DEFINITION OF MySQL TABLES
In GlobalFungi database
https://globalfungi.com/

IMPORTANT TABLES →

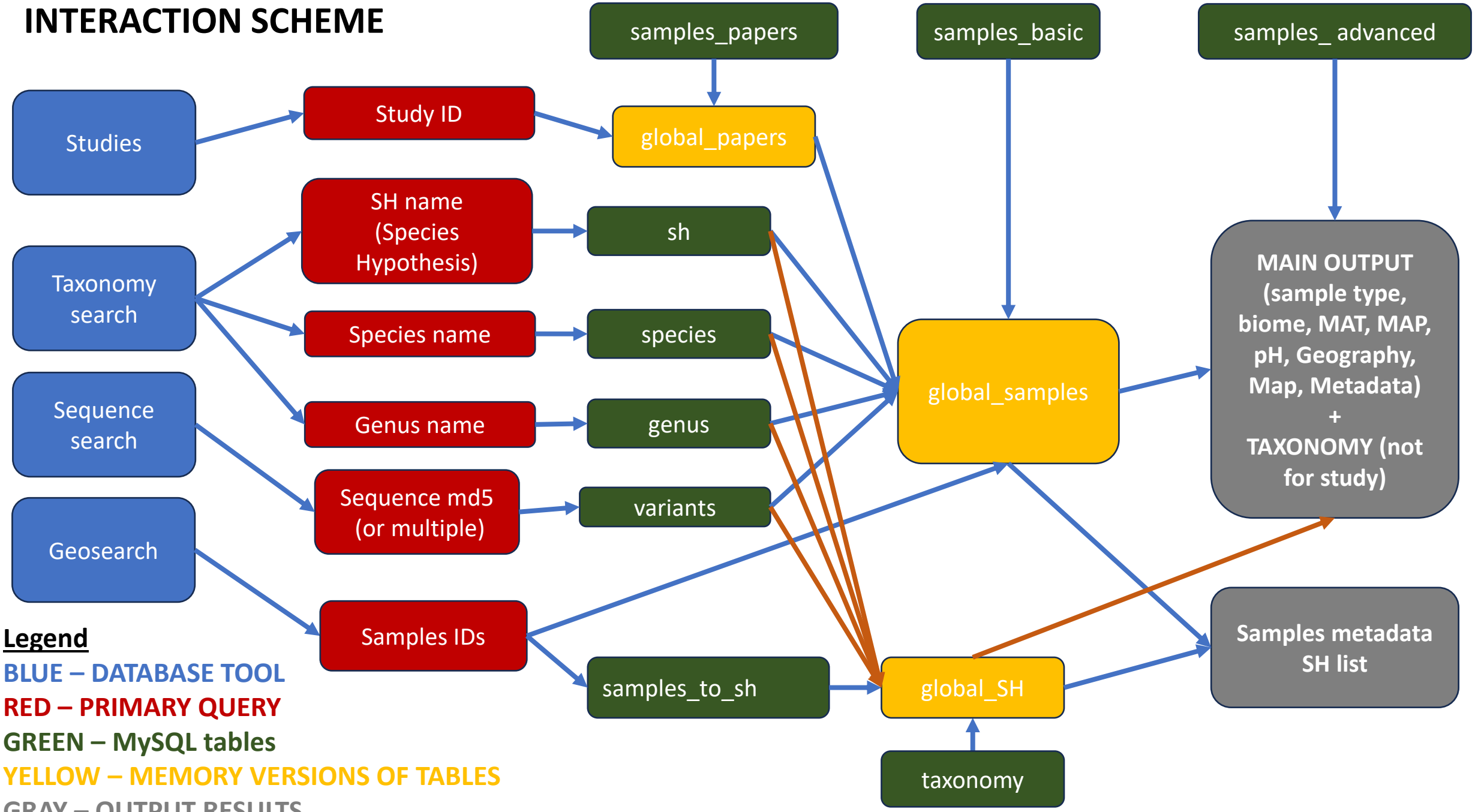
samples_advanced
samples_basic
samples_papers

taxonomy

variants
sh
genus
species

samples_to_sh

INTERACTION SCHEME



Legend

BLUE – DATABASE TOOL

RED – PRIMARY QUERY

GREEN – MySQL tables

YELLOW – MEMORY VERSIONS OF TABLES

GRAY – OUTPUT RESULTS

```
####################
# SAMPLES ADVANCED #
####################

CREATE TABLE IF NOT EXISTS `samples_advanced` (
  `id` int NOT NULL PRIMARY KEY,
  `sample_name` VARCHAR(128) NOT NULL,
  `sample_description` TEXT NOT NULL,
  `sequencing_platform` VARCHAR(16) NOT NULL,
  `target_gene` VARCHAR(7) NOT NULL,
  `primers_sequence` VARCHAR(256) NOT NULL,
  `sample_seqid` VARCHAR(256) NOT NULL,
  `sample_barcode` VARCHAR(128) NOT NULL,
  `elevation` INT,
  `MAT_study` FLOAT,
  `MAP_study` FLOAT,
  `Biome_detail` VARCHAR(64) NOT NULL,
  `country` VARCHAR(64) NOT NULL,
  `month_of_sampling` VARCHAR(32) NOT NULL,
  `day_of_sampling` VARCHAR(16) NOT NULL,
  `plants_dominant` TEXT NOT NULL,
  `plants_all` TEXT NOT NULL,
  `area_sampled` FLOAT,
  `number_of_subsamples_from` INT,
  `number_of_subsamples_to` INT,
  `sampling_info` TEXT NOT NULL,
  `sample_depth_from` FLOAT,
  `sample_depth_to` FLOAT,
  `extraction_DNA_mass_from` FLOAT,
  `extraction_DNA_mass_to` FLOAT,
  `extraction_DNA_size` VARCHAR(256) NOT NULL,
  `extraction_DNA_method` VARCHAR(256) NOT NULL,
  `total_C_content` FLOAT,
  `total_N_content` FLOAT,
  `organic_matter_content` FLOAT,
  `pH_study` FLOAT,
  `pH_method` VARCHAR(12) NOT NULL,
  `total_Ca` FLOAT,
  `total_P` FLOAT,
  `total_K` FLOAT,
  `sample_info` TEXT NOT NULL,
  `location` VARCHAR(256) NOT NULL,
  `area_GPS_from` FLOAT,
  `area_GPS_to` FLOAT,
  `ITS1_extracted` INT NOT NULL,
  `ITS2_extracted` INT NOT NULL
);
ALTER TABLE samples_advanced ADD INDEX(id);
```

```
#################
# SAMPLES BASIC #
#################

CREATE TABLE IF NOT EXISTS `samples_basic` (
  `id` int NOT NULL PRIMARY KEY,
  `paper` int NOT NULL,
  `permanent_id` VARCHAR(16) NOT NULL,
  `sample_type` VARCHAR(32) NOT NULL,
  `latitude` float NOT NULL,
  `longitude` float NOT NULL,
  `continent` VARCHAR(14) NOT NULL,
  `year_of_sampling_from` int,
  `year_of_sampling_to` int,
  `Biome` VARCHAR(32) NOT NULL,
  `primers` VARCHAR(128) NOT NULL,
  `MAT` FLOAT,
  `MAP` FLOAT,
  `pH` FLOAT,
  `SOC` FLOAT,
  `ITS_total` int NOT NULL,
  `manipulated` TINYINT(1) NOT NULL
);
```

```
################
# PAPERS TABLE #
################

CREATE TABLE IF NOT EXISTS `samples_papers` (
  `id` int NOT NULL PRIMARY KEY,
  `add_date` VARCHAR(10) NOT NULL,
  `year` int NOT NULL,
  `title` TEXT NOT NULL,
  `authors` TEXT NOT NULL,
  `journal` VARCHAR(128) NOT NULL,
  `doi` VARCHAR(64) NOT NULL,
  `contact` TEXT NOT NULL,
  `manipulated` TINYINT(1) NOT NULL
);
```

SAMPLES METADATA TABLES

Tables describing all the metadata used in the database.
The **samples_basic** and **samples_papers** tables are loaded directly to the database's memory for quick access. I thought accessing this metadata through R would be faster than using an SQL query. However, this is causing some delays when the database webpage is accessed for the first time.

```
##################
# TAXONOMY TABLE #
##################

CREATE TABLE IF NOT EXISTS `taxonomy` (
  `SH` varchar(32) NOT NULL,
  `Kingdom` varchar(64) NOT NULL,
  `Phylum` varchar(64) NOT NULL,
  `Class` varchar(64) NOT NULL,
  `Order` varchar(64) NOT NULL,
  `Family` varchar(64) NOT NULL,
  `Genus` varchar(64) NOT NULL,
  `Species` varchar(64) NOT NULL,
  `SH_id` int NOT NULL
);

ALTER TABLE taxonomy ADD INDEX(SH);
ALTER TABLE taxonomy ADD INDEX(SH_id);
```

TAXONOMY TABLE

The **taxonomy** table is also loaded directly into the database's memory for quick access. This helps generate a list of options on the form, which suggests predefined terms to help you find the correct word.

```
##################
# VARIANTS TABLE #
##################

CREATE TABLE IF NOT EXISTS `variants` (
  `hash` varchar(32) NOT NULL,
  `samples` MEDIUMTEXT NOT NULL,
  `abundances` MEDIUMTEXT NOT NULL,
  `marker` varchar(4) NOT NULL,
  `SH` int NOT NULL,
  `sequence` TEXT NOT NULL
);

ALTER TABLE variants ADD INDEX(hash);
ALTER TABLE variants ADD INDEX(SH);
```

VARIANTS TABLE
(currently ~600 000 000 records)

This table holds all unique sequence variants
hash – md5 generated from nucleotide sequence
samples – “;” separated IDs of samples
abundances - “;” separated abundances of sequences according to samples
marker – ITS1 or ITS2
SH – ID of taxonomical classification to Species Hypotheses of UNITE database if any
sequence – nucleotide sequence

```
############
# SH TABLE #
###########

CREATE TABLE IF NOT EXISTS `sh` (
  `sh` varchar(32) NOT NULL,
  `samples` MEDIUMTEXT NOT NULL,
  `abundances` MEDIUMTEXT NOT NULL,
  `vars` int NOT NULL
);

ALTER TABLE sh ADD INDEX(sh);
```

```
##############
# GENUS TABLE #
##############

CREATE TABLE IF NOT EXISTS `genus` (
  `genus` varchar(32) NOT NULL,
  `samples` MEDIUMTEXT NOT NULL,
  `abundances` MEDIUMTEXT NOT NULL,
  `vars` int NOT NULL
);

ALTER TABLE genus ADD INDEX(genus);
```

```
################
# SPECIES TABLE #
################

CREATE TABLE IF NOT EXISTS `species` (
  `species` varchar(64) NOT NULL,
  `samples` MEDIUMTEXT NOT NULL,
  `abundances` MEDIUMTEXT NOT NULL,
  `vars` int NOT NULL
);

ALTER TABLE species ADD INDEX(species);
```

TAXONOMICAL GROUPS SUB-TABLES

All those tables could be derived from the largest variants table. This will help to speed up the samples retrieval…
First variable is taxon name (sh, genus, species)
samples – “;” separated IDs of samples
abundances - “;” separated abundances of sequences according to samples
vars – number of different sequence variants classified as this taxon

```
#######################
# SAMPLES TO SH TABLE #
#######################

CREATE TABLE IF NOT EXISTS `samples_to_sh` (
  `sample` int NOT NULL,
  `SHs` MEDIUMTEXT NOT NULL
);

ALTER TABLE samples_to_sh ADD INDEX(sample);
```

SAMPLES TO SH TABLE

Table summarizing all the species hypotheses found in each sample. This table is used for Geosearch tool of the database.

sample – ID of sample

SHs - “;” separated IDs of Species Hypotheses

```
##################
# TRACKING TABLE #
##################

CREATE TABLE IF NOT EXISTS `traffic` (
  `id` int unsigned NOT NULL auto_increment PRIMARY KEY,
  `session` int NOT NULL,
  `category` varchar(32),
  `value` varchar(64),
  `date` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
);

######################
# MAILING LIST TABLE #
######################

CREATE TABLE IF NOT EXISTS `maillist` (
  `id` int unsigned NOT NULL auto_increment PRIMARY KEY,
  `name` TEXT NOT NULL,
  `email` TEXT NOT NULL,
  `date` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
);

SELECT * FROM maillist;

##################
# MESSAGES TABLE #
##################

CREATE TABLE IF NOT EXISTS `messages` (
  `id` int unsigned NOT NULL auto_increment PRIMARY KEY,
  `email` TEXT NOT NULL,
  `subject` TEXT NOT NULL,
  `message` TEXT NOT NULL,
  `processed` boolean not null default 0,
  `date` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP
);
```

```
##################
# INFO TABLE #
##################

CREATE TABLE IF NOT EXISTS `info` (
  `id` int unsigned NOT NULL auto_increment PRIMARY KEY,
  `name` VARCHAR(24) NOT NULL,
  `version` VARCHAR(6) NOT NULL,
  `release` VARCHAR(4) NOT NULL,
  `unite_version` VARCHAR(24) NOT NULL,
  `its_variants_count` BIGINT NOT NULL,
  `its1_raw_count` BIGINT NOT NULL,
  `its2_raw_count` BIGINT NOT NULL,
  `info` TEXT CHARACTER SET utf8,
  `citation` VARCHAR(128) CHARACTER SET utf8,
  `date` VARCHAR(10) NOT NULL
);
```

INFORMATIVE TABLES (NOT IMPORTANT)

These tables are used for leaving messages to administrators, adding users to the mailing list, monitoring traffic, and summarizing the release of the database.

```
#################
# INSERT TABLES #
#################

CREATE TABLE IF NOT EXISTS `study` (
  `hash` varchar(32) NOT NULL PRIMARY KEY,
  `contributor` TEXT NOT NULL,
  `email` TEXT NOT NULL,
  `affiliation_institute` TEXT NOT NULL,
  `affiliation_country` TEXT NOT NULL,
  `ORCID` TEXT NOT NULL,
  `title` TEXT NOT NULL,
  `authors` TEXT NOT NULL,
  `year` TEXT NOT NULL,
  `journal` TEXT NOT NULL,
  `volume` TEXT NOT NULL,
  `pages` TEXT NOT NULL,
  `doi` TEXT NOT NULL,
  `repository` TEXT NOT NULL,
  `include` TEXT NOT NULL,
  `coauthor` TEXT NOT NULL,
  `email_confirmed` int NOT NULL,
  `submission_finished` int NOT NULL,
  `date` varchar(32) NOT NULL
);
```

```
CREATE TABLE IF NOT EXISTS `metadata` (
  `id` int unsigned NOT NULL auto_increment PRIMARY KEY,
  `paper_study` varchar(32) NOT NULL,
  `longitude` float NOT NULL,
  `latitude` float NOT NULL,
  `elevation` varchar(32) NOT NULL,
  `continent` varchar(32) NOT NULL,
  `country` TEXT NOT NULL,
  `location` TEXT NOT NULL,
  `sample_type` TEXT NOT NULL,
  `Biome` TEXT NOT NULL,
  `Biome_detail` TEXT NOT NULL,
  `MAT_study` varchar(32) NOT NULL,
  `MAP_study` varchar(32) NOT NULL,
  `sample_name` TEXT NOT NULL,
  `area_sampled` varchar(32) NOT NULL,
  `area_GPS` varchar(32) NOT NULL,
  `number_of_subsamples` varchar(32) NOT NULL,
  `sample_depth` varchar(32) NOT NULL,
  `year_of_sampling` varchar(32) NOT NULL,
  `month_of_sampling` varchar(32) NOT NULL,
  `day_of_sampling` varchar(32) NOT NULL,
  `sampling_info` TEXT NOT NULL,
  `sample_description` TEXT NOT NULL,
  `sequencing_platform` varchar(32) NOT NULL,
  `target_gene` varchar(32) NOT NULL,
  `extraction_DNA_mass` varchar(32) NOT NULL,
  `extraction_DNA_size` TEXT NOT NULL,
  `extraction_DNA_method` TEXT NOT NULL,
  `primers` TEXT NOT NULL,
  `primers_sequence` TEXT NOT NULL,
  `pH` varchar(32) NOT NULL,
  `pH_method` varchar(64) NOT NULL,
  `organic_matter_content` varchar(32) NOT NULL,
  `total_C_content` varchar(32) NOT NULL,
  `total_N_content` varchar(32) NOT NULL,
  `total_P` varchar(32) NOT NULL,
  `total_Ca` varchar(32) NOT NULL,
  `total_K` varchar(32) NOT NULL,
  `plants_dominant` TEXT NOT NULL,
  `plants_all` TEXT NOT NULL,
  `sample_info` TEXT NOT NULL,
  `sample_seqid` TEXT NOT NULL,
  `sample_barcode` TEXT NOT NULL
);
```

TABLES FOR NEW STUDY SUBMISSION (NOT IMPORTANT)

These tables are used to describe the new study. The records are not directly incorporated into the database; they are added to the waiting list and manually curated for the next release.

DATABASE INITIALIZATION

The basic sample metadata is held in memory of the database to fast access…
I thought accessing the samples metadata through R is faster than through SQL query…
This is causing some delays when the database webpage is accessed…

module_load.R (initiation after launch of the database)

- load basic samples metadata:

```
query <- sprintf(paste0("SELECT * FROM samples_basic"))
global_samples <- data.table(sqlQuery(query))
```

- load papers:

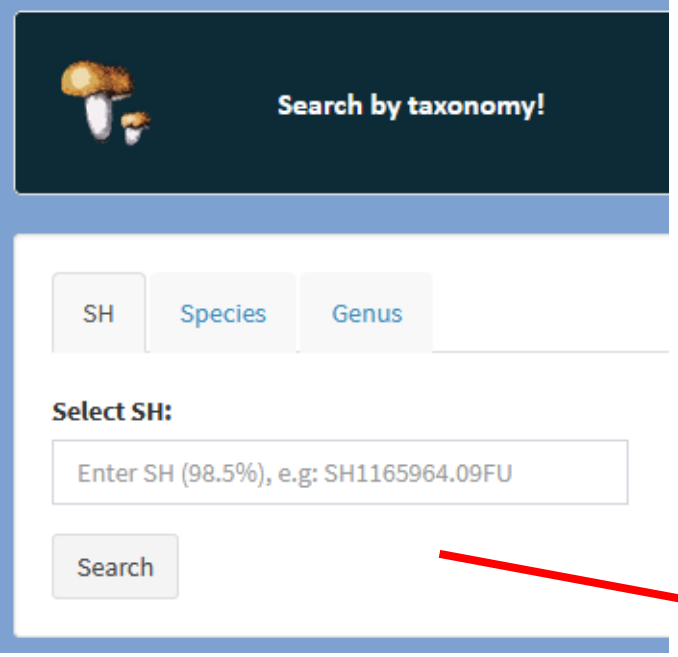
```
query <- sprintf(paste0("SELECT * FROM samples_papers"))
global_papers <- data.table(sqlQuery(query))
```

- load SH taxonomy table

```
query <- sprintf(paste0("SELECT * FROM taxonomy"))
global_SH <- sqlQuery(query)print(nrow(global_SH))
```

- get selection options:

```
global_SH_list <- global_SH$SH
global_species_list <- sort(unique(global_SH$Species))
global_species_list <- global_species_list[!global_species_list %in% grep(" sp.", global_species_list, value = T)]
global_genus_list <- sort(unique(global_SH$Genus))
```

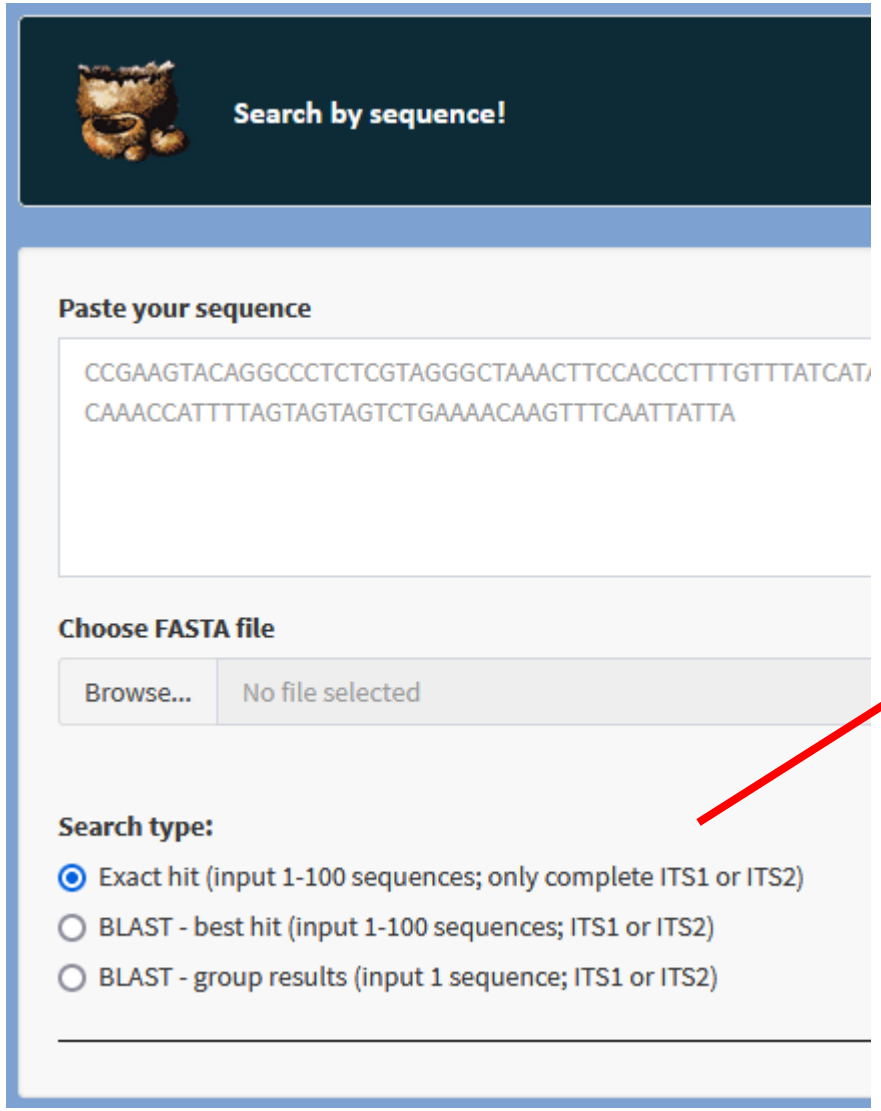


The output of this tool is send to "**module_results.R**" as query…

<u>SH</u>
A search of the desired Species Hypothesis in the database…
The query is a name of SH e.g: "SH1165964.09FU"

<u>Species</u>
Search for the desired Species name in the database…
The query is the name of species e.g: "Russula ochroleuca"

<u>genus</u>
Search for the desired Genus name in the database…
The query is a name of genus e.g: "Russula"



The output of this tool is send to "**module_results.R**" as query…

<u>Exact hit (exact)</u>

Search if the sequence variant is presented in the database. Firstly check if the sequence is in the **variants** table by sequence md5…

*query <- paste0("SELECT * from **variants** WHERE `hash` IN ",key_string)*

If it is presented then you can continue to the results…

The query is md5 of sequence e.g.: "b8a0144a2884cf50350eccd56bcde66d"

<u>Blast – best hit (single-blast)</u>

Blast input sequence against variants dataset (using BLASTn) and return name of the best hit from blast results (md5 code). Then you can continue to the results…

The query is md5 of sequence derived from the blast output e.g: "b8a0144a2884cf50350eccd56bcde66d"

<u>Blast – group results (multi-blast)</u>

Blast input sequence against variants dataset (using BLASTn) and return titles of "n" closest hits (md5 codes)

The query are md5s of sequences derived from the blast output e.g: "b8a0144a2884cf50350eccd56bcde66d, 90f8663ccf99fd8c5b3537f8c530e355"

Studies

Show 10 entries

Search:

	Id	Inserted	Title	Authors	Journal	Year	DOI	manipulated	Actions
1	834	27.03.2020	454 Pyrosequencing analyses of forest soils reveal an unexpectedly high fungal diversity.	Buée, M., Reich, M., Murat, C., Morin, E., Nilsson, R.H., Uroz, S. and Martin, F.	New Phytologist	2009	10.1111/ j.1469-8137.2009.03003.x	No	Show
2	695	27.03.2020	454 Pyrosequencing and Sanger sequencing of tropical mycorrhizal fungi provide similar results but reveal substantial methodological biases.	Tedersoo, L., Nilsson, R.H., Abarenkov, K., Jairus, T., Sadam, A., Saar, I., Bahram, M., Bechem, E., Chuyong, G. and Koljalg, U.	New Phytologist	2010	10.1111/ j.1469-8137.2010.03373.x	No	Show

The output of this tool is send to "**module_results.R**" as query…

<u>Show samples of the study (study)</u>
The query is ID of the study e.g.: 834

module_results.R (main module where the output results are generated)

Here are processed the query from “search by taxonomy” and “search by sequence”…samples IDs (“;” separated) are retrieved from the database table by this SQL queries:

<u>„SH“ (text is SH name e.g.: e.g: “SH1165964.09FU”)</u>
*query <- paste0("SELECT * from **sh** WHERE `sh` = '",**text**,"'")*

<u>"species„ (text is name of species e.g: “Russula ochroleuca”)</u>
*query <- paste0("SELECT * from **species** WHERE `species` = '",**text**,„‘“)*

<u>"genus„ (text is name of genus e.g: “Russula”)</u>
*query <- paste0("SELECT * from **genus** WHERE `genus` = '",**text**,"'")*

<u>"study“</u>
This will get the info about the study from “**global_papers**” table and all the samples belonging to that study from “**global_samples**” table based on “**study ID**”.

<u>"sequence“</u> or **<u>"single-blast“</u>** or **<u>"multi-blast“ (key is md5 of sequence e.g.: “b8a0144a2884cf50350eccd56bcde66d”)</u>**
*query <- paste0("SELECT `hash`,`samples`,`abundances`,`SH`,`marker`,`sequence` from **variants** WHERE `hash` IN ",key)*

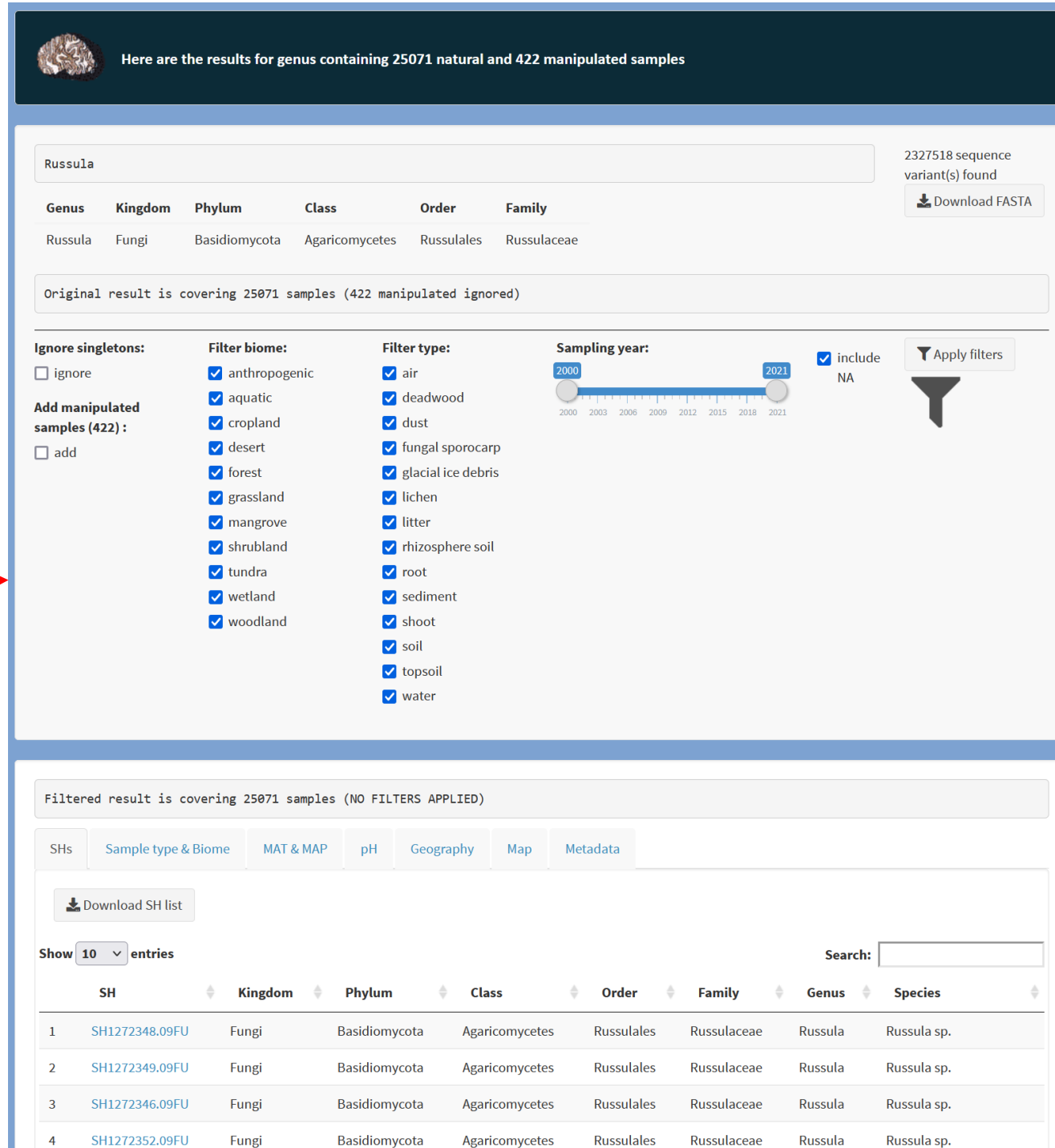
module_results.R
(main module where the output results are generated)

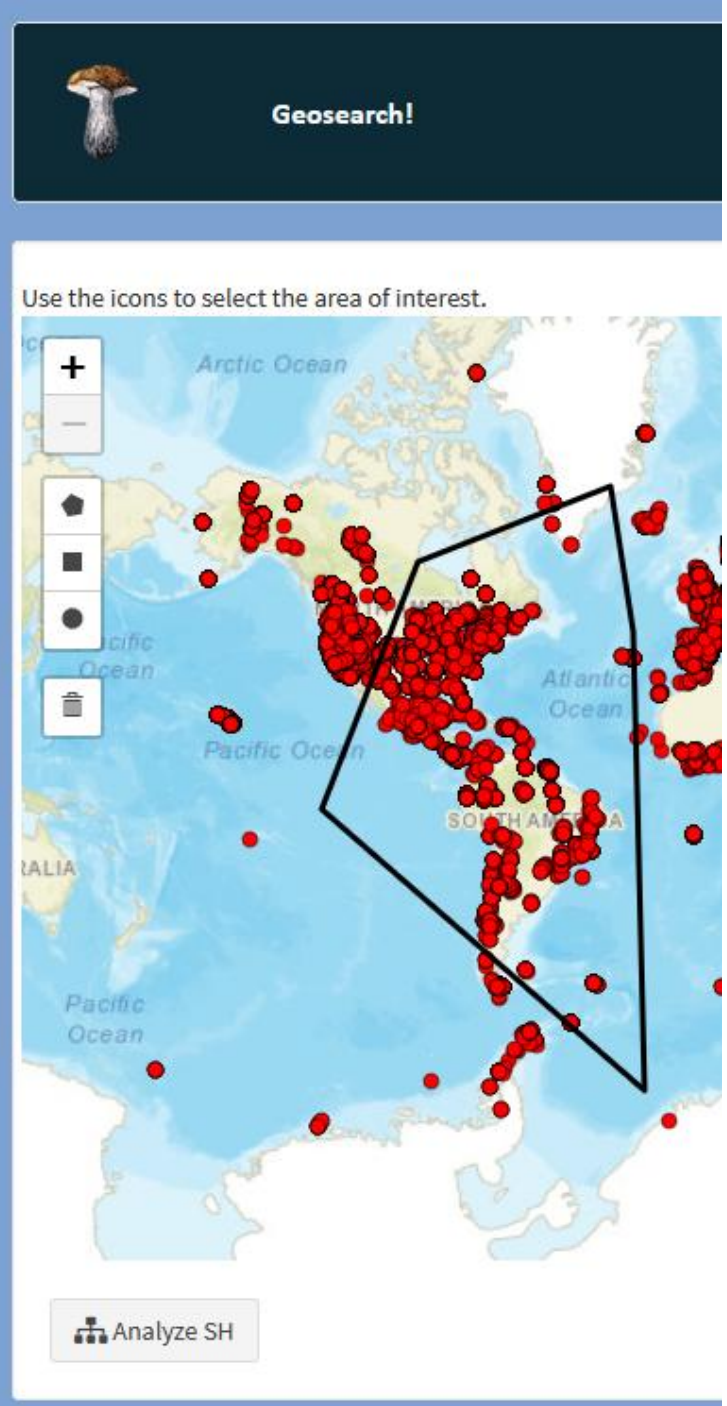
Then the IDs are used to get the samples metadata for final results…

```
samples <- strsplit(variants$samples, ';', fixed=TRUE)
abundances <- strsplit(variants$abundances, ';', fixed=TRUE)

sample_tab <- global_samples[which(global_samples$id %in% samples),]
```

This is an example of the result





This tool selects list of SHs (taxa) from selected samples based on this query:

key_string – is list of sample IDs based on the selected area on the map

*query <- paste0("SELECT * from **samples_to_sh** WHERE `sample` IN ",key_string)*

Then the taxonomy and samples metadata are selected from the memory tables **global_SH** and **global_samples**.