

Loading Data in ICE

ICE includes a dedicated high-performance loader that differs from the standard MySQL Loader. The Infobright Loader is designed for speed, but supports less load syntax than the MySQL Loader, and only supports variable length text formatted load files.

Infobright Loader Syntax

Import your data into an Infobright table by using the following load syntax (all other MySQL Loader syntax is not supported):

```
LOAD DATA INFILE '/full_path/file_name'  
  
    INTO TABLE tbl_name  
    [FIELDS  
  
        [TERMINATED BY 'char']  
        [ENCLOSED BY 'char']  
        [ESCAPED BY 'char']  
  
    ];
```

The data is committed when the load completes if AUTOCOMMIT is set to on. This is default setting, but you can make it explicit by setting:

```
Set AUTOCOMMIT=1;
```

If you want to check the data via select before committing, then set AUTOCOMMIT to off:

```
Set AUTOCOMMIT=0;
```

New data can be seen by the loading session even though its not committed. Complete the load using an explicit COMMIT:

```
COMMIT;
```

FIELDS Clause

FIELDS subclauses are optional. If not specified the default values are used:

CLAUSE	DEFAULT VALUE
FIELDS TERMINATED BY	';' (semicolon)
FIELDS ENCLOSED BY	'"' (double quote)
FIELDS ESCAPED BY	' ' (none)

Field delimiters must be single characters and not two (ie, not '//'). The Loader does not have a default value for the escape character, so if you are using one it needs to be stated explicitly.

MySQL full syntax is available on <http://dev.mysql.com/doc/refman/5.1/en/load-data.html>

Loading Data in ICE

FIELDS TERMINATED BY

The input file may be delimited using, for example, a semicolon, comma, pipe (|) or tab (\t) - it must be a single character (ie, not '//'). It is important that the character used as a delimiter does not appear in the actual data unless it is specifically escaped or enclosed (see details in FIELDS ESCAPED BY further on).

FIELDS ENCLOSED BY

The input file may have fields enclosed by a character as long as it is stated explicitly, otherwise the default enclosure of " is assumed. If there is no enclosure, then either ENCLOSED BY 'NULL' needs to be stated explicitly as the enclosure type, or alternatively, the ENCLOSED BY clause can be omitted. It is important that an enclosure character does not appear in the actual data unless it is specifically escaped (see details in FIELDS ESCAPED BY further on).

FIELDS ESCAPED BY - Case 1: Delimiters

If a character that is used as a delimiter appears in the actual data it must either be escaped or the entire field must be enclosed.

For example if we want to import a text field of [one, two or three] where the data fields are also terminated by ',' as in:

```
1,one,two or three,1234
```

then we can either use ESCAPED BY '\\' which requires adding the \ escape character to the data, or we can use ENCLOSED BY "" which requires the text field to be enclosed by "".

The input file and corresponding load statement:

```
1,one\, two or three,1234
```

```
LOAD DATA INFILE '/usr/tmp/file1.txt' INTO TABLE test_table1 FIELDS
TERMINATED BY ',' ENCLOSED BY 'NULL' ESCAPED BY '\\';
```

is equivalent to an input file and corresponding load statement of:

```
1,"one, two or three",1234
```

```
LOAD DATA INFILE '/usr/tmp/file2.txt' INTO TABLE test_table1 FIELDS
TERMINATED BY ',' ENCLOSED BY '";
```

so these two files and load statements will result in:

```
mysql> select * from test_table1;
+-----+-----+-----+
| id  | textfield          | numerical |
+-----+-----+-----+
| 1  | one, two or three | 1234      |
| 1  | one, two or three | 1234      |
+-----+-----+-----+
2 rows in set (0.00 sec)
```

Loading Data in ICE

FIELDS ESCAPED BY - Case 2: Enclosures

If a character used as an enclosure appears in the actual data, it must be escaped otherwise it will not load.

For example if we want to import the text field [one "and" two], then the input file required is:

```
1,"one \"and\" two",1234
```

and the corresponding load statement is:

```
LOAD DATA INFILE '/usr/tmp/file3.txt' INTO TABLE test_table1 FIELDS
TERMINATED BY ',' ENCLOSED BY '"' ESCAPED BY '\\';
```

so this third file was read in as:

```
mysql> select * from test_table1;
+-----+-----+-----+
| id   | textfield          | numerical |
+-----+-----+-----+
| 1   | one, two or three | 1234     |
| 1   | one, two or three | 1234     |
| 1   | one "and" two     | 1234     |
+-----+-----+-----+
3 rows in set (0.00 sec)
```

Note: There is currently a bug being tracked in ICE where un-escaped embedded enclosures are handled in IB differently than in MySQL. If an even number of un-escaped embedded enclosures is included in the text, the data loads without requiring an escape character and is handled as expected as per MySQL. However, if there are an odd number of un-escaped embedded enclosures within the text then that row of data does not load and the load completes at the previous row; no error message is returned. With MySQL this case would load the data. This behaviour is being updated to reflect how MySQL handles this case.

ESCAPE CHARACTERS

If a text field includes escape characters, these must be escaped explicitly, otherwise the string is read in as straight text.

For the following input file:

```
2,other \t\t\ttext,4567
```

using two different load commands (one with and one without an ESCAPED BY):

```
LOAD DATA INFILE '/usr/tmp/file4.txt' INTO TABLE test_table1 FIELDS
TERMINATED BY ',' ENCLOSED BY 'NULL' ESCAPED BY '\\';
```

```
LOAD DATA INFILE '/usr/tmp/file4.txt' INTO TABLE test_table1 FIELDS
TERMINATED BY ',' ENCLOSED BY 'NULL';
```

Loading Data in ICE

the data will be read in differently and produce different results respectively:

```
mysql> select * from test_table1;
+-----+-----+-----+
| id   | textfield                | numerical |
+-----+-----+-----+
| 2    | other                    | text      | 4567 |
| 2    | other \t\t\ttext        |           | 4567 |
+-----+-----+-----+
2 rows in set (0.00 sec)
```

LINES TERMINATED Clause

It is important to note that IB loader ignores the LINES TERMINATED BY clause. Instead it detects how records are terminated based on the data in the input file.

There are two supported EOL formats:

1. Windows specific - '\r\n'
2. Unix specific - '\n'