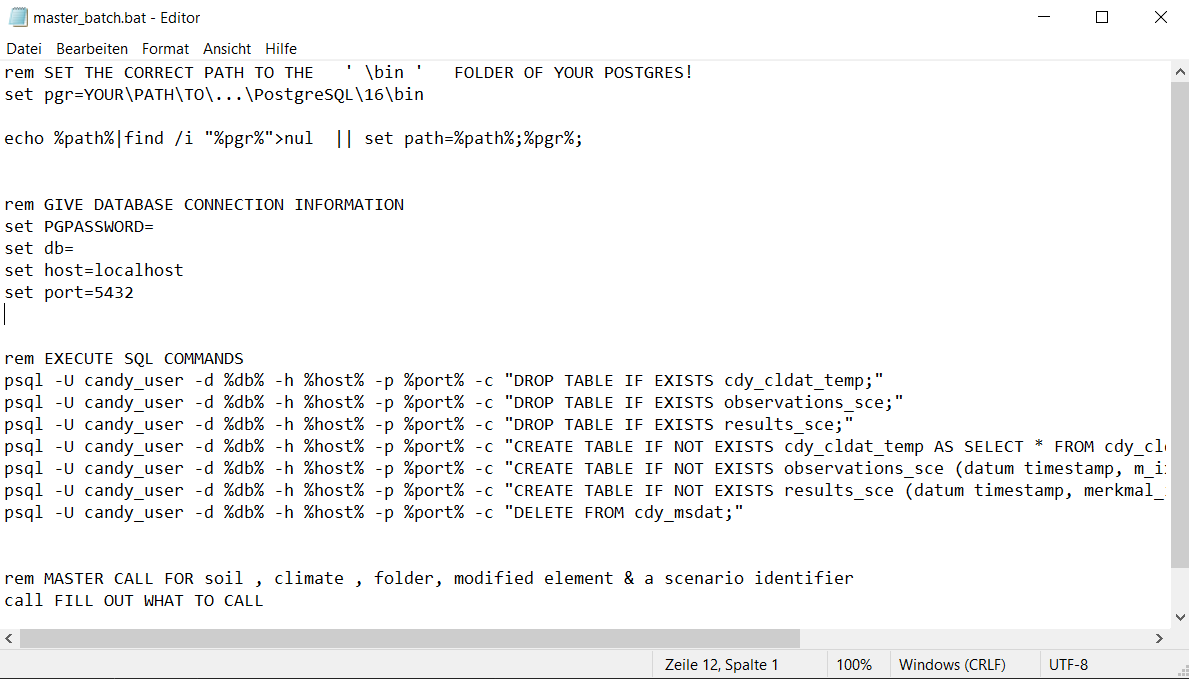
**!!! DO NOT DOUBLE CLICK THIS FILE !!! (yet)**

**RIGHT CLICK on the file -> edit**



A double click will EXECUTE the file! We will do that later. After you opend the batch file, we need to fill out the missing parts!

**FILL OUT the missing parts (marked red boxes):**



**SOLUTION:**

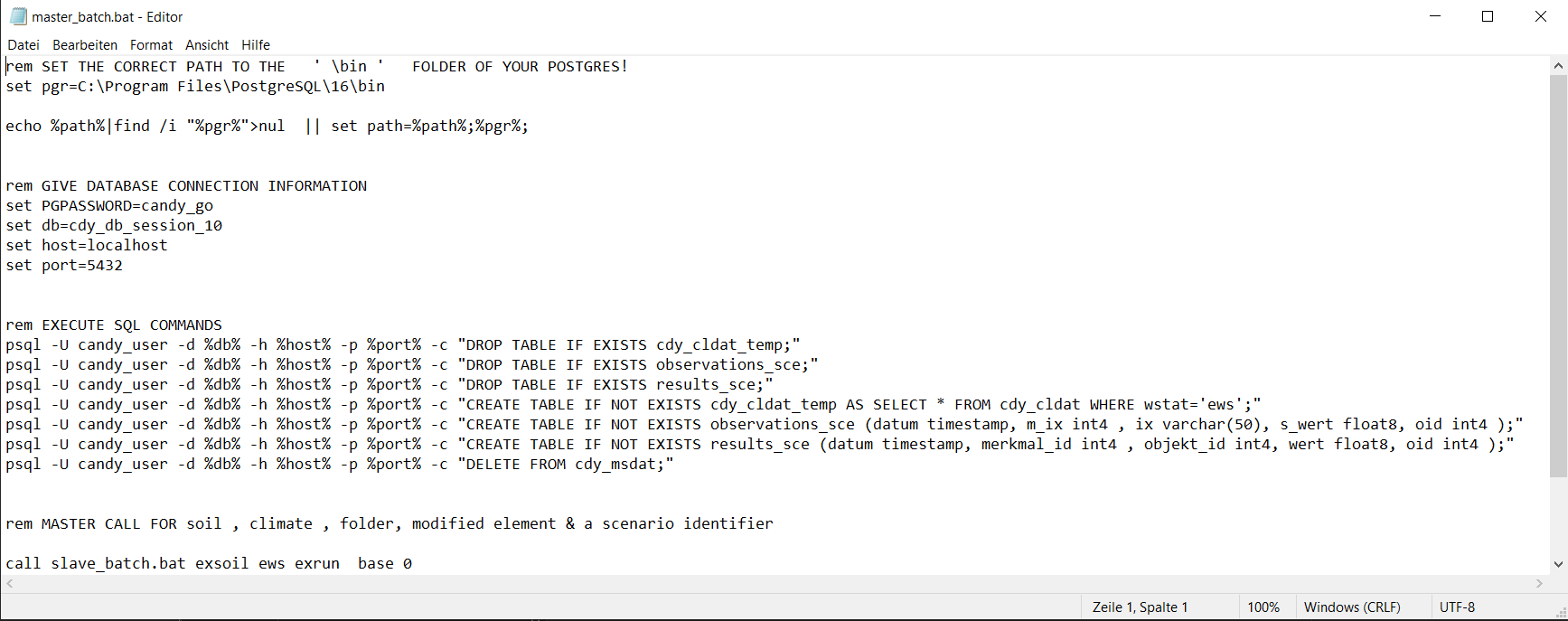
Set pgr=*provide folderpath to …/PostgresSQL/16/bin*

set PGPASSWORD=candy\_go

set db =cdy\_db\_session\_10

call slave\_batch.bat exsoil ews exrun base 0

THE FINALISED MASTER BATCH SHOULD NOW LOOK SOMETHING LIKE THIS



**PART 1**

**PART 2**

**PART 3**

**PART 4**

**From top to bottom, this file now:**

**PART 1 - defines the path to the \bin folder of postgres**

INFO: ***‘set’*** assigns everything that follows the equal sign (=) to the value of variable

For example: *set variable=CANDY\_love* - now variable contains the string ‘CANDY\_love’; to use it, we simply add a *%* in front and behind the variable: *%variable%*

**PART 2 - define the database connection parameters**

**PART 3 - execute .sql scripts or commands with *psql***

INFO: ‘psql’ is a software that comes with the postgres installation. With it you can do database manipulations outside of DBeaver, PgAdmin4 or R-Studio!

-c execute a **SQL command**

-f execute a **file** (a prewritten .sql file for example)

**PART 4 - calls the slave batch with defined parameters**

The parameters listed behind the slave\_batch.bat file (separated by simple whitespace) are passed on from the master batch to the slave file. In the slave file we can access these parameters by position!

For example: *call slave\_batch.bat* parameter\_1 parameter\_2 parameter\_3 … parameter\_n

* in the slave batch writing %1 then refers to parameter\_1, %3 to parameter\_3 …

**helpful CMD – Syntax**

**set** –assigns everything that follows the equal sign (=) to the value of variable

**rem** –similar to # in R – is used for comments in the file

**echo** –prints everything behind it into the command shell

**cd** –change directory

**/d** – addition to **cd** to specify a drive change (e.g. if CANDY is located on C:/ but your workingfolder is on D:/, you need to add **/d** after **cd**

**call** – calls the command and WAITS until finished bevor jumping to the next

**run** – calls the command WITHOUT waiting if the previous one is done. Continues to call commands until all runs are finished, or a **call** command is detected. Great for simultaneous simulation runs.

**SLAVE BATCH**

There is an already prepared, partially filled out slave batch which you can find in folder ***01\_Batch\_files/batches\_windows***

**!!! DO NOT DOUBLE CLICK THIS FILE !!! (yet)**

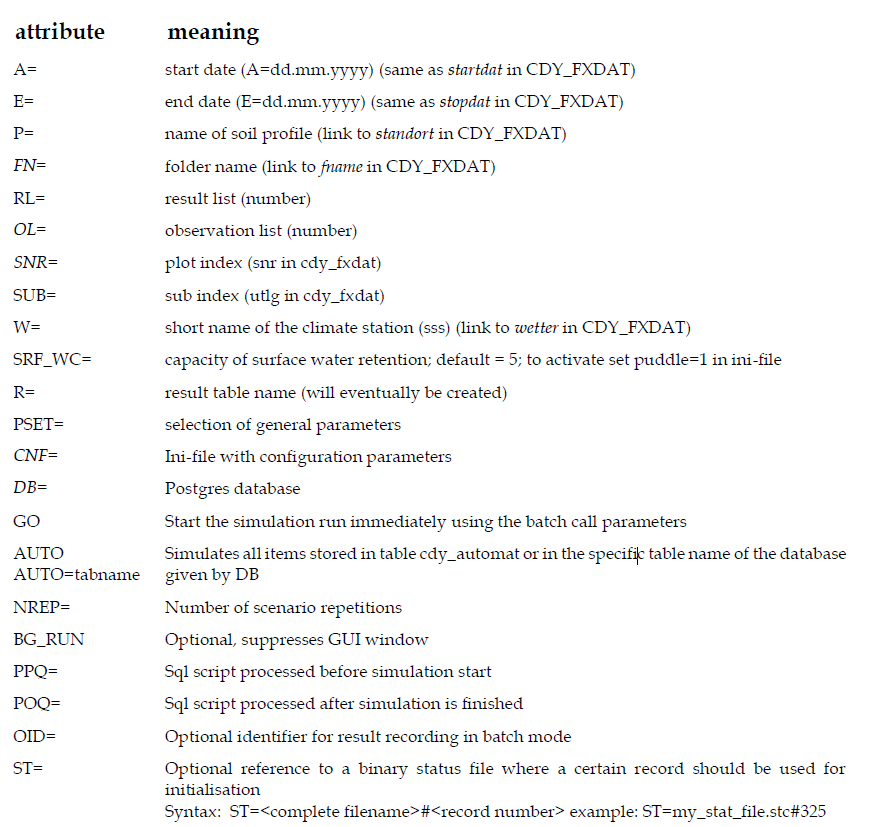
**RIGHT CLICK on the file -> edit**

When opened, you only have to fill out the actual command line, which calls the CANDY exe with respective model switches and general simulation information.

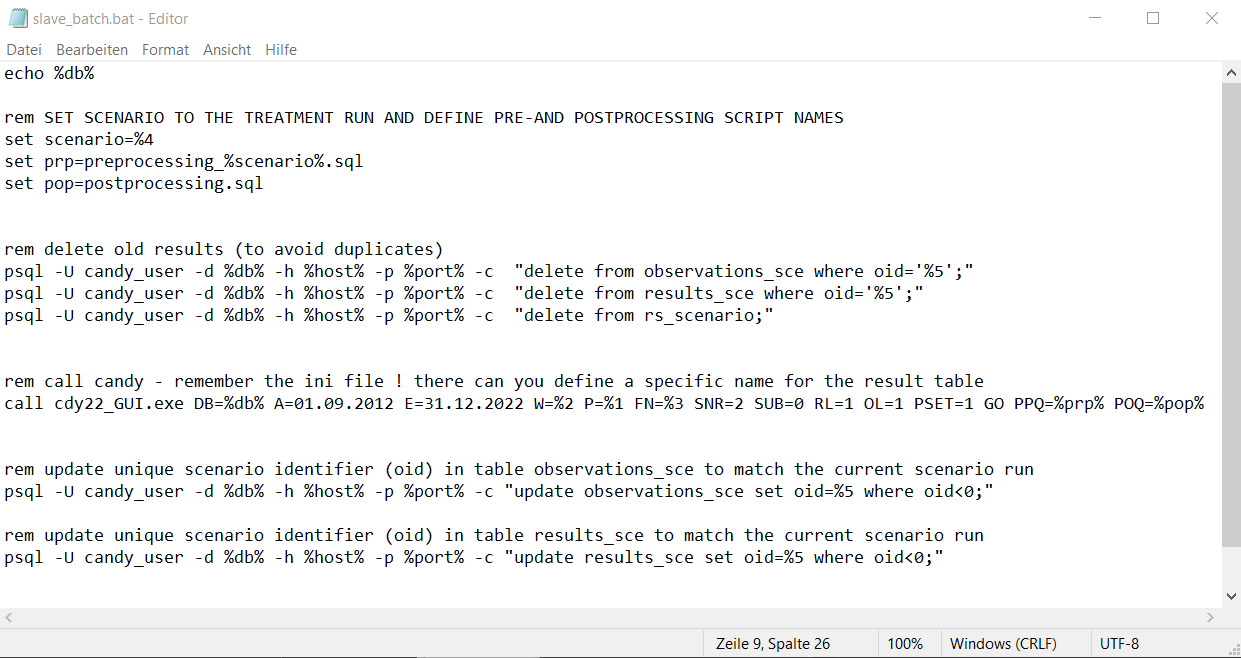
**FILL OUT the missing parts (marked red boxes) with the following listed entries. You can write them in ONE line separated just by a whitespace (hit your spacebar).**

1. Parameters to call CANDY and set correct switches; copy the***CURSIVE***parts:
   1. ***call cdy22\_GUI.exe*** *-* calls the CANDY exe
   2. ***DB=%db%***- database, defined in the master\_batch.bat
   3. ***A=01.09.2012*** *-* startdate of simulation [dd.mm.yyyy]
   4. ***E=31.12.2023*** - enddate of simulation [dd.mm.yyyy]
   5. ***W=%2*** - second entry in master\_batch.bat
   6. ***P=%1*** - first entry in master\_batch.bat
   7. ***FN=%3*** - third entry in master\_batch.bat
   8. ***SNR=2***  - plot number
   9. ***SUB=0***  - sub plotnumber
   10. ***RL=1*** - result list to use (from Session 10)
   11. ***OL=1*** - observation list to use (from Session 10)
   12. ***PSET=1*** - the standard parameterset
   13. ***GO*** - starts the simulation
   14. ***PPQ=%prp%*** - execute preprocessing script
   15. ***POQ=%pop%*** - execute postprocessing script

Helpful explanations of CANDY batch model switches:



Your slave batch file should now look like this:



**ALL DONE?** **SAVE THE FILE**

**CAN YOU, from top to bottom, explain what is done in this file?**

Now let us add some more simulation runs additional to our base simulation, followed by pre-and postprocessing scripts.

For our singular base simulation run, we actually do not really need a pre or postprocessing script. However, since we want to add MORE runs later on, we will already prepare the scripts for the base simulation

**PRE-PROCESSING**

For this pre-processing script, we need to create a new .sql file! Again, you can find a pre-written file in the folder XXX – don’t worry, here you can safely double click to open it.

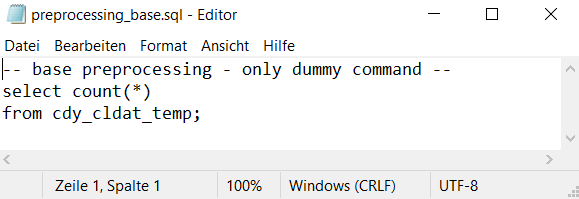
Since we need no pre-processing for the base run, we will add a dummy command. However, we will create THREE MORE later with actual functions!

**OPEN THE FILE and copy the following:**

*select count(\*)*

*from cdy\_cldat\_temp;*

**SAVE THE FILE and RENAME to *preprocessing\_base.sql***



**POST-PROCESSING**

For the post-processing script, we again need to create a new .sql file! You can find a pre-written file in the folder XXX – double click to open it.

ALL runs will use the same post-processing script, so we only need ONE, other than the pre-processing.

After each simulation run the post processing script will:

1. delete manipulated climate from cdy\_cldat
2. insert UNMANIPULATED climate from our temporary climate table (cdy\_cldat\_temp) into our actual climate table (cdy\_cldat)
3. copy the virtual observations from cdy\_msdat into a new table, observations\_sce. Every new run stored here gets a unique identification key!
4. copy the results from rs\_scenario into a new table, results\_sce.  
   Every new run stored here gets a unique identification key!

Step 3) and 4) are MANDATORY when executing multiple simulation runs with batch files. Cdy\_msdat and the result table (in this case rs\_scenario) WILL ALWAYS be overwritten with the simulations of the last run!

**OPEN THE FILE and COPY:**

*-- delete manipulated climate --*

*delete from cdy\_cldat*

*where wstat='ews';*

*-- insert climate from temporary table into cdy\_cldat --*

*Insert into cdy\_cldat*

*select \* from cdy\_cldat\_temp;*

*-- copy virtual results --*

*Insert into observations\_sce (datum,m\_ix,ix,s\_wert,oid)*

*select datum,m\_ix,ix,s\_wert,-99 as oid*

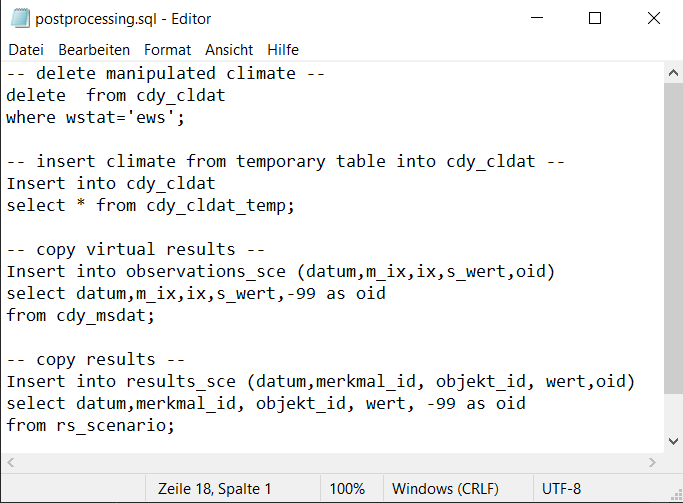
*from cdy\_msdat;*

*-- copy results --*

*Insert into results\_sce (datum,merkmal\_id, objekt\_id, wert,oid)*

*select datum,merkmal\_id, objekt\_id, wert, -99 as oid*

*from rs\_scenario;*

**ALL DONE? Save the file as *postprocessing.sql***

**ADD NEW SIMULATION RUN SCENARIOS TO THE BATCH CALL**

For the next simulation runs, we want to assess, how future climate will impact Carbon and Nitrogen Dynamics. Since we don't have actual future climate data, we will perform some weather manipulations with pre-and postprocessing to see how isolated temperature, isolated precipitation, and their combined effects influence our simulation results.

To start, we add three new calls to our master batch. Three new calls for three weather manipulations!

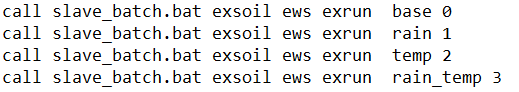
**OPEN THE master\_batch.bat we previously created! Remember, NO DOUBLE CLICK!**

ADD THREE NEW ENTRIES for the new runs UNDER the base run:

*slave\_batch.bat exsoil ews exrun rain 1*

*slave\_batch.bat exsoil ews exrun temp 2*

*slave\_batch.bat exsoil ews exrun rain\_temp 3*



**ALL DONE? Good, save the file and close it**

We do not have to change any entries in the slave batch file, since we provide all the necessary parameters to change the run with the master batch. However, we need THREE more pre-processing files to manipulate the climate:

* Increased precipitation - base precipitation x1.3
* Increased temperature – base temperature + 1.5
* Increased precipitation AND temperature – both of the above

Let’s start by writing our pre-processing scripts:

***CREATE THREE NEW preprocessing scripts***

1. *COPY the previously created pre-processing script*
2. *REPLACE the old entry by COPYING the following SQL statement into the file:*

*-- update cdy\_cldat --*

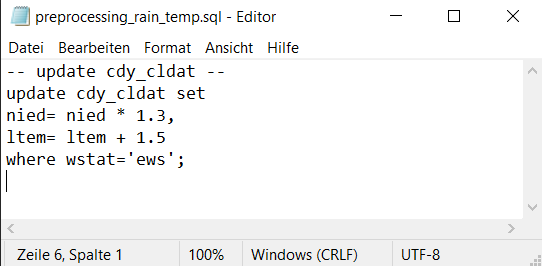
*update cdy\_cldat set*

*nied= nied \* 1.3,*

*ltem= ltem + 1.5*

*where wstat='ews';*

1. *SAVE the new file as:* ***preprocessing\_rain\_temp.sql***
2. *COPY THE BASE file TWO MORE times and update the SQL statement to match the other manipulations and save files accordingly as:*
   1. *preprocessing\_temp.sql*
   2. *preprocessing\_rain.sql*



**ALL DONE? Good! Now we are very close to finally be able to double click that master\_batch.bat file!! Only two more little steps!**

Remember what I told you earlier about data manipulation in a databank when you are still inexperienced? Make a backup of cdy\_cldat by executing following script in a DBeaver SQL script (we will do it together).

Ein Bild, das Text, Screenshot, Software, Multimedia-Software enthält.

Automatisch generierte Beschreibung

-- backup cdy\_cldat as cdy\_cldat\_backup --

create table cdy\_cldat\_backup

as table cdy\_cldat;

Ein Bild, das Text, Software, Multimedia-Software, Grafiksoftware enthält.

Automatisch generierte Beschreibung

Now that we updated our master and slave batch files and created our pre-and postprocessing scripts, lets open them one by one and check if all entries are correct and matching.

IF YOU WERE NOT ABLE TO CREATE ANY OF THE FILES, JUST USE THE FILES STORED UNDER …01\_Batch\_files/batches\_windows/solution.

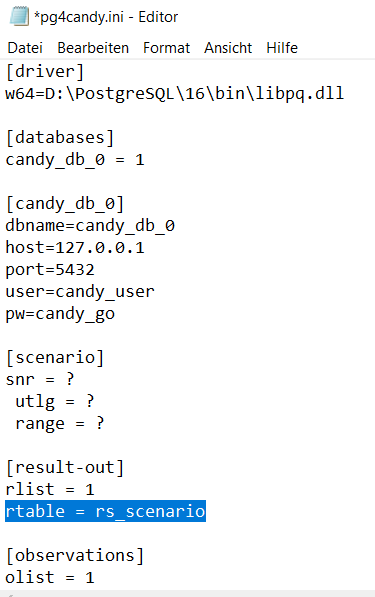
As a last step, we have to copy all the new files into our working folder where the CANDY exe and its initialization file is located:

[yourDataPathUpToHere]**/Session\_11/00\_executable**

**Speaking of .ini file, there is one last step we have to do before starting the runs!** We must define the name of the result\_table!

**Open pg4candy.ini file and set the name of the result table accordingly!**

*rtable = rs\_scenario*

****

**DONE? GOOD!! Save the file**

NOW, THE MOMENT HAS COME! **DOUBLE CLICK THAT master\_batch.bat** AND ENJOY AUTOMATED CANDY RUNS!

**The next part of the session will be done in R!**

**OPEN R-SCRIPT session\_11\_result\_sorting.R   
and together we will continue**

**CANDY AUTOMAT**

With recent Versions of CANDY, there is an even more efficient way to execute multiple simulation runs in the form of the automat table. If we still have time and capacity, we may explore this simulation setup at the end.

The same problem can be handled using an autolist: **cdy\_automat**

Of course, we first need to create and FILL it with our simulation run scenarios. The sql script create\_automat.sql includes TWO steps:

First, the EMPTY cdy\_automat will be created (also useful for your future adventures)

SECOND, the table is filled with the example runs from this session.

The example shows the standard table cdy\_automat plus two additional columns to control the climate data manipulations.

The necessary preprocessing relates to the table cdy\_automat and uses the parameters of the currently simulated task (simtag=-1) to update climate and delete former results from observations\_sce.

**Preprocessing\_automat.sql:**

*update cdy\_cldat set nied=cdy\_cldat.nied\*(select prec\_manip from cdy\_automat where simtag=-1),*

*ltem=cdy\_cldat.ltem+(select temp\_manip from cdy\_automat where simtag=-1)*

*from cdy\_automat where cdy\_automat.simtag=-1 ;*

*delete from observations\_sce where oid=(select obj\_id from cdy\_automat where simtag=-1);*

Here again, we need postprocessing (postprocessing\_automat.sql) to revert the changes of climate data and to collect the results in cdy\_msdat into our permanent result table observations\_sce:

**Postprocessing\_automat.sql:**

*delete from cdy\_cldat where wstat='ews';*

*Insert into cdy\_cldat select \* from cdy\_cldat\_temp;*

*Insert into observations\_sce (datum, m\_ix, ix, s\_wert, oid)*

*select datum, m\_ix, ix, s\_wert, (select obj\_id from cdy\_automat where simtag=-1) as oid*

*from cdy\_msdat;*

**WITH THAT you are best prepared to automize your simulation runs! You can try to realise the particular example in cdy\_db\_session\_10!**

**SIMPLY COPY the batch + pre-and postprocessing files into a folder, where the CANDY.exe as well as the pg4Candy.ini file is located.**

**For more information about batches and especially the cdy\_automat, refer to the CANDY22\_manual :)!**

**THANK YOU VERY MUCH FOR STICKING UNTIL THE END! WE HOPE YOU LEARNED A LOT AND ARE NOW ABLE TO DO SOME AWESOME SIMULATION RUNS 😊!**

