- The FIELDS [OPTIONALLY] ENCLOSED BY character
- The first character of the FIELDS TERMINATED BY and LINES TERMINATED BY values
- ASCII NUL (the zero-valued byte; what is actually written following the escape character is ASCII 0, not a zero-valued byte)

The FIELDS TERMINATED BY, ENCLOSED BY, ESCAPED BY, or LINES TERMINATED BY characters *must* be escaped so that you can read the file back in reliably. ASCII NUL is escaped to make it easier to view with some pagers.

The resulting file need not conform to SQL syntax, so nothing else need be escaped.

If the FIELDS ESCAPED BY character is empty, no characters are escaped and NULL is output as NULL, not \N. It is probably not a good idea to specify an empty escape character, particularly if field values in your data contain any of the characters in the list just given.

INTO OUTFILE can also be used with a TABLE statement when you want to dump all columns of a table into a text file. In this case, the ordering and number of rows can be controlled using ORDER BY and LIMIT; these clauses must precede INTO OUTFILE. TABLE ... INTO OUTFILE supports the same *export_options* as does SELECT ... INTO OUTFILE, and it is subject to the same restrictions on writing to the file system. An example of such a statement is shown here:

```
TABLE employees ORDER BY lname LIMIT 1000
    INTO OUTFILE '/tmp/employee_data_1.txt'
    FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"', ESCAPED BY '\'
    LINES TERMINATED BY '\n';
```

You can also use SELECT ... INTO OUTFILE with a VALUES statement to write values directly into a file. An example is shown here:

```
SELECT * FROM (VALUES ROW(1,2,3),ROW(4,5,6),ROW(7,8,9)) AS t
    INTO OUTFILE '/tmp/select-values.txt';
```

You must use a table alias; column aliases are also supported, and can optionally be used to write values only from desired columns. You can also use any or all of the export options supported by SELECT ... INTO OUTFILE to format the output to the file.

Here is an example that produces a file in the comma-separated values (CSV) format used by many programs:

```
SELECT a,b,a+b INTO OUTFILE '/tmp/result.txt'
  FIELDS TERMINATED BY ',' OPTIONALLY ENCLOSED BY '"'
  LINES TERMINATED BY '\n'
  FROM test_table;
```

If you use INTO DUMPFILE instead of INTO OUTFILE, MySQL writes only one row into the file, without any column or line termination and without performing any escape processing. This is useful for selecting a BLOB value and storing it in a file.

TABLE also supports INTO DUMPFILE. If the table contains more than one row, you must also use LIMIT 1 to limit the output to a single row. INTO DUMPFILE can also be used with SELECT * FROM (VALUES ROW()[, ...]) AS *table_alias* [LIMIT 1]. See Section 13.2.14, “VALUES Statement”.

Note

Any file created by INTO OUTFILE or INTO DUMPFILE is owned by the operating system user under whose account mysqld runs. (You should *never* run mysqld as root for this and other reasons.) As of MySQL 8.0.17, the umask for file creation