CSCI-UA.60-1, Fall 2019

Database Design and Implementation - MySQL

Course Materials and Readings:

Using LOAD DATA in MySQL

There are sometimes problems with importing data from a .txt or .csv file into a MySQL table, so here are three easy steps you should take so everything runs smoothly.

(1) Regardless of what format your data is in, make sure you check your file on the server if you are having problems importing it into your table. To do that, log in to i6, and use pico to check that your file is how it should be. The reason you should do this is that if you upload your text file from an application like WordPad or TextEdit, your file will most likely include extra formatting characters, which will cause errors within MySQL. Sometimes the result is that MySQL imports only one line of your data file.

If you work on a Mac, you can use TextWrangler to check your text. Open the file in TextWrangler … go to FILE / SAVE AS … and be sure to select the “UNIX-LF” option under “Line Breaks”. Then upload your newly saved file up to your i6 account to run LOAD DATA again.

If you are working in Windows or a Mac, you can use SUBLIME TEXT to be sure that you have Unix Line-Feeds in your file. Open your dataset file in SUBLIME TEXT and navigate to VIEW / LINE-ENDINGS and be sure that UNIX is selected.

(2) Know where your files are. If you created a folder for your uploaded database files, make sure to include that folder when importing to MySQL. For example:

```
LOAD DATA LOCAL INFILE "/folder/myfile.txt" INTO TABLE tablename
```

If your data file is in the same directory as your script, you could write:

```
LOAD DATA LOCAL INFILE “myfile.csv” INTO TABLE tablename
```

(3) If your .txt file is tab delimited, then you can use the simple LOAD command:

```
LOAD DATA LOCAL INFILE "myfile.txt" INTO TABLE tablename;
```

However, if you are using a comma delimited file, either .txt or .csv, then you should use the following command (the example assumes you have two columns in your table):

```
LOAD DATA LOCAL INFILE "myfile.csv" INTO TABLE tablename
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
(column name, column name);
```

If your fields are further enclosed by quotation marks, here is another sample:

```
LOAD DATA LOCAL INFILE "myfile.txt"
INTO TABLE tablename
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
(column name, column name);
```

Be sure that MySQL knows exactly how to read your file, and where to put each field.

Documentation on the LOAD DATA command: http://dev.mysql.com/doc/refman/5.1/en/load-data.html (http://dev.mysql.com/doc/refman/5.1/en/load-data.html)

Class Notes: Warnings to watch for when using LOAD DATA

Warnings

Warnings are generated when the data are not imported correctly. The most common problems are insufficient data to fill up all of the fields (which might be correct but you would want to double-check); data for too many fields (definately an error); or data type mismatches (which are a good indicator that your fields might be in a different order in the table than your LOAD DATA command anticipates or problems in the data.)Use *SHOW WARNINGS* right after the LOAD FILE command to see them: For example:

```
Query OK, 11 rows affected, 20 warnings (0.01 sec)
Records: 11 Deleted: 0 Skipped: 0 Warnings: 19

mysql> SHOW WARNINGS;
+---------+------+------------------------------------------
------+
| Level | Code | Message |
+---------+------+------------------------------------------
------+
| Warning | 1261 | Row 6 doesn't contain data for all columns
|
| Warning | 1261 | Row 10 doesn't contain data for all
columns |
| Warning | 1366 | Incorrect integer value: '' for column
'objectID' at row 11 |
.....
```

CSCI-UA.60-1 Database Design and Implementation, Fall 2019