Redmine - Defect #21398

Mysql: 500 server error when submitting 4 bytes utf8 (to be saved in the 'notes' field)

2015-12-02 01:01 - Deoren Moor

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:	Database	**Estimated time:**	0.00 hour
Target version:			
Resolution:	Duplicate	**Affected version:**	3.1.2

Description

Hi,

We're running multiple Redmine installations, many of them several years old and all setup as described on the RedmineInstall wiki page:

```
CREATE DATABASE redmine CHARACTER SET utf8;
CREATE USER 'redmine'@'localhost' IDENTIFIED BY 'my_password';
GRANT ALL PRIVILEGES ON redmine.* TO 'redmine'@'localhost';
```

As far as I know today is the first case where we ran into a problem submitting text content where one of us wasn't attempting to submit more content than a field would hold.

The error messages:

```
Mysql2::Error: Incorrect string value: '\xF0\x9F\x98\x81' for column 'notes' at row 1: INSERT INTO
 `journals` (`journalized_id`, `journalized_type`, `user_id`, `notes`, `created_on`) VALUES
 (35747, 'Issue', 3, '¦~_~X~A', '2015-12-01 16:14:16')

ActiveRecord::StatementInvalid (Mysql2::Error: Incorrect string value: '\xF0\x9F\x98\x81' for colu
mn 'notes' at row 1: INSERT INTO `journals` (`journalized_id`, `journalized_type`, `user_id
`, `notes`, `created_on`) VALUES (35747, 'Issue', 3, '¦~_~X~A', '2015-12-01 16:14:16')):
```

Details from bin/about:

```
Environment:
  Redmine version                3.1.2.stable.14882
  Ruby version                   1.9.3-p0 (2011-10-30) [i686-linux]
  Rails version                  4.2.4
  Environment                    production
  Database adapter               Mysql2
SCM:
  Subversion                     1.6.17
  Git                            1.7.9.5
  Filesystem
Redmine plugins:
  no plugin installed
```

After turning to Google it appears that the character is an emoticon described as, "Grinning Face With Smiling Eyes"and has the UTF-8 hex code of F0 9F 98 81. The character displays properly within the text field and when choosing to preview the text/formatting, but not when attempting to save to the database.

Looking at other tickets here I see several others similar to this one:

- #20636 (not enough info to be sure it's the same)
- #20143 (mail handler related)
- #10772#note-7 (workaround was to convert to utf8mb4)

and the general response appears to be to use utf8mb4, but as noted on #10772#note-7 it appears that the results of that conversion resulted in data loss. This strikes me as odd since the MySQL 5.5 reference manual[1] utf8mb4 is a superset of utf8:

> For a supplementary character, utf8 cannot store the character at all, while utf8mb4 requires four bytes to store it. Since utf8

cannot store the character at all, you do not have any supplementary characters in utf8 columns and you need not worry about converting characters or losing data when upgrading utf8 data from older versions of MySQL.

- Is it safe to upgrade the character set for the database and tables as described on #10772#note-7?
  - i.e., will future upgrades be problematic due to a conversion of utf8 to utf8mb4 set?
- Should Redmine be expected to filter out invalid characters that do not match the character set of the database storing the data?
- Should utf8mb4 be used at the outset?
  - I ask because that's not noted in the guide (that I can find).

Thanks for your time.

[1] http://dev.mysql.com/doc/refman/5.5/en/charset-unicode-utf8mb4.html

Related issues:

Related to Redmine - Patch #19742: RedmineInstall: MySQL: collation_database	**Closed**
Related to Redmine - Defect #24116: Tickets are not created if the email cont...	**Closed**
Related to Redmine - Feature #31921: Changes to properly support 4 byte chara...	**Closed**

History

#1 - 2015-12-14 09:32 - Peter Pfläging

I've got the same problem. I've tested a conversion to UTF8MB4, which ist working.

I've posted a (german) article how I converted my instance to the correct behaving.

http://www.pflaeging.net/blog/archives/938

If someones interested I translate it to english ;-)

#2 - 2015-12-14 11:45 - Toshi MARUYAMA

- Related to Patch #19742: RedmineInstall: MySQL: collation_database added

#3 - 2016-09-13 20:09 - Jean-Marc Lagacé

Peter Pfläging wrote:

> If someones interested I translate it to english ;-)

I'm more than interested. Google Translate did most of the work but with this issue plaguing me right now (one of my dev is trying to store emojis in our database because we are tracking social media work).

#4 - 2016-09-14 05:48 - Toshi MARUYAMA

- Subject changed from 500 server error when submitting invalid string values (to be saved in the 'notes' field) to 500 server error when submitting 4 byte utf8 (to be saved in the 'notes' field)

#5 - 2016-09-14 05:49 - Toshi MARUYAMA

- Subject changed from 500 server error when submitting 4 byte utf8 (to be saved in the 'notes' field) to Mysql: 500 server error when submitting 4 byte utf8 (to be saved in the 'notes' field)

#6 - 2016-10-21 14:49 - Toshi MARUYAMA

- Related to Defect #24116: Tickets are not created if the email contains some of the special character added

#8 - 2017-03-10 08:46 - Silvio Geller

Hello,

I ran into the same Probleme as Peter Pfläging. I don't know why, but many people want to use emoji in their notes. So we also run into this 500 server error. As I tested a fresh installation of Redmine with utf8mb4, it fails. So it is not possible to do this without any changes. So as Peter Pfläging I have the same questions:

> Is it safe to upgrade the character set for the database and tables as described on #10772#note-7?
> i.e., will future upgrades be problematic due to a conversion of utf8 to utf8mb4 set?
> Should Redmine be expected to filter out invalid characters that do not match the character set of the database storing the data?

Should utf8mb4 be used at the outset?

Best regards,
Silvio Geller

#9 - 2017-03-13 04:51 - Toshi MARUYAMA

- Subject changed from Mysql: 500 server error when submitting 4 byte utf8 (to be saved in the 'notes' field) to Mysql: 500 server error when submitting 4 bytes utf8 (to be saved in the 'notes' field)

#10 - 2017-03-27 12:52 - Peter Pfläging

Ok, here a short solution in English for the problem:

Everything lies in two facts:
- MySQL has to handle all tables as UTF8MB4.
- INNODB large prefix has to be defined in MySQL (or MariaDB)

You must also be sure that every new table which is created by redline is also with these parameters!

Good! Now for the tasks during installation.

You have to be sure, that MySQL or MariaDB have the correct settings. So in /etc/ or /usr/local/etc search for my.cnf and put the following in:

```
[Mysqld]
innodb_file_per_table = 1
innodb_file_format = barracuda
innodb_large_prefix = 1
```

Create DB with mysql -u root -p:

```
CREATE DATABASE redmine CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;
CREATE USER 'redmine'@'localhost' IDENTIFIED BY 'my_password';
GRANT ALL PRIVILEGES ON redmine.* TO 'redmine'@'localhost';
```

Putz the following Ruby code in your redline installation /opt/redmine/config/initializers/enable_urf8mb4.rb with the following content:

```
ActiveSupport.on_load :active_record do
  module ActiveRecord::ConnectionAdapters

    class AbstractMysqlAdapter
      def create_table_with_innodb_row_format(table_name, options = {})
        table_options = options.reverse_merge(:options => 'ENGINE=InnoDB ROW_FORMAT=DYNAMIC')
        create_table_without_innodb_row_format(table_name, table_options) do |td|
          yield td if block_given?
        end
      end
      alias_method_chain :create_table, :innodb_row_format
    end

  end
end
```

That's everything apart from a standard installation of RedMine. It worked for me from 3.0 up to 3.3.2.

Here are my original two German articles to the problem:

http://www.pflaeging.net/blog/archives/1058
http://www.pflaeging.net/blog/archives/938

Greetings

:peter

#11 - 2018-04-05 11:40 - Tobias Fischer

#27361

#26386

#12 - 2018-04-05 11:47 - Peter Pfläging

OK, I've got a second solution ;-)

I've switched all my installations to postgresql 9.6 which solves all character set based problems, ...

Sorry for the comment, but this is in various ways the best solution.

#13 - 2018-04-18 12:48 - Martin Denizet (redmine.org team member)

After converting my tables to utf8mb4 and updating my database.yml accordingly, I could test that new tables created by Rails migrations were using utf8mb4, ROW_FORMAT=DYNAMIC and would not crash with emojis.
Using MySQL 5.7.21 and Redmine 3.4.5

#14 - 2018-11-05 11:12 - Juozapis Juozapauskiksi

Martin Denizet (redmine.org team member) wrote:

> After converting my tables to utf8mb4 and upadting my database.yml <...>

Could you provide more details on this? Did your environment already had data? What script did you use? What updates carried out in database.yml?

#15 - 2018-11-06 21:27 - Martin Denizet (redmine.org team member)

Juozapis Juozapauskiksi wrote:

> Martin Denizet (redmine.org team member) wrote:
>
> > After converting my tables to utf8mb4 and updating my database.yml <...>
>
> Could you provide more details on this? Did your environment already had data? What script did you use? What updates carried out in database.yml?

As far as I can remember (of course, make sure you have good backup before trying that!):

1. I created a file *db/migrate/20180418115300_encoding_convertion.rb* (adapt the timestamp to work with your install):

```
class EncodingConvertion < ActiveRecord::Migration
  #https://railsmachine.com/articles/2017/05/19/converting-a-rails-database-to-utf8mb4.html
  def db
    ActiveRecord::Base.connection
  end

  def up
    #return if Rails.env.staging? or Rails.env.production?

    execute "ALTER DATABASE `#{db.current_database}` CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;"
    db.tables.each do |table|
      execute "ALTER TABLE `#{table}
` ROW_FORMAT=DYNAMIC CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci;"

      db.columns(table).each do |column|
        case column.sql_type
          when /([a-z]*)text/i
            default = (column.default.blank?) ? '' : "DEFAULT '#{column.default}'"
            null = (column.null) ? '' : 'NOT NULL'
            execute "ALTER TABLE `#{table}` MODIFY `#{column.name}` #{column.sql_type.upcase}
 CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci #{default} #{null}$
          when /varchar\(([0-9]+)\)/i
            sql_type = column.sql_type.upcase
            default = (column.default.blank?) ? '' : "DEFAULT '#{column.default}'"
            null = (column.null) ? '' : 'NOT NULL'
            execute "ALTER TABLE `#{table}` MODIFY `#{column.name}`
#{sql_type} CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci #{default} #{null};"
        end
      end
    end
  end
end
```

2. Database migration:

```
RAILS_ENV=production bundle exec rake db:migrate
```

3. Then modified my *database.yml*:

```
production:
  adapter: mysql2
  database: redmine
  host: localhost
  username: root
  password: "password"
-  encoding: utf8
+  encoding: utf8mb4
```

4. And restarted the app

Credits: https://railsmachine.com/articles/2017/05/19/converting-a-rails-database-to-utf8mb4.html

#16 - 2019-02-19 23:26 - Sven Culley

This does not work for me on Redmine 4.0.1, any ideas?
Error: NameError: uninitialized constant Utf8mb4

#17 - 2019-07-31 10:18 - Tobias Fischer

Unfortunately the script does not work for me in Redmine 3.4.11:

```
/var/www/projekte/redmine# rake db:migrate RAILS_ENV=production
rake aborted!
SyntaxError: /var/www/projekte/redmine-3.4/db/migrate/20190731075300_encoding_convertion.rb:23: syntax error,
unexpected tCONSTANT,
expecting keyword_end
...default.blank?) ? '' : "DEFAULT '#{column.default}'"
...                               ^
/var/www/projekte/redmine-3.4/db/migrate/20190731075300_encoding_convertion.rb:25: syntax error, unexpected tC
ONSTANT, expecting keyword_end
            execute "ALTER TABLE `#{table}` MODIFY `#{c...
                          ^
/var/www/projekte/redmine-3.4/db/migrate/20190731075300_encoding_convertion.rb:25: syntax error, unexpected tC
ONSTANT, expecting keyword_end
... "ALTER TABLE `#{table}` MODIFY `#{column.name}` #{sql_type}...
...                               ^
/var/www/projekte/redmine-3.4/db/migrate/20190731075300_encoding_convertion.rb:30: syntax error, unexpected ke
yword_end, expecting end-of-input
/usr/local/rvm/gems/ruby-2.2.1/gems/polyglot-0.3.5/lib/polyglot.rb:65:in `require'
/usr/local/rvm/gems/ruby-2.2.1/gems/polyglot-0.3.5/lib/polyglot.rb:65:in `require'
/usr/local/rvm/gems/ruby-2.2.1/gems/activesupport-4.2.11.1/lib/active_support/dependencies.rb:274:in `block in
 require'
/usr/local/rvm/gems/ruby-2.2.1/gems/activesupport-4.2.11.1/lib/active_support/dependencies.rb:240:in `load_dep
endency'
/usr/local/rvm/gems/ruby-2.2.1/gems/activesupport-4.2.11.1/lib/active_support/dependencies.rb:274:in `require'
/usr/local/rvm/gems/ruby-2.2.1/gems/activerecord-4.2.11.1/lib/active_record/migration.rb:777:in `load_migratio
n'
/usr/local/rvm/gems/ruby-2.2.1/gems/activerecord-4.2.11.1/lib/active_record/migration.rb:773:in `migration'
/usr/local/rvm/gems/ruby-2.2.1/gems/activerecord-4.2.11.1/lib/active_record/migration.rb:768:in `disable_ddl_t
ransaction'
/usr/local/rvm/gems/ruby-2.2.1/gems/activerecord-4.2.11.1/lib/active_record/migration.rb:1076:in `use_transact
ion?'
/usr/local/rvm/gems/ruby-2.2.1/gems/activerecord-4.2.11.1/lib/active_record/migration.rb:1068:in `ddl_transact
ion'
/usr/local/rvm/gems/ruby-2.2.1/gems/activerecord-4.2.11.1/lib/active_record/migration.rb:1022:in `execute_migr
ation_in_transaction'
/usr/local/rvm/gems/ruby-2.2.1/gems/activerecord-4.2.11.1/lib/active_record/migration.rb:984:in `block in migr
ate'
/usr/local/rvm/gems/ruby-2.2.1/gems/activerecord-4.2.11.1/lib/active_record/migration.rb:980:in `each'
/usr/local/rvm/gems/ruby-2.2.1/gems/activerecord-4.2.11.1/lib/active_record/migration.rb:980:in `migrate'
/usr/local/rvm/gems/ruby-2.2.1/gems/activerecord-4.2.11.1/lib/active_record/migration.rb:823:in `up'
/usr/local/rvm/gems/ruby-2.2.1/gems/activerecord-4.2.11.1/lib/active_record/migration.rb:801:in `migrate'
/usr/local/rvm/gems/ruby-2.2.1/gems/activerecord-4.2.11.1/lib/active_record/tasks/database_tasks.rb:139:in `mi
grate'
/usr/local/rvm/gems/ruby-2.2.1/gems/activerecord-4.2.11.1/lib/active_record/railties/databases.rake:44:in `blo
ck (2 levels) in <top
(required)>'
/usr/local/rvm/gems/ruby-2.2.1/gems/rake-12.3.2/exe/rake:27:in `<top (required)>'
/usr/local/rvm/gems/ruby-2.2.1/bin/ruby_executable_hooks:15:in `eval'
```

```
/usr/local/rvm/gems/ruby-2.2.1/bin/ruby_executable_hooks:15:in `<main>'
Tasks: TOP => db:migrate
(See full trace by running task with --trace)
```

Did it work for someone else? What's the problem here?

#18 - 2019-08-17 18:15 - Marius BĂLTEANU

- Related to Feature #31921: Changes to properly support 4 byte characters (emoji) when database is MySQL added

#19 - 2019-10-14 09:57 - agile tuan

- File redmine_convert_utf8_to_utf8mb4.log added

Tobias Fischer wrote:

> Unfortunately the script does not work for me in Redmine 3.4.11:
>
> [...]
>
> Did it work for someone else? What's the problem here?

Hi Tobias Fischer,

I solved (please, view attachment):

By line number 19th:

```
execute "ALTER TABLE `#{table}` MODIFY `#{column.name}` #{column.sql_type.upcase} CHARACTER SET utf8mb4 COLLAT
E utf8mb4_unicode_ci #{default} #{null}$
```

missing close character

```
"
```

in end of line.

And, if has error by version of mysql,
you can remove dollar character in line 19th.

```
execute "ALTER TABLE `#{table}` MODIFY `#{column.name}` #{column.sql_type.upcase} CHARACTER SET utf8mb4 COLLAT
E utf8mb4_unicode_ci #{default} #{null}"
```

#20 - 2023-11-21 00:16 - Marius BĂLTEANU

- Status changed from New to Reopened

- Resolution set to Duplicate

This issue, #31921, #32140 and HowTo_convert_a_database_from_utf8_to_utf8mb4 page contain enough information to identify the issues caused by utf8 and to migrate your database to utf8mb4.

Closing this as duplicate.

#21 - 2023-11-21 00:19 - Marius BĂLTEANU

- Status changed from Reopened to Closed

Files

redmine_convert_utf8_to_utf8mb4.log	26 KB	2019-10-14	agile tuan