



Your network is more powerful
than you think

Migration MySQL latin1 vers UTF-8

Meetup Viadeo / LeMUG.fr, Paris 16-11-2011



+ Plan Thèmes abordés

- Pourquoi migrer en UTF-8
- Charset et collation ?
- Les obstacles rencontrés
- Les solutions trouvées, approuvées et éprouvées
- Rolling upgrade
- Switchover
- A retenir... (résumé pour ceux qui vont avoir un coup de barre)



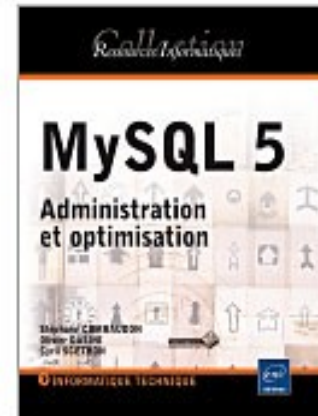
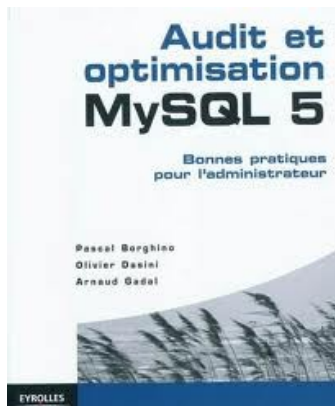
+ Me, myself & I

➤ Olivier DASINI

- Expert MySQL chez Viadeo
- Mon blog de vulgarisation des technologies MySQL
 - <http://dasini.net/blog/>
- Co-fondateur du MySQL User Group Francophone (LeMug.fr)
 - <http://lemug.fr>

➤ Co-auteur des livres

- Audit et optimisation – MySQL 5, Bonnes pratiques pour l'administrateur
 - Eyrolles, ISBN-13: 978-2212126341
- MySQL 5 – Administration et optimisation
 - ENI, ISBN-13: 978-2-7460-5516-2



+ Mission migration UTF-8

➤ Mission principale

- Convertir les données en UTF-8
 - Viadeo utilisé dans plus de 200 pays
 - Dont la Chine, l'Inde, la Russie, le moyen orient, ...

➤ Missions secondaires

- Convertir les tables en InnoDB
 - Performances, Sauvegarde, Exploitabilité,...
- Tuning de la configuration des serveurs
 - InnoDB tuning différent de MyISAM tuning



+ Charset & collation 1/4

あ A	か KA	さ SA	た TA	な NA
い I	き KI	し SI	ち TI	に NI
う U	く KU	す SU	つ TU	ぬ NU
え E	け KE	せ SE	て TE	ね NE
お O	こ KO	そ SO	と TO	の NO
だ DA	ぢ DJI	づ ZUZE	で DE	ど DO

Charset

- Peut être défini comme l'encodage d'un alphabet
- 39 jeux de caractères différents dans MySQL 5.5
- **Latin1** (ISO/CEI 8859-1) est le charset par défaut dans MySQL
 - « Presque » optimal pour l'Europe occidentale
 - 1 caractère = 1 octet
- **utf8** est le charset que nous voulons (devons) utiliser
 - Charset « universel »
 - 1 caractère = 1, 2 ou 3 octet(s)

```
SHOW CHARACTER SET WHERE Charset='latin1';
```

Charset	Description	Default collation	Maxlen
latin1	cp1252 West European	latin1_swedish_ci	1

```
SHOW CHARACTER SET WHERE Charset='utf8';
```

Charset	Description	Default collation	Maxlen
utf8	UTF-8 Unicode	utf8_general_ci	3

+ Charset & collation 2/4

↳ **Latin1 ne reconnaît pas tout les caractères**

```
CREATE TABLE t_latin1 (
  nom varchar(10) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1 COLLATE=latin1_swedish_ci
```

```
CREATE TABLE t_utf8 (
  nom varchar(10) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_swedish_ci
```

```
SELECT * FROM t_latin1;
```

nom
??

 **Latin1 ne peut encoder du mandarin**

À	Á	Â	Ã	Ä	Å	Æ	Ç	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß
a	á	â	ã	ä	å	æ	ç	ð	ñ	ó	o	ô	õ	ö	ø	ù	ú	ü	ý	þ	ÿ	
í	j	k	l	ly	m	n	ny	o	ó	ö	ó	ö	ó	p								
r	s	sz	t	ty	u	ú	ü	ú	v	z	zs	ak										

```
SELECT * FROM t_utf8;
```

nom
谢谢

+ Charset & collation 3/4

➤ Collation

- Peut être défini comme l'ensemble des règles qui permettent de comparer et d'ordonner les symboles d'un alphabet.
- Sert principalement lors des tris
- Elle est liée à un jeu de caractères
 - Pour le latin1 la collation par défaut est **latin1_swedish_ci**
 - Pour l'utf8 la collation par défaut est **utf8_general_ci**
- Attention aux différences de comportement et de performances...



+ Charset & collation 4/4

➤ La collation sert principalement lors des tris

```
SELECT * FROM table ORDER BY col COLLATE latin1_xxxxx_ci;
```

```
SELECT * FROM c1;
```

c
û
u
ü
ù

```
ORDER BY c COLLATE latin1_swedish_ci;
```

c
û
u
ù
ü

```
ORDER BY c COLLATE latin1_german1_ci;
```

c
û
u
ü
ù

```
ORDER BY c COLLATE latin1_german2_ci;
```

c
û
u
ù
ü

TO BE CONTINUED...

+ Méthode prévue

➤ Convertir les tables

- Charset : UTF-8
- Collation : utf8_swedish_ci
- Storage : InnoDB

➤ Très simple avec MySQL

- 1 seule commande : ALTER TABLE
- Se fait les doigts dans le nez !

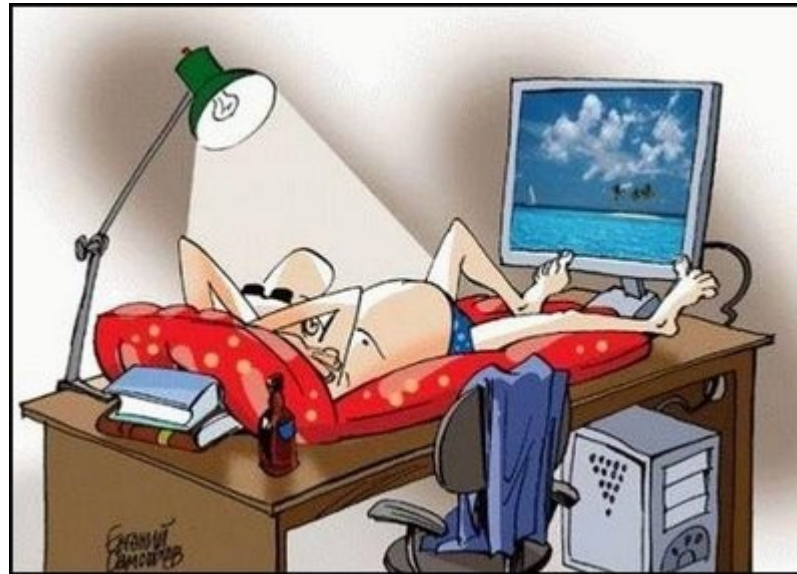
```
ALTER TABLE ma_table ENGINE = INNODB, CHARSET = utf8 COLLATE  
utf8_swedish_ci
```

Ça fonctionne ?



+ The end ?

Merci pour votre attention !





Pas si simple...

➤ Contraintes Viadeo

- Volumétrie : un important volume de données est géré
 - 1 000 000 de nouveaux membres / mois
 - 250 000 demandes de relations par jour
 - 165 000 discussions activés dans les hubs
 - 80 000 nouveaux membres de hubs / mois
 - 18 000 articles partagés / jour
 - 1 250 événements organisés / semaine
- Minimiser le downtime
 - Arrêt de service = diminution des recettes
- Surprise du chef !
 - Les données ne sont pas « clean »
 - Héritage du passé...

➤ Contraintes MySQL

- Taille maximale des index
- Caractéristiques liées aux CHARSET & COLLATION
- => devenu subitement des problèmes lors de la migration...

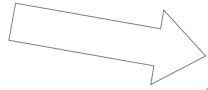
+ Mission impossible ?



+ Contraintes Viadeo

➤ MySQL @ Viadeo (prod OLTP) : volume de donnée important

- Une trentaine d'instances réparties sur 5 « clusters » ie réplication Master / Slaves
- 2 To de données (pour MySQL)
- Environs 1000 tables
 - 70% de MyISAM
- 5 milliards d'enregistrements



➤ 20% de l'effort

- Nécessite de bonnes connaissances MySQL
- Du savoir faire en scripting

➤ Minimiser le downtime grâce à la réplication

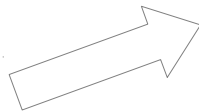
- => *Rolling upgrade*
 - La migration est effectuée sur les *slaves* en premier
 - A permet de tester et de valider la procédure
- => *Switchover*
 - Un *slave* est promu *master*

➤ 80% de l'effort

- Nécessite de bonnes connaissances métier
- Gros consommateur de temps et d'énergie

➤ Data legacy

- Pour des raisons historiques, une partie des données n'était pas « clean »
 - Merci aux anciens.....



- La migration applicative effectuée plusieurs mois auparavant
- => mix de données latin1 et utf8 dans des tables latin1
- => Fastidieux nettoyage à la main
- => Effectué grâce à une bonne connaissance du code
 - Merci aux anciens !



+ 80% de l'effort

- **Nécessite de bonnes connaissances métier**
- **Gros consommateur de temps et d'énergie**
 - 3 semaines de taf
 - Le coté obscur de la force...



+ Data legacy - Taille max des index

↳ Limité à 767 octets pour InnoDB (1000 octets pour MyISAM)

↳ Specified key was too long; max key length is 767 bytes

- ↳ Warning 1071 : pour un index non unique
 - MySQL index alors les 255 premiers caractères
 - KEY `idx_url` (`url`(255))
- ↳ ERROR 1071 (42000) : pour un index unique
 - Pas de solutions génériques.
 - Hautement dépendant de la logique métier
 - => Traité au cas par cas.

```
ALTER TABLE _table CONVERT TO CHARACTER SET utf8;
ERROR 1071 (42000): Specified key was too long; max key length is 767 bytes
```

```
CREATE TABLE _table (
  info varchar(256) NOT NULL,      256 x 3 = 768 > 767 ... le compte n'est PAS bon !
  PRIMARY KEY (info)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

```
ALTER TABLE _table2 CONVERT TO CHARACTER SET utf8;
Query OK, 0 rows affected, 2 warnings (0.33 sec)
Records: 0 Duplicates: 0 Warnings: 2
```

```
CREATE TABLE _table2 (
  info varchar(256) NOT NULL,
  KEY info (info(255)) ← Ajouté par MySQL
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

+ Data legacy - Duplicate entry 1/3

Process

- Lancer les tests sur un échantillon des données
- Comprendre le(s) problème(s)
 - **ERROR 1062 (23000): Duplicate entry**
 - => Charset différent = comportement différent du serveur
- Identifier les enregistrements problématiques
 - Les caractères posant problème :
 - Ne semblent pas poser problèmes ou sont parfois invisible
 - Avec des requêtes SQL : simple mais parfois long, souvent très très long
- Traiter le problème
 - Comprendre la donnée
 - Comprendre MySQL
 - Difficilement automatisable => FASTIDIEUX

+ Data legacy - Duplicate entry 2/3

- **Présence de caractères bizarres dans les données**
 - A0 / A020 / 20A0 / C2A0 / ... à la fin de certains enregistrements

Opération main propre !



```
SELECT ID, hex(url) FROM _table WHERE LEFT(reverse(Url),2) LIKE  
CONCAT(UNHEX('A0'),'%') ;
```

```
for col in postID; do  
  for carac in A0 A020 20A0; do  
    mysql -ugrantless -uP4S5 -B -N -e"SELECT ID FROM _table  
WHERE LEFT(reverse($col),2) LIKE CONCAT(UNHEX('$carac'),'%');";  
  done;  
done;
```

+ Data legacy - Duplicate entry 2/3

- **ERROR 1062 (23000): Duplicate entry 'pykachu' for key 'surnom'**
 - Alors qu'il n'y a (visiblement) pas de doublons...

```
SELECT surnom... LIKE 'p_kachu';
```

surnom
pykachu
pÿkachu

← Index unique



?

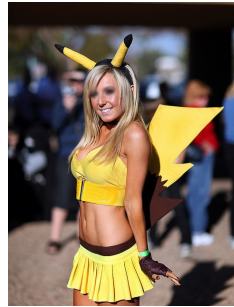
=



```
SELECT surnom... LIKE 'p_kachu';
```

surnom
pykachu
pykachu

← Index unique



?

=



- **Caractères différents mais identiques...**
 - Dépend de la collation

ÿ = ÿ
 ² = 2
 @ = a
 ß = ss

+ Charset & collation (encore) 1/4

➤ 2 lettres différentes peuvent être identiques...

```
SELECT 'u' = 'ü' COLLATE utf8_general_ci;
+-----+
| 'u' = 'ü' COLLATE utf8_general_ci |
+-----+
|                                1 |
+-----+
```

```
SELECT 'e' = 'ë' COLLATE utf8_general_ci;
+-----+
| 'e' = 'ë' COLLATE utf8_general_ci |
+-----+
|                                1 |
+-----+
```

```
SELECT 'u' = 'ü' COLLATE utf8_swedish_ci;
+-----+
| 'u' = 'ü' COLLATE utf8_swedish_ci |
+-----+
|                                0 |
+-----+
```

```
SELECT 'e' = 'ë' COLLATE utf8_swedish_ci;
+-----+
| 'e' = 'ë' COLLATE utf8_swedish_ci |
+-----+
|                                1 |
+-----+
```

```
SELECT 'u' = 'ü' COLLATE utf8_bin;
+-----+
| 'u' = 'ü' COLLATE utf8_bin |
+-----+
|                                0 |
+-----+
```

```
SELECT 'e' = 'ë' COLLATE utf8_bin;
+-----+
| 'e' = 'ë' COLLATE utf8_bin |
+-----+
|                                0 |
+-----+
```

+ Charset & collation (encore) 2/4

➤ Comportement parfois différent entre *latin1_swedish_ci* & *utf8_swedish_ci*

```
SELECT 'u' = 'ü' COLLATE latin1_swedish_ci;
+-----+
| 'u' = 'ü' COLLATE utf8_general_ci |
+-----+
|                                0 |
+-----+
```

```
SELECT 'u' = 'ü' COLLATE utf8_swedish_ci;
+-----+
| 'u' = 'ü' COLLATE utf8_swedish_ci |
+-----+
|                                0 |
+-----+
```

Comportement identique
u est différent de ü

```
SELECT 'e' = 'ë' COLLATE latin1_swedish_ci;
+-----+
| 'e' = 'ë' COLLATE utf8_general_ci |
+-----+
|                                0 |
+-----+
```

```
SELECT 'e' = 'ë' COLLATE utf8_swedish_ci;
+-----+
| 'e' = 'ë' COLLATE utf8_swedish_ci |
+-----+
|                                1 |
+-----+
```

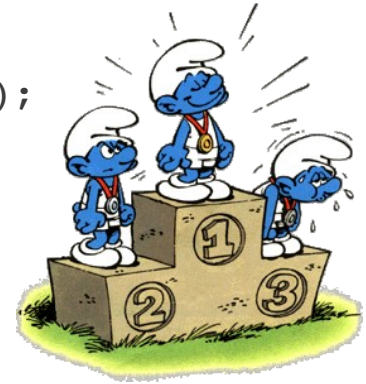
Comportement différent

- e est différent de ë en *latin1_swedish_ci*
- e est identique à ë en *utf8_swedish_ci*

+ Charset & collation (encore) 3/4

➤ Collation : attention aux différences de performances

```
SELECT BENCHMARK(1000000000, (select 'u'='ü' collate utf8_bin));
+-----+
| BENCHMARK(1000000000, (select 'u'='ü' collate utf8_bin)) |
+-----+
|                                                                 0 |
+-----+
1 row in set (20.62 sec)
```



```
SELECT BENCHMARK(1000000000, (select 'u'='ü' collate utf8_swedish_ci));
+-----+
| BENCHMARK(1000000000, (select 'u'='ü' collate utf8_swedish_ci)) |
+-----+
|                                                                 0 |
+-----+
1 row in set (57.53 sec)
```

```
SELECT BENCHMARK(1000000000, (select 'u'='ü' collate utf8_general_ci));
+-----+
| BENCHMARK(1000000000, (select 'u'='ü' collate utf8_general_ci)) |
+-----+
|                                                                 0 |
+-----+
1 row in set (27.71 sec)
```

+ Charset & collation (encore) 4/4 - Illegal mix

↳ Collation : attention aux mélanges de collation

```
CREATE TABLE City (  
...  
) DEFAULT CHARSET=utf8 COLLATE=utf8_swedish_ci  
  
CREATE TABLE Country (  
...  
) DEFAULT CHARSET=utf8 ← collate=utf8_general_ci
```

```
SELECT City.name FROM City JOIN Country USING(name) ;
```

```
ERROR 1267 (HY000): Illegal mix of collations (utf8_swedish_ci,IMPLICIT)  
and (utf8_general_ci,IMPLICIT) for operation '='
```

```
SELECT City.name FROM City Ci JOIN Country Co ON Ci.name=Co.name COLLATE  
utf8_general_ci ;
```

Permet de contourner le problème

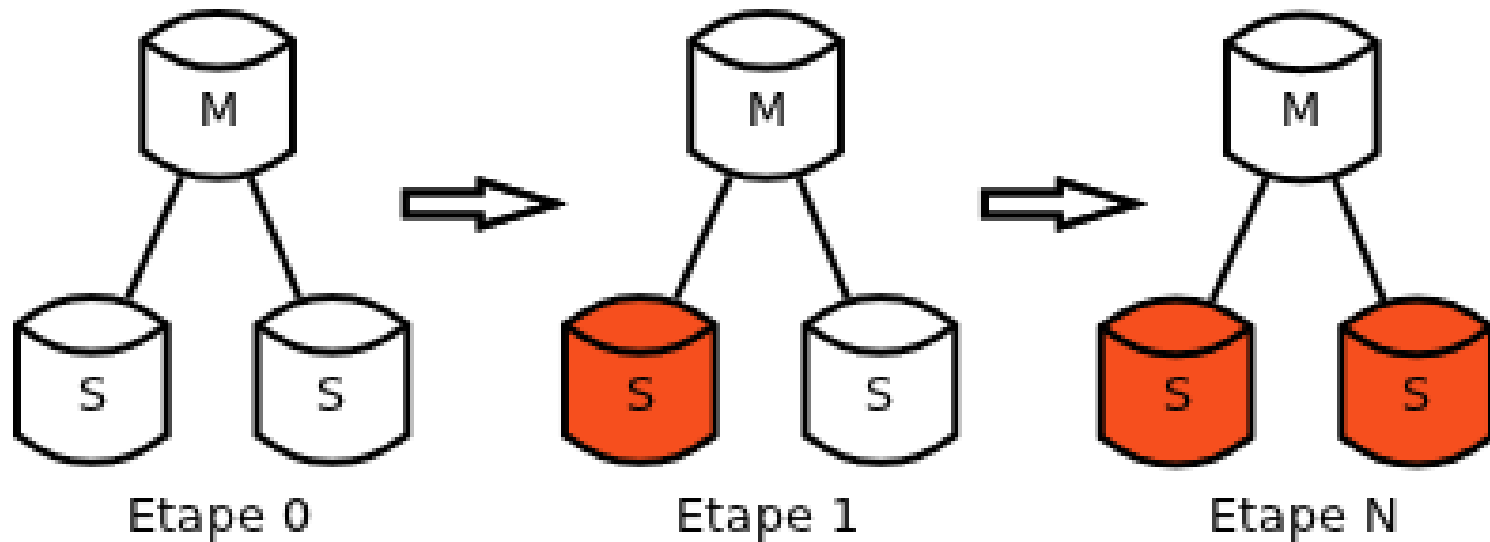
+ 20% de l'effort

- **Nécessite de bonnes connaissances MySQL**
- **Du savoir faire en scripting**
 - 3 jours de taf
 - Coté clair de la force



+ Minimiser le downtime - Rolling upgrade

➤ Mise à jour testée et validée sur les slaves



DB blanche : latin1
DB orange : UTF-8

+ Rolling upgrade 1/4

➤ Optimiser les perfs de la migration

- Minimiser les I/O disque
- Maximiser l'utilisation des buffers

➤ Extraction, transformation et chargement des données

- mysqldump
- ALTER TABLE
- mysqlimport
- REPAIR TABLE (MyISAM)

Powered by bash !

➤ Tuning serveur

- Remettre les buffers à leur bonne valeur
- Optimisation diverses

➤ La réplication

- La stopper avant la migration
- La démarrer après la migration

+ Rolling upgrade 2/4

➤ Optimiser les perfs de la migration

- Minimiser les I/O disque
- Maximiser l'utilisation des buffers

Désactiver les logs

```
SET SESSION SQL_LOG_BIN=0;
SET GLOBAL slow_query_log=0;
SET GLOBAL general_log=0;
```

Tuning InnoDB

Réduction des I/O disque

```
SET GLOBAL innodb_flush_log_at_trx_commit = 0;
SET GLOBAL innodb_support_xa = 0;
SET GLOBAL unique_checks=0;
SET GLOBAL foreign_key_checks=0;
```

Tuning MyISAM

```
SET GLOBAL myisam_sort_buffer_size = N;
```

Buffer pour le tri des index MyISAM :
REPAIR TABLE / CREATE INDEX / ALTER TABLE

```
SET GLOBAL bulk_insert_buffer_size = N;
```

Optimisation : LOAD DATA INFILE

Tuning Server

```
SET GLOBAL read_buffer_size = N;
```

Sert pour les « full table scan »

Charset & collation par défaut sur la DB

```
ALTER DATABASE DEFAULT CHARACTER SET utf8 COLLATE utf8_swedish_ci;
```

+ Rolling upgrade 3/4

↳ Extraction, transformation et chargement des données

- ↳ mysqldump
- ↳ ALTER TABLE
- ↳ mysqlimport
- ↳ REPAIR TABLE (MyISAM)

Sauvegarder la structure et les données en UTF8:

```
mysqldump --default-character-set=utf8 --hex-blob DB table1 table2 -T /path/to/bck/  
1m22.406s
```

Passer les tables en InnoDB, utf8, utf8_swedish_ci:

```
ALTER TABLE _table ENGINE=InnoDB, CONVERT TO CHARACTER SET utf8 COLLATE  
utf8_swedish_ci;
```

```
Query OK, 12193106 rows affected (31 min 14.77 sec)
```

```
Records: 12193106 Duplicates: 0 Warnings: 0
```

← **A faire sans les données !!!**

Charger les données en UTF8:

```
time mysql --default-character-set=utf8 DB < data.sql  
real 26m4.613s
```

VS

```
time mysqlimport --default-character-set=utf8 DB _table.txt  
DB._table: Records: 12193106 Deleted: 0 Skipped: 0 Warnings: 0
```

```
real 13m40.607s
```

← **mysqlimport (load data infile) est le plus rapide**

+ Rolling upgrade 4/4

↳ Tuning serveur

- ↳ Remettre les buffers à leur bonne valeur
- ↳ Optimisation diverses

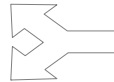
Refactoring my.cnf + Tuning Server

```
default_storage_engine=InnoDB
```

```
# Charset & Collation
```

```
character_set_server=utf8
```

```
collation_server=utf8_swedish_ci
```



Limite les mauvaises surprises

```
# Réplication
```

```
Skip-slave-start
```

```
# Changement algo de hash (REHASH des passwords)
```

```
old_passwords=0
```

L'algorithme de hachage des mots de passe n'est pas sécurisé (< 4.1)

```
$ time ./poc XXxxXxxXXXxXxXXX
```

```
mysql crack POC (c) 2006 Philippe Vigier & www.sqlhack.com
```

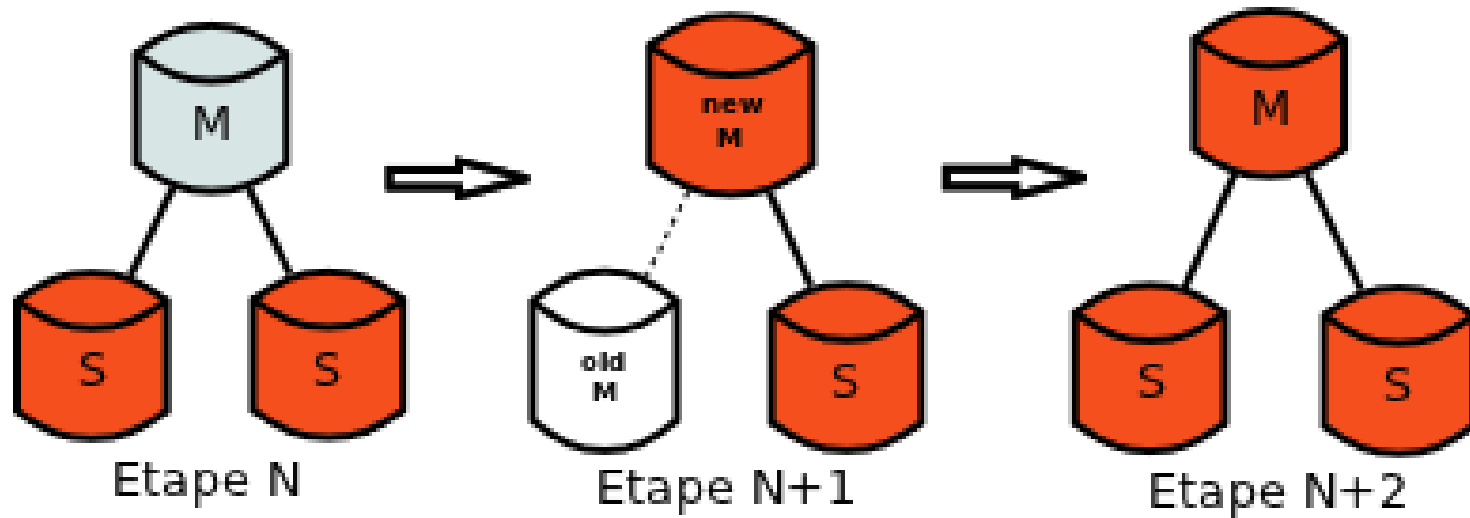
```
password for footprint XXxxXxxXXXxXxXXX = '_____'
```

real 38m47.400s

+ Minimiser le downtime - Switchover

➤ Switchover permet de minimiser interruption de service

- 1 slave devient master
- Le master devient 1 slave



+ Switchover - Pseudo code 1/3

```
/* the master should be in READ ONLY mode*/
```

Ne fonctionne pas avec le droit SUPER

```
$mA_cmd = "SET GLOBAL read_only='ON';";
```

```
$mA_cmd = "FLUSH NO_WRITE_TO_BINLOG TABLES WITH READ LOCK;";
```

```
/*Retrieve old master binlog info*/
```

```
$mA_cmd = "SHOW MASTER STATUS;";
```

Permet au slave de s'arrêter à la position exact

```
$old_master_file = $row["File"];
```

```
$old_master_pos = $row["Position"];
```

```
/* Kill the pending old master's connexions */
```

```
SELECT sleep(2);
```

```
SELECT ID, USER, HOST FROM information_schema.PROCESSLIST WHERE  
TIME > 2;
```

```
KILL CONNECTION ...
```

Tuer les éventuelles connexions résiduelles sur l'ancien master

+ Switchover - Pseudo code 2/3

```
/*For all the slaves*/
```

```
/*Wait for the slave up to date*/
```

```
$m[B|C]_cmd = "SELECT master_pos_wait('$old_master_file',  
$old_master_pos);"; Donne la main quand le slave est synchronisé avec le master
```

```
/*Stop the replication*/
```

```
$m[B|C]_cmd = "STOP SLAVE"; Arrêt de la réplication
```

```
/*Retrive new master binlog info*/
```

```
$mB_cmd = "SHOW MASTER STATUS;"; Sur le serveur promu master.
```

```
$new_master_file = $row["File"]; Sert à paramétrer les slaves sur le nouveau master  
$new_master_pos = $row["Position];
```

+ Switchover - Pseudo code 3/3

```
/*For all the new slaves*/
```

```
/*reset the old slave configuration*/
```

```
$mC_cmd = "RESET SLAVE;"; Réinitialiser les infos de la réplication sur les autres slaves
```

```
/*Configure the new master*/
```

```
$mC_cmd = "CHANGE MASTER TO MASTER_HOST = '$new_master_host',  
MASTER_USER = '$new_master_user', MASTER_PASSWORD =  
'$new_master_pwd', MASTER_PORT=$new_master_port,  
MASTER_LOG_FILE='$new_master_file',  
MASTER_LOG_POS=$new_master_pos;"; Indiquer aux slaves qui est le nouveau master
```

```
/*Start the replication*/
```

```
$mC_cmd = "START SLAVE"; Démarrer la réplication sur tout les slaves
```

```
/*reset the replication info for the new master*/
```

```
$mB_cmd = "RESET SLAVE"; Nettoyage du nouveau master
```

+ Switchover - inspiration

➤ Audit et optimisation, MySQL 5 --- éditions Eyrolles

➤ Page 239

Soit A le serveur master, B le slave de A qui va être promu master et C un autre slave de A

1/ Interdire les écritures sur A:

```
SET GLOBAL read_only='ON';
```

 verrouille en lecture seule

```
FLUSH TABLES WITH READ LOCK;
```

 verrouille en lecture seule pour les autres comptes avec le droit SUPER

2/ Sauvegarder le numéro du journal binaire et la position de A :

```
SHOW MASTER STATUS;
```

paramètres : File, Position

3/ Laissez le serveur B & C rattraper leur retard au niveau de la réplication

```
SELECT master_pos_wait('mysql-bin.xxxxxx',N);
```

paramètres : File & Position de A (point 2).

La fonction rendra la main une fois l'esclave à jour

3/ Une fois B à jour,

assurez vous qu'il est configuré en master

log binaire

server-id unique

Utilisateur de réplication

exécutez `SHOW MASTER STATUS;`

File & Position serviront à reconfigurer A et C.

4/ Routez les clients sur B qui devient alors le serveur actif

5/ Reconfigurer les serveurs A et C pour qu'ils soient esclave de B et qu'il reparte sur le bon fichier

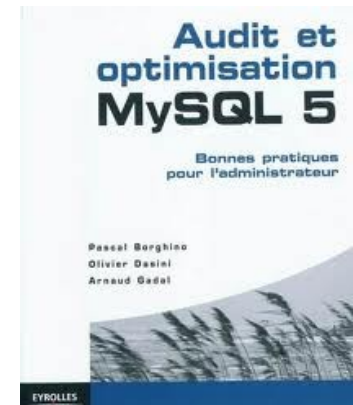
binaire et à la bonne position:

```
STOP SLAVE;
```

```
CHANGE MASTER TO MASTER_HOST="", MASTER_PORT=, MASTER_USER="", MASTER_PASSWORD="",  
MASTER_LOG_FILE = 'mysql-bin.xxxxxx', MASTER_LOG_POS = N;
```

A doit être configuré comme un slave

... **la suite dans le livre !)**



+ Résumé

➤ Enseignements à tirer de cette ~~douloureuse~~ expérience

- Connaître ses données
 - Comprendre et connaître la logique métier
 - Respecter la data
 - => Vérifiez les avant de les stocker (dans un monde idéal)
 - => Sinon vous le paierez tôt ou tard... (dans le vrai monde)
- Connaître MySQL
 - Le charset qu'il me faut, la collation que je choisi
 - Ses limites => RTFM
 - Rolling upgrade
 - Switchover
- Migrer la DB avant de migrer l'applicatif
 - Voir point 1...
- Automatiser tout ce qui est possible
 - Scripting for ever

+ Merci

➤ A la team Viadeo

- Anna
- Christophe
- Élodie
- Lionel
- Lourdes
- Marie Anne
- Nicolas
- Pierre(s)
- Sabri et son équipe
- Sandy
- Séverine
- Stéphane
- Sylvain
- Yorick
- ...

Thank
You

➤ Aux conférenciers

- Stéphane Combaudon
- Olivier Dasini
- Cédric Peintre
- Marc Thomas

➤ A vous

viadeo 

LEMUG.fr
MySQL User Group

viadeo 

+ Questions

▸ Ou trouver les slides du meetup ?

- ▾ <http://www.lemug.fr/>
- ▾ <http://dasini.net/blog/>
- ▾ <http://www.dbnewz.com/>
- ▾ <http://www.mysqlplus.net/>

