

## Modul 105: Daten analysieren und strukturieren

### Unity und SQLite Crashkurs

#### Ausgangslage

Nachdem Sie sich v.a. theoretisches Wissen über Daten und Datenbanken erworben haben, ist es nun an der Zeit, praktisch zu arbeiten. Einige werden bereits über Grundkenntnisse des Betriebssystems Linux verfügen. Die wenigsten werden SQLite oder andere relationale Datenbanksysteme kennen. Weder das eine noch das andere ist Voraussetzung für dieses Arbeitsblatt. Wir beginnen ganz von vorne, entweder handelt es sich um Neuland für Sie oder Sie können die Aufgaben als Repetition betrachten.

#### Ziele

- Kennenlernen von Ubuntu Linux und Unity
- Sich im Dateisystem zurechtfinden
- Arbeiten mit «Dateien» (grafischer Dateimanager)
- Arbeiten mit der Konsole
- Arbeiten mit SQLite3
- Arbeiten mit SQLiteman

#### Vorgehen

Gehen Sie dieses Arbeitsblatt Schritt für Schritt durch. Konsultieren Sie die Unterlagen, suchen Sie zusätzliche Informationen im Internet oder fragen Sie Ihre Lehrperson.

#### Unterlagen / Quellen

- Linux Kurzreferenz auf Moodle: Linux-auf-einem-Blatt.pdf
- SQL Referenz auf Moodle: SQL-Referenz.pdf
- Offizielle SQLite Dokumentation: <https://sqlite.org/docs.html>
- SQLite Tutorials:
  - <http://www.w3ii.com/de/sqlite/default.html>
  - <https://www.tutorialspoint.com/sqlite/>
  - <http://www.sqlitetutorial.net/>

**Zeit:** 2 Lektionen

#### Übersicht SQLite



#### SQLite – Datenbank mit Potenzial

SQLite ist eine kommandozeilenbasierte SQL Datenbank für Linux, MacOS und Microsoft Betriebssysteme. *Eine kleine Allzweckwaffe der Datenspeicherung!*

SQLite ist eine **lizenzfreie, intrinsische (aus sich selbst bestehende), plattformunabhängige, installations- und konfigurationsfreie** Datenbank, die das **Transaktionsprinzip von SQL** unterstützt. Wo andere Datenbanksysteme Migrationsschnittstellen benötigen, migriert oder besser "verschiebt" SQLite Datenbanken und Daten mit Kopieren und Einsetzen (ctrl+c und ctrl+v). Die benötigten Dateien sind sehr klein.

SQLite bringt viele Vorteile und wird in Applikationen als Speicherkomponente benutzt. Es ist **sicher, schnell** und vor allem „**reduced to the max**“. Alles was nicht nötig ist und Kilobytes von Speicher oder Performance kostet wurde eliminiert. Folgende Hersteller unterstützen SQLite finanziell:



Mehr zu SQLite unter <https://www.sqlite.org/>

## Ubuntu



Ubuntu ist ein grafisches Linux Betriebssystem. Smartlearn setzt lediglich Long Term Support (LTS) Distributionen ein, welche 5 Jahre aktualisierbar und gewartet sind.

<https://ubuntuusers.de/>

<https://wiki.ubuntuusers.de/KDE>

<https://wiki.ubuntuusers.de/Long Term Support>

## Unity



Unity ist im Moment die Standard-Desktop-Umgebung für Ubuntu. Unity ist keine Programmsammlung und nutzt v.a. Gnome-Anwendungen. Vor Kurzem wurde bekannt, dass Ubuntu Unity aufgibt und möglicherweise zu seiner ursprünglichen Desktop-Umgebung Gnome zurückkehren wird.

<https://wiki.ubuntuusers.de/Unity/>

[https://de.wikipedia.org/wiki/Unity \(Benutzeroberfläche\)](https://de.wikipedia.org/wiki/Unity_(Benutzeroberfläche))

## smartlearn-Lernumgebung

Wir verwenden in diesem Modul die virtuelle Maschine **bmLP1** mit Ubuntu und Unity.



### Ubuntu

#### Applikationen:

- Dateien
- Konsole
- SQLite3
- Sqliteman

#### Datenbank:

- schueler.db

## Crash-Kurs Unity und Konsole

Konsole = Kommandozeile = Terminal

Es gab mal eine Zeit, da wurden alle Computer via Terminal bedient. Ein Computer braucht keine grafische Benutzeroberfläche, fast alle Aufgaben lassen sich wunderbar via Terminal erledigen. Warum also kam man auf die Idee einer grafischen Benutzeroberfläche? Aus einem simplen Grund: Damit wird es einfacher für den Menschen, den Computer zu bedienen. Natürlich gibt es auch Aufgaben, die sich nur via grafische Oberfläche erledigen lassen, wie z.B. die Bearbeitung von Bildern.

Die meisten Aufgaben wie die Verwaltung von Dateien oder das Verwalten und Bedienen von Datenbanken lassen sich sehr gut mit dem Terminal erledigen. Wenn man nun einen grafischen Dateimanager verwendet, dann dient dies nur der vereinfachten Bedienung. Im Hintergrund werden genau dieselben Befehle ausgeführt, die man via Konsole eingibt (oder eingeben würde...).

Wir werden bei der Dateienverwaltung mit der Konsole und der grafischen Oberfläche arbeiten. Die Konsole ist wichtig für das Verständnis, was wie funktioniert und was bei der grafischen Oberfläche im Hintergrund abläuft. Und es kann sein, dass Sie mal aus der Ferne Zugriff auf einen Server benötigen und dazu nur ein Terminal zur Verfügung haben.

### 1. Starten

Starten Sie die virtuelle Maschine bmLP1.

### 2. Anmelden

Melden Sie sich mit dem Benutzer *vmadmin* und dem Passwort *sml12345* an.

### 3. Ubuntu-Schreibtisch

Sie landen auf dem Ubuntu-Schreibtisch (Desktop), die für uns wichtigsten Verknüpfungen sind auf der Kontrollleiste bereits vorhanden.

### 4. Dateimanager «Dateien»



- Starten Sie «Dateien».
- Sie befinden sich in «Persönlicher Ordner».
- Erstellen Sie das Verzeichnis «Modul105».

### 5. Terminal



Starten Sie die Konsole («Terminal»). Die wichtigsten Linux-Befehle entnehmen Sie dem Dokument *Linux-auf-einem-Blatt.pdf*.

Die für uns wichtigsten Befehle werden im nächsten Kapitel aufgeführt und erklärt.

### 6. Rechner durchsuchen



Den Rechner können Sie über dieses Symbol durchsuchen. So finden Sie sowohl Dateien als auch Anwendungen.



## Terminal

Die Befehle in der folgenden Tabelle sind für uns wichtig. Öffnen Sie das Terminal, führen Sie jeweils den Befehl gemäss Beispiel durch und notieren Sie die Bedeutung des Befehls!

Befehl	Beispiel	Bedeutung
cd	cd Modul105	
	cd ..	
	cd ~	
pwd	pwd	
ls	ls	
	ls -al	
	ls --help	
man	man cd	
	man ls	
	man man	
Stellen Sie sicher, dass Sie sich in Ihrem Home-Verzeichnis befinden.		
mkdir	mkdir test	
rmdir	rmdir test	
Kopieren Sie die Datei <i>AB105-01.sql</i> vom Moodle ins Verzeichnis <i>Modul105</i> (Drag & Drop). Gehen Sie im Terminal ins Verzeichnis <i>Modul105</i> .		
cp	cp AB105-01.sql 01.tmp	
mv	mv 01.tmp 02.tmp	
find	Find	
	find ../	
Gehen Sie mit <code>cd ..</code> oder <code>cd ~</code> in Ihr Home-Verzeichnis.		
	find -name 008.tmp	
	find -name *.tmp	
	find -name *.txt	
rm	rm Modul105/02.tmp	
exit	exit	

Im Terminal ist es schwieriger, sich zurechtzufinden, als mit einer grafischen Oberfläche.

**Deshalb ist es wichtig, dass Sie immer wissen, wo sie sich befinden!**

Beispiel: Wenn Sie SQLite starten und eine Datei ausführen oder importieren wollen, dann muss sich diese Datei im aktuellen Verzeichnis befinden. Oder Sie müssen zum Dateinamen auch den Pfad angeben – in diesem Fall müssen Sie genau wissen, wie dieser Pfad lautet.

## Crash-Kurs SQLite und Sqliteman

Eine kurze Übersicht, was SQLite ist, haben Sie am Anfang dieses Dokumentes erhalten. SQLite selber wird über die Konsole bedient und bietet keine grafische Benutzerschnittstelle an. Es gibt jedoch Dritthersteller, die SQLite eine grafische Oberfläche "aufgepfropft" haben. Ein Beispiel einer solchen Anwendung ist *Sqliteman*, die wir in diesem Modul verwenden werden.

Wie bei der Dateienverwaltung werden wir SQLite sowohl mit der Konsole als auch mit der grafischen Oberfläche benutzen. Und auch hier gilt: Die Konsole ist für das Verständnis wichtig, was wie funktioniert und was bei der grafischen Oberfläche im Hintergrund abläuft. Vielleicht sind Sie mal gezwungen, SQLite via Konsole zu bedienen. Dann werden Sie froh sein, zumindest Grundkenntnisse der SQLite-Befehle zu haben.

Wir erstellen nun eine erste Datenbank mit der Konsole:

1. Vergewissern Sie sich, dass Sie im Verzeichnis *Modul105* sind.
2. Starten Sie SQLite:  
`vmadmin@bmLP1:~/Modul105$ sqlite3 training.sqlite`
3. Überprüfen Sie die aktuelle Datenbank mit:  
`sqlite> .databases`
4. Führen Sie die SQL-Befehle aus, die sich in der Datei *AB105-01.sql* befinden. Dabei werden zwei Tabellen angelegt und mit Daten gefüllt:  
`sqlite> .read AB105-01.sql`
5. Führen Sie folgenden SQL-Befehl aus:  
`sqlite> SELECT * FROM ORT;`

Nachfolgend die für uns wichtigsten SQLite-Befehle. Tragen Sie die Bedeutung ein!

Befehl	Beispiel	Bedeutung
<code>.help</code>	<code>.help</code>	
<code>.databases</code>	<code>.databases</code>	
<code>.tables</code>	<code>.tables</code>	
<code>.schema</code>	<code>.schema ORT</code>	
<code>.fullschema</code>	<code>.fullschema</code>	
<code>.read</code>	<code>.read AB105-01.sql</code>	
<code>.import</code>	(diesen Befehl benutzen wir später im Modul)	
<code>.quit</code>	<code>.quit</code> oder CTRL+d	
<code>.exit</code>	<code>.exit</code>	
<code>SELECT</code>	<code>SELECT * FROM ORT;</code>	

### Structured Query Language SQL

SELECT ist ein SQL-Befehl. SQLite unterstützt einen Grossteil der im SQL-92-Standard festgelegten SQL-Sprachbefehle. Ein paar Befehle sind nicht vorhanden, da SQLite möglichst klein, schlank und portabel bleiben will, was uns aber nicht weiter kümmert.

Sämtliche von SQLite unterstützten SQL-Befehle können sowohl in der Konsole als auch via *Sqliteman* eingegeben werden. Es sind die identischen Befehle, d.h. das Wissen, das Sie jeweils erwerben, ist Ihnen in beiden SQLite-Benutzeroberflächen vom selben Nutzen. Ganz zu schweigen davon, dass Sie dieselben SQL-Befehle auch in späteren Modulen und anderen Datenbanken wiederverwenden können!

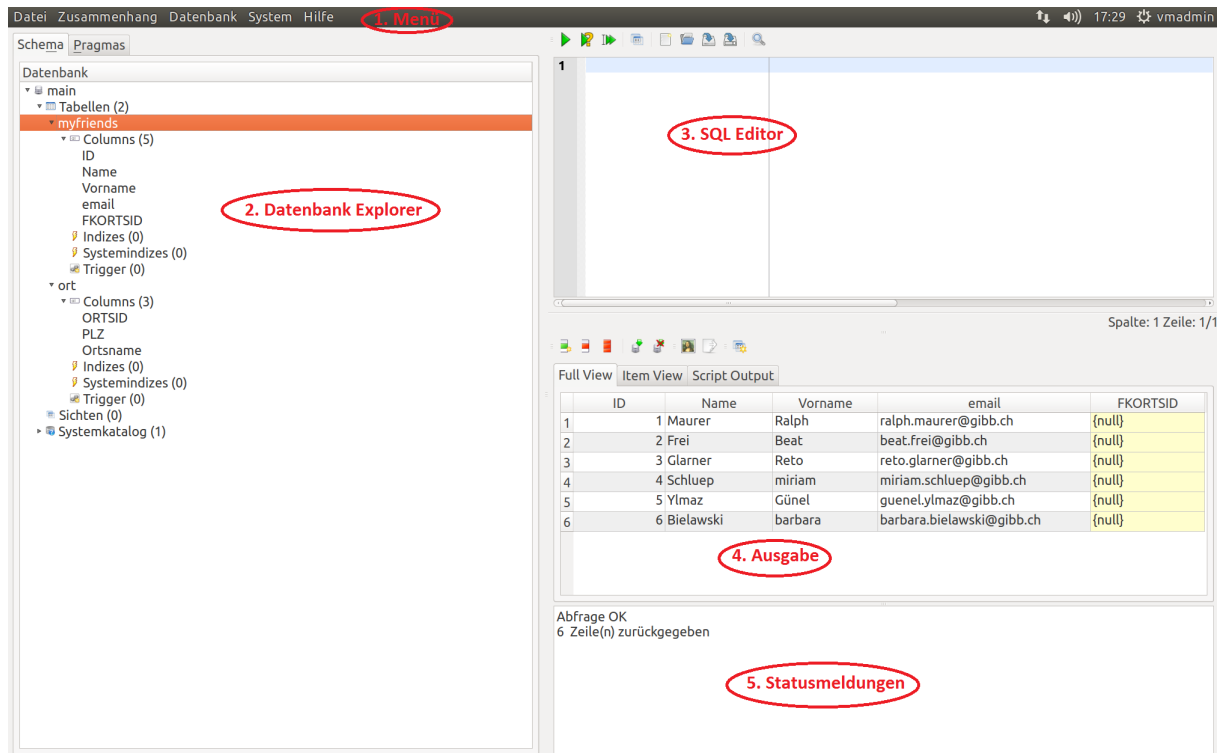
**Eine Einführung in SQL erhalten Sie im nächsten Arbeitsblatt105-02!**

## Sqliteman



Starten Sie Sqliteman über dieses Symbol. Um in der Anwendung Zugriff auf das Menü zu erhalten, fahren Sie mit der Maus oben über den Fenstertitel *Sqliteman*.

### Applikationsfenster



1. Menü: Wenn Sie mit der Maus über den Titelbalken fahren, erscheint das Menü
2. Datenbank-Explorer: Zeigt alle Datenbankobjekte der aktuellen Datenbank. Beim Klick auf die Pfeile werden die jeweiligen Objekte, Attribute bzw. Eigenschaften angezeigt.
3. SQL-Editor: Hier werden die SQL-Befehle eingegeben. Die Befehle sind identisch mit denjenigen, die Sie über die Konsole eingeben.
4. Ausgabe: Zeigt das Resultat des SQL-Befehls an (Editor) bzw. das Resultat der Aktion im Datenbank-Explorer.
5. Statusmeldungen: Zeigt Fehler- und Erfolgsmeldungen an.

Führen Sie nun folgende Aktionen durch:

1. Öffnen Sie die vorhin erstellte Datenbank *training.sqlite* (Menü *Datei* – *Öffnen...*).
2. Klicken Sie auf den Pfeil neben *Tabellen*.
3. Doppelklicken Sie auf die Tabelle *myfriends*. Alle Datensätze der Tabelle werden ausgegeben.
4. Klicken Sie auf alle Pfeile im Datenbank-Explorer, bis alle Objekte und Attribute sichtbar sind. Nun sollte *Sqliteman* wie oben aussehen.

### Dateitypen

In SQLite arbeiten wir vorläufig mit zwei Dateitypen, und **es ist wichtig, dass Sie den Unterschied kennen:**

- Die Datenbank selber mit allen Definitionen und Daten ist eine binäre Datei, die Sie nur über die Konsole oder einem Tool wie *Sqliteman* betrachten und bearbeiten können. Die Dateierweiterung ist normalerweise *sqlite* oder *db*, kann jedoch auch davon abweichen. *Sqliteman* fragt

nie, ob Sie diese Datei speichern wollen – alles, was Sie in der Datenbank machen, wird automatisch gespeichert oder Sie werden direkt bei einer Aktion aufgefordert, auf *Verändern* zu klicken (z.B. beim Anpassen einer Tabelle).

- SQL-Befehle werden in Textdateien gespeichert (SQL-Dateien). Die Endung ist normalerweise *sql*, jedoch kann sie auch anders sein. Diese Dateien können in der Konsole ausgegeben oder mit jedem Texteditor betrachtet und bearbeitet werden. Im *Sqliteman* werden sie in den SQL-Editor geladen und dort auch wieder gespeichert.

### Bedienung von Sqliteman

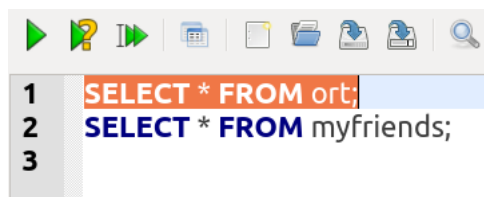
#### Starten, Datenbank öffnen und Beenden

- Wenn Sie Sqliteman starten, werden automatisch die letzte Datenbank und SQL-Datei geöffnet. Die Datenbank sehen Sie im Datenbank-Explorer, die SQL-Datei im SQL-Editor.
- Eine andere Datenbank öffnen Sie über *Datei – Letzte Dateien* oder *Datei – Öffnen*.
- Schliessen Sie Sqliteman über *Datei – Beenden*. Falls nicht bereits erledigt, werden Sie gefragt, ob Sie die Änderungen im SQL-Editor speichern wollen. **Diese Frage betrifft immer die SQL-Befehle im SQL-Editor und nie die Datenbank!**

#### Datenbank-Explorer

Den Datenbank-Explorer erkunden Sie am besten selber. Sie können Tabellen erstellen, anpassen, Indices erstellen oder eine Aktion auslösen, die eine Ausgabe erzeugt. Ein paar Übungen dazu erfolgen weiter unten.

#### SQL-Editor



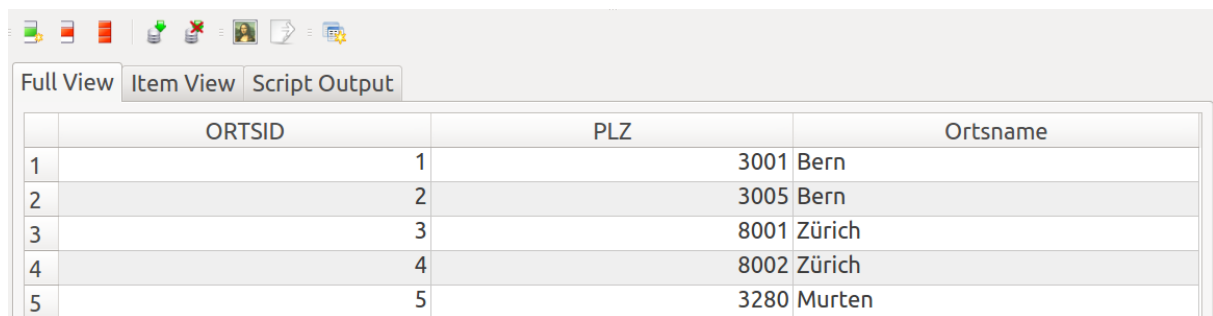
Die gespeicherten SQL-Befehle der letzten Sitzung werden beim Start von *Sqliteman* automatisch angezeigt.

Symbol	Aktion	Bedingung, Bemerkungen
	SQL-Befehl ausführen	Der SQL-Befehl muss markiert sein, nur 1 Befehl möglich.
	Mehrere SQL-Befehle ausführen	Ausführen von allen SQL-Befehlen ab Cursorposition bis Ende.
	Editorfenster leeren	Mit leerem Fenster beginnen bzw. weiterfahren
	Datei öffnen	Öffnen und anzeigen einer SQL-Datei
	Eingaben speichern	Alle Eingaben im SQL-Editor werden gespeichert. Beim 1. Speichern müssen Pfad und Dateiname angegeben werden.
	Eingaben speichern unter	Alle Eingaben im SQL-Editor werden in eine neue Datei gespeichert. Pfad und Dateiname müssen angegeben werden.

**Hinweis:** Falls sich Daten im SQL-Editor befinden, die noch nicht gespeichert worden sind, erfolgt eine entsprechende Frage bei den Aktionen 3-6.



**Achtung:** Wenn Sie beim Speichern aufgefordert werden, Pfad und Dateiname anzugeben, dann wählen Sie immer eine neue Datei im Format *name.sql*. **Speichern Sie die Daten niemals in eine bestehende Datei *name.sqlite* oder *name.db*!** Denn sonst überschreiben Sie eine bestehende (möglicherweise sogar die aktuelle) Datenbank mit SQL-Befehlen!

## Ausgabe



	ORTSID	PLZ	Ortsname
1	1	3001	Bern
2	2	3005	Bern
3	3	8001	Zürich
4	4	8002	Zürich
5	5	3280	Murten

Die Symbole sind nur aktiviert, wenn Sie im Datenbank-Explorer eine Tabelle doppelklicken und alle Datensätze angezeigt werden.

Symbol, Tab	Aktion	Bedingung, Bemerkungen
	Fügt einen neuen Datensatz hinzu	Der Datensatz wird hinten in der Tabelle angehängt. Klicken Sie auf die Felder, um die Attributwerte einzutragen.
	Löscht den markierten Datensatz	Ein Datensatz muss ausgewählt sein, sonst passiert nichts.
	Löscht alle Datensätze der Tabelle	Sicherheitsfrage, ob wirklich alle Datensätze gelöscht werden sollen.
	Transaktion in die Datenbank schreiben	Das Einfügen bzw. Löschen muss mit Klicken auf dieses Symbol bestätigt werden. Erst danach werden die Änderungen in die Datenbank geschrieben.
	Transaktion verwerfen	Das Einfügen bzw. Löschen wird rückgängig gemacht bzw. nicht ausgeführt.
Full View	Anzeige des gesamten Ergebnisses	Das gesamte Ergebnis wird in diesem Reiter angezeigt und kann bearbeitet werden.
Item View	Anzeige eines Datensatzes	Ein einzelner Datensatz wird angezeigt und kann bearbeitet werden. Mit den Pfeilen navigieren Sie zwischen den Datensätzen.
Script Output	Anzeige des Batch-Status	Wenn mehrere Befehle auf einmal ausgeführt werden, wird automatisch dieser Tab mit dem Status der Batch-Verarbeitung angezeigt.

## Übungen mit dem Sqlliteman

1. Starten Sie mit geschlossenem *Sqlliteman*.
  2. Gehen Sie im Dateimanager *Dateien* ins Verzeichnis *Modul105*.
  3. Doppelklicken Sie auf die Datei *training.sqlite*. Was passiert?
- 
4. Erstellen Sie im Datenbank-Explorer die Tabelle *company* mit folgenden Attributen:
    - COMPANYID INTEGER PRIMARY KEY
    - Firmenname TEXT
    - Email TEXT



5. Markieren Sie die Tabelle im Datenbank-Explorer und führen Sie den Befehl *Tabelle beschreiben* aus (via Menü oder rechte Maustaste). Was wird in der Ausgabe angezeigt?

---

---

---

---

6. Fügen Sie in der Ausgabe folgende Datensätze ein:

- 1, Swisscom AG, info@swisscom.ch
- 2, Post CH AG, post@post.ch
- 3, Bedag, admin@bedag.ch
- 4, RUAG, info@ruag.ch
- 5, Band-Genossenschaft, band@genossenschaft.ch

7. Fügen Sie 5 weitere Datensätze hinzu.  
8. Zeigen Sie alle Datensätze an, indem Sie im Datenbank-Explorer auf den Tabellennamen doppelklicken.  
9. Geben Sie im SQL-Editor ein: `SELECT * FROM company`  
Was ist der Unterschied zum Ergebnis in Aufgabe 8? Gibt es überhaupt einen Unterschied?

---

---

10. Geben Sie im SQL-Editor auf einer neuen Zeile ein:

```
SELECT * FROM myfriends  
SELECT * FROM ort
```

11. Führen Sie jetzt im SQL-Editor den Befehl "SELECT \* FROM myfriends" aus. Wie gehen Sie vor?

---

---

12. Speichern Sie die Befehle im SQL-Editor in eine Datei namens *AB105-07SELECT.sql* im Verzeichnis *Modul105* ab.

13. Leeren Sie den SQL-Editor.

14. Geben Sie im SQL-Editor folgenden Befehl ein und führen ihn aus:

```
INSERT INTO company VALUES (11, "BiCT AG", "info@bict.ch")
```

Was ist das für ein Befehl und was passiert in der Datenbank?

---

---

15. Speichern Sie den Befehl in eine Datei namens *AB105-07INSERT.sql* im Verzeichnis *Modul105* ab.

16. Öffnen Sie die vorher gespeicherte Datei *AB105-07SELECT.sql*.  
Was wird im SQL-Editor angezeigt?

---

---

## Zusatzaufgaben

1. Erstellen Sie die neue Tabelle *firma*. Diese Tabelle soll genau gleich aussehen wie die Tabelle *company* (identische Tabellendefinition). Auch der Inhalt der Tabelle (total 11 Datensätze) soll identisch sein. Diese Aufgabe soll vollständig mit SQL-Statements gelöst werden, entweder im SQL-Editor des Sqliteman oder via Konsole!  
Notieren Sie alle notwendigen SQL-Befehle.
2. Erkunden Sie den *Sqliteman*, indem Sie möglichst viele Funktionen ausprobieren!