
**Maatkit-Werkzeuge für Datenbankadministratoren**

# Ordnung statt Chaos

Die Tool-Sammlung Maatkit hilft beim Performance-Tuning und anderen Aufgaben für die MySQL-, aber auch für andere Datenbanken. Dieser Artikel gibt einen Überblick über ihre Funktionen. Falko Benthin

**Maatkit [1]**, benannt nach Maat, der ägyptischen Göttin für Ordnung und Gerechtigkeit, gibt Administratoren und Entwicklern eine umfangreiche, in Perl implementierte Werkzeugsammlung für die Kommandozeile in die Hand, mit der sie ihr Datenbank-Managementsystem um etliche Funktionalitäten erweitern und leichter verwalten.

Sie können damit etwa prüfen, ob die Replikation korrekt funktioniert, fehlerhafte Daten aufspüren und beheben, Tabellen schnell in Dateien überführen und wieder einlesen oder sich die Rechte der einzelnen Datenbanknutzer ausgeben lassen. Die meisten Funktionen sind für MySQL **[2]** erstellt, doch Maatkit unterstützt auch andere freie Datenbank-Managementsysteme, unter anderem PostgreSQL **[3]** und Memcached **[4]**.

Maatkit wurde 2006 von Baron Schwartz entwickelt. Seit 2008 führt hauptsächlich Daniel Nichter die Entwicklung fort, doch Schwartz liefert noch Anregungen für neue Werkzeuge und Features. Sowohl Schwartz als auch Nichter arbeiten beim Datenbankspezialisten Percona, der auch mehrfach Performance-steigernde Patches zu MySQL beigesteuert hat **[5]**.

## Reiche Tool-Sammlung

Maatkit lässt sich mit wenigen Tastenanschlägen installieren. Oft ist die Werkzeugsammlung schon in den Paket-Repositories der Distributionen vertreten. Ist das nicht der Fall, stehen im Downloadbereich des Projekts Quelltexte oder Pakete im Deb- und RPM-Format bereit. Ausgewählte Werkzeuge sind auch rasch mit »wget http://www.maatkit.org/get/*Toolname*« auf die Platte geholt und sofort einsetzbar. Aktuell umfasst Maatkit 30 Werkzeuge, das ADMIN-Magazin hat einige davon herausgepickt und genauer unter die Lupe genommen.

Sobald irgendwo ein neues System in Betrieb genommen werden soll, muss es vorher ausgiebig getestet werden, am besten mit realen Daten. Da die Daten meist vorhanden sind, bietet es sich an, die einzelnen Datenbanken auf das zu testende System zu übertragen. MySQL bringt zum Sichern von Datenbanken das Werkzeug »mysqldump« mit, das die Inhalte einer Datenbank oder aller Datenbanken in eine Datei schreibt, die auf einem anderen System nur noch eingelesen werden muss. »mysqldump« erledigt diese Aufgabe zuverlässig, aber langsam, denn es sichert alle Datenbanken und Tabellen sequenziell.

Bei einem System mit sieben Datenbanken, von denen nur eine etwas größer ist, misst »time mysqldump --skip-opt --create-options --database datenkrake


**Abbildung 1: »mk-log-player« extrahiert Anfragen aus Logfiles, die der Admin zum Test parallel und ohne Zeitverzögerungen an die Datenbank schicken kann.**

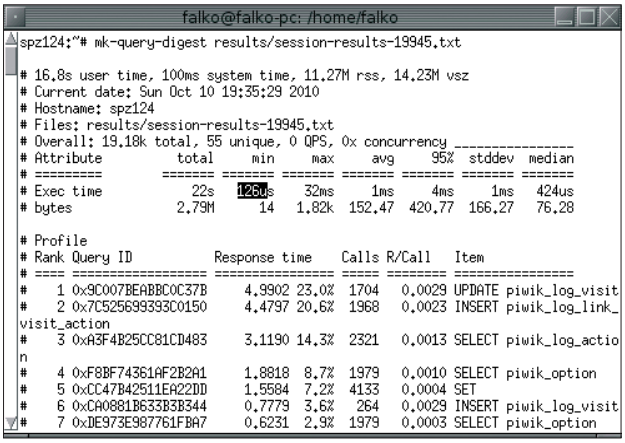


**Abbildung 2: »mk-query-digest« zeigt, wozu das DBMS in der Lage ist.**

wikkawiki piwik sprzblog mysql limesurvey -uroot -ppassword > backup.sql« insgesamt etwa 21 Sekunden:

```
real    0m21.626s
user    0m0.000s
sys     0m0.000s
```

Bei umfangreicheren Datenbanken auf nicht replizierten Servern ist »mysqldump« kaum mehr zu gebrauchen, denn die zu sichernden Tabellen werden für die Zeit der Sicherung gesperrt (um die Konsistenz sicherzustellen). Um solchen Abläufen etwas Dampf zu machen, enthalten die Maatkit-Tools das Werkzeug »mk-parallel-dump«, das die Daten, ein Mehrkernsystem vorausgesetzt, parallel in Dateien überträgt.

Standardmäßig nutzt »mk-parallel-dump« zwei Threads, um die Datenbanken zu kopieren. Die Anzahl kann bei entsprechenden Systemvoraussetzungen durch den Schalter »--threads« erhöht werden, wodurch sich der Kopiervorgang noch einmal spürbar beschleunigen lässt. Auf dem Testsystem erledigte »mk-parallel-dump« die gleiche Arbeit fast doppelt so schnell wie »mysqldump« (Listing 1).

## Alles im Speicher: Swap-Gefahr

Baron warnt davor, »mk-parallel-dump« als Backup-Lösung einzusetzen. Wer mit richtig großen Tabellen hantiert, merkt schnell warum. Das Werkzeug greift auf »mysqldump« zurück, ohne allerdings zwingend dessen Quick-Option zu nutzen. Das bedeutet, dass die Daten in den Arbeitsspeicher passen müssen, um in eine Datei geschrieben zu werden. Sind Tabellen zu umfangreich, beginnt das System zu swappen oder »mk-parallel-dump« notiert einen Fehler, bevor es weitermacht. Wenn der Administrator die Fehlermeldung überliest oder »mk-parallel-dump« von einem Cronjob ausgeführt wird, kann bei einer Wiederherstellung eine böse Überraschung warten. Das Gegenstück zu »mk-parallel-dump« ist »mk-parallel-restore«, das die kopierten Tabellen wieder in eine Datenbank schreibt, ebenfalls unter Nutzung mehrerer Threads.

## Stresstest fürs DBMS

Wer die Leistung seines Datenbank-Managementsystems testen möchte, findet in der Maatkit-Sammlung das Tool »mk-log-player«. Das Spiel, das sich hinter dem Werkzeug verbirgt, geht wie folgt. Mit

```
mk-log-player --split Thread_id --type ➥
genlog --only-select --base-dir=stresstest ➥
/var/log/mysql/mysql.log
```

durchforstet Maatkit die MySQL-Logdatei nach lesenden Zugriffen und erstellt im Verzeichnis »stresstest« nach Threads zusammengehörige Session-Dateien, die der Anwender dann parallel auf das Datenbank-Managementsystem (DBMS) loslassen kann (Abbildung 1). Sinnvoll sind solche Tests, um beispielsweise schnell Aussagen zu unterschiedlichen Konfigurationen des DBMS machen zu können.

Diese Sessions kann der Administrator anschließend mit »mk-log-player --play -uroot -ppassword ./stresstest --base-dir results h=localhost« an die Datenbank schicken. »mk-log-player« sendet standardmäßig in zwei parallelen Threads alle in den Sessions abgelegten Anfragen an die Datenbank, registriert, wie lange das DBMS für die Bearbeitung der Anfragen braucht, und schreibt die Ergebnisse für jeden Thread in eine eigene Datei im Slowlog-Format, die sich komfortabel mit »mk-query-digest« auslesen lässt.

## Langsame Queries finden

Sobald genügend Logdaten zusammengekommen sind, können Admins diese mit »mk-query-digest« auswerten. Per Default liest das Werkzeug die Daten im Slowquery-Format, kann aber auch die normale Logdatei sowie das Binärlog interpretieren oder, wenn kein Zugriff auf den Datenbankserver besteht, den mit TCPDump abgefangenen Traffic auswerten. Jede Abfrage ist mit einem Fingerprint versehen, sodass sich identische Abfragen, beispielsweise Selects oder Updates, gruppieren und separat auswerten lassen.

Die Ausgabe von »mk-query-digest« am Beispiel des Stresstests aus Abbildung 2 zeigt, dass im Test insgesamt zirka 19 200 Anfragen bestehend aus 55 un-

**Listing 1: Parallel gedumpt**

```
01 $ mk-parallel-dump -uroot -ppassword --database da
   tenkrake,wikkawiki,piwik,sprzblog,mysql,limesurvey
   --base-dir=db-backup/
02
03 CHUNK  TIME  EXIT  SKIPPED DATABASE.TABLE
04    db 11.43     0        0 datenkrake
05    db  1.34     0        0 limesurvey
06    db  2.75     0        0 sprzblog
07    db  2.67     0        0 mysql
08    db  3.45     0        0 piwik
09    db  2.50     0        0 wikkawiki
10   all 13.63     0        0 -
```

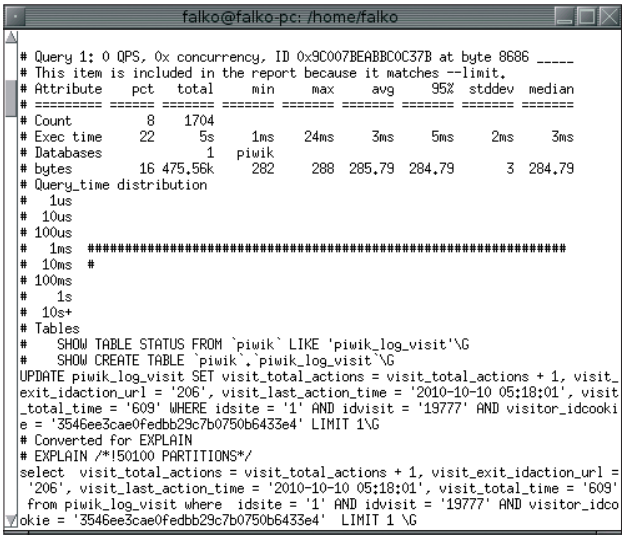


**Abbildung 3: »mk-query-digest« zeigt, an welchen Anfragen das System am meisten zu knabbern hat.**

terschiedlichen Kommandos abgesetzt wurden. Für die Bearbeitung benötigte das System, ein Server mit 768 MByte RAM und einem Pentium 3, insgesamt 22 Sekunden, die längste Beantwortung dauerte 32 Millisekunden, die kürzeste 126 Mikrosekunden, im Durchschnitt bearbeitete der Server Requests innerhalb einer Millisekunde.

## Verbrauchsstatistik

Interessant wird es weiter unten, wo das Tool auflistet, auf welche 19 Anfragen die meiste Serverzeit entfällt. Im Beispiel ist das DBMS die meiste Zeit damit beschäftigt, dem Webanalysewerkzeug Piwik zu dienen (Abbildung 3). Nach der Gesamtübersicht wird für jeden der 19 „Topverbraucher“ eine genauere Auswertung angeboten, sodass sich Admins ein Bild davon machen können, wo es eventuell hängt und ob vielleicht eine Optimierung vorzunehmen ist.

»mk-query-digest« bietet über 50 Kommandozeilen-Optionen an, mit denen DB-Admins beispielsweise angeben, ob sie nur Abfragen einer bestimmten Datenbank, eines selbst definierten Zeitraums oder in einer anderen Reihenfolge angezeigt bekommen möchten.

**Listing 2: »mk-loadavg«**

```
01 mk-loadavg -uroot -ppassword --watch "Status:innodb:Innodb_buffer_
   pool_pending_reads:>:15" --daemonize --pid /var/run/mk-loadavg.
   pid/--execute-command 'echo "DB-Server hängt" | mail -s "falko@web.de"
   017966666666@sms.web.de'
```

Bei Datenbankservern kann vorkommen, dass sie die Anfragen extrem langsam oder gar nicht mehr ausführen, weil irgendetwas im System hakt. Das Werkzeug »mk-loadavg« ist dann in der Lage, eine Vielzahl von Server-Werten auszulesen und bei interessanten Ereignissen den Administrator zu informieren beziehungsweise weitere Systemwerte abzufragen und zu Auswertungszwecken zu protokollieren.

Zu den interessanten Ereignissen kann beispielsweise zählen, wenn die laufenden MySQL-Threads eine definierte Grenze überschreiten, der Server zu swappen beginnt, der Prozessor zu Höchstleistungen aufläuft oder das DBMS nicht mehr fließend arbeitet.

Der Aufruf aus Listing 2 läuft als Daemon, fragt alle 60 Sekunden den Inno-DB-Status ab. Wenn mehr als 15 Leseanfragen auf ihre Bearbeitung warten, sendet das Tool über ein Mail-to-SMS-Gateway eine SMS an den Administrator, damit er sich schnell um Abhilfe bemühen kann.

## Ungenutzte Indexe

Indexe sollen den Zugriff auf in Tabellen abgelegte Datensätze beschleunigen. Wächst ein System oder wird es verändert, geraten früher benutzte Indexe bald in Vergessenheit. Ihre Erstellung und Aktualisierung benötigt aber weiterhin Zeit und Speicherplatz, sodass es aufschlussreich sein kann zu wissen, welche Indexe für Abfragen ungenutzt bleiben. Dazu bedarf es lediglich einer Logdatei im Slowlog-Format. Gibt die Datei »mysql-slow.log« nicht allzu viel her, besteht die Möglichkeit, eine normale Logdatei mittels »mk-query-digest« in das Slowlog-Format zu überführen, etwa durch:

```
mk-query-digest --type genlog --print?
 /var/log/mysql/mysql.log.1 > index.log
```

Danach analysiert »mk-index-usage index.log -uroot -password --host localhost« die Logdatei und prüft, welche Indexe nicht verwendet werden. Als Ergebnis schlägt das Tool vor, ungenutzte Non-Unique-Indexe zu entfernen, und liefert auch gleich die zugehörige Alter-Table-Anweisung (Abbildung 4). »mk-index-usage« ist nicht auf Non-Unique-Indexe beschränkt, der Anwender kann die Kriterien mit dem Schalter »--drop« auch auf Indexe der Typen »primary« und »unique« ausweiten. Das Werkzeug lässt sich auch auf einige ausgewählte Tabellen beschränken und kann die Ergebnisse in einer Datenbank speichern.

## Herzschlag

Beim Einsatz replizierter Systeme fällt es nicht undedingt sofort auf, wenn ein Slave sich verabschiedet. Mit »mk-heartbeat« existiert ein Maatkit-Werkzeug, das einmalig oder fortlaufend prüft, ob ein Slave noch am Leben ist. Dazu schreibt »mk-heartbeat --database --update --uroot -ppassword« kontinuierlich (standardmäßig jede Sekunde) in eine zu definierende Tabelle des Masters (Abbil-

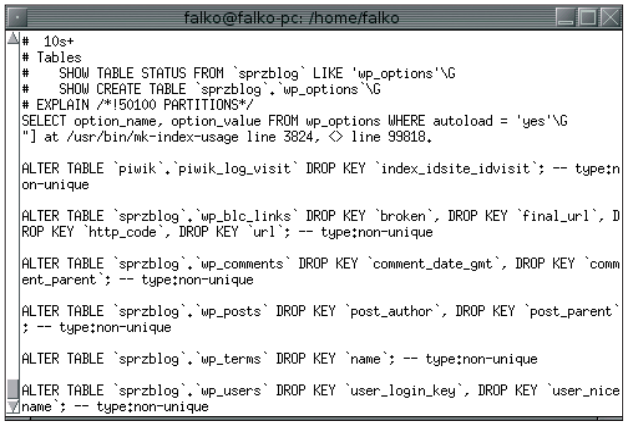


**Abbildung 4: »mk-index-usage« listet nicht genutzte Indexe auf und liefert gleich die SQL-Anweisung zum Löschen mit.**

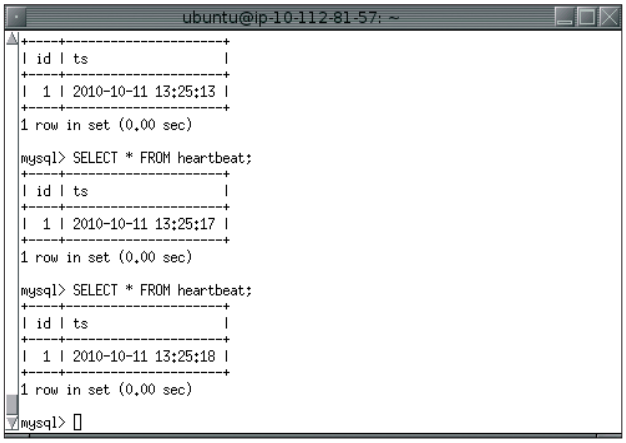


**Abbildung 5: Eine winzige Tabelle im Master gibt bei der Replikation mit »mk-heartbeat« den Takt vor.**

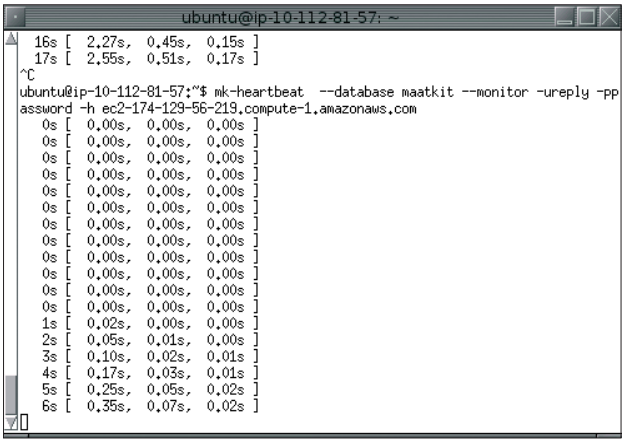


**Abbildung 6: »mk-heartbeat« registriert sofort, wenn bei der Replikation Verzögerungen auftreten.**

dung 5), um in einem (parallelen) zweiten Schritt mit »mk-heartbeat --database *DB* --monitor -u *Benutzer* -p *Passwort* -h *Slave*« zu prüfen, wann der Slave die Änderung übernimmt. Die Heartbeat-Tabelle legt das Programm mit dem Parameter »--create-table« selbst an, sonst muss dies der Admin von Hand machen.

Das Tool listet auf, wie schnell der Server durchschnittlich reagiert (Abbildung 6). Für die Berechnung werden per Default Zeitrahmen von 1, 5 und 15 Minuten herangezogen, die der Anwender aber auf Wunsch ändern kann. Fällt irgendeine Komponente aus oder erfolgt die Synchronisierung zu langsam, fällt das sehr schnell auf. Neben der kontinuierlichen Beobachtung ist mit »--check« anstelle von »--monitor« auch eine Einzelabfrage möglich, »--daemonize« schiebt die Überwachung des Herzschlags in den Hintergrund.

## Rechte und Pflichten

Mitarbeiter und Praktikanten kommen und gehen, Anwendungen werden in Rente geschickt und schnell kann es passieren, dass verflossene Anwender samt ihren Privilegien in der Datenbank vergessen werden. Wenn Nutzer mit ihrem Ausscheiden aus einer Organisation nicht auch automatisch ihre Passwörter vergessen, kann sich solch ein Fall irgendwann als Sicherheitslücke herausstellen.

Ein anderes Szenario wäre, dass die Rechte schnell von einem DBMS auf ein anderes übertragen werden sollen. Das Werkzeug »mk-show-grants« ermöglicht es, innerhalb von Sekunden die gesetzten Rechte abzufragen. Dabei ordnet es die Ausgabe im Gegensatz zu einem SQL-Select stets gleich, was ein Audit der Rechte vereinfacht.

Als Ergebnis erhalten Admins – wie auch bei anderen Maatkit-Tools – eine Reihe von SQL-Anweisungen. Handelt es sich um ein umfangreiches DBMS, können Admins in kommaseparierten Listen angeben, nur die Rechte bestimmter Nutzer auszugeben (»--only«) oder Anwender von der Erfassung ausschließen (»--ignore«). Der Schalter »--revoke« erstellt zu den Grant- passende Revoke-Anweisungen (Abbildung 7)

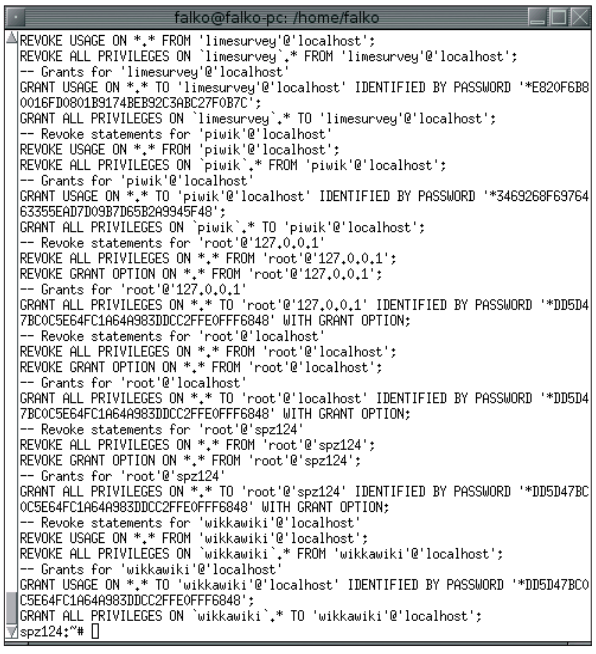


**Abbildung 7: »mk-show-grants« zeigt nicht nur, wer im DBMS welche Rechte hat, sondern liefert auf Wunsch auch den SQL-Befehl, um diese zu entfernen.**

## Fazit

Die hier erwähnten Werkzeuge zeigen nur einen kleinen Teil des Umfangs der Maatkit-Sammlung, deren Dokumentation zu durchstöbern durchaus lohnenswert sein kann. Die Werkzeuge sind alle in der Praxis entstanden, wurden vielfach getestet und können Administratoren jede Menge Arbeit abnehmen. *(ofr)* ■

### Infos

[1] Maatkit [http://www.maatkit.org]
[2] MySQL: [http://www.mysql.com]
[3] PostgreSQL: [http://www.postgresql.org]
[4] Tim Schürmann, „Memcached als verteilter Datenbank-Cache“: ADMIN 02/2009, [http://www.admin-magazin.de/content/memcached-als-verteilter-datenbank-cache]
[5] Caspar Clemens Mierau, „Moderne MySQL-Forks und Patches“: ADMIN 03/2010, [http://www.admin-magazin.de/content/moderne-mysql-forks-und-patches]

### Der Autor

Falko Benthin ist LPIC-zertifizierter Administrator und studiert Gesundheitsinformatik. In seiner Freizeit verschlingt er Kriminalromane und berichtet in seinem Blog [e-leseratte.de] über elektronische Lesegeräte.