Partitioning a Zabbix MySQL database

And why you should...
Who am I?

Nathan Liefting
IT Consultant / Zabbix Trainer

@larcorba / larcorba.com
Our presentation today

1 – Why would you partition the database

2 – How to partition the database

3 – Manage the partitions with Perl

4 – Manage the partitions with Stored Procedures

Let’s wrap things up

More content like this
1 – Why would you partition the database
Housekeeper can’t keep up

- Housekeeper internal process more than 75% busy all the time
- Looks up old data in the database and deletes this entry by entry
- Impossible to tune the housekeeper process at one point
Partition before you Zabbix

• When building a good Zabbix setup, design is where it all starts
• Partitioning is part of your design

• Expect a server to grow? PARTITION
• Only a proof of concept that might go into production? PARTITION

• Partitioning a big database? It’s not fun
2 – How to partition the database
Let’s prepare to partition

- Partitioning can take a long time
- Use the Linux `screen` command!

- Partition History tables -> by day
- Partition Trends tables -> by month

<table>
<thead>
<tr>
<th>Table name</th>
<th>Purpose</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>history</td>
<td>Keeps raw history</td>
<td>Numeric (float)</td>
</tr>
<tr>
<td>history_uint</td>
<td>Keeps raw history</td>
<td>Numeric (unsigned)</td>
</tr>
<tr>
<td>history_str</td>
<td>Keeps raw short string data</td>
<td>Character</td>
</tr>
<tr>
<td>history_text</td>
<td>Keeps raw long string data</td>
<td>Text</td>
</tr>
<tr>
<td>history_log</td>
<td>Keeps raw log strings</td>
<td>Log</td>
</tr>
<tr>
<td>trends</td>
<td>Keeps reduced dataset (trends)</td>
<td>Numeric (float)</td>
</tr>
<tr>
<td>trends_uint</td>
<td>Keeps reduced dataset (trends)</td>
<td>Numeric (unsigned)</td>
</tr>
</tbody>
</table>
Prepare your partitioning commands - time

- Get your current timestamp

```
SELECT FROM_UNIXTIME(MIN(clock)) FROM history_uint;
```

- Command output example: **2020-12-19 08:09:17**
- Today’s date: **2021-02-19 09:00:00**
Prepare your partitioning commands - history

```sql
ALTER TABLE history_uint PARTITION BY RANGE ( clock )
(PARTITION p2020_12_19 VALUES LESS THAN (UNIX_TIMESTAMP("2020-12-20 00:00:00")) ENGINE = InnoDB,
PARTITION p2020_12_20 VALUES LESS THAN (UNIX_TIMESTAMP("2020-12-21 00:00:00")) ENGINE = InnoDB,
PARTITION p2020_12_21 VALUES LESS THAN (UNIX_TIMESTAMP("2020-12-22 00:00:00")) ENGINE = InnoDB,
PARTITION p2020_12_22 VALUES LESS THAN (UNIX_TIMESTAMP("2020-12-23 00:00:00")) ENGINE = InnoDB,
... 
PARTITION p2021_02_18 VALUES LESS THAN (UNIX_TIMESTAMP("2021-02-19 00:00:00")) ENGINE = InnoDB,
PARTITION p2021_02_19 VALUES LESS THAN (UNIX_TIMESTAMP("2021-02-20 00:00:00")) ENGINE = InnoDB,
PARTITION p2021_02_20 VALUES LESS THAN (UNIX_TIMESTAMP("2021-02-21 00:00:00")) ENGINE = InnoDB,
PARTITION p2021_02_21 VALUES LESS THAN (UNIX_TIMESTAMP("2021-02-22 00:00:00")) ENGINE = InnoDB,
PARTITION p2021_02_22 VALUES LESS THAN (UNIX_TIMESTAMP("2021-02-23 00:00:00")) ENGINE = InnoDB);
```

- We’ll prepare the same for the **history, history_str, history_text, history_log** tables
Prepare your partitioning commands - trends

```sql
ALTER TABLE trends_uint PARTITION BY RANGE ( clock )
(PARTITION p2020_10 VALUES LESS THAN (UNIX_TIMESTAMP("2020-11-01 00:00:00")) ENGINE = InnoDB,
PARTITION p2020_11 VALUES LESS THAN (UNIX_TIMESTAMP("2020-12-01 00:00:00")) ENGINE = InnoDB,
PARTITION p2020_12 VALUES LESS THAN (UNIX_TIMESTAMP("2021-01-01 00:00:00")) ENGINE = InnoDB,
PARTITION p2021_01 VALUES LESS THAN (UNIX_TIMESTAMP("2021-02-01 00:00:00")) ENGINE = InnoDB,
PARTITION p2021_02 VALUES LESS THAN (UNIX_TIMESTAMP("2021-03-01 00:00:00")) ENGINE = InnoDB,
PARTITION p2021_03 VALUES LESS THAN (UNIX_TIMESTAMP("2021-04-01 00:00:00")) ENGINE = InnoDB);
```

- We’ll prepare the same for the trends table
Now it’s time to partition

- Again do not forget to open a **screen**
- Login to MySQL
- Execute the prepared command table by table
- Have patience!
3 – Manage the partitions with Perl
Getting the script

• Download the script here: https://github.com/OpensourceICTSolutions/zabbix-mysql-partitioning-perl

• Save it to:

/usr/share/zabbix/

• Make it executable:

chmod +x /usr/share/zabbix/mysql_zbx_part.pl
Edit the script

• Make sure to open the script with your editor:

```
vim /usr/share/zabbix/mysql_zbx_part.pl
```

• Make sure to setup your MySQL login details:

```
my $dsn = 'DBI:mysql:'.dbname.'@mysql_socket=/var/lib/mysql/mysql.sock';
my $db_user_name = 'zabbix';
my $db_password = 'password';
```

• Then add how long you want to keep data:

```
my $ables = {  
  'history' => { 'period' => 'day', 'keep_history' => '60' },
  'history_log' => { 'period' => 'day', 'keep_history' => '60' },
  'history_str' => { 'period' => 'day', 'keep_history' => '60' },
  'history_text' => { 'period' => 'day', 'keep_history' => '60' },
  'history_uint' => { 'period' => 'day', 'keep_history' => '60' },
  'trends' => { 'period' => 'month', 'keep_history' => '12' },
  'trends_uint' => { 'period' => 'month', 'keep_history' => '12' },
```
Edit the script

- Add the correct timezone:

```perl
my $curr_tz = 'Europe/Amsterdam';
```

- You’re done, unless you use MySQL 5.5 (or earlier). Comment the following:

```perl
my $sth = $dbh->prepare(qq{SELECT plugin_status FROM information_schema.plugins WHERE plugin_name = 'partition'});

return 1 if $row eq 'ACTIVE';
```
Edit the script

- And uncomment:

```perl
#my $sth = $dbh->prepare(qq{SELECT variable_value FROM
  information_schema.global_variables WHERE variable_name = 'have_partitioning'});

#return 1 if $row eq 'YES';
```
Edit the script

• Last, but not least install the Perl script dependencies:

```bash
yum install perl-DateTime perl-Sys-Syslog
```

```bash
apt-get install libdatetime-perl liblogger-syslog-perl
```

• Then add a Cronjob to execute the script with:

```bash
crontab -e
```

• Add the following line:

```bash
0 23 * * * /usr/share/zabbix/mysql_zbx_part.pl >/dev/null 2>&1
```
Execute / monitor the script

- We can execute the script with:

  perl /usr/share/zabbix/mysql_zbx_part.pl

- To check if the script ran use:

  journalctl -t mysql_zbx_part

Congratulations, your database is partitioned and will remain so.
4 – Manage the partitions with Stored Procedures
Stored procedures, don’t use them (for this)!

Cons
• Hard to troubleshoot
• Messy to setup
• ‘Hidden’ inside your database
• Requires additional knowledge

Pros
• Some organisations don’t allow external scripts
• We included a guide in the blog post
Let’s wrap things up

- Preferably partition before you put Zabbix into production
- Prepare your partition commands
- Partition the database, use **screen**
- Setup your partition management (Perl Script / Stored Procedures)
- Keep an eye on your partitions for a few days
- Enjoy!

For a complete text guide see: [https://blog.zabbix.com/partitioning-a-zabbix-mysql-database-with-perl-or-stored-procedures/13531/](https://blog.zabbix.com/partitioning-a-zabbix-mysql-database-with-perl-or-stored-procedures/13531/)
Questions?