

# Self-Healing Databases

Managing schema updates in the field

*Jonathan Oxer* <[jon@ivt.com.au](mailto:jon@ivt.com.au)>



## Dealing drugs to Sakila

Brilliant Web Applications.

[www.ivt.com.au](http://www.ivt.com.au)

# The problem



Applications are not static.

New versions mean schema changes.

App / schema mismatches are bad.

Schema changes mean pain.

# Obvious solution



# Update scripts

# Update scripts



~~Run manually~~

# Update scripts



~~Statically defined~~



# A better way?



# Self- Healing Databases

# Reasons for change



New tables required.

New columns required.

Alterations to columns.

Alterations to contents of tables.

# Failure modes



New tables required.

***“Unknown table”***

New columns required.

***“Unknown column”***

Alterations to columns.

***?***

Alterations to contents of tables.

***?***





# Reactive, not proactive

# Smart error trapping



1. Run queries blindly.
2. Detect failure conditions.
3. Fix them.
4. Profit!

# But...



...if you don't have a db  
abstraction layer you're

# stuffed!

# Build, Borrow or Steal



One central  
query executor

# MySQL errors



MySQL has built-in error reporting: use it!

In PHP:

```
$errno = mysql_errno($link);  
$error = mysql_error($link);
```

Specify the link or you'll get the value from the last opened connection, not the last error from your connection.

# MySQL errors



Check for specific errors, such as:

1146: Table doesn't exist

1054: Unknown column

[dev.mysql.com/doc/refman/5.0/en/error-handling.html](http://dev.mysql.com/doc/refman/5.0/en/error-handling.html)

# Missing table



- Store reference schemas in app
- Trap “1146” errors
- Examine error to determine table name
- Load reference schema
- Create table
- Rerun original query
- Return result

The user never even notices a glitch :-)

# Missing table



Embed reference schemas into your app.

```
[modulename]/sql/articles.sql:
```

```
CREATE TABLE `articles` (  
  `Serial` INT NOT NULL AUTO_INCREMENT PRIMARY KEY ,  
  `Title` VARCHAR( 255 ) NOT NULL ,  
  `Article` TEXT NOT NULL  
) ENGINE = MYISAM ;
```

# Missing column



- Record schema changes in “alter” file
- Trap “unknown column” errors
- Load and execute alter file
- Rerun original query
- Return result

No harm, no foul.

# Missing column



Make execution of “alter” file idempotent.

```
[modulename]/sql/alter.php:
```

```
if (!$dbase->field_exists("news", "Modified"))
{
    $s = "ALTER TABLE news ADD `Modified` TIMESTAMP
        NOT NULL";
    $dbase->query($s);
}
```

# Missing trigger



Missing triggers  
are a problem:  
**silent death**

# Missing trigger



```
[modulename]/sql/alter.php:
```

```
if (!$dbase->trigger_exists("UPDATESTOCK"))  
{  
    $s = "CREATE TRIGGER UPDATESTOCK ...";  
    $dbase->query($s);  
}
```

# Missing procedure



- Put procedure additions in “alter” file
- Trap “1106”, errors
- Load and execute alter file
- Rerun original query
- Return result

(Note: trap “1107” and “1108” errors too, for handling altered procedures)

# Missing procedure



```
[modulename]/sql/alter.php:
```

```
if (!$dbase->procedure_exists("DISCOUNTCALC"))  
{  
    $s = "CREATE PROCEDURE DISCOUNTCALC ...";  
    $dbase->query($s);  
}
```



# That's not all, folks!



# Problem:



Multiple module  
instances require  
data partitioning

# Solution:



# Three-tier dynamic table naming scheme

# Dynamic table names



- 1: Module instance
- 2: Module name
- 3: Specific table

**hotstuff\_news\_articles**  
**hotstuff\_news\_comments**

# Benefits



Storage of schema with module: error handler can deduce path from table.

Upgrade of tables when you don't know their name.

# Schema templates



Placeholders in reference schemas

[modulename]/sql/articles.sql:

```
CREATE TABLE <articles> (  
  `Serial` INT NOT NULL AUTO_INCREMENT PRIMARY KEY ,  
  `Title` VARCHAR( 255 ) NOT NULL ,  
  `Article` TEXT NOT NULL  
) ENGINE = MYISAM ;
```

# “Alter” templates



Make table names in “alter” file dynamic

[modulename]/sql/alter.php:

```
$articles = $instance.'_news_articles';  
if (!$dbase->field_exists($articles, “Modified”))  
{  
    $s = “ALTER TABLE $articles ADD `Modified` TIMESTAMP  
        NOT NULL”;  
    $dbase->query($s);  
}
```

# Benefits



Stop caring about:

- App / schema mismatches
- Knowing what tables are called
- Telling users to run upgrade scripts

# Caveat:



this messes with

# triggers

and

# procedures

# Is This A Fairy Tale?



# Is this a fairy tale?



Technique in production use in the SiteBuilder web application framework and modules for more than 7 years:

- 1.2 million lines of PHP
- 149 modules
- 11,779 SQL statements
- 1,247 embedded table schemas

# Is this a fairy tale?



Deployments include:

Siemens national intranet with over 5,000 dynamically managed tables

Shaver Shop e-commerce system with tens of thousands of transactions / year

Gift Store Online with over 800k users

Brisbane Airport security credential system with 10k users / 30k cards

# Self-healing databases



# Thankyou :-)

These slides: [jon.oxer.com.au/talks](http://jon.oxer.com.au/talks)

We're hiring: [www.ivt.com.au/jobs](http://www.ivt.com.au/jobs)

Flames: Jonathan Oxer ([jon@ivt.com.au](mailto:jon@ivt.com.au))