```
PURGE BINARY LOGS BEFORE '2019-04-02 22:46:26';
```

The `BEFORE` variant's *`datetime_expr`* argument should evaluate to a `DATETIME` value (a value in *`'YYYY-MM-DD hh:mm:ss'`* format).

This statement is safe to run while replicas are replicating. You need not stop them. If you have an active replica that currently is reading one of the log files you are trying to delete, this statement does not delete the log file that is in use or any log files later than that one, but it deletes any earlier log files. A warning message is issued in this situation. However, if a replica is not connected and you happen to purge one of the log files it has yet to read, the replica cannot replicate after it reconnects.

To safely purge binary log files, follow this procedure:

1. On each replica, use `SHOW REPLICA | SLAVE STATUS` to check which log file it is reading.
2. Obtain a listing of the binary log files on the source with `SHOW BINARY LOGS`.
3. Determine the earliest log file among all the replicas. This is the target file. If all the replicas are up to date, this is the last log file on the list.
4. Make a backup of all the log files you are about to delete. (This step is optional, but always advisable.)
5. Purge all log files up to but not including the target file.

`PURGE BINARY LOGS TO` and `PURGE BINARY LOGS BEFORE` both fail with an error when binary log files listed in the `.index` file had been removed from the system by some other means (such as using `rm` on Linux). (Bug #18199, Bug #18453) To handle such errors, edit the `.index` file (which is a simple text file) manually to ensure that it lists only the binary log files that are actually present, then run again the `PURGE BINARY LOGS` statement that failed.

Binary log files are automatically removed after the server's binary log expiration period. Removal of the files can take place at startup and when the binary log is flushed. The default binary log expiration period is 30 days. You can specify an alternative expiration period using the `binlog_expire_logs_seconds` system variable. If you are using replication, you should specify an expiration period that is no lower than the maximum amount of time your replicas might lag behind the source.

13.4.1.2 RESET MASTER Statement

```
RESET MASTER [TO binary_log_file_index_number]
```

Warning

Use this statement with caution to ensure you do not lose any wanted binary log file data and GTID execution history.

`RESET MASTER` requires the `RELOAD` privilege.

For a server where binary logging is enabled (`log_bin` is `ON`), `RESET MASTER` deletes all existing binary log files and resets the binary log index file, resetting the server to its state before binary logging was started. A new empty binary log file is created so that binary logging can be restarted.

For a server where GTIDs are in use (`gtid_mode` is `ON`), issuing `RESET MASTER` resets the GTID execution history. The value of the `gtid_purged` system variable is set to an empty string (`''`), the global value (but not the session value) of the `gtid_executed` system variable is set to an empty string, and the `mysql.gtid_executed` table is cleared (see mysql.gtid_executed Table). If the GTID-enabled server has binary logging enabled, `RESET MASTER` also resets the binary log as described above. Note that `RESET MASTER` is the method to reset the GTID execution history even if the GTID-enabled server is a replica where binary logging is disabled; `RESET REPLICA | SLAVE` has no effect on the GTID execution