

# MySQL Performance Tuning

## DOAG Conference 2011, Nürnberg

**Oli Sennhauser**

Senior MySQL Consultant, FromDual GmbH

[oli.sennhauser@fromdual.com](mailto:oli.sennhauser@fromdual.com)



# FromDual GmbH

- **FromDual bietet neutral und unabhängig:**
  - Beratung für MySQL (vor Ort und remote)
  - Remote-DBA / MySQL Betrieb
  - Support für Galera (synchrone Replikation)
  - Support für MySQL (Basic und Silber)
  - Schulung für MySQL
- **Consulting Partner der Open Database Alliance (ODBA.org)**
- **Oracle Silber Partner (OPN)**
- **Mehr Informationen unter:**



<http://www.fromdual.com>

[www.fromdual.com](http://www.fromdual.com)



# Kunden



# Inhalt

## Performance Tuning

- › **FromDual Performance Waage**
- › **Kritische Ressourcen**
- › **Hardware / OS**
- › **Datenbank Parameter**
- › **Applikations-Tuning**
- › **Architektur & Design**
- › **Messen**



# Der FromDual Weg des PT

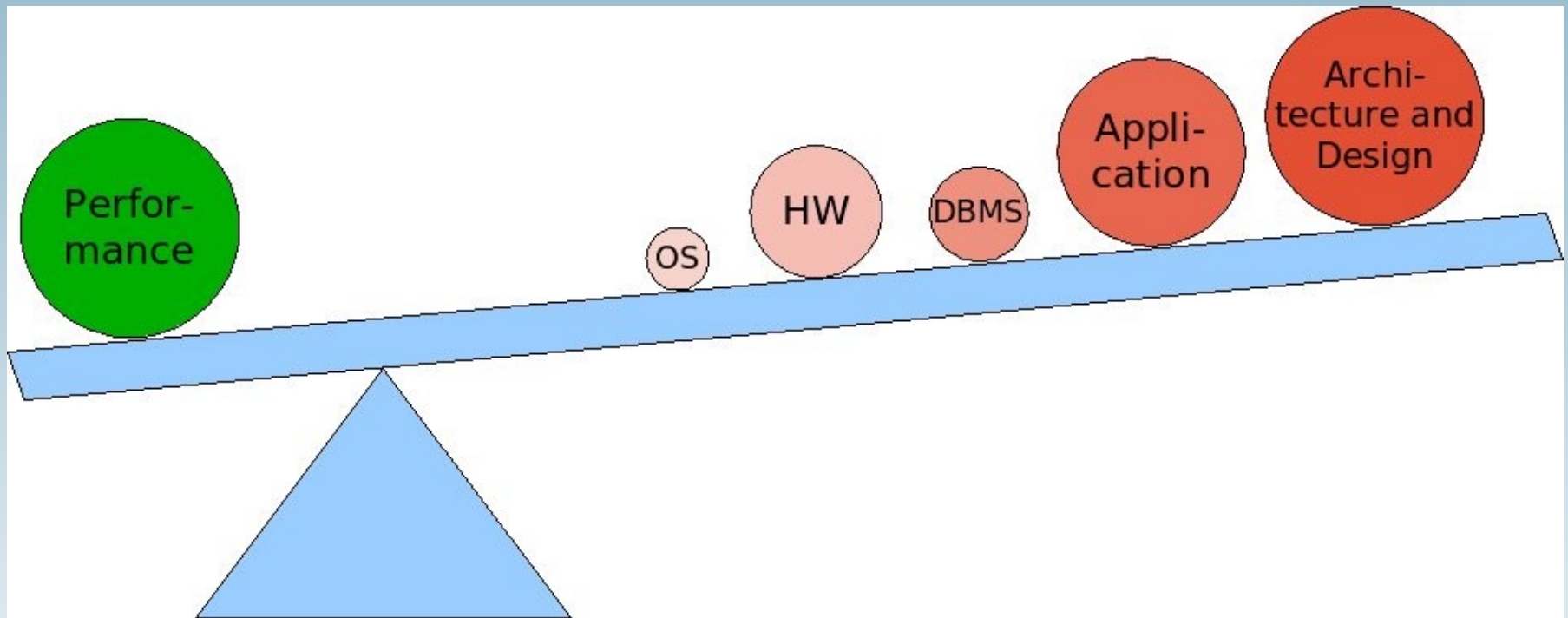
- **Wer von Euch hat oder hatte Performance Probleme mit MySQL?**
- **Wer von Euch hat ein systematisches Vorgehen für Performance Tuning?**

→ **Viele Wege führen nach Rom!**

- **Ausgangslage: Kunde schreit, weil er ein Performance Problem hat!**



# FromDual Performance Waage



# Zusammentragen von Fakten

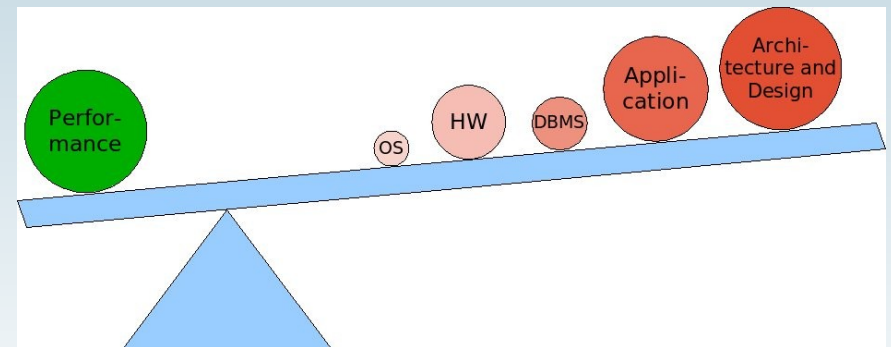
- **Wie sieht das Problem aus?**
    - DB ist plötzlich, manchmal oder schon immer langsam?
    - Was ist genau langsam?
    - Neuer Release wurde eingespielt?
    - Jemand hat „rumgefummelt“?
    - Wir sind kurz vor Produktionseinführung und viel zu langsam?
  - **Haben wir historische Messdaten?**
  - **Am besten wenn:**
    - man das Problem (gezielt) simulieren kann
    - es vorhersagbar oder periodisch auftritt
- **Finde das Muster!**



# Kritische Ressourcen

## Hardware

OS





# Kritische Ressourcen

- Finde den Flaschenhals / die limitierende Ressource:
- Zum Glück „nur“:
  - CPU
  - Speicher (RAM)
  - I/O (IOPS, Durchsatz)
  - Netzwerk (FpS, Durchsatz)
- Aber wie? → finde die kritische Ressource!



# Messen: CPU

- **top**

```
Cpu0  :  7.1%us, 12.8%sy,  0.0%ni, 71.4%id,  1.5%wa,  0.0%hi,  7.2%si,  0.0%st
Cpu1  : 16.5%us,  3.4%sy,  0.0%ni, 79.4%id,  0.0%wa,  0.0%hi,  0.7%si,  0.0%st
Cpu2  : 99.8%us,  0.1%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.1%si,  0.0%st
Cpu3  :  8.5%us,  2.3%sy,  0.0%ni, 58.5%id, 28.2%wa,  2.3%hi,  0.2%si,  0.0%st
```

- **vmstat**

```
# vmstat 1
procs -----memory-----  ---swap--  ----io----  -system--  ----cpu----
 r  b   swpd   free   buff   cache    si   so    bi   bo    in   cs  us  sy  id  wa
 1  0   96148  56096  35936  548792    0    0     0   656   379  343  5  38  57  0
 0  0   96148  56096  35936  548792    0    0     0    0   260  357  5  34  61  0
 0  0   96148  56096  35936  548792    0    0     0    0   306  399  9  29  62  0
 3  0   96148  49192  35940  549808    0    0  1020    0   289  431  91  4  3  2
 1  0   96148  47424  35944  551572    0    0   896    0   310  378  98  2  0  0
 1  0   96148  45656  35944  553344    0    0   896    0   260  359  98  1  0  1
 2  0   96148  43948  35944  555112    0    0   896    0   280  355  97  3  0  0
 1  0   96148  42056  35952  556884    0    0   904    0   260  374  99  0  0  1
 1  0   96148  40288  35984  558672    0    0   896  3772  312  398  97  3  0  0
 1  0   96148  38520  35984  560424    0    0   896    0   259  365  97  1  0  2
```

- **oder mpstat**

- **Welcher Prozess braucht denn CPU?**



# Messen: Speicher (RAM)

- **free / top:**

```
#free
          total    used      free  shared  buffers  cached
Mem:      1036016  983864    52152        0   35484  547432
-/+ buffers/cache: 400948  635068
swap:      4202112  96148  4105964
```

- **ps**

```
# ps -eo user,pid,%cpu,%mem,vsz,rsz,comm --sort -vsz | \
  egrep 'mysql|COMMAND'
USER      PID  %CPU  %MEM   VSZ   RSZ  COMMAND
mysql    1361  0.0   1.5 108368 16444 mysql
mysql    1210  0.0   0.1  4536  1956  bash
mysql    1289  0.0   0.1  4060  1444  safe_mysql
mysql    1204  0.0   0.1  4048  1404  su
```



# Messen: I/O

- **vmstat**

```
# vmstat 1
procs ---swap-- -----io----- ----cpu----
 r  b   si   so    bi    bo us sy id wa
 0  0    3    3   94   143 21 21 56  2
 0  0    0    0    0     4  9 37 54  0
```

- **iostat** (→ sysstat package)

```
# iostat -x 1
avg-cpu:  %user   %nice %system %iowait  %steal   %idle
           5.88    0.00   34.31   2.94    0.00   56.86

Device:            r/s    w/s  rkB/s  wkB/s  await  svctm   %util
hda                 0.00   0.00   0.00   0.00   0.00   0.00    0.00
hdc                 0.00   2.94   0.00  23.53  14.67  12.00   3.53
```

- **pidstat**



# Messen: Netzwerk

- `dstat`

```
# dstat
----total-cpu-usage---- -dsk/total- -net/total- ---paging-- ---system--
usr  sys  idl  wai  hig  sig  read  writ  recv  send  in   out  int  csw
 21   6   56   2   0   14  25k   39k   0     0   764B 880B 129  762
  9   2   55   0   0   34   0     0   262B 1680B 0     0   297  374
  6   2   59   0   0   33   0     0  1075B 1467B 0     0   284  372
  8   3   54   5   1   29   0    208k 1046B  884B 0     0   309  377
 14   2   54   0   1   29   0    236k 3479B 3669B 0     0   333  362
 18   5   47   1   0   29   0    164k 2800B 3632B 0     0   351  2257
 30  69   0   0   0   1   0     0  1807B 1181B 0     0   651  243k
 24  74   2   0   0   0   0     0  2380B 2183B 0     0   685  240k
```

- `watch -d -n 1 'ifconfig'`

- Frames pro Sekunde (80k – 1.5M) /  
Durchsatz (1 Gbit/s → 120 Mbyte/s)?



# Zusammenfassung

- CPU

- Welcher Prozess
- wie viele Cores?  
→ meist SQL Abfragen

- Speicher

- Welcher Prozess
- Swapping?  
→ Über- oder Unterallo-  
zierung von DB Caches!

- I/O

- Durchsatz oder IOPS
- welches Device?
- Random oder Sequential
- Read oder Write  
→ Caches zu klein, tmp  
Tabellen?

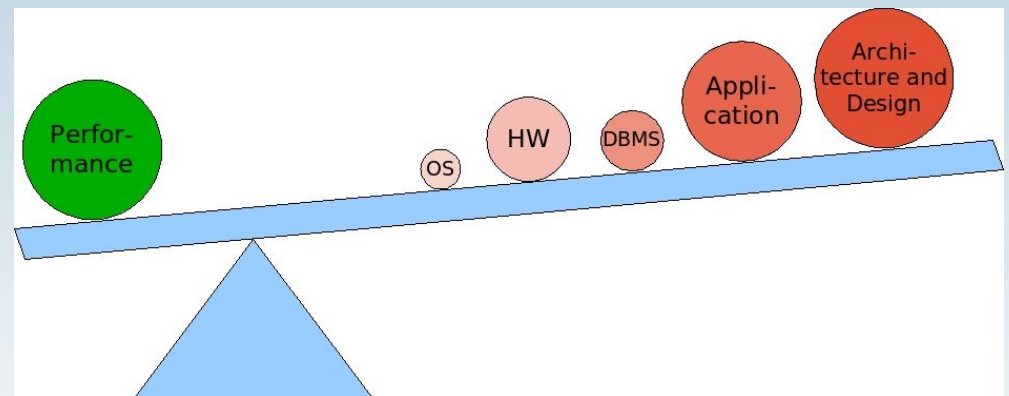
- Netzwerk

- Errors / Drops?
- Durchsatz
- FpS





# Datenbank Tuning





# MySQL Tuning

- **Welche Storage Engine verwendet Ihr zur Zeit?**
- **Welchen MySQL Release? (→ 5.1 und neuer)**
- **Zur Zeit: ca. 330 MySQL Parameter**
  - aber nur ca. 8 davon sind primär signifikant!
  - **Grob-Tuning**
- **Alle anderen nur nach ausführlichem Benchmarken**
  - **Fine-Tuning**



# InnoDB Grob-Tuning

- `innodb_buffer_pool_size`
  - ca. 80% vom RAM auf dedizierter Maschine
  - `SHOW STATUS LIKE 'Innodb_buffer_pool_pages%';`
- `innodb_log_file_size`
  - Grösser = schneller, aber längere Recovery Zeiten → 2 x 256 M
- `innodb_flush_log_at_trx_commit`
  - 0, 2 für Performance, 1 für Sicherheit
- `sync_binlog`
  - `!= 0` → langsam(er)



# MySQL Grob-Tuning

- **key\_buffer\_size**
  - ca. 25 – 33% vom RAM auf dedizierter Maschine
  - `SHOW STATUS LIKE 'Key_blocks_%';`
- **table\_open\_cache**
  - Running connections x used tables → 2 – 4k ist nicht ungewöhnlich! Siehe `Open[ed]_tables`.
- **table\_definition\_cache**
  - Siehe `Open[ed]_table_definitions` → 512 – 4096 ist nicht ungewöhnlich!
- **query\_cache\_type/query\_cache\_size**
  - Nicht zu gross machen ( $\leq 128$  M), bei sehr hoher Concurrency aber schädlich!

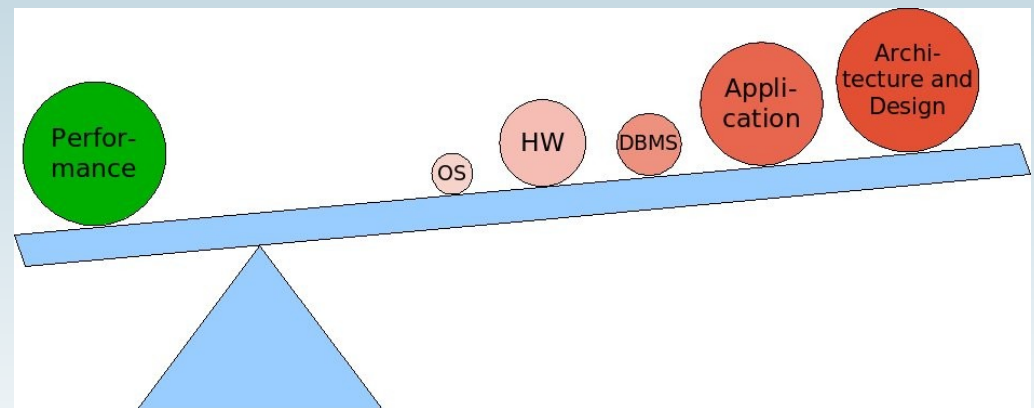


# Weitere Hilfe

- **Wie messen?**
  - `SHOW GLOBAL STATUS;`
  - `SHOW ENGINE INNODB STATUS\G`
- **ca. 330 Variablen**
- **ca. 310 Status Informationen**
- **MySQL Database Health Check:**
  - <http://www.fromdual.com/mysql-database-health-check>
- **MySQL Doku, Server Status Variablen:**
  - <http://dev.mysql.com/doc/refman/5.5/en/server-status-variables.html>
- **Das wärs soweit!**



# Applikations-Tuning



# Applikations-Tuning

- **Index Tuning**
  - **Primary Key (InnoDB) → Länge der Sekundärindices!**
  - **(partiell) Redundante Indices weg**
  - **Indices mit einer geringen Kardinalität weg!?!**
- **Query Tuning**
  - **SHOW PROCESSLIST;**
  - **Slow Query Log (ab 5.1 dynamisch!)**
  - **log\_queries\_not\_using\_indexes = 1**
  - **mysqldumpslow -s t slow.log > slow.log.profile**
  - **EXPLAIN SELECT ...**



# EXPLAIN

- Der Schlüssel zur Wahrheit!
- Query Tuning: x schneller möglich!

```
EXPLAIN SELECT * FROM test;
```

id	select_type	table	type	possible_keys	key	key_len	ref	rows	Extra
1	SIMPLE	test	ALL	NULL	NULL	NULL	NULL	261369	

- **EXPLAIN Output Format:**

<http://dev.mysql.com/doc/refman/5.5/en/explain-output.html>



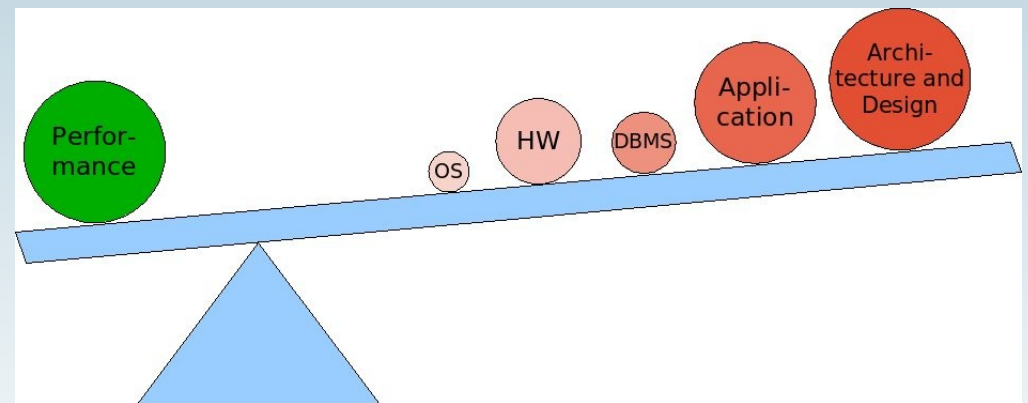
# Applikations-Tuning

- **Loslassen!**
  - Spalten, welche nicht gebraucht werden → weg (`char(0)` oder `ausNULLen`)!
  - Alte Daten → weg (archivieren)!
- **Schema Tuning**
  - `mysqldump --no-data > structure_dump.sql`
  - Richtige Datentypen und richtige Längen!
    - `int(1)` → 4 byte int!
  - `utf8` → nur wenn nötig
  - `NULL` or `NOT NULL`
- **Lokalität der Daten**
  - InnoDB Primary Key
  - V-Partitioning / (H-)Partitioning

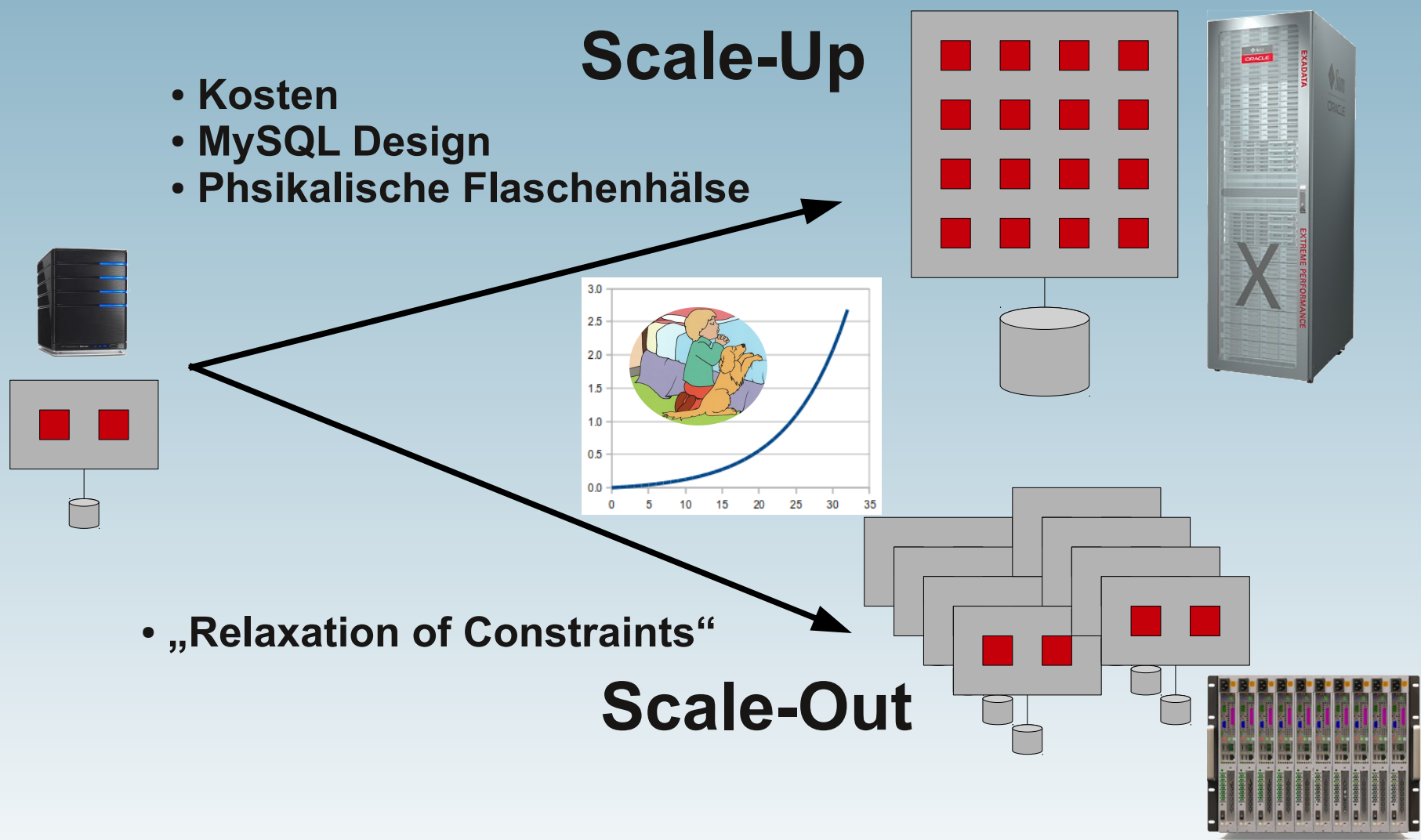




# Architektur & Design



# MySQL Scale-Out vs Scale-Up



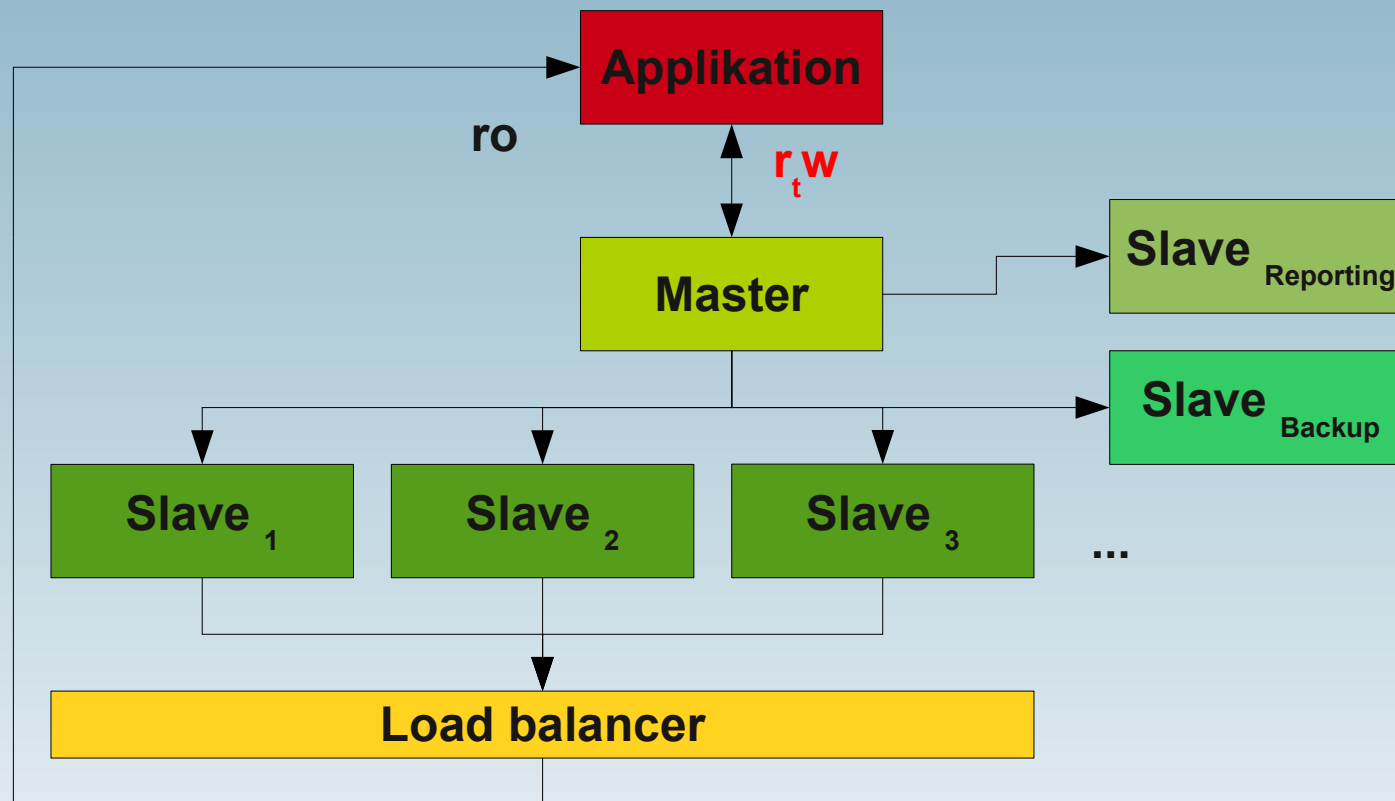
- Kosten
- MySQL Design
- Phsikalische Flaschenhalse

- „Relaxation of Constraints“

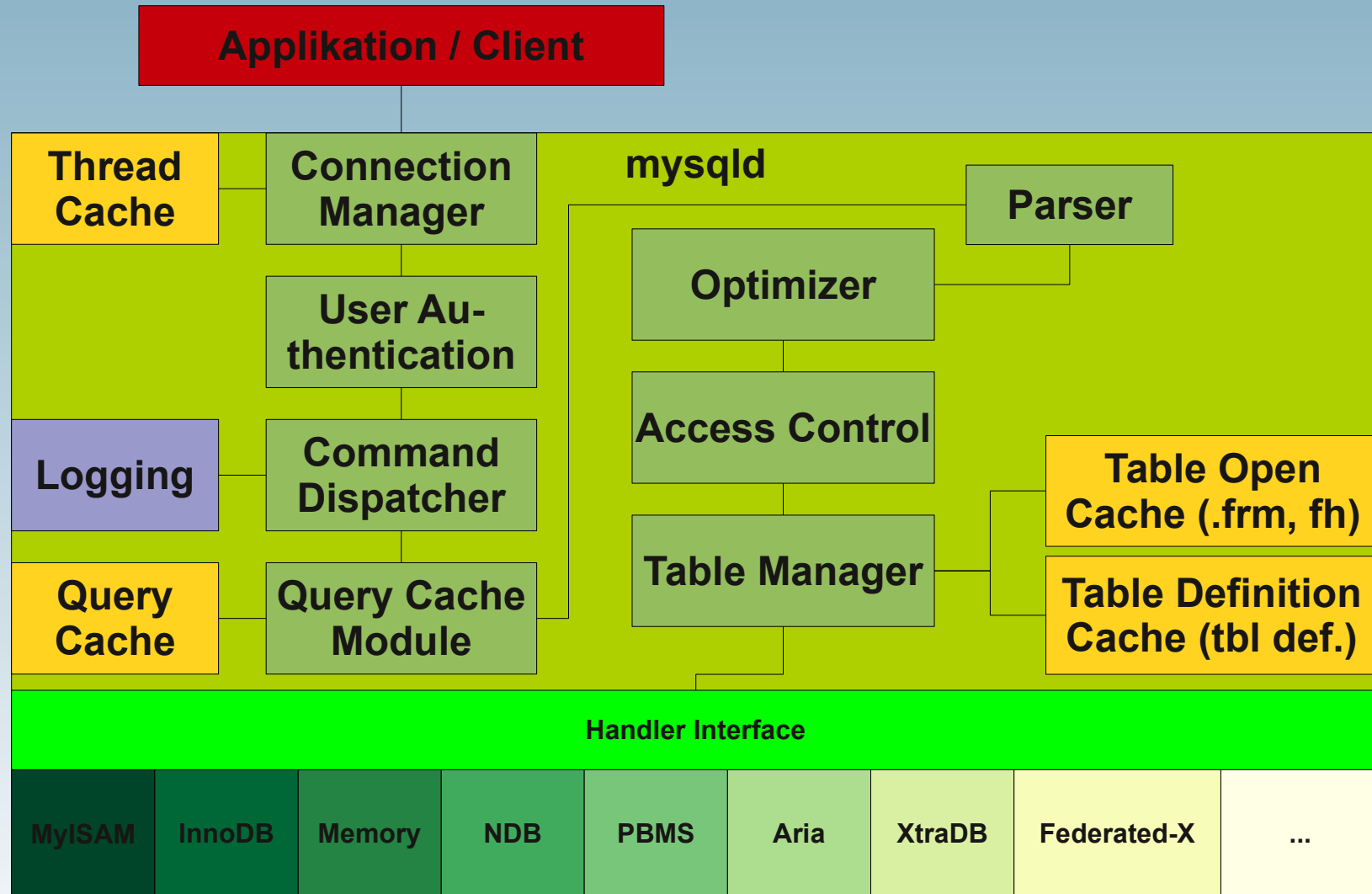
**Scale-Out**



# Der MySQL Scale-Out Ansatz



# MySQL Architektur



# Architektonische Möglichkeiten

- **RDBMS sind eine langsame Technologie!**
  - Daher: Cachen (heisse Daten in Cache!)
  - MySQL Memcached Plugin / Memcached
  - HandlerSocket
  - MySQL Cluster
- **Abstraktionslayer (ORM, Frameworks, etc.)**
  - Schneller entwickeln aber
  - Standard == „nicht optimiert“ → schlecht für Performance!
- **BLOB sind ungeeignet für RDBMS**
  - auf Filer legen
  - Blob Streaming (PBMS)
- **Sharding / verteilen**
  - „von Hand“
  - Spider SE
  - MySQL Replikation (r/w Traffic Split)
  - MySQL Cluster
  - Synchroner Replikation mit Galera

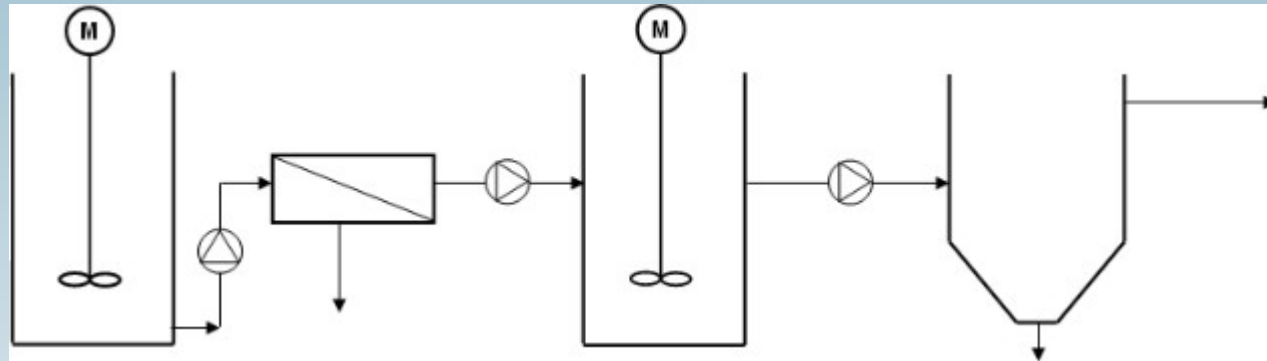


# Monitoring



# Was passiert, wenn...?

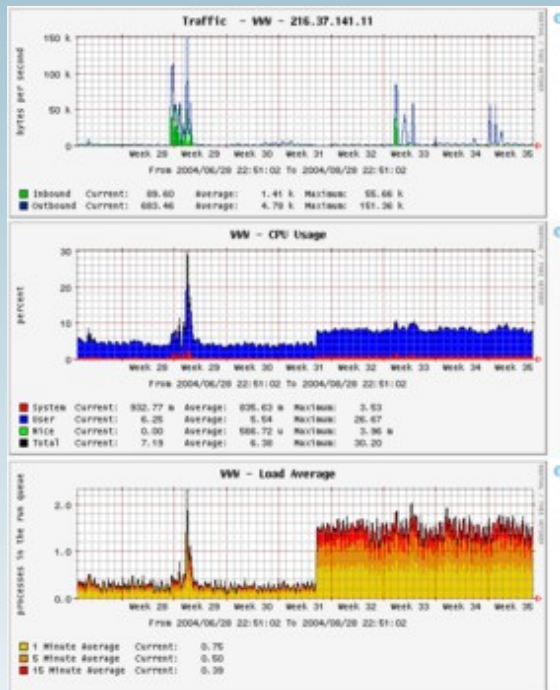
- Kunde fragt mich: Kann mein System 30% mehr Last vertragen?
- Chemische Verfahrenstechnik:



- Gibt es Unterschiede zu eine DB basierten System?
- Was brauche ich um diese Frage zu beantworten?

# Messen, messen, messen...

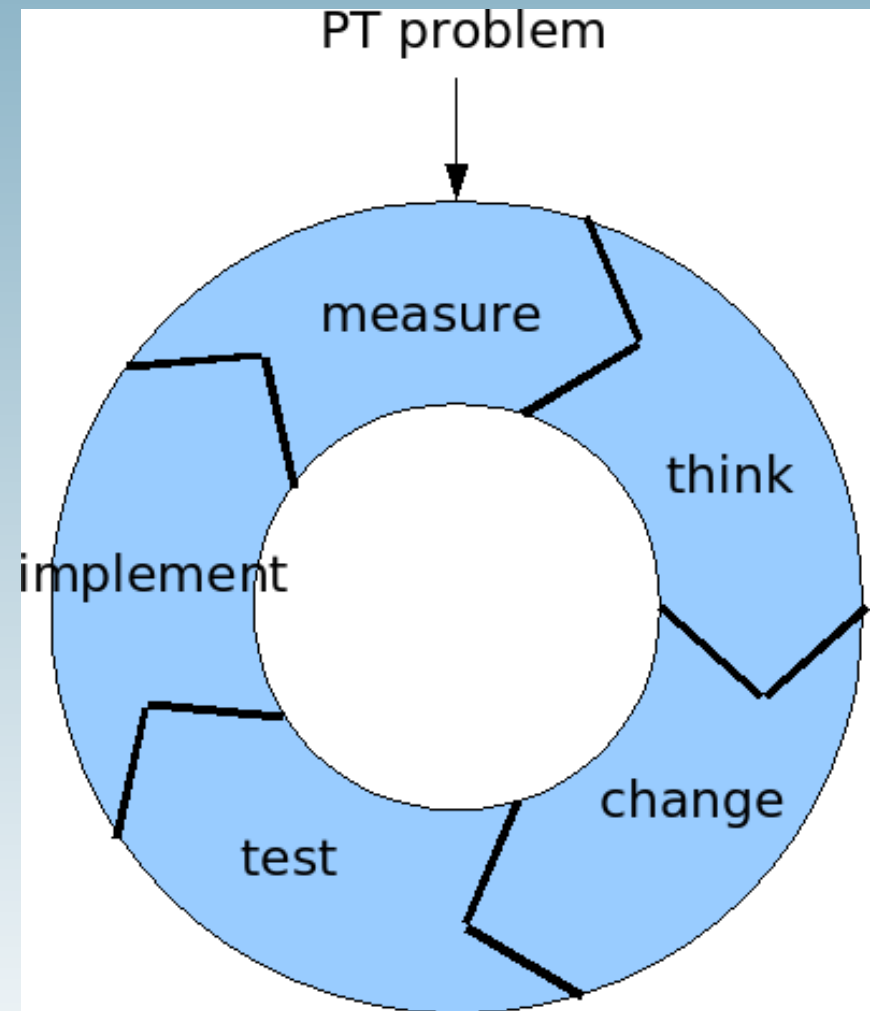
- Messen
- Simulieren → Kapazitätsplanung





# Der Bogen schliesst sich...

- Im Idealfall: Nur eine Änderung aufs Mal!



# Q & A

**Fragen ?**

**Diskussion?**

**Wir haben noch Zeit für persönliche und individuelle Beratungen...**

