MariaDB Database Documentation: `petljakdb`

Author: Luka Culibrk

Overview

The petljakdb is a MariaDB database. MariaDB is an open-source implementation of the MySQL Relational Database Management system. Relational databases store tables of data that are related to each other through specific values in those tables. For example, one table could describe metadata relating to an entire project, and another table could describe sample-level metadata. In this example, the samples table could simply have a column that matches a column in the project table to indicate the project that the sample belongs to.

This document describes the structure of the petljakdb MariaDB database. It includes details about tables, columns, relationships, and usage information.

Usage information

Getting started using the DB

Note that the following must be done on BigPurple/Ultraviolet at NYULH.

First you will need a DB account, created by Luka. Once that's set up, the easiest way to authenticate is to create a plain text file `.my.cnf` in your home directory (`/gpfs/home/{your-user-name}/.my.cnf`) using your favorite text editor (vi, emacs, VScode, etc), structured according to the below code block. You may simply copy and paste the below code block, but replace the text in {} with your own - e.g. `{your_username}` corresponds to culibl01 for me, and the passwords corresponds to the password set during setup of your database account.

```
[client]
user={your_username}
password={your_password}
port=33100
host=db
default-character-set=utf8mb3
```

Enter the CLI interface:

Click here for a basic SQL tutorial!

Run these in bash on the cluster

```
module load mariadb
mysql petljakdb
```

You should now enter a mySQL command-line interface where you can run mySQL-formatted queries. mySQL/broadly SQL is a rich programming language with the capability to perform a large variety of complex queries. Some simple ones are provided below to get you started, but providing a full guide on SQL is outside the scope of this document.

Some example queries:

Broadly, the database can be queried via mySQL/mariaDB syntax (click me).

These need to be run in the mySQL command line interface (see previous section).

View the table of all projects:

```
SELECT * FROM studies;
```

a quick breakdown of the above:

- `SELECT` is a command to select (print out/return) rows of the table
- `*` is a wildcard. This is telling mySQL to return all columns.
- `FROM` refers to which table we are querying
- `studies` refers to the table, in this case we are querying the `studies` table.

You can simply query other tables by using a table name that is not `studies`, for example you could use `cells` to get the table of cells.

You can do a filtered selection based on the value of columns by adding a `WHERE` statement:

Select the study with ID `my_id` and query its samples:

```
SELECT *
FROM studies
WHERE study_id=my_id;
```

We can showcase the relational nature of the database. The `samples` table has a `study_id` column, which points to the `id` column of `studies`. So we can show all the samples associated with a specific study like this:

```
SELECT *
FROM samples
WHERE study_id=my_id;
```

Multiply filtered queries and specific columns

There are more advanced queries that can be done. This one uses multiple filters (see the `AND`) and selects specific columns `id,analysis_dir`. This returns the path to all the raw alignment data for a given study:

```
SELECT id,analysis_dir
FROM analyses
```

```
WHERE study_id=my_id
AND analysis_name="GATK_BAM";
```

More advanced, the path to all the mutation calls for JSC-1 cells, regardless of study, using an `INNER JOIN` operation. Notice the nested queries:

```
SELECT *
FROM samples
  INNER JOIN analyses
  ON analyses.samples_id=samples.id
  WHERE pipeline_name="MUTECT_CELLLINE"
  AND cell_id=(
    SELECT id
    FROM cells
    WHERE rname="JSC-1"
    );
```

For more information on queries refer to the previously linked mySQL/mariaDB documentation.

Dumping a query to a text file

Sometimes you might want to save the output of an SQL query as a tab-separated file. The below command, run in bash, will do this:

```
mysql petljakdb --execute "SQL QUERY HERE" -B > output_file.txt
```

The `--execute` flag signals that the next string in quotes `""` is an SQL query, formatted as in the previous section. The `-B` flag formats the output to separate the columns with tabs. Finally, `> output_file.txt` redirects the output from your terminal to a text file. You can name the output file whatever you want, not just `output_file.txt`

Tables Overview

The database contains the following tables:

1. `analyses`
2. `cells`
3. `patients`
4. `runs`
5. `samples`
6. `studies`

Table Details

The columns of each table are described in this section. Data types, indexes, and foreign keys are not vital for database users, but are important for database developers and are therefore described below. If you are unsure which you are, you are a user, and the pertinent information should be the columns and their description.

Technical details

The data types for each column are explicit and strictly defined in mariaDB, and are listed below:

- `Primary Key`: the "main column" of the table. The values are always unique and are the main way to index (select) from this table. Compatible with most data types, usually an `int` however. Every table must have a primary key.
- `int` (`signed/unsigned`): The column value will always be an integer. `unsigned` means that the values must be positive. All `int` columns in this database are unsigned.
- `varchar`: **var**iable **char**acter strings. The number (e.g. 20 for `varchar(20)`) refers to the length of the string.
- `datetime`: Date and time
- `enum`: Restricted vocabulary strings. For example the `analyses` table has an `input_table` column that must be one of `studies`, `samples`, or `runs`
- `Foreign Key`: This column contains the primary key of another table and is used to refer to a related row elsewhere. For example the `analyses` table has a column `studies_id` that relates to the `id` column of the table `studies`.

Indexes are listed for each table. For all tables the `id` column serves as the primary key/index, and foreign keys use this primary key.

Table description (for everybody)

Below is a technical description of each table. Each table is briefly described, and each of its columns are listed and briefly described. Columns of other tables are named as `table.column`, for example `runs.id` means the `id` column of the `runs` table.

1. analyses

- **Description**: Stores information about various analyses.
- **Columns**:
  - `id` (int unsigned, Primary Key): Unique auto-assigned numeric identifier for each analysis.
  - `pipeline_name` (varchar(20)): Name of the pipeline used.
  - `pipeline_version` (varchar(20)): Version of the pipeline.
  - `analysis_type` (varchar(20)): Type of analysis.
  - `input_table` (enum): Source table (`studies`, `samples`, or `runs`).
  - `studies_id` (int unsigned, Foreign Key): Linked to `studies.id`.
  - `samples_id` (int unsigned, Foreign Key): Linked to `samples.id`.

  - `runs_id` (int unsigned, Foreign Key): Linked to `runs.id`.
  - `cells_id` (int unsigned, Foreign Key): Linked to `cells.id`. Not currently used.
  - `analysis_time` (datetime): Timestamp of the analysis.
  - `analysis_dir` (varchar(250)): Directory for the analysis output.
  - `analysis_complete` (enum): Status of analysis (`True` for complete or `False` for ongoing).
  - `reference_genome` (varchar(255)): Reference genome used.
- **Indexes**:
  - `PRIMARY KEY`: `id`
  - Foreign Key Indexes: `studies_id`, `samples_id`, `runs_id`
- **Foreign Keys**:
  - `studies_id` → `studies.id`
  - `samples_id` → `samples.id`
  - `runs_id` → `runs.id`

---

2. `cells`

- **Description**: Contains information about cells.
- **Columns**:
  - `id` (int unsigned, Primary Key): Unique identifier for each cell.
  - `rname` (varchar(150), Unique): Unique cell name.
- **Indexes**:
  - `PRIMARY KEY`: `id`
  - `UNIQUE`: `rname`

---

3. `patients`

- **Description**: Stores patient-related data.
- **Columns**:
  - `id` (int, Primary Key): Unique identifier for each patient.
  - `rname` (varchar(255), Unique): Unique patient name.
  - `germline_sample` (int unsigned, Foreign Key): Linked to `samples.id`. Refers to the single authoratative germline sample for the patient.
  - `study_id` (int unsigned, Foreign Key): Linked to `studies.id`.
- **Indexes**:
  - `PRIMARY KEY`: `id`
  - `UNIQUE`: `rname`
  - Foreign Key Indexes: `germline_sample`, `study_id`
- **Foreign Keys**:
  - `germline_sample` → `samples.id`
  - `study_id` → `studies.id`

---

4. `runs`

- **Description**: Stores details about sequencing runs.
- **Columns**:
  - `id` (int unsigned, Primary Key): Unique identifier for each run.
  - `rname` (varchar(150), Unique): Unique run name.
  - `sample_id` (int unsigned, Foreign Key): Linked to `samples.id`.
  - `study_id` (int unsigned, Foreign Key): Linked to `studies.id`.
  - `source` (varchar(150)): Source of the sequencing data (ERA, SRA, local, etc).
  - `ega_id` (varchar(20)): EGA ID, if applicable.
  - `sra_id` (varchar(20)): SRA ID, if applicable.
  - `local_path` (varchar(150)): Path to local data, if applicable. Deprecated.
  - `sequencing_strategy` (varchar(50)): Sequencing strategy used, such as WGS, WXS, RNA.
  - `fastq_path` (varchar(500)): Path to FASTQ files.
- **Indexes**:
  - `PRIMARY KEY`: `id`
  - `UNIQUE`: `rname`
  - Foreign Key Indexes: `sample_id`, `study_id`
- **Foreign Keys**:
  - `sample_id` → `samples.id`
  - `study_id` → `studies.id`

---

5. `samples`

- **Description**: Contains sample-related data.
- **Columns**:
  - `id` (int unsigned, Primary Key): Unique identifier for each sample.
  - `rname` (varchar(150), Unique): Unique sample name.
  - `study_id` (int unsigned, Foreign Key): Linked to `studies.id`.
  - `biosample_id` (varchar(50)): Biosample identifier.
  - `treatment` (varchar(1000)): Treatment or relevant genotype details.
  - `sample_parent_id` (int unsigned, Foreign Key): Linked to `samples.id`.
  - `cell_id` (int unsigned, Foreign Key): Linked to `cells.id`.
  - `culture_days` (int unsigned): Number of culture days for cell line models.
  - `patient_id` (int, Foreign Key): Linked to `patients.id` for patient data.
- **Indexes**:
  - `PRIMARY KEY`: `id`

  - `UNIQUE`: `rname`
  - Foreign Key Indexes: `study_id`, `cell_id`, `sample_parent_id`, `patient_id`
- **Foreign Keys**:
  - `study_id` → `studies.id`
  - `cell_id` → `cells.id`
  - `sample_parent_id` → `samples.id`
  - `patient_id` → `patients.id`

---

6. `studies`

- **Description**: Stores information about studies.
- **Columns**:
  - `id` (int unsigned, Primary Key): Unique identifier for each study.
  - `rname` (varchar(150), Unique): Unique study name.
  - `study_pmid` (varchar(150)): PubMed ID for the study.
  - `ncbi_bioproject_id` (varchar(50)): NCBI BioProject ID.
- **Indexes**:
  - `PRIMARY KEY`: `id`
  - `UNIQUE`: `rname`

Relationships Overview

- **`analyses`** references `studies`, `samples`, and `runs`. Is referenced by none.
- **`patients`** references `samples` and `studies`. Is referenced by `samples`
- **`runs`** references `samples` and `studies`. Is referenced by `analyses`
- **`samples`** references `studies`, `cells`, and `patients`. Is referenced by `analyses` and `runs`
- **`cells`** references no tables. Is referenced by `samples`.
- **`studies`** serves as a foundational table referenced by `analyses`, `patients`, `runs`, and `samples`

Entity-relationship diagram (ERD)

Above you will find an ERD (click me), describing the tables in the database for a visual reference of the various interactions between the tables.

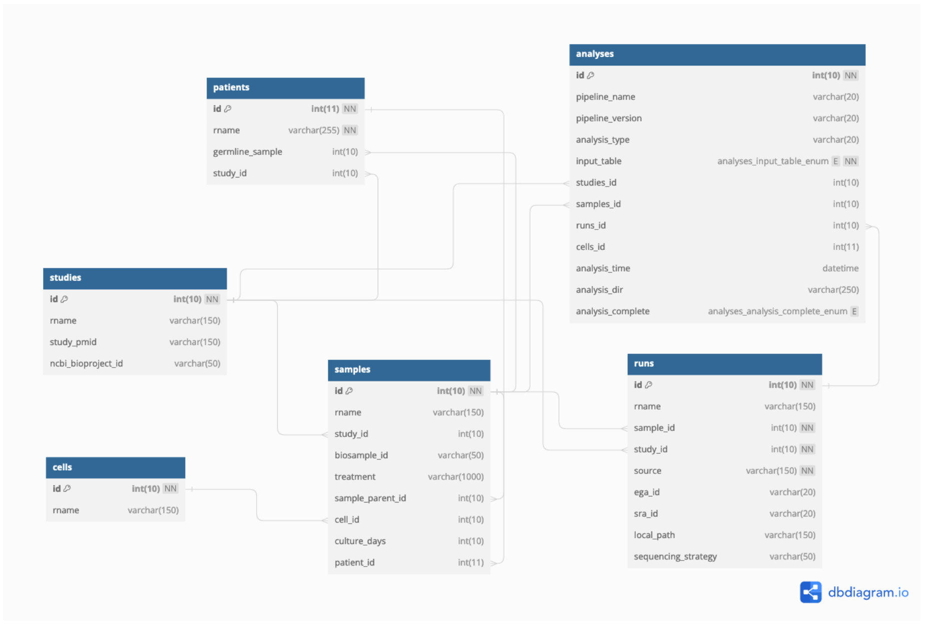
patients
id int(11) NN
rname varchar(255) NN
germline_sample int(10)
study_id int(10)
analyses
id int(10) NN
pipeline_name varchar(20)
pipeline_version varchar(20)
analysis_type varchar(20)
input_table analyses_input_table_enum E NN
studies_id int(10)
samples_id int(10)
runs_id int(10)
cells_id int(11)
analysis_time datetime
analysis_dir varchar(250)
analysis_complete analyses_analysis_complete_enum E
studies
id int(10) NN
rname varchar(150)
study_pmid varchar(150)
ncbi_bioproject_id varchar(50)
samples
id int(10) NN
rname varchar(150)
study_id int(10)
biosample_id varchar(50)
treatment varchar(1000)
sample_parent_id int(10)
cell_id int(10)
culture_days int(10)
patient_id int(11)
runs
id int(10) NN
rname varchar(150)
sample_id int(10) NN
study_id int(10) NN
source varchar(150) NN
ega_id varchar(20)
sra_id varchar(20)
local_path varchar(150)
sequencing_strategy varchar(50)
cells
id int(10) NN
rname varchar(150)
dbdiagram.io

Figure 1: ERD