# Parser Testing

## A brief overview of the process of testing the parsers during development of the QC project as a whole

### Testing data parsing and storage

The parsers and database were tested by parsing and loading the data from all of the available raw datasets into the database. The accuracy of the data was then tested by querying the data and comparing it to the data as displayed in the Illumina SAV software using a single dataset, as before with the testing of the parsers alone. Graphs were plotted using the matplotlib Python library to compare against the graphs in the SAV as an additional check(1,2).

Changes were made to the length and type of some database fields (e.g. INT to VARCHAR) to accommodate information found in some runs that was in different format to the original test run. It was also noted that variation between runs, such as differences in cycle number, or number of tiles would need to be incorporated into calculations and plots in order to represent the data accurately. Additional information (Table 1) was required to achieve this which was not stored in the database. A script was created to parse the raw data, extract the required additional information and append the data to the miseqrun relation in the database.

Table 1: Required additional data

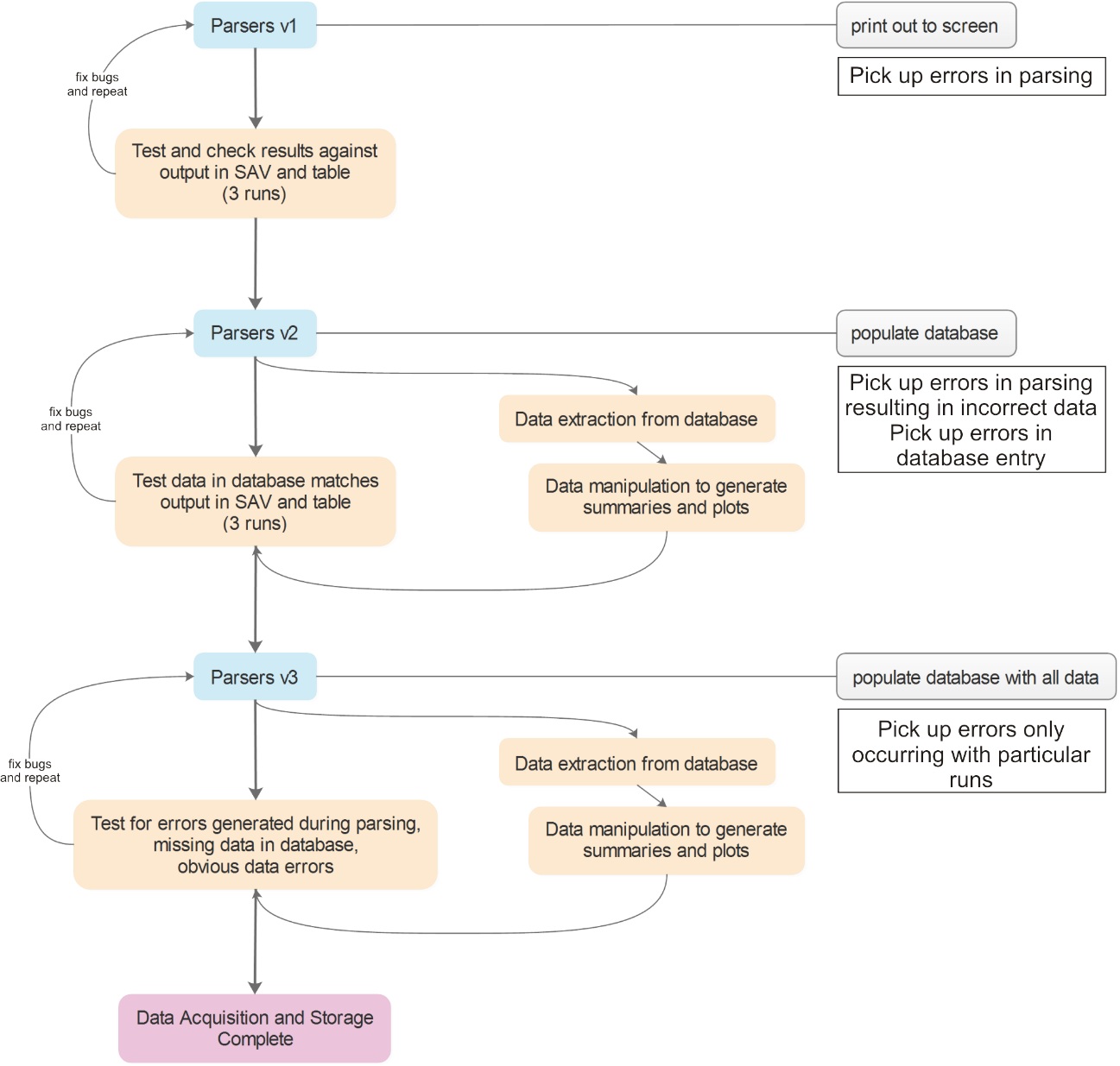
|  |  |  |
| --- | --- | --- |
| Data | Location (file) | Rationale |
| Number of tiles | runParameters.xml | Required to enable averaging over tiles for extracting information on a per-cycle basis.. Used to identify the end of read 1 and the beginning of read 2. |
| Number of swaths | runParameters.xml | This is currently always 1 on the MiSeq, but is incorporated for future proofing and possible extension to the HiSeq and other Illumina instruments. |
| Number of lanes | runParameters.xml | This is currently always 1 on the MiSeq, but is incorporated for future proofing and possible extension to the HiSeq and other Illumina instruments. |
| Number of surfaces | runParameters.xml | Required to enable averaging over tiles, as the total number of entries for which imaging data (the raw data generated by the Illumina sequencing method) is available is the number of tiles multiplied by the number of surfaces imaged. |

Table 1: Additional data identified as being required to accurately extract metrics and plot graphs from sequencing runs using different versions of kits and flowcells.

During this testing process, it was identified that one of the runs was incomplete at the time of acquisition of the raw data, which resulted in missing data in the database. It was therefore necessary to re-acquire the raw data and re-parse this run after completion of the analysis process to ensure that the captured data in the database was accurate. As the current design of the parsers and database skips entering data for runs if there is already any data in the miseqrun relation corresponding to that run identifier, all data associated with this run needed to be removed from the database prior to the parsers being re-run on the re-acquired complete raw dataset. Implementation of this functionality also has the additional advantage of allowing specific datasets to be removed from the database for other reasons. Using a cascading delete option on the foreign keys resulted in a lock wait time-out error, possibly due to the number of entries to be deleted at one time(3,4). Therefore, to overcome this, the entry for the desired run in each relation was deleted individually in order of dependencies to achieve deletion of all data associated with a run. Entries in the rds table were only deleted if they were unique to that particular run, as other runs may also share the same entry in the rds relation due to being of the same experimental design.

The entire testing process workflow undertaken to ensure correct data extraction from the file parsers and population of the database is summarised in Figure 1.

Figure 1: Testing process for data extraction and storage

Figure 1: A summary of the testing process undergone to ensure that the parsers extract the required data accurately and enter it into the database correctly. Initially the parsers were tested using 3 randomly selected datasets, initially by printing out values to the console and then more extensively by entering data into the database, extracting the data and comparing plots made using the extracted data and the matplotlib library against the SAV(1,2). Finally, all of the available datasets were imported, parsed and entered into the database. This test was to identify issues in parsing and data entry, perhaps associated with only specific runs, which had not previously been identified, however, some limited check on plots and summary statistics were used to verify the data.

## References

1. matplotlib development team. matplotlib [Internet]. 2016 [cited 2016 Feb 28]. Available from: http://matplotlib.org/

2. Illumina Technical Assistance. Sequencing Analysis Viewer v1.10 Software Guide. (October 2015).

3. Rocjoe. Delete single row from large MySql table results in “lock timeout” [Internet]. 2011 [cited 2016 Feb 28]. Available from: http://stackoverflow.com/questions/5902084/delete-single-row-from-large-mysql-table-results-in-lock-timeout

4. James. MySQL - Preventing lock wait timeout when deleting rows from self [Internet]. 2015 [cited 2016 Feb 28]. Available from: http://dba.stackexchange.com/questions/103982/mysql-preventing-lock-wait-timeout-when-deleting-rows-from-self