LogAnalyser - Benutzerhandbuch

Autor: Nils Eckert

Stand: 30.08.2016

# Motivation

Der LogAnalyser ist ein Java Programm für die Performance-Analyse einer Webseite. Es zerlegt Logfiles und importiert die Daten in eine relationale Datendank. So sind beliebige per SQL formulierbare Abfragen und Auswertungen möglich. Das Tool verfügt über eine einer grafischen Benutzerschnittstelle, so dass es einfach bedient werden kann und für einen breiten Anwenderkreis einsetzbar ist.

Im Folgenden wird die Installation und Benutzung des Tools beschrieben.

# Installation

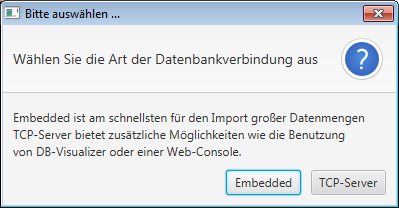
Die Software wird als ZIP Datei bereitgestellt. Zur Installation muss die ZIP Datei (z.B. LogAnalyser-1.2.1-bin.zip) auf die lokale Festplatte kopiert werden. Es wird dringend davon abgeraten die Anwendung auf einem Netzlaufwerk zu starten, da das Tool über eine integrierte Datenbank verfügt und ein Importvorgang nur auf einer lokalen (schnellen) Festplatte effizient und reibungslos funktionieren kann.

Für die Installation muss das ZIP-Archiv lediglich auf der lokalen Festplatte entpackt werden. In der ZIP Datei ist das Programm bereits in einem Ordner zusammengefasst. Es ist daher nicht nötig für das Extrahieren extra einen neuen Ordner anzulegen.

# Erster Start

* In den Ordner wechseln, in dem die ZIP Datei entpackt wurde.
* In den Ordner „LogAnalyser-<version>“ und dort in den „bin“ Ordner wechseln.
* Die Datei „start.bat“ per Doppelklick starten.

Es öffnet sich zunächst eine Windows Eingabeaufforderung und nach kurzer Zeit sollte der folgende Dialog erscheinen.



Sollte dieser Dialog nicht erscheinen und sich die Eingabeaufforderung nach kurzer Zeit ohne Fehler selbst beenden, so findet sich eventuell in Abschnitt 8 eine Lösung.

## Datenbank Modus Embedded und TCP-Server

Alle Log-Einträge werden in eine lokale Datenbank eingelesen, die in die Anwendung integriert ist. Diese Datenbank kann auf zwei Arten gestartet werden. „Embedded“ und als „TCP-Server“, der auch Verbindungen von außen erlaubt.

Beide Varianten haben Vor- und Nachteile, so dass der Anwender bei beim Start der Anwendung wählen kann. Der Hauptvorteil des Embedded Mode ist die mehr als doppelt so hohe Verarbeitungsgeschwindigkeit beim Import. Der TCP-Server Mode bietet jedoch eine übersichtlichere Benutzerschnittstelle für die spätere Auswertung. Die Empfehlung ist, für den Import größerer Datenmengen den „Embedded“ Modus zu verwenden und anschließend für die Auswertung das Programm zu beenden und im „TCP-Server“ Modus neu zu starten.

## Mehrere Installationen

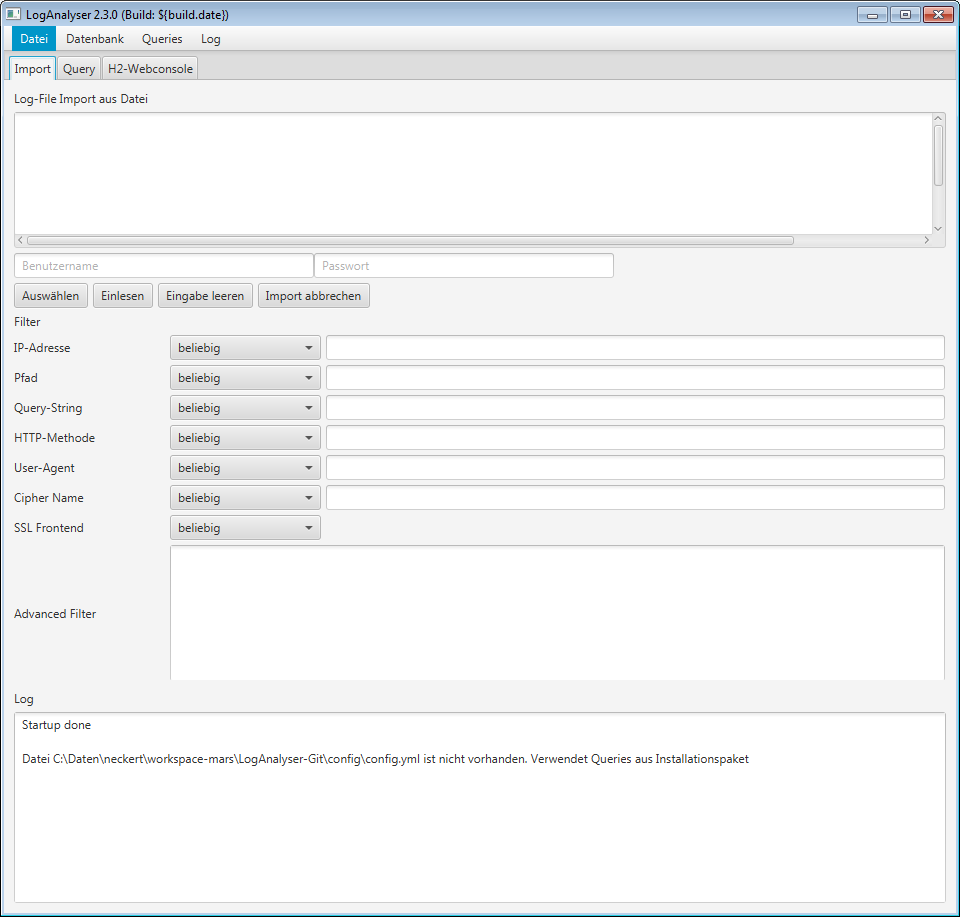
Es ist möglich die Anwendung mehrfach zu installieren, also die ZIP Datei mehrfach zu extrahieren und jeweils umzubenennen. Abhängig vom Modus (siehe 3.1) ist es nicht immer möglich mehrere Installationen zeitgleich zu starten.

Im Embedded Modus können mehrere Instanzen der Anwendung gleichzeitig gestartet werden. Es ist aber nicht möglich dieselbe Installation doppelt zu starten, da die Datenbank den Zugriff auf die internen Daten sperrt, um Datenverluste zu vermeiden. Auch die Menge des lokalen Speichers (RAM) kann die maximale Anzahl parallel startbarer Installationen begrenzen.

Der TCP-Server Modus startet den Datenbank Server und bindet dazu zwei TCP-Ports. Aus diesem Grund wird jeder weitere Start der Anwendung fehlschlagen, da die Ports bereits in Verwendung sind.

# Benutzer-Schnittstelle

Die Benutzerschnittstelle teilt sich in drei wesentliche Bereiche. Im oberen Bereich des Fensters befindet sich das Hauptmenü, am unteren Bildschirmrand befindet sich ein Textfeld mit Log-Ausgaben und der Hauptbereich ist in der Mitte der Anwendung sichtbar. Der Hauptbereich verfügt über mehrere Ansichten. Diese kann man über die Reiter „Import“, „Query“ und „H2-Webconsole“ anwählen.



Log-Ausgaben

Hauptmenü

Hauptbereich

Im Folgenden wird der Aufbau des Hauptmenüs und der drei Tabs im Hauptbereich näher beschrieben.

## Hauptmenü

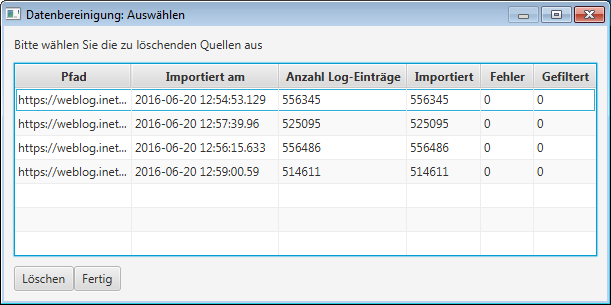
### Datei: Beenden

Beendet die Anwendung. Die Importierten Daten bleiben erhalten.

### Datenbank: Datenbereinigungs-Assistent...

Über dieses Menü öffnet sich ein neues Fenster mit einer Tabelle, die eine Übersicht über alle bisher durchgeführten Importvorgänge auflistet. Zu jedem Import wird der Zeitpunkt angegeben, an dem die Daten eingespielt wurden sowie die Anzahl der Log-Einträge.

Ein oder mehrere Datensätze können ausgewählt und über den Button „Löschen“ aus der Datenbank entfernt werden. Der Button „Fertig“ schließt den Dialog ohne weitere Aktionen.



### Datenbank: Alle Daten löschen

Über diesen Befehl kann schnell der gesamte Datenbestand gelöscht werden.

Das Verfahren ist erheblich schneller als alle Daten über den Bereinigungsassistenten zu löschen.

### Datenbank: Indizies anlegen / entfernen

Über einen Datenbank-Index können Abfragen und Auswertungen erheblich beschleunigt werden, der Import-Vorgang wird aber durch einen Index auch erheblich ausgebremst. Aus diesem Grund bietet die Anwendung die Möglichkeit den Index über das Menü anzulegen und zu entfernen.

Sinnvollerweise wird der Index vor einem großen Import entfernt und anschließend neu erstellt (angelegt).

Das Anlegen des Index kann bei großen Datenmengen mehrere Minuten dauern. Das wiederholte anlegen oder löschen wird abgefangen und beschädigt die Daten nicht.

### Queries

Dieses Menü zeigt eine Liste von nützlichen Queries als Vorlage für Datenbank-Auswertungen. Das Menü wird dynamisch erzeugt und zeigt alle Queries, die in der Datei queries.xml konfiguriert sind an.

Beim Klick auf einen Menüpunkt wird die hinterlegte Query sowohl in das Eingabefeld im Reiter „Query“ als auch in das Eingabefeld in der „H2-Webconsole“ kopiert. Dabei wird bei beiden Eingabefeldern die bisherige Eingabe überschrieben.

Details dazu finden sich Abschnitt 5.1 Queries.xml.

### Log: Log leeren

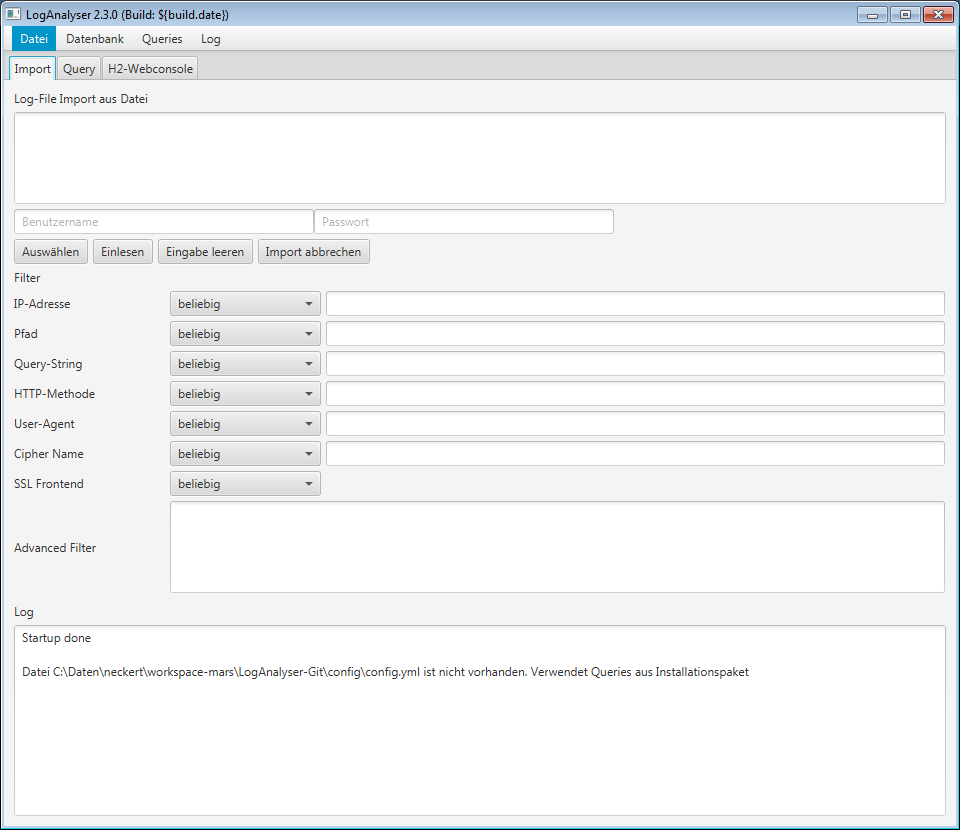
Löscht alle bisher im unteren Teil der Seite protokollierten Log-Meldungen.

## Import

Über den Reiter „Import“ können einzelne oder mehrere Log-Dateien eingelesen werden.

Dazu müssen im Eingabefeld die kompletten Pfade oder URLs eingetragen werden. Pro Zeile darf immer nur eine URL oder Datei stehen.

Die Dateien oder URLs können händisch eingetragen und per Copy&Paste eingefügt werden. Zusätzlich besteht die Möglichkeit über den Button „Auswählen“ einen Dateiauswahldialog zu starten. In diesem Dialog können mehrere Dateien ausgewählt werden. Die ausgewählten Dateien werden an die bisherige Eingabe angefügt. Duplikate werden dabei automatisch entfernt.



Die Eingabe von Benutzername und Passwort ist optional, für den direkten Import einem Webserver aber in der Regel erforderlich.

Der Import startet beim Klick auf „Einlesen“. Dabei werden GZip Dateien automatisch entpackt, wenn der Pfad oder die URL auf „.gz“ endet.

Ein laufender Import kann über den Button „Import abbrechen“ gestoppt werden. Bisher importierte Daten werden jedoch nicht wieder gelöscht.

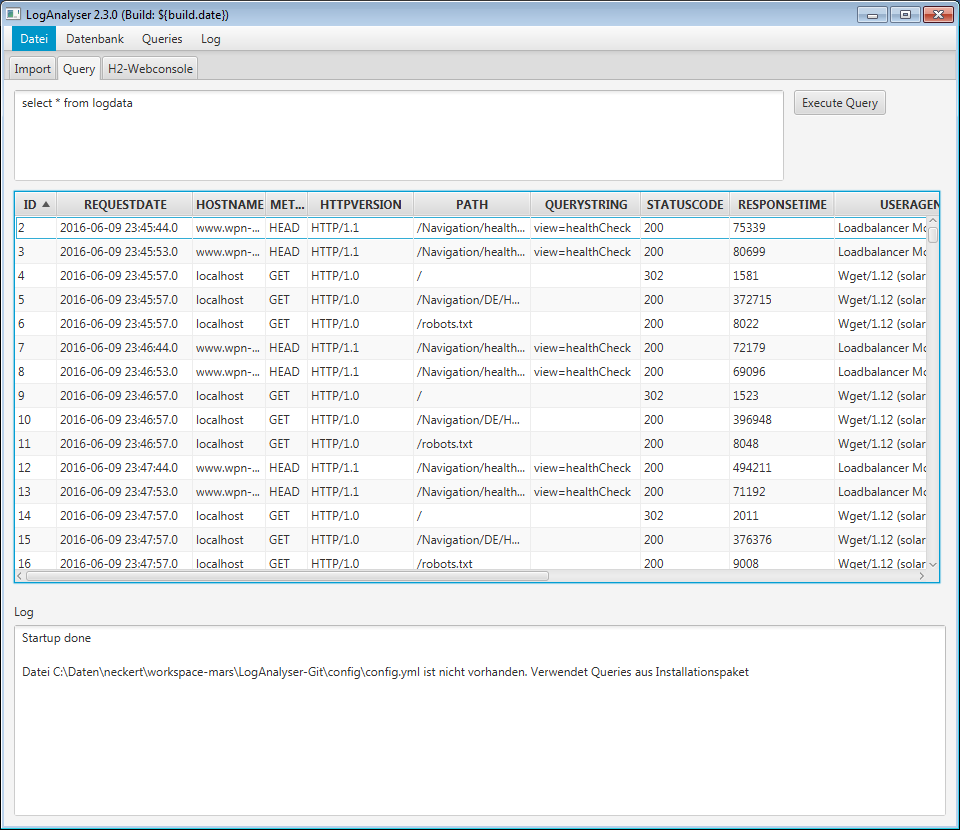
In Abschnitt 9 befindet sich eine Sammlung mit Import URLs.

### Filter

Über die Eingabemaske „Filter“ im unteren Bereich der Seite können bereits vor dem Einfügen in die Datenbank Logeinträge herausgefiltert werden, die nicht den angegebenen Kriterien entsprechen.

So ist es zum Beispiel möglich nur alle Request auf die Startseite zu importieren. Dadurch wird die Datenmenge erheblich kleiner und die Queries laufen schneller.

## Query

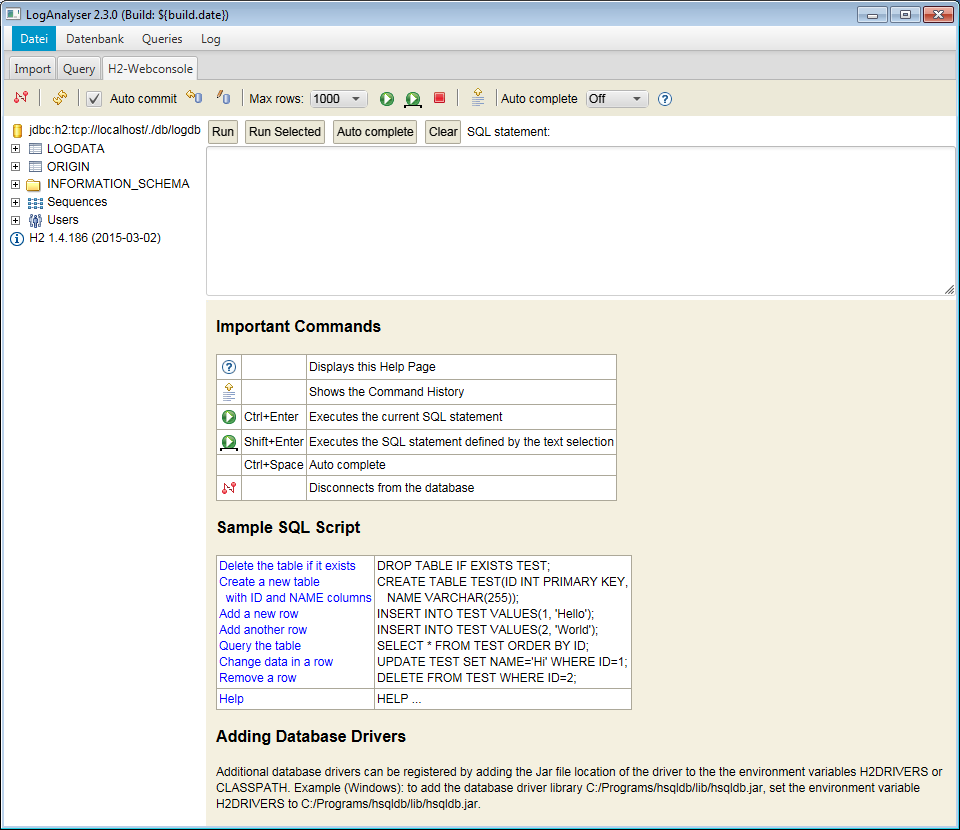


Diese Maske bietet eine stark vereinfachte DB-Abfrage und funktioniert auch im Embedded-Mode. Zum Beispiel kann diese Eingabemaske benutzt werden um Indizes anzulegen, zu löschen oder einfache Queries abzusetzen.

Für aufwendige Auswertungen wird die Verwendung der H2-Webconsole (Abschnitt 4.4) empfohlen.

## H2-Webconsole

Im TCP-Server Modus wird neben dem Datenbank-Server noch der H2-Webserver auf Port 10500 gestartet. Dort bietet die H2 Datenbank eine webbasierte Schnittstelle zum Abfragen der Datenbank an.



Die Webconsole kann auch, solange das Programm läuft, aus dem Browser über die URL <http://localhost:10500/> gestartet werden. Dort erscheint jedoch zunächst ein Login-Dialog, in dem die Datenbank Zugangsdaten abgefragt werden. Um diesen Schritt zu vereinfachen wurde die Webconsole in Anwendung integriert und der Login automatisiert, so dass man als Anwender direkt das Query-Tool sieht.

Die über das Hauptmenü Queries hinterlegten Statements werden auch direkt in das Eingabefeld der H2-Webconsole kopiert.

**Hinweis**: Die H2-Webconsole arbeitet mit Sessions, die nach einer gewissen Zeit ablaufen. Ist dies der Fall leitet die Webconsole auf die Login-Seite um. Das wird aber von der Anwendung erkannt und der Login wird automatisch durchgeführt. Es erscheint sofort wieder die Standard-Eingabemaske. Die Query ist aber nicht mehr sichtbar und wurde auch nicht ausgeführt. Ebenso hat der Klick auf „Logout“ keine Funktion, da die Anwendung sich automatisch sofort wieder anmeldet.

# Konfiguration

## config.yml

Über die Datei config.yml können, ohne die Anwendung neu zu kompilieren, die vordefinierten Queries aktualisiert und erweitert werden. Ebenso ist es möglich im Rahmen der technischen Möglichkeiten das Logformat anzupassen. Die Datei wird beim Start der Anwendung eingelesen. Nach einer Änderung muss die Anwendung folglich neu gestartet werden.

### Aufbau der Datei

Die Datei muss im yml Format vorliegen.

In der obersten Ebene wird ein logformat und ein queries attribut erwaretet.

Beispiel

---

logformat: '%{X-Forwarded-For}i %l %u %t "%r" %>s %b "%{Referer}i" "%{User-Agent}i" "%{Host}i" %D "%{Age}o" %{If-Modified-Since}i "%{X-CipherName}i" "%{X-CipherVersion}i" "%{X-CipherBits}i" "%{SSL-FRONTEND}i"'

queries:

- name: "Anzahl DB Einträge"

description: "Ermittelt die Anzahl der Einträge in der LOGDATA Tabelle."

query: |+

select

count(\*)

from logdata

comment: Kommentar

queries hat vier Attribute:

* name: Der Name der Query, der im Menü angezeigt wird (erforderlich)
* query: Die SQL-Query oder eine Vorlage (erforderlich)
* description: Eine Beschreibung, was die Query macht. Wird als Tooltip (Mouse-Over) im Menü verwendet. (optional)
* comment: Ein Kommentar. Wird in der Anwendung nicht verwendet. (optional)

**Tipp**: Bei mehrzeiligen SQL Abfragen sollte CDATA verwendet werden. Beispiel:

query: |+

select

count(\*)

from logdata

# Datenmodell

In der Datenbank gibt es zwei Tabellen. Die Tabelle LOGDATA hält die aufbereiteten Einträge der eingelesenen LOG-Files. Die Tabelle ORIGIN verknüpft diese Daten mit den Importvorgängen.

## LOGDATA

|  |  |  |
| --- | --- | --- |
| Spalte | Datentyp | Kommentar |
| ID | BIGINT(19) | Technischer Schlüssel |
| CACHEAGE | INTEGER(10) | Cache-Age Header |
| CACHEAGEPRESENT | BOOLEAN(1) | Wurde der Parameter im Logfile erkannt |
| CIPHERBITS | VARCHAR(255) | X-CipherBits Header Value |
| CIPHERNAME | VARCHAR(255) | X-CipherName Header-Value |
| CIPHERVERSION | VARCHAR(255) | X-CipherVersion Header-Value |
| FORWARDEDFORIP | VARCHAR(128) | Lokale IP Adresse im LAN des Clients. I.d.R. nur bei Zugriff über Firmen-Proxy im Netz des Client. |
| HTTPVERSION | VARCHAR(10) | Die HTTP Version (z.B. HTTP/1.1) |
| IP | VARCHAR(128) | IP Adresse des Clients (kann auch IPv6 sein) |
| METHOD | VARCHAR(16) | Die HTTP Methode (GET, POST, PUT, …) |
| PATH | VARCHAR(2048) | Die Angefrage URL. Query-Parameter und die SessionID sind entfernt, so dass nach dieser Spalte gruppiert werden kann. Achtung: Die SessionID wird nicht entfernt, wenn Sie nicht ordnungsgemäß erkannt wurde. |
| QUERYSTRING | VARCHAR(2048) | Der Teil der angefragten URL, der nach dem Fragezeichen in einer GET oder POST anfrage übermittelt wurde. |
| REFERER | VARCHAR(2048) | Der REFERER-String aus dem HTTP Header |
| REQUESTDATE | TIMESTAMP(23, 10) | Der Zeitstempel der HTTP Anfrage |
| RESPONSETIME | INTEGER(10) | Die Zeit, die zum beantworten der Frage benötigt wurde in **Mikrosekunden**. |
| SESSIONID | VARCHAR(128) | Die HTTP-SessionID (sofern erkannt) |
| SSLFRONTEND | BOOLEAN(1) | SSL-Frontend Request Header |
| STATUSCODE | INTEGER(10) | Der HTTP Status-Code (z.B. 200, 404, 301) |
| TRANSFEREDBYTES | INTEGER(10) | Die Anzahl der übermittelten Bytes |
| USERAGENT | VARCHAR(1024) | Der User-Agent Header |
| ORIGIN\_ID | BIGINT(19) | Eine Referenz auf den Datensatz, mit dem dieser Logeintrag importiert wurde. |

# Unterstützte Log-Elemente

|  |  |
| --- | --- |
| Log Element | Beschreibung |
| %{X-Forwarded-For}i | Request Header X-Forwarded-For |
| %t | Zeitpunkt des Requests |
| %r | First line of request |
| %{User-Agent}i | User-Agent |
| %{Referer}i | Referer String |
| %{Host}i | Hostname (Host-Header) |
| %v | The canonical ServerName of the server serving the request |
| %D | The time taken to serve the request, in microseconds |
| %{Age}o | Age Response Header (Cache Age) |
| %s | StatusCode (before mod\_rewrite) |
| %>s | Final StatusCode |
| %b | Transferred bytes |
| %{X-CipherName}i | Wert des Request Headers X-CipherName |
| %{X-CipherBits}i | Wert des Request Headers X-CipherBits |
| %{X-CipherVersion}i | Wert des Request Headers X-CipherVersion |
| %{SSL-FRONTEND}i | Wert des Request Headers SSL-FRONTEND |

# Tipps

## Datenmenge klein halten

Je größer die Datenmenge in der Datenbank, desto langsamer werden die Abfragen oder das Anlegen von Indizes. Es empfiehlt sich daher immer nur so viele Daten zu importieren, wie für die Bearbeitung der aktuellen Aufgabe benötigt werden.

Sollte man parallel an mehreren unabhängigen Aufgaben mit unterschiedlichen Datensätzen arbeiten, so empfiehlt es sich dafür auch getrennte Datenbanken zu verwenden. So werden die Abfragen einfacher, da man diese nicht auch nicht auf die relevanten Eingangsdaten beschränken muss. Zudem ist die Performance erheblich besser.

Das ganze lässt sich am einfachste so umsetzen, dass man eine Kopie vom LogAnalyser für jede Aufgabe verwendet. Dazu sollte am einfachsten das ZIP-File neu entpackt werden. Wenn man eine bestehende Installation kopiert, sollte man darauf achten das „db“-Verzeichnis nicht mit zu kopieren, da dieses Verzeichnis in der Regel mehrere hundert bis Gigabyte groß ist.

Es ist aber aktuell nicht mehrere Instanzen der Anwendung im TCP-Server Modus gleichzeitig geöffnet zu haben.

## Empfehlung: Vorgang Import

Vor dem Import von Daten sollten zunächst die folgenden Schritte durchgeführt werden:

* Programm beenden und im Embedded Mode neu starten
* Löschen aller alten und nicht für die geplante Recherche relevanten Daten
* Index entfernen
* Alle relevanten Daten importieren
* Programm beenden und im TCP-Server Mode neu starten
* Index anlegen
* Abfragen ausführen

## Datenbankauswertungen

Datenbankauswertungen sollten nach Möglichkeit mit der H2-Webconsole durchgeführt werden. Dazu muss die Anwendung im TCP-Server Mode gestartet werden.

Die H2-Datenbank bietet einen WebServer an, der ein einfaches DB-Query-Tool integriert hat. Dieses Query-Tool bietet wesentlich mehr Funktionen als das integrierte „Query“-Modul der Anwendung. So kann z.B. die Struktur der Tabellen angezeigt werden, um entsprechende Queries zu formulieren.

Folgende Features sind vorhanden:

* Historie der bisherigen DB-Abfragen
* Nur einen Teil der Query ausführen
* Struktur der Tabellen anzeigen
* Einstellen, wie viele Zeilen angezeigt werden sollen
* Langsame Query abbrechen
* …

## Wichtige H2 Funktionen

### Datum formatiert ausgeben

Diese Funktion wird bspw. benötigt, um Daten nach Stunden oder Tagen zu gruppieren.

FORMATDATETIME(requestdate, 'yyyy-MM-dd HH')

### Count, Mittelwert, Min, Max, Summe

select

count(\*),

avg(responsetime),

min(responsetime),

max(responsetime),

sum(responsetime)

from LOGDATA

### Übersicht über alle Funktionen

Eine vollständige Dokumentation der in H2 integrierten Funktionen findet sich hier:

<http://www.h2database.com/html/functions.html>

## Zusätzlichen Index erstellen oder Löschen

Anlegen:

CREATE INDEX IF NOT EXISTS IDX\_<SPALTENNAME> ON LOGDATA(<SPALTENNAME>);

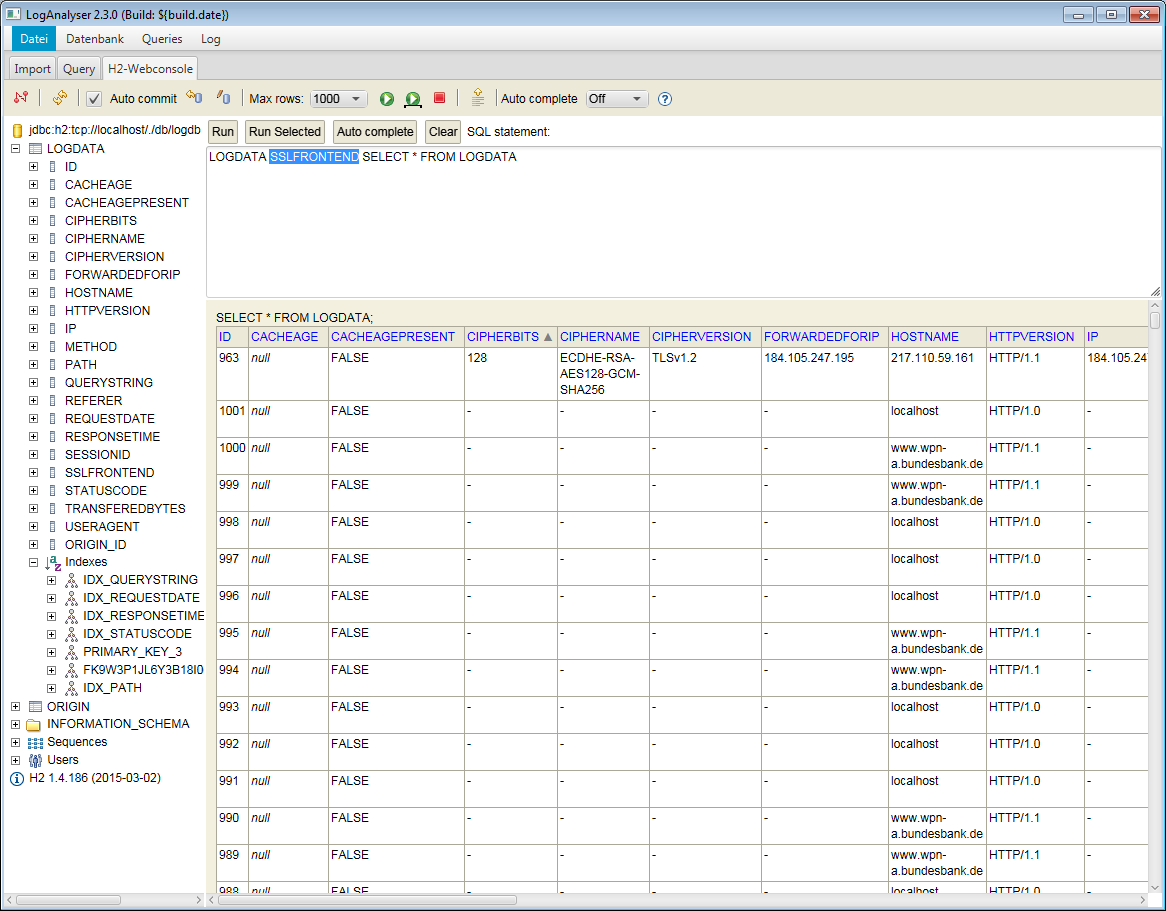
Löschen:

DROP INDEX IF EXISTS IDX\_<SPALTENNAME>;

Alle Indizies anzeigen:

select INDEX\_NAME from INFORMATION\_SCHEMA.INDEXES where index\_name like 'IDX\_%'

Alternativ können Indizies auch visuell über die H2-Webconsole ausgelesen werden:



## Import vieler Log-Files über Template-Muster

Muss man viele Log-Files importieren, dann besteht die Gefahr, dass beim Copy&Paste sich Tippfehler einschleichen. Um das zu vermeiden, unterstützt die Eingabe-Box auf der Import Maske auch einen Template-Mechanismus.

Es kann also bspw. das Datum in einer Variable gespeichert und dann in allen sechs URLs genutzt werden. So muss noch eine Stelle angepasst werden, wenn man einen weiteren Tag einlesen muss.

<#assign day = "2015-07-07" >

https://weblog.domain.de/weblog/host1/path/access.log.${day}.gz

https://weblog.domain.de/weblog/host2/path/access.log.${day}.gz

https://weblog.domain.de/weblog/host3/path/access.log.${day}.gz

Im Extremfall lassen sich auch die Hostnames in einem Array definieren und die Liste der URLs über eine Schleife erstellen.

<#assign

baseUrl = "https://weblog.domain.de/weblog"

hosts=["host1", "host2", "host3", "host4"]

day = "2015-07-07"

>

<#list hosts as host>

${baseUrl}/${host}/path/access.log.${day}.gz

</#list>

Der Template-Mechanismus stammt aus Freemarker. Die vollständige Dokumentation findet sich hier: <http://freemarker.org/docs/dgui_quickstart_template.html>

# Nützliche Queries

## Abfragen pro Stunden

Um alle Anfragen pro Sekunde, Minute oder Stunde auszuwerten, müssen die Daten nach der Spalte „REQUESTDATE“ gruppiert werden. Das geht entweder über die Methoden YEAR, MONTH, WEEK, HOUR, MINUTE oder über die entsprechend formatierte Ausgabe des REQUEST Date.

Die Variante über die Datums-Funktionen HOUR, MINUTE, SECOND ist etwas schneller und bietet einfache Möglichkeiten die Daten weiter einzuschränken. Die FORMATDATETIME Variante ist eine etwas kompaktere Query.

In der Variante mit HOUR, MINUTE und SECOND lassen sich durch die vielen separaten Bedingungen auch einzelne Einschränkungen der WHERE-Clause durch auskommentieren (-- am Anfang der Zeile) entfernen.

Variante 1:

select

year(requestdate),

week(requestdate),

hour(requestdate),

count(\*)

from logdata

where

1 = 1

and year(requestdate) = 2015

and week(requestdate) = 11

and hour(requestdate) between 2 and 5

-- and minute(requestdate) between 0 and 59

group by

year(requestdate),

week(requestdate),

hour(requestdate)

order by

year(requestdate) asc,

week(requestdate) asc,

hour(requestdate) asc

Variante 2:

select

FORMATDATETIME(requestdate, 'yyyy-MM-dd HH'),

count(\*)

from logdata

group by

FORMATDATETIME(requestdate, 'yyyy-MM-dd HH')

order by

FORMATDATETIME(requestdate, 'yyyy-MM-dd HH')

Die Formate richten sich nach dem Java SimpleDateFormat. Hier findet sich eine Übersicht: <http://docs.oracle.com/javase/7/docs/api/java/text/SimpleDateFormat.html>.

# Probleme und Lösungen

## Programm startet nicht

Ursache Beim Doppelklick auf „start.bat“ öffnet sich eine Eingabeaufforderung. Das Fenster verschwindet aber sofort wieder.

Lösung Manuell eine Eingabeaufforderung starten und in das Verzeichnis wechseln, in dem der LogAnalyser entpackt / installiert wurde. Dort ins bin-Verzeichnis wechseln und start.bat in der Eingabeaufforderung starten. Nun erscheint die eigentliche Fehlermeldung die den Start abgebrochen hat.

## Fehler: Active Transaction

Ursache Bei der nebenläufigen Verarbeitung kam es zu einem Fehler. Das Programm hat eine Datenbank-Transaktion nicht ordnungsgemäß beendet.

Lösung Programm beenden und neu starten.

## Datenbank neu aufbauen

Ursache Es ist möglich, dass die Datenbank durch einen Programmabsturz korrupt wird oder die Datenmenge so groß ist, dass die Datenbank nicht reagiert oder das Programm nicht mehr startet.

Lösung Die Datenbank kann ganz einfach neu aufgebaut werden. Dazu muss zunächst das Programm gestoppt werden. Im Anschluss löscht man das Verzeichnis „db“ im Basis-Verzeichnis des LogAnalyser. Beim nächsten Start der Anwendung wird die Datenbank neu initialisiert.

## H2-Webconsole: Query wird nicht ausgeführt. Maske wird geleert.

Fehler Nach dem Klick auf „Run“ oder „Run Selected“ werden alle Queries in der Eingabe gelöscht und es erscheint kein Ergebnis.

Ursache Die H2-Webconsole ist eigentlich eine eigenständige Web-Anwendung, die der H2-Server über einen eigenen Webserver bereitstellt. Um dieses Query-Tool innerhalb der Anwendung verfügbar zu machen und die Anmeldung zu automatisieren wurde das Tool über einen integrierten Web-Browser eingebunden.

Dennoch läuft nach einiger Zeit Inaktivität die Session ab. Sendet man nach dieser Zeit eine Query ab, so wird die Eingabe verworfen und man wird auf die Login-Seite umgeleitet.

Der LogAnalyser erkennt die Login-Seite und führt automatisch den Login erneut durch. Das geht so schnell, dass man dies als Anwender gar nicht bemerkt. Dennoch geht die Query verloren.

Lösung Queries sollten am besten hin und wieder in einen externen Text-Editor kopiert werden, um den Verlust zu vermeiden.