SQL Handbuch für Einsteiger

Der leichte Weg zum SQL-Experten

Paul Fuchs

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie;
detaillierte bibliografische Informationen sind im Internet über http://dnb.d-nb.de abrufbar.

www.bmu-verlag.de
fuchs@bmu-verlag.de

Lektorat: Matthias Kaiser
Einbandgestaltung: Pro ebookcovers Angie
Druck und Bindung: Wydawnictwo Poligraf sp. zo.o. (Polen)

Taschenbuch-ISBN: 978-3-96645-070-6
Hardcover-ISBN: 978-3-96645-071-3
E-Book-ISBN: 978-3-96645-069-0

SQL Handbuch für Einsteiger

Inhaltsverzeichnis

Downloadhinweis

Alle Programmcodes aus diesem Buch sind als PDF zum Download verfügbar. Dadurch müssen Sie sie nicht abtippen:
https://bmu-verlag.de/sql

Außerdem erhalten Sie die eBook Ausgabe zum Buch im PDF Format kostenlos auf unserer Website:

https://bmu-verlag.de/sql
Downloadcode: siehe Kapitel 21

Kapitel 1
Einführung

Die Erhebung und Auswertung von Daten gewinnt im Wirtschaftsleben immer mehr an Bedeutung. Einige der weltweit größten Unternehmen – beispielsweise Google oder Facebook – verwenden ein Geschäftsmodell, das im Wesentlichen darauf basiert, die Daten der Nutzer auszuwerten und gewinnbringend einzusetzen. Schlagworte wie Big Data stehen für eine neue technische Entwicklung, die darauf abzielt, viele Millionen Werte aufzunehmen und auszuwerten.

Hinzu kommen die klassischen Anwendungen wie Mitarbeiterverzeichnisse oder Kundendateien, die wahrscheinlich in jedem Unternehmen anzutreffen sind. Während diese Informationen früher in Papierform festgehalten wurden, kommen hierfür mittlerweile wohl in fast allen Betrieben elektronische Speichermedien zum Einsatz.

Sowohl bei großen Datenmengen mit vielen Millionen Einträgen als auch bei einer einfachen Kunden- oder Mitarbeiterliste ist es wichtig, auf eine sichere und effiziente Speicherung der Daten zu achten. Ein Verlust der Informationen könnte häufig hohe finanzielle Schäden nach sich ziehen. Es kommt hinzu, dass oftmals auch datenschutzrechtliche Aspekte relevant sind. Ein Unternehmen muss personenbezogene Inhalte beispielsweise vor einem unberechtigten Zugriff schützen. Da immer mehr Daten gespeichert werden, sind auch ein effizienter Zugriff und ein gut strukturierter Aufbau sehr wichtig. Ist dies nicht gegeben, ist viel Arbeitszeit für die Verwaltung notwendig.

In vielen Fällen bietet es sich an, hierfür eine Datenbank zu verwenden. Diese ermöglicht eine sichere Aufbewahrung und einen schnellen Zugriff. Sie zeichnen sich selbst bei großen Datenmengen durch eine hohe Effizienz aus. Außerdem eignet sie sich sehr gut für die gleichzeitige Benutzung durch mehrere Anwender. SQL ist ein praktisches Werkzeug, um Datenbanken zu erstellen und zu verwalten.

1.1 SQL – was ist das eigentlich?

Dieses Buch befasst sich mit SQL. Daher steht an erster Stelle erst einmal die Frage, was SQL eigentlich ist. Hierbei handelt es sich um eine Datenbanksprache. Diese dient der Kommunikation mit einer Datenbank. Dabei ist es wichtig, darauf zu achten, dass SQL selbst jedoch keine Datenbank ist. Hierbei handelt es sich um ein eigenes System. SQL stellt die Verbindung zwischen der Datenbank und dem Anwender dar.

SQL ist auch keine Programmiersprache. Der Befehlsumfang ist hierbei deutlich geringer. Damit ist es nicht möglich, ein Computerprogramm mit einem eigenständigen Ablauf zu erstellen. Vielmehr handelt es sich dabei um einzelne Befehle, die die Datenbank dazu bringen, eine bestimmte Aktion durchzuführen.

SQL kann dabei vielfältige Aufgaben übernehmen. Die Sprache ermöglicht es, die grundlegenden Strukturen einer Datenbank zu erstellen. Damit lässt sich die Datenbank selbst erzeugen und es ist möglich, Tabellen mit unterschiedliche Spalten für die Aufnahme bestimmter Werte zu gestalten. Darüber hinaus lassen sich mit SQL konkrete Werte in diese Strukturen einfügen.

Des Weiteren dient SQL der Abfrage der Werte. Damit ist es möglich, einen einzelnen Wert abzurufen oder Gruppen von Werten gemeinsam abzufragen. Darüber hinaus bietet die Datenbanksprache vielfältige Möglichkeiten, um die Ergebnisse zu sortieren und zu strukturieren.

SQL bietet verschiedene Anwendungsmöglichkeiten. Die meisten Datenbanken verfügen über eine Benutzeroberfläche, über die sich die SQL-Befehle direkt eintragen lassen. Das erlaubt eine manuelle Verwaltung der Daten. Sehr häufig kommt die Sprache jedoch auch in Verbindung mit einem Computerprogramm zum Einsatz. Fast alle Programmiersprachen erlauben es, SQL-Befehle zu integrieren und an eine Datenbank zu übermitteln. Auf diese Weise ist es möglich die Daten automatisch verwalten zu lassen. Das ist besonders effizient. Die Einbindung von SQL in ein Computerprogramm wird in diesem Buch zwar nicht behandelt. Gute Kenntnisse in dieser Datenbanksprache führen

jedoch dazu, dass es mit grundlegenden Programmierfähigkeiten recht einfach ist, entsprechende Programme zu gestalten.

Um SQL zu verwenden, ist es erforderlich, dass die Datenbank diese Sprache unterstützt. Das ist zwar bei den meisten gängigen Datenbanken der Fall, da es sich hierbei um die mit großem Abstand am häufigsten verwendete Datenbanksprache handelt. Allerdings gibt es auch einige Systeme, bei denen die Kommunikation über SQL nicht möglich ist. Um diese Datenbanksprache anzuwenden, ist es daher immer notwendig, ein System auszuwählen, das sie unterstützt.

1.2 Die Definition einer Datenbank

Bereits in den ersten Abschnitten dieses Buchs fiel mehrmals der Begriff *„Datenbank"*. Das zeigt bereits, welche Bedeutung die Datenbank für die Arbeit mit SQL hat. Der ausschließliche Zweck dieser Sprache besteht darin, eine Verbindung zu einer Datenbank zu ermöglichen. Daher ist es sehr wichtig, zunächst einmal zu vermitteln, was eine Datenbank genau ist.

Hierbei handelt es sich um ein System für die elektronische Datenverwaltung. Die Aufgabe der Datenbank besteht darin, die Informationen dauerhaft, sicher, effizient und korrekt abzuspeichern.

Eine Datenbank ist immer in zwei Ebenen aufgebaut. Zum einen gibt es die physische Ebene des Datenspeichers. Hier sind die Daten in elektronischer Form abgespeichert. Im Gegensatz zu gewöhnlichen Dateien gibt es auf dieser Ebene jedoch häufig kein klares Ordnungssystem. Die Reihenfolge muss dabei nicht den Strukturen entsprechen, die wir für unsere Tabellen verwendet haben.

Die zweite Ebene ist das Datenbankmanagementsystem (DBMS). Dieses ist für die Verwaltung des physischen Datenspeichers verantwortlich. Wenn man einen Wert in eine Datenbank einträgt, wählt das DBMS einen passenden Speicherort dafür aus. Bei der Abfrage einer Information muss es diese an der richtigen Stelle abrufen und an den Anwen-

der übermitteln. Die Organisationsformen können dabei je nach verwendeter Software ganz unterschiedlich sein.

Darüber hinaus stellt das DBMS die Schnittstelle zwischen den Daten und dem Anwender dar. Es ist dazu in der Lage, Befehle entgegenzunehmen und die zugehörigen Aktionen durchzuführen. Hierfür kommt wie bereits ausgeführt eine Datenbanksprache zum Einsatz – in den meisten Fällen SQL. Das DBMS bietet normalerweise die Möglichkeit, die Befehle direkt einzugeben. Die Software stellt hierfür ein geeignetes Eingabefenster bereit. Hinzu kommt eine Schnittstelle, die die Anbindung eines Computerprogramms erlaubt. Schließlich bieten die meisten Angebote auch noch eine grafische Benutzeroberfläche an. Diese ermöglicht es, die Datenbank ohne entsprechende SQL-Kenntnisse zu verwalten.

Eine weitere wichtige Aufgabe dieser Software besteht bei den meisten Systemen auch darin, den Zugriff auf die Daten zu kontrollieren. Das ist für die Sicherheit sehr wichtig – insbesondere wenn externe Nutzer mit der Datenbank interagieren können. In diesem Fall muss das DBMS den Nutzernamen und das Passwort abfragen, bevor es den Zugriff gestattet. Dabei erlaubt es das System, die Rechte detailliert vorzugeben. Es ist beispielsweise möglich, einem Anwender nur Rechte für die Nutzung eines bestimmten Bereichs zu gewähren. Darüber hinaus kann der Administrator bei jedem Nutzer vorgeben, ob nur ein lesender oder auch ein schreibender Zugriff gestattet ist. Das erlaubt es, detaillierte Vorgaben zu machen. In einem Unternehmen ist es auf diese Weise beispielsweise möglich, die Zugriffsrechte genau an den Tätigkeiten der Mitarbeiter auszurichten. Auf diese Weise lässt sich ein reibungsloser Ablauf der Arbeitsprozesse mit einem guten Schutz der Daten verbinden.

Um eine Datenbank zu nutzen, ist daher immer ein passendes Datenbanksystem erforderlich. Dabei handelt es sich um eine Software, die das DBMS enthält. Hierfür stehen vielfältige Angebote zur Auswahl. Diese unterscheiden sich nicht nur hinsichtlich der Interaktionsmöglichkeiten mit dem Anwender und den Maßnahmen für die Sicherheit der Daten. Darüber hinaus können sie unterschiedliche Datenbankmodelle anwenden. Dabei handelt es sich um die grundlegenden Ord-

nungsstrukturen für die in der Datenbank enthaltenen Informationen. Eine ausführliche Erklärung zu diesem Thema folgt noch später in diesem Buch. Doch wird daran bereits deutlich, dass die Auswahl des Datenbanksystems große Auswirkungen auf die Verwaltung der Informationen hat.

1.3 Die Ursprünge von SQL

Die Geschichte von SQL beginnt bereits in den 70er Jahren. Damals begannen Donald D. Chamberlin und Raymond F. Boyce damit, eine Sprache für die Verwaltung einer Datenbank zu entwickeln. Beide hatten zuvor bereits mit dem britischen Informatiker Edgar F. Codd zusammengearbeitet, der bereits seit den 60er Jahren das *relationale Datenbankmodell* entwarf und die theoretischen Grundlagen dafür erforschte. Dieses Strukturmodell war zu diesem Zeitpunkt jedoch noch neu und es gab keine praktischen Anwendungen dafür. Chamberlin und Boyce lernten diese Idee jedoch durch die Arbeit mit Codd kennen und beschlossen, sie in die Praxis umzusetzen.

Beide Informatiker waren zu dieser Zeit bei IBM beschäftigt. Dieser Konzern hatte bereits zuvor ein Datenbankmanagementsystem mit der Bezeichnung System R entworfen. Dabei handelte es sich zwar genau genommen noch nicht um ein relationales Datenbanksystem. Dennoch setzte dieses bereits zahlreiche Grundsätze dieser Idee um. Chamberlin und Boyce machten sich daran, eine Sprache zu entwickeln, mit der es möglich ist, Daten aus diesem System abzurufen, sie zu speichern oder sie zu verändern.

Diese Datenbanksprache erhielt zunächst die Bezeichnung SEQUEL – als Abkürzung für Structured English Query Language. Als sie jedoch an Bekanntheit gewann, kam es schon bald zu einer Auseinandersetzung über die Namensrechte. Der Grund dafür bestand darin, dass bereits lange zuvor ein anderes Unternehmen eine Marke mit dieser Bezeichnung registrieren lassen hatte. Aus diesem Grund mussten die Forscher ihre Datenbanksprache umbenennen. Sie wählten daraufhin die Bezeichnung SQL. Aufgrund der Ursprünge ist jedoch insbesondere

im englischsprachigen Raum bis heute die Aussprache SEQUEL üblich. Viele Informatiker geben als Auflösung dieser Abkürzung den Begriff Structured Query Language an. Allerdings ist im offiziellen SQL-Standard angegeben, dass es sich hierbei nicht um eine Abkürzung, sondern um eine eigenständige Bezeichnung ohne weitere Bedeutung handelt.

Nachdem Chamberlin und Boyce die Sprache entwickelt hatten, führte IBM umfangreiche Tests mit ihr durch. Dabei stellte sich heraus, dass sie eine sehr effiziente Kommunikation mit einer Datenbank ermöglichte. Deshalb begann das Unternehmen damit, kommerzielle Produkte zu entwickeln, die SQL verwendeten. Das erste von ihnen trug den Namen System/38 und erschien 1979. Dabei handelte es sich um ein Datenbanksystem, das in erster Linie in großen Unternehmen und Forschungseinrichtungen zum Einsatz kam. 1981 und 1983 folgten mit SQL/DS und DB2 zwei weitere kommerzielle Produkte, die SQL verwendeten.

Das relationale Datenbankmodell hatte in der Zwischenzeit jedoch auch andere Entwickler auf sich aufmerksam gemacht. Das Unternehmen Relational Software, Inc. – das mittlerweile in Oracle umbenannt wurde und zu den wichtigsten Anbietern für Datenbanksysteme gehört – begann bereits Ende der 70er Jahre damit, ein eigenes DBMS zu entwickeln, das auf SQL basierte. Dabei überholte es sogar die eigentlichen Entwickler und präsentierte 1979 – noch vor dem Erscheinen von System/38 – mit Oracle V2 das erste kommerzielle Datenbanksystem, das auf SQL beruhte.

Das relationale Datenbankmodell erwies sich schnell als eine besonders effiziente Möglichkeit für die Verwaltung von Daten. Daher entwickelte es sich bereits im Laufe der 80er Jahre zum am häufigsten verwendeten Datenbankmodell weltweit. Da SQL die wesentliche Sprache für dieses Modell ist, stieg auch hierbei die Nutzung stark an. 1986 wurde schließlich der erste offizielle SQL-Standard verabschiedet. Bis heute ist SQL die Datenbanksprache, die mit großem Abstand am häufigsten genutzt wird.

Kapitel 2

Theoretische Hinführung: das relationale Datenbankmodell

Bereits im Einleitungs-Kapitel wurde mehrfach erwähnt, dass SQL eine Sprache für relationale Datenbanken ist und dass ihre Entwicklung eng mit diesem Modell verbunden ist. Doch wurde dabei noch nicht erklärt, was ein Datenbankmodell ist und wodurch sich relationale Datenbanken auszeichnen. Das ist jedoch sehr wichtig, um die Funktionsweise von SQL zu verstehen.

Bevor wir uns der praktischen Anwendung von SQL widmen können, ist es daher wichtig, sich kurz mit den Grundzügen dieses Ordnungssystems auseinanderzusetzen. Um möglichst bald mit der Arbeit mit Datenbanken beginnen zu können, werden diese Ausführungen jedoch so kurz wie möglich gehalten.

2.1 Was ist ein Datenbankmodell?

Das *Datenbankmodell* beschreibt die Struktur, in der Daten innerhalb einer Datenbank abgespeichert werden. Dieses Ordnungsprinzip wird stets durch das verwendete Datenbankmanagementsystem vorgegeben. Für jedes einzelne Datenbankmodell ist eine eigene Software notwendig, die die entsprechenden Strukturen umsetzt.

Das Datenbankmodell hat auch einen großen Einfluss darauf, auf welche Weise wir auf die Daten zugreifen können. Da für den Zugriff eine Datenbanksprache zum Einsatz kommt, muss diese am verwendeten Datenbankmodell ausgerichtet sein. SQL ermöglicht beispielsweise die Verwaltung relationaler Datenbanken. Dabei handelt es sich heutzutage um das mit großem Abstand am häufigsten genutzte Datenbankmodell. Einer der entscheidenden Vorteile dieses Systems besteht darin, dass es recht einfach verständlich ist. Außerdem bietet es sehr

effiziente Möglichkeiten für die Abfrage der Werte. Es gibt jedoch noch einige weitere Möglichkeiten.

Das älteste Datenbankmodell ist das hierarchische System. Hierbei gibt es ein Ausgangs-Element, das bestimmte Kind-Elemente enthält. Auf diese Weise entsteht eine hierarchische Beziehung. Weitere Möglichkeiten stellen das objektrelationale, das objektorientierte und das dokumentenorientierte Datenbankmodell dar. Außerdem gibt es Netzwerkdatenbanken.

Diese kurze Ausführung zeigt, dass es neben dem relationalen Datenbankmodell noch viele weitere Möglichkeiten gibt. Um SQL zu verstehen, ist es daher notwendig, sich mit den Strukturen der verschiedenen Alternativen zu beschäftigen. Daher befasst sich der folgende Abschnitt damit, wie diese verschiedenen Datenbanken aufgebaut sind. Zum Abschluss des Kapitels wird dann noch das relationale Datenbankmodell – das die Grundlage für SQL darstellt – etwas detaillierter erklärt.

2.2 Die grundlegenden Eigenschaften verschiedener Datenbankmodelle

Bevor wir uns dem relationalen Datenbankmodell zuwenden, sollen in diesem Abschnitt die übrigen Alternativen zu diesem System kurz präsentiert werden. Das macht es später einfacher, die Unterschiede zu erkennen und die Eigenheiten relationaler Datenbanken zu verstehen.

Die ersten Datenbanken, die in der Informatik zum Einsatz kamen, verwendeten das *hierarchische Datenbankmodell*. Dieses verwendet eine Baumstruktur, über die die einzelnen Informationen erreichbar sind. Diese ist in Abbildung 2.1 zu erkennen. Hierbei gibt es immer genau ein Ausgangselement, das man sich als die Wurzel des Baums vorstellen kann. Jede Datenbankabfrage beginnt an dieser Stelle.

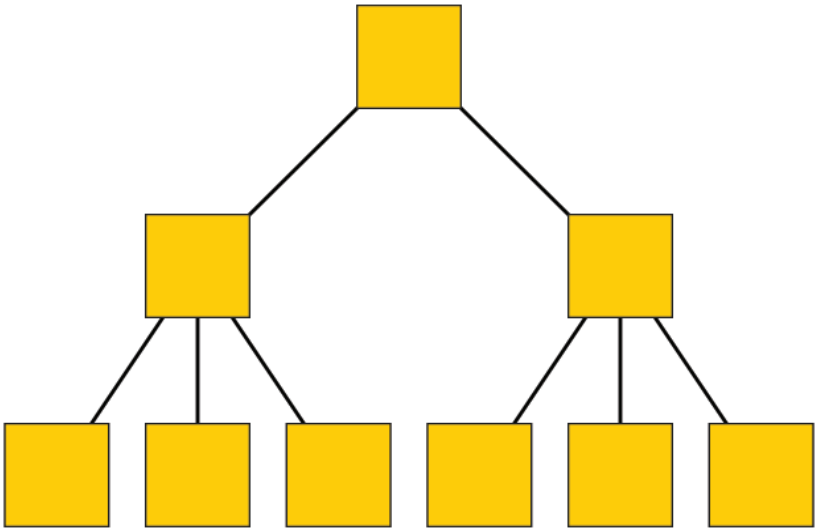

Abb. 2.1 Das hierarchische Datenbankmodell

Das Ausgangselement kann mehrere Kind-Elemente aufweisen, um weitere Daten zu speichern. Diese Kind-Elemente können dann wiederum weitere Kinder enthalten. Auf diese Weise entsteht eine fein verzweigte Struktur. Bei manchen Datenbanken dürfen die Elemente nur maximal zwei Kinder haben. Meistens ist die Anzahl der Kind-Elemente jedoch nicht begrenzt. Dabei hat jedoch jedes Element genau ein Eltern-Element. Das ist eine der wesentlichen Eigenschaften einer hierarchischen Datenbank. Die einzige Ausnahme stellt hierbei das Ausgangs-Element dar, das kein Eltern-Element aufweist.

Wenn man nun einen bestimmten Eintrag in der Datenbank abrufen will, ist es notwendig, sich über diese Baumstruktur zu den verschiedenen Ebenen vorzuarbeiten. Jedes Element verfügt über die Information, welche Kinder hierbei vorhanden sind. Auf diese Weise ist es möglich, genau die richtige Stelle zu erreichen.

Ein Problem dieses Modells besteht darin, dass sich damit nur die Beziehungen zwischen Eltern- und Kind-Elementen abbilden lassen. Es ist nicht möglich, Beziehungen über mehrere Ebenen hinweg zu definieren. Das stellt eine erhebliche Einschränkung bei der Anwendung dar und ist einer der wesentlichen Gründe dafür, dass dieses Datenbankmodell mittlerweile nur noch selten zum Einsatz kommt.

Eine weitere Möglichkeit stellt das *Netzwerkdatenbankmodell* dar. Dessen Struktur ist in Abbildung 2.2 zu sehen. Es entstand etwa zeitgleich zum relationalen Datenbankmodell. Es basiert auf dem hierarchischen Datenbankmodell und hat das Ziel, dessen wesentlichen Nachteil – die Einschränkung der Beziehungen zwischen den verschiedenen Elementen – auszugleichen. Hierbei gibt es ebenfalls Abhängigkeiten zwischen den Elementen. Um eines von ihnen zu erreichen, ist es notwendig, sich über die anderen Knotenpunkte zu ihm vorzuarbeiten. Dabei herrschen jedoch keine hierarchischen Strukturen. Es ist möglich, dass ein Element mehrere übergeordnete Knotenpunkte aufweist. Auf diese Weise ist es möglich, sehr vielfältige Beziehungen zwischen den einzelnen Einträgen zu definieren. Das hat jedoch zur Folge, dass es hierbei keinen klar definierten Suchweg gibt. Das kann die Effizienz dieses Systems beeinträchtigen.

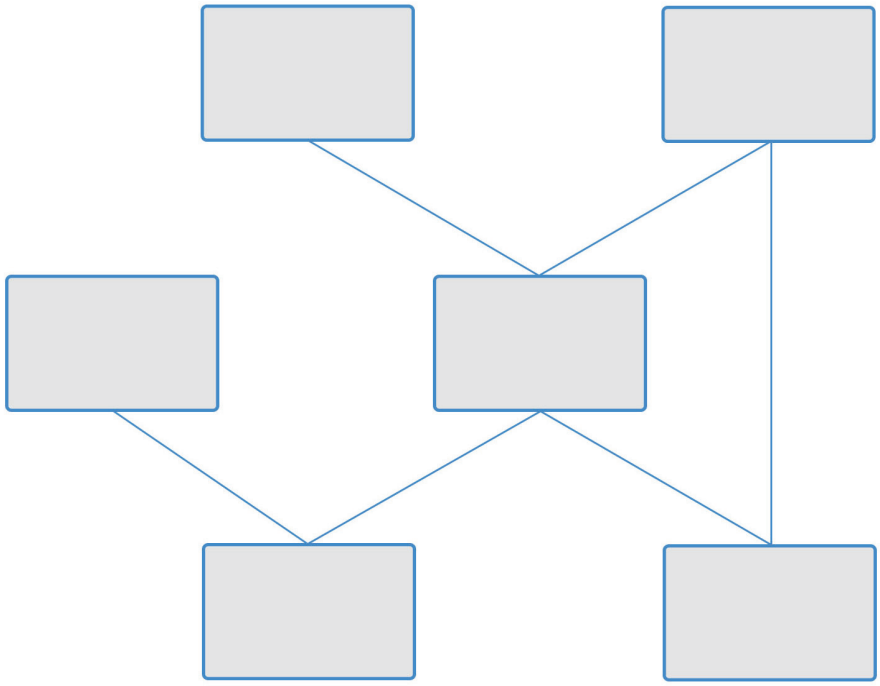

Abb. 2.2 Das Netzwerkdatenbankmodell

Eine neuere Entwicklung stellt die *objektorientierte Datenbank* dar. Diese definiert Objekte, denen sie bestimmte Attribute zuweist. In der Informatik hat sich die objektorientierte Programmierung in den letz-

ten Jahrzehnten immer stärker durchgesetzt. Diese ermöglicht es, Objekte zu gestalten, die Gegenständen in der realen Welt nachempfunden sind. Die Objektdatenbank überträgt diese Vorgehensweise auf die Speicherung der Daten. Das ermöglicht ein einfaches und effizientes Zusammenspiel mit objektorientierten Computerprogrammen. Eine Besonderheit besteht hierbei darin, dass man in der Datenbank auch Methoden definieren kann. Diese ermöglichen den Zugriff auf die Inhalte eines Objekts. Die Struktur ist in Abbildung 2.3 dargestellt. Mit diesem System ist es möglich, sehr komplexe Objekte zu verwalten.

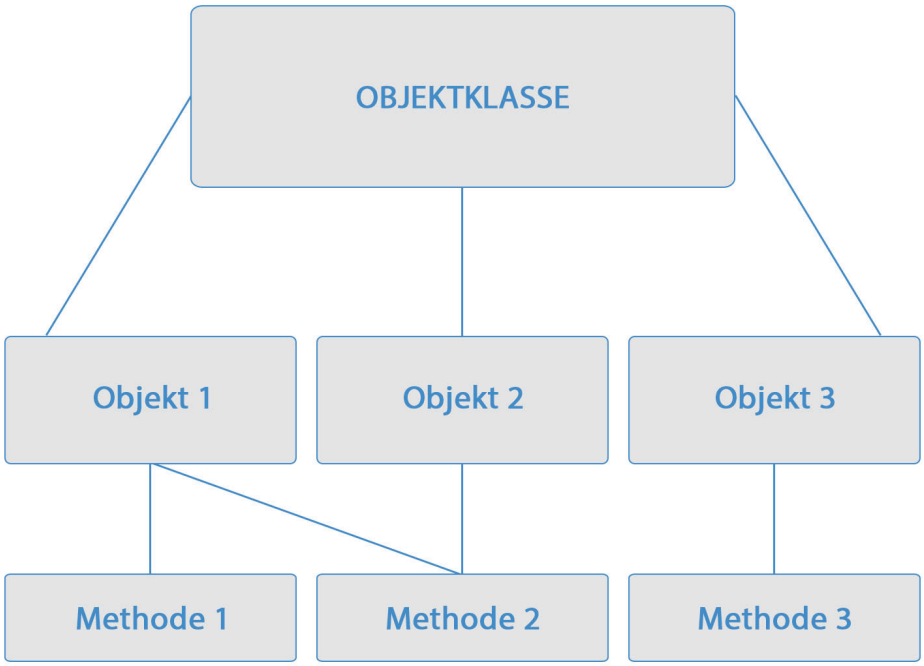

Abb. 2.3 Die objektorientierte Datenbank

Eine weitere Entwicklung stellt die *objektrelationale Datenbank* dar. Hierbei handelt es sich um eine Mischform zwischen der objektorientierten und der relationalen Datenbank. Sie ermöglicht es, Objekte zu erzeugen und Relationen zwischen ihnen zu definieren.

Schließlich gibt es noch das *dokumentorientierte Datenbankmodell*. Hierbei stellen einzelne Dokumente die Grundlage für die Datenspeicherung dar. Für den Aufbau dieser Dokumente gibt es jedoch keine festen Vorgaben. Es ist möglich, hierbei ganz unterschiedliche Informationen abzuspeichern. Jedes Dokument enthält einen Identifikator,

über den es sich eindeutig bestimmen lässt und der den Zugriff auf die Informationen erlaubt. Dabei ist es jedoch nicht möglich, Beziehungen zwischen verschiedenen Dokumenten herzustellen. Das stellt einen der wesentlichen Unterschiede zu relationalen Datenbanken dar. Das macht Abbildung 2.4 deutlich. Dokumentorientierte Datenbanken zeichnen sich dadurch aus, dass sie sehr einfach aufgebaut sind. Das gestaltet den Zugriff und die Auswertung der Daten einfach und effizient. Allerdings sorgen diese Systeme nicht dafür, dass die Informationen konsistent sind und außerdem kann es hierbei zu einer redundanten Speicherung kommen. Das führt dazu, dass der Anwender selbst – beziehungsweise die Programme, die die Datenbank verwalten – sich um diese Aufgabe kümmern müssen. Das gestaltet den Umgang mit diesen Datenbanken manchmal schwierig.

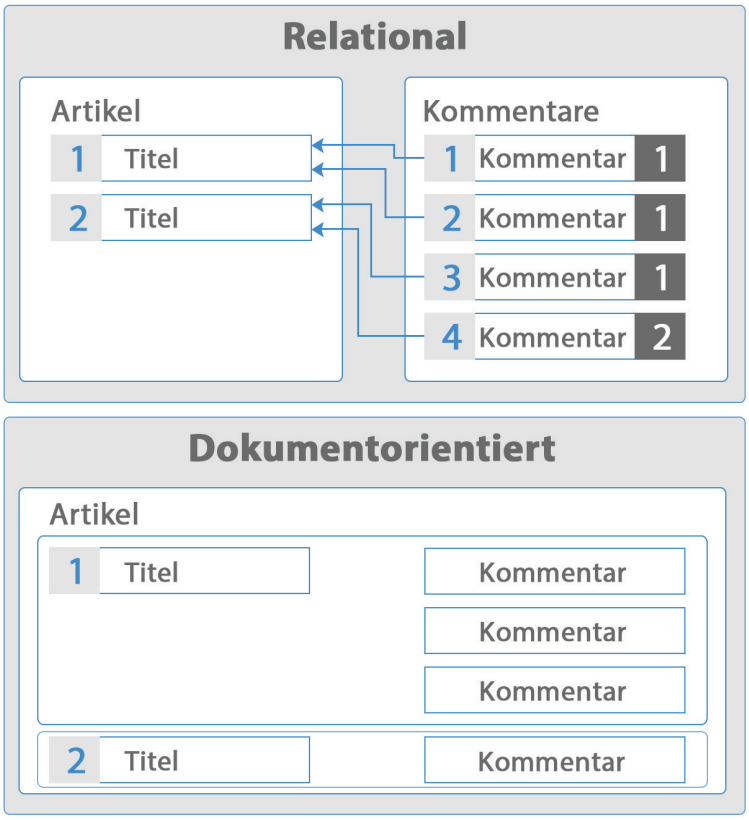

Abb. 2.4 Der Unterschied zwischen relationalen und dokumentorientierten Datenbanken

2.3 Das relationale Datenbankmodell im Detail erklärt

Das Thema dieses Buchs ist die Datenbanksprache SQL. Diese dient der Bearbeitung relationaler Datenbanken. Die theoretische Grundlage hierfür stellt das relationale Datenbankmodell dar. Daher ist es wichtig, auf dessen Eigenschaften etwas detaillierter einzugehen.

Relationale Datenbanken beruhen auf Tabellen. Jede Zeile enthält dabei einen zusammengehörigen *Datensatz*, der sich auf einen bestimmten Gegenstand oder auf eine bestimmte Person bezieht. Abbildung 2.5 zeigt die Struktur der Tabellen auf. Wenn es sich beispielsweise um die Mitgliederliste eines Sportvereins handelt, dann bezieht sich die Zeile auf ein einzelnes Mitglied. Sie wird auch als Tupel bezeichnet. Dieser Begriff steht für eine Sammlung von Werten mit fester Ordnung.

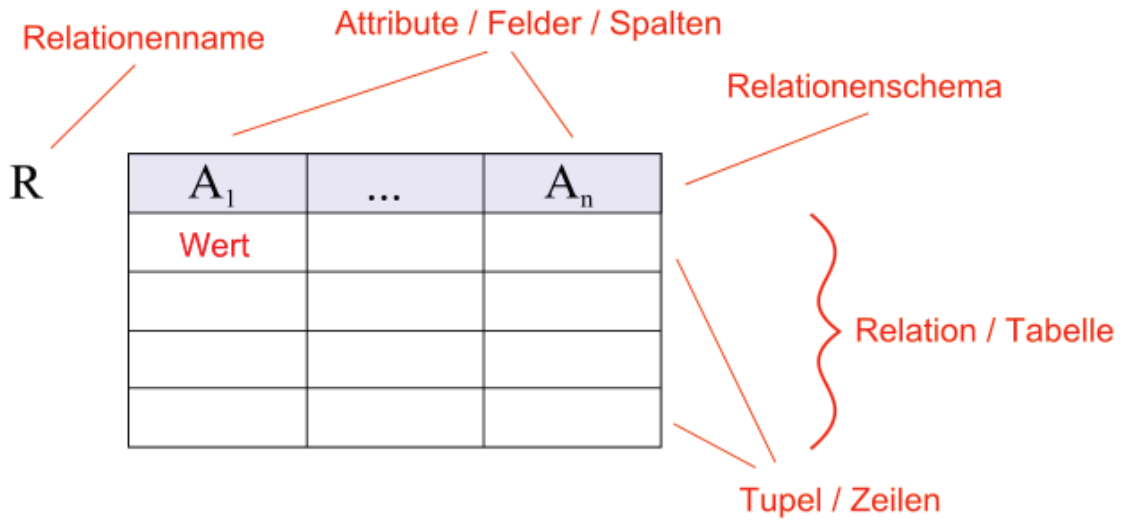

Abb. 2.5 Eine Tabelle in einer relationalen Datenbank

Die einzelnen Felder der Zeile enthalten dann die *Attributswerte*. Bei der bereits erwähnten Mitgliederliste könnte es sich hierbei um die Mitgliedsnummer, den Vornamen, den Nachnamen, die Telefonnummer, das Alter und um einige weitere Werte handeln. Dabei ist es wichtig, zu beachten, dass innerhalb einer Tabelle alle Datensätze das gleiche Relationsschema beachten müssen. Dieses bestimmt die Anzahl und die Reihenfolge der einzelnen Attributswerte. Das bedeutet, dass jede Zeile die gleichen Eigenschaften in der gleichen Reihenfolge be-

inhalten muss. Diese werden in einer relationalen Datenbank über die Festlegung der Spalten definiert. Jede Spalte erhält einen Titel, der angibt, welchen Wert sie aufnehmen soll. Außerdem ist hier festgehalten, welchen Datentyp sie dafür verwenden soll. Die Festlegung der Spalten bestimmt daher die Struktur der Tabelle. Wenn wir eine neue Tabelle anlegen, besteht deshalb der erste Schritt immer darin, sich zu überlegen, welche Spalten sie erhalten soll und welche Datentypen dafür jeweils notwendig sind.

Wichtig ist es bei der Gestaltung der Tabelle außerdem, dass jeder Datensatz eindeutig identifizierbar ist. Hierfür kommt ein Schlüssel zum Einsatz. In manchen Tabellen finden auch mehrere dieser Schlüssel gleichzeitig Verwendung. Diese müssen immer eindeutig sein. Das heißt, dass sie sich nicht wiederholen dürfen. Um wieder das Beispiel des Sportvereins aufzugreifen, wäre hierbei die Verwendung der Mitgliedsnummer als Schlüssel geeignet, da hierbei jede Nummer nur ein einziges Mal vergeben wird.

Eine wichtige Eigenschaft relationaler Datenbanken besteht darin, dass hierbei auch Beziehungen zwischen den verschiedenen Tabellen möglich sind. Um das zu verdeutlichen, soll das Beispiel des Sportvereins nun erweitert werden. Neben der Mitgliederliste fügen wir eine weitere Tabelle ein, die die einzelnen Angebote enthält – beispielsweise Fußball. Volleyball und Leichtathletik. Nun ist es möglich, eine weitere Tabelle zu gestalten, in der festgehalten ist, welches Mitglied welche Angebote nutzt. Das bedeutet, dass diese Tabelle sowohl auf die Mitglieder als auch auf die einzelnen Sportangebote Bezug nimmt. Diese Beziehung wird in der Regel über die Schlüssel der einzelnen Tabellen hergestellt. Um die Mitglieder in die neue Liste einzutragen, ist es lediglich notwendig, ihre Mitgliedsnummer anzugeben. Bei Bedarf ist es über diese Beziehung dann möglich, auch alle weiteren Informationen, die in der Tabelle zu den Mitgliedern abgelegt sind, abzurufen.

Diese vielfältigen Möglichkeiten der Beziehungen zwischen den einzelnen Tabellen führt dazu, dass hierbei häufig sehr komplexe Zusammenhänge entstehen – insbesondere wenn es sich um eine größere Datenbank handelt. Aus diesem Grund ist für das Erstellen einer relationalen Datenbank die Datenmodellierung von großer Bedeutung. Dazu ist es

notwendig, ein Schema zu erstellen, das zum einen festlegt, welche Informationen in die Datenbank aufgenommen werden sollen und zum anderen die Beziehungen zwischen diesen beschreibt.

Kapitel 3

Die Vorbereitungsmaßnahmen für die Arbeit mit Datenbanken

Wenn wir SQL in der Praxis anwenden möchten, ist es notwendig, hierfür ein Datenbankmanagementsystem zu installieren. Wie bereits beschrieben, ist diese Software für die Erstellung der Datenbank notwendig und außerdem ist sie für die Verwaltung der Daten verantwortlich. Ohne DBMS ist es daher nicht möglich, eine Datenbank zu nutzen.

Es gibt zahlreiche Datenbankmanagementsysteme, die SQL unterstützen. Das bedeutet, dass sich viele verschiedene Angebote für diesen Zweck eignen, sodass die Auswahl hierfür sehr groß ist. Der erste Schritt besteht daher darin, sich einen Überblick über die verschiedenen Möglichkeiten zu verschaffen und anschließend ein passendes Angebot auszuwählen. Daraufhin wird in diesem Kapitel gezeigt, wie die Installation dieser Software abläuft. Schließlich wird die grafische Benutzeroberfläche kurz vorgestellt, die es ermöglicht, Datenbanken ohne SQL-Kenntnisse zu erstellen.

3.1 Eine passende Datenbank-Software auswählen

In der Einleitung zu diesem Kapitel wurde bereits erwähnt, dass es sehr viele Datenbankmanagementsysteme gibt, die SQL als Datenbanksprache verwenden. Hier sollen nun einige Möglichkeiten vorgestellt werden. Aus diesen Angeboten wollen wir dann eine geeignete Software für die Verwendung im Rahmen dieses Lehrbuchs auswählen.

Einer der Marktführer im Bereich der Datenbankanwendungen ist das Unternehmen Oracle. Das zugehörige Produkt trägt den Namen Oracle Database. Die erste Version wurde bereits 1979 veröffentlicht – als erste kommerzielle SQL-Anwendung überhaupt. Oracle DB zeichnet sich dadurch aus, dass diese Software neben dem relationalen Datenbankmodell auch einige weitere Alternativen unterstützt. Insbesonde-

re bei großen Datenmengen bietet diese Software eine hervorragende Effizienz. Für unseren Kurs ist dieses DBMS jedoch nicht geeignet. Das liegt daran, dass es sich hierbei um eine proprietäre Software handelt. Die Lizenzkosten hierfür sind sehr hoch. Daher kommt die Oracle DB in erster Linie in einem gewerblichen Umfeld zum Einsatz.

Eine weitere Möglichkeit stellt die IBM DB2 dar. Auch hierbei handelt es sich um ein System, das bereits seit vielen Jahren auf dem Markt ist. Die erste Version erschien 1993. Das DBMS ist für verschiedene Betriebssysteme erhältlich und es kommt in vielen Unternehmen zum Einsatz. Darüber hinaus bietet IBM noch weitere Datenbankmanagementsysteme an – beispielsweise Informix. Doch auch hierbei handelt es sich um eine kommerzielle Software, für deren Nutzung erhebliche Lizenzgebühren anfallen.

Sehr beliebt ist außerdem der Microsoft SQL Server. Dieses Datenbankmanagementsystem ist ebenfalls proprietär und auf die Nutzung in einem gewerblichen Umfeld ausgerichtet. Auch diese Software zählt zu den am häufigsten verwendeten SQL-Datenbanken.

Hierbei handelt es sich nur um einen kleinen Ausschnitt aus kommerziellen Datenbanksystemen, die sich für die Anwendung in einem Unternehmen sehr gut eignen. Daneben gibt es jedoch auch einige Open-Source-Alternativen. Diese zeichnen sich dadurch aus, dass sie kostenfrei sind. Daher bieten sie sich auch für private Anwendungen und für gemeinnützige Organisationen hervorragend an. Auch zu Übungszwecken lassen sie sich sehr gut verwenden. Dabei handelt es sich jedoch nicht um die einzigen Einsatzmöglichkeiten. Auch immer mehr kommerziell ausgerichtete Unternehmen entscheiden sich dafür, für die Verwaltung ihrer Daten Open-Source-Datenbanken zu nutzen. Das stellt in vielen Fällen eine besonders kosteneffiziente Lösung dar. Damit für die Bearbeitung der Aufgaben in diesem Buch keine zusätzlichen Ausgaben anfallen, ist es sinnvoll, hierfür ein Open-Source-Datenbanksystem zu verwenden.

Das wohl am häufigsten verwendete Datenbanksystem weltweit ist *MySQL*. Hierbei handelt es sich um Open-Source-Software, die kostenfrei genutzt werden darf. Darüber hinaus sind jedoch auch lizenzpflichtige Enterprise-Versionen verfügbar. Die häufige Verwendung und die

kostenfreie Nutzung stellen zwei entscheidende Vorteile für die Projekte in diesem Buch dar. Daher wollen wir für die Umsetzung MySQL verwenden.

Der Vollständigkeit halber sollen nun jedoch noch einige weitere Open-Source-Datenbanksysteme vorgestellt werden. Immer beliebter wird beispielsweise MariaDB. Hierbei handelt es sich um eine Abspaltung von MySQL. Als der amerikanische Software-Konzern Oracle MySQL übernahm, gestaltete der ursprüngliche Entwickler Michael Widenius eine eigene Version, die jedoch auf MySQL beruht. Darüber hinaus gibt es noch einige weitere Möglichkeiten für Open-source-Datenbanken, die jedoch deutlich seltener zum Einsatz kommen – beispielsweise MonetDB oder PostgreSQL.

3.2 MySQL – eines der beliebtesten Datenbanksysteme

Die Entwicklung von MySQL begann 1994. Zu dieser Zeit hatten sich relationale Datenbanken und damit SQL bereits etabliert und stellten eine sehr beliebte Möglichkeit dar, um Daten in Unternehmen und Organisationen zu verwalten. Allerdings handelte es sich hierbei fast ausschließlich um kommerzielle Software.

Daher beschlossen die schwedischen Entwickler Michael Widenius, David Axmark und Allan Larson, eine quellenoffene Version zu entwickeln. Diese erhielt den Namen MySQL. My ist der Name der Tochter von Michael Widenius. SQL steht für die bekannte Datenbanksprache. 1995 gründeten die Entwickler das Unternehmen MySQL AB, um diese Software zu vertreiben. Noch in diesem Jahr erschien die erste Version von MySQL.

Der durchschlagende Erfolg von MySQL ist eng mit der Ausbreitung des Internets verbunden. Als gegen Ende der 90er Jahre die Zahl der dynamischen Internetseiten stark anstieg, war für den Betrieb eine passende Datenbanksoftware notwendig. Damals kamen hierfür vorwiegend proprietäre Programme zum Einsatz. Ein viel beachteter Artikel in der Computer-Fachzeitschrift c't zeigte 1998 jedoch auf, dass Open-Source-Software hierfür eine gleichwertige Alternative darstellt und die Kosten für das Betreiben einer Internetseite dadurch deutlich senken

kann. In diesem Rahmen entstand der Begriff LAMP. Dieser steht als Akronym für das Betriebssystem Linux, den Webserver Apache, das Datenbanksystem MySQL und die Programmiersprache PHP. Dabei handelt es sich um vier lizenzfreie Techniken, mit denen sich aber dennoch dynamische Internetseiten einfach und effizient gestalten lassen. Mittlerweile setzen viele Millionen von Internetangeboten auf LAMP und damit auf MySQL. Das führte dazu, dass sich diese Software zum am häufigsten verwendeten Datenbanksystem entwickelte. Sogar viele weltbekannte Internetangebote mit Millionenumsätzen verwenden diese Open-Source-Lösung. Bekannte Seiten, die MySQL nutzen, sind beispielsweise Facebook, YouTube, Twitter und Flickr.

Dieser Erfolg führte dazu, dass das schwedische Unternehmen die Aufmerksamkeit großer Konzerne auf sich zog. 2008 erwarb der US-amerikanische Softwarehersteller Sun Microsystems MySQL AB für rund eine Milliarde Dollar. Bereits im darauffolgenden Jahr übernahm Oracle Sun. Mit diesem Schritt waren jedoch viele der ursprünglichen MySQL-Entwickler nicht einverstanden. Unter der Leitung von Michael Widenius spaltete sich ein erheblicher Teil der Programmierer von Oracle ab und entwarf MariaDB als neues Open-Source Datenbanksystem.

3.3 Download und Installation

Bevor wir mit der Arbeit mit Datenbanken beginnen können, ist es notwendig, das Datenbankmanagementsystem herunterzuladen und auf dem Rechner zu installieren. Dieses nimmt dann die SQL-Befehle entgegen, die wir später kennenlernen werden und führt die zugehörigen Aktionen durch.

Um MySQL herunterzuladen, ist es notwendig, den folgenden Link zu öffnen:

https://bmu-verlag.de/sql1

Auf dieser Seite ist es möglich, aus verschiedenen Optionen für den Download auszuwählen. MySQL ist für zahlreiche Betriebssysteme verfügbar. Das bedeutet, dass es möglich ist, die Aufgaben in diesem Kurs auch mit Rechnern zu erledigen, die Linux oder macOS verwenden.

Hierfür ist es an dieser Stelle notwendig, eine passende Option auszuwählen. Da die meisten Leser jedoch wahrscheinlich Windows verwenden, soll der Download- und Installationsprozess für dieses Betriebssystem etwas ausführlicher beschrieben werden. Hierfür ist es notwendig, in der Auswahlliste oben auf der Seite das Betriebssystem Windows auszuwählen. Danach ist es empfehlenswert, den „MySQL Installer for Windows" anzuklicken. Abbildung 3.1 zeigt diese Download-Seite.

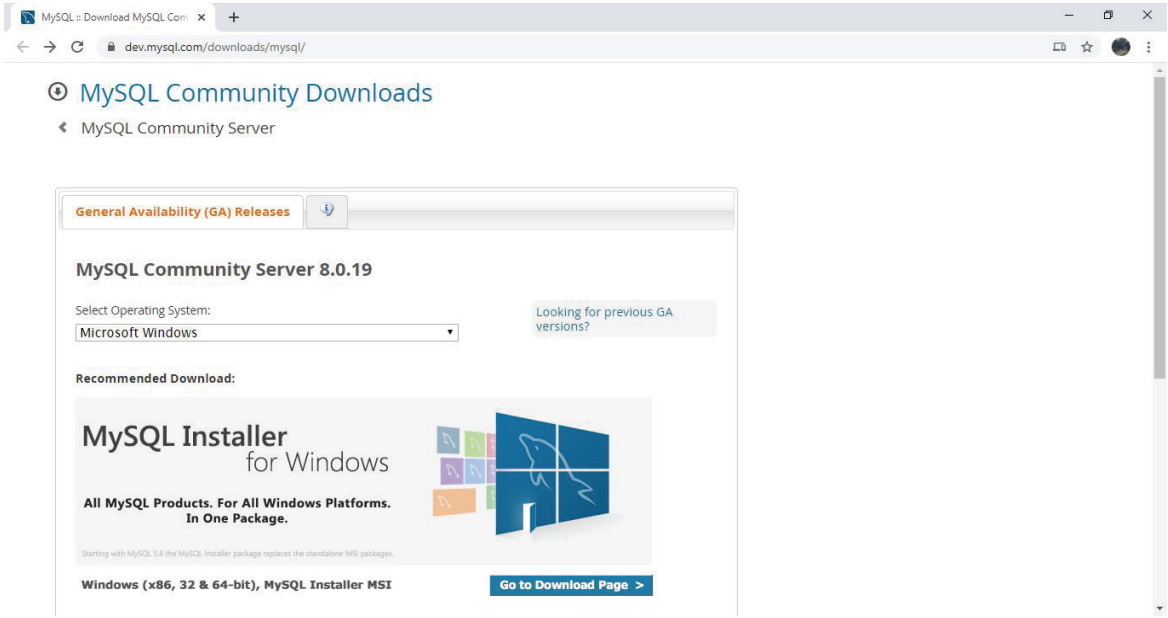

Abb. 3.1 Die Download-Seite für die Windows-Version

Daraufhin erscheint eine neue Seite. Hier sind zwei verschiedene Download-Möglichkeiten verfügbar. In der Regel ist es sinnvoll, den ersten Button auszuwählen. In diesem Fall wird ein Installer installiert, der alle weiteren Dateien eigenständig herunterlädt. In Abbildung 3.2 ist zu sehen, dass hierbei angegeben wird, dass es sich um die 32-bit-Version handelt. Doch auch Leser mit einem 64-bit-Rechner können diesen Link verwenden. Diese Angabe bezieht sich nur auf den Installer selbst. Dieser wählt dann jedoch selbstständig die passenden Version für den entsprechenden Rechner aus.

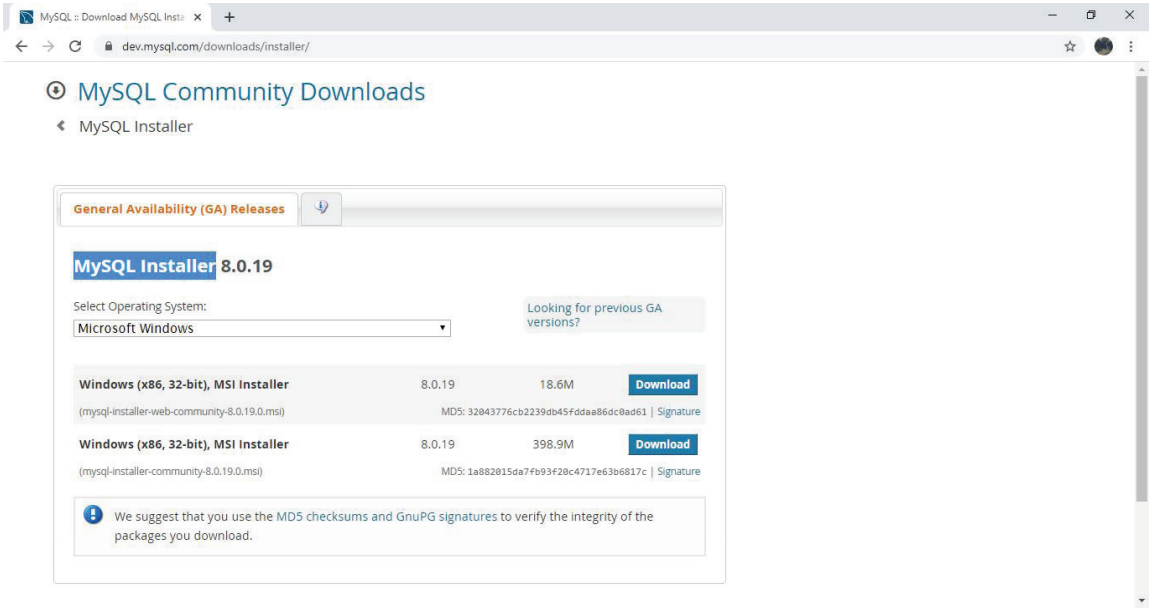

Abb. 3.2 Den Installer herunterladen

Nun erscheint eine Seite, die den Besucher dazu auffordert, sich mit einem Oracle-Account einzuloggen beziehungsweise diesen anzulegen. Leser, die daran Interesse haben, können hier kostenlos ein Benutzerkonto anlegen. Allerdings ist dies nicht notwendig. Weiter unten auf der Seite erscheint auch die Option, die Software ohne Login herunterzuladen – so wie dies in Abbildung 3.3 zu sehen ist.

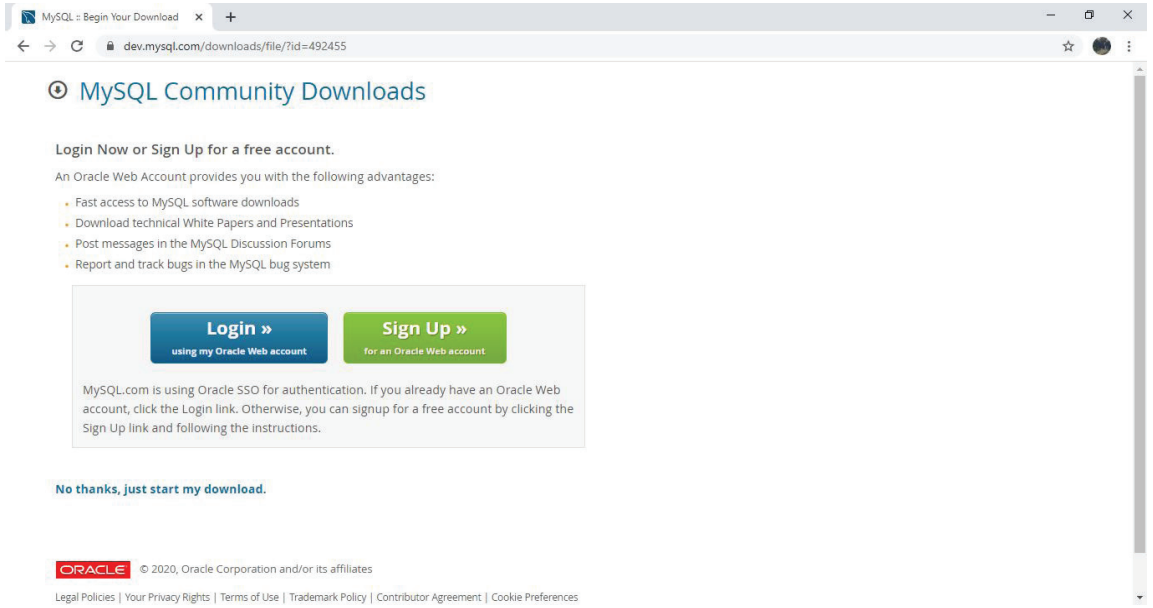

Abb. 3.3 Der Download ist mit oder ohne Benutzerkonto möglich

Nachdem der Installer heruntergeladen wurde, ist es lediglich notwendig, diesen auszuführen. Dabei kommt es zu mehreren Abfragen der gewünschten Einstellungen für die Konfiguration. Für die Aufgaben in diesem Buch ist hierbei jedoch keine spezielle Auswahl notwendig. Daher ist es am einfachsten, stets die Standardeinstellung zu übernehmen.

Etwas Vorsicht ist lediglich an der Stelle geboten, an der der Anwender dazu aufgefordert wird, ein Passwort auszuwählen. Das entsprechende Fenster ist in Abbildung 3.4 zu sehen. Hierbei ist es wichtig, sich dieses gut zu merken, da es später für den Zugriff auf die Datenbank notwendig ist. Da für die Beispiele in diesem Buch keine hohe Sicherheit erforderlich ist, wählen wir hierfür „xyz123". Dabei kann jedoch jeder Leser selbst entscheiden, welche Zugangsdaten er verwendet. Wichtig ist es lediglich, sich das gewählte Passwort zu notieren.

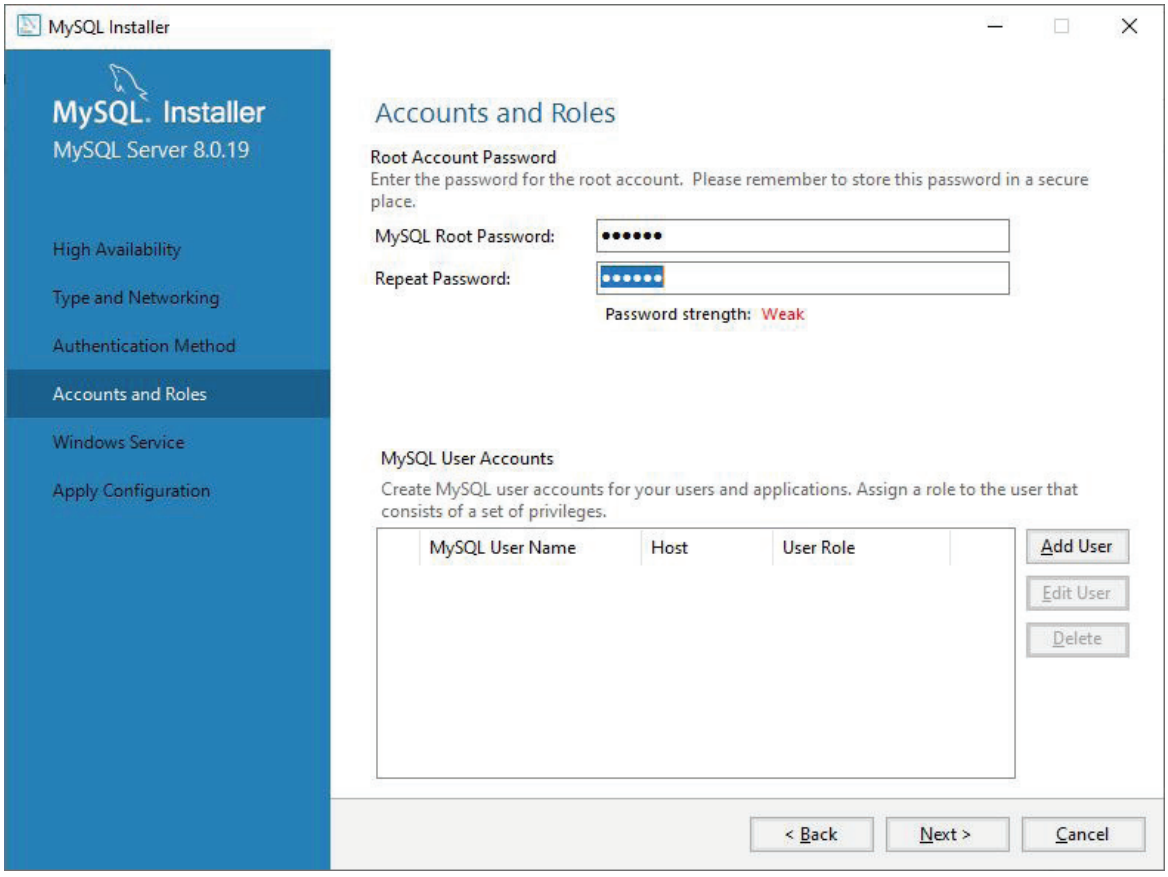

Abb 3.4 Die Eingabe des Passworts

Dieses Passwort ist bereits bei einem der weiteren Installations-Schritte erforderlich. Wenn sich das Fenster mit der Bezeichnung „Connect to Server" öffnet, das in Abbildung 3.5 zu sehen ist, ist es notwendig, genau dieses Passwort einzugeben und die Verbindung herzustellen.

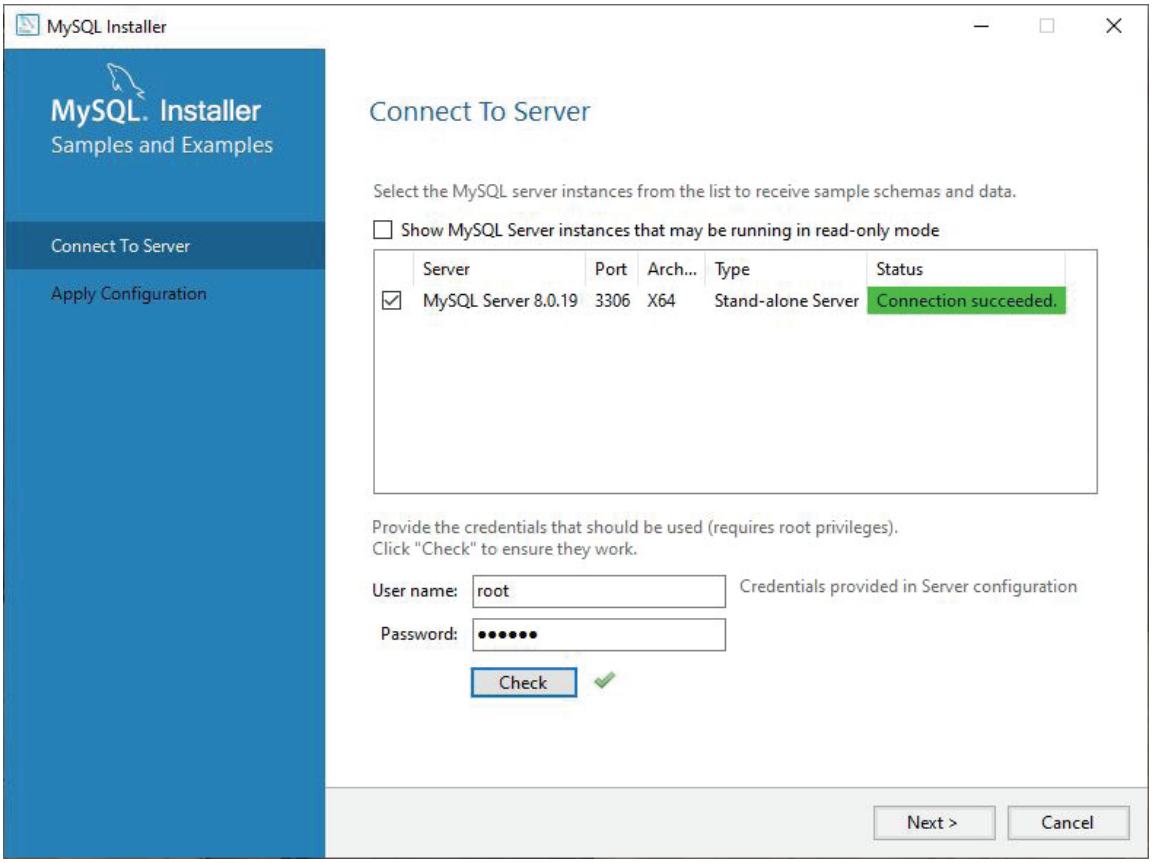

Abb. 3.5 Die Passworteingabe für die Verbindung zum Server

3.4 Die verschiedenen Benutzeroberfläche kennenlernen

Um mit MySQL zu arbeiten, gibt es mehrere Möglichkeiten. Diese Werkzeuge zeichnen sich durch erhebliche Unterschiede bei der Bedienung und durch verschiedene Funktionen aus. Dieser Abschnitt dient dazu, diese kurz vorzustellen und auszuprobieren. Auf diese Weise erhalten wir einen Überblick über die Vor- und Nachteile der verschiedenen Alternativen. Dabei werden wir auch bereits die ersten SQL-Befehle einfügen, um die Funktionsweise deutlich zu machen. Das dient jedoch

nur der Einführung. Es ist noch nicht notwendig, sich diese zu merken. Eine ausführliche Erklärung dazu folgt dann in den nächsten Kapiteln.

MySQL enthält beispielsweise Möglichkeiten, deren Erscheinungsbild an einen Kommandozeileninterpreter erinnert. Hier ist es notwendig, jeden einzelnen SQL-Befehl von Hand einzutippen. Durch die Betätigung der Eingabetaste wird dieser dann ausgeführt. Eine dieser Oberflächen ist unter der Bezeichnung MySQL Command Line Client verfügbar. Um sie kennenzulernen, rufen wir sie nun auf. Dabei erscheint das Fenster, das in Abbildung 3.6 zu sehen ist.

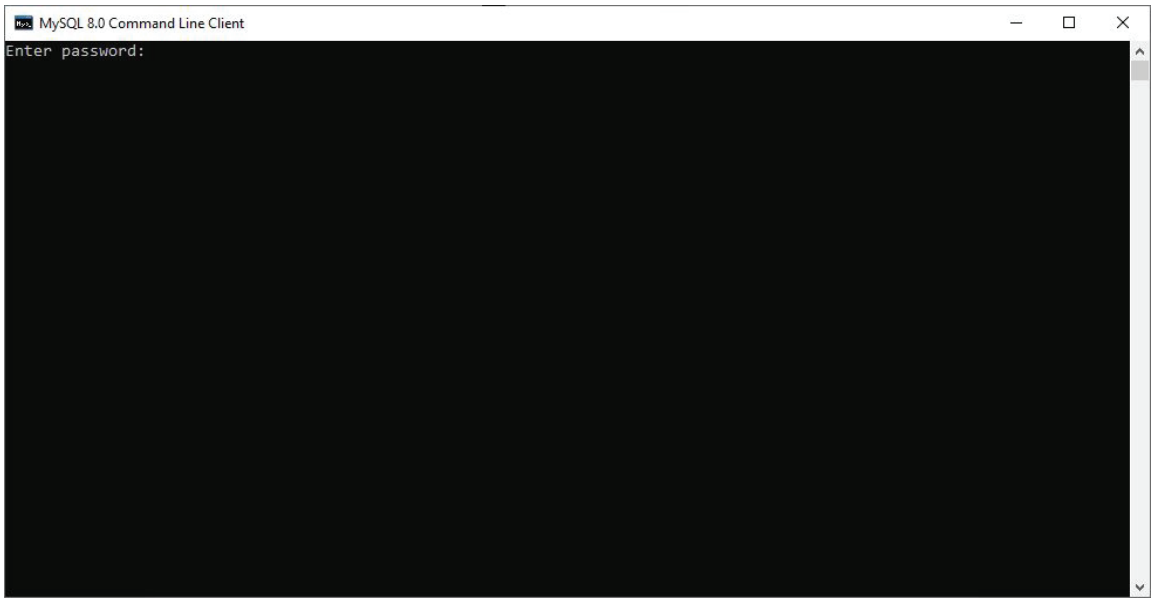

Abb. 3.6 Der MySQL Command Line Client

Hier ist es nun zunächst notwendig, das Passwort einzugeben, das wir uns im vorigen Abschnitt notiert haben. Daraufhin erscheint eine Begrüßung zum Programm. Nun können wir einmal den Begriff SHOW `DATABASES`; eingeben. Dabei ist es wichtig, darauf zu achten, dass am Ende des Befehls immer ein Semikolon stehen muss. Es ist üblich, SQL-Befehle in Großbuchstaben zu schreiben. Das ist jedoch nicht unbedingt notwendig. Es wäre auch möglich, den entsprechenden Befehl klein zu schreiben. Auf die Ausführung hat das keinerlei Einfluss. Dennoch ist es sinnvoll, sich an diese Vorgabe zu halten, da sie zur Übersichtlichkeit des Codes beiträgt.

Nach der Bestätigung durch die Eingabetaste erscheint die Anzeige, die in Abbildung 3.7 zu sehen ist. Hier sind alle Datenbanken aufgelistet, die als Anschauungsbeispiele bereits in MySQL vorinstalliert sind.

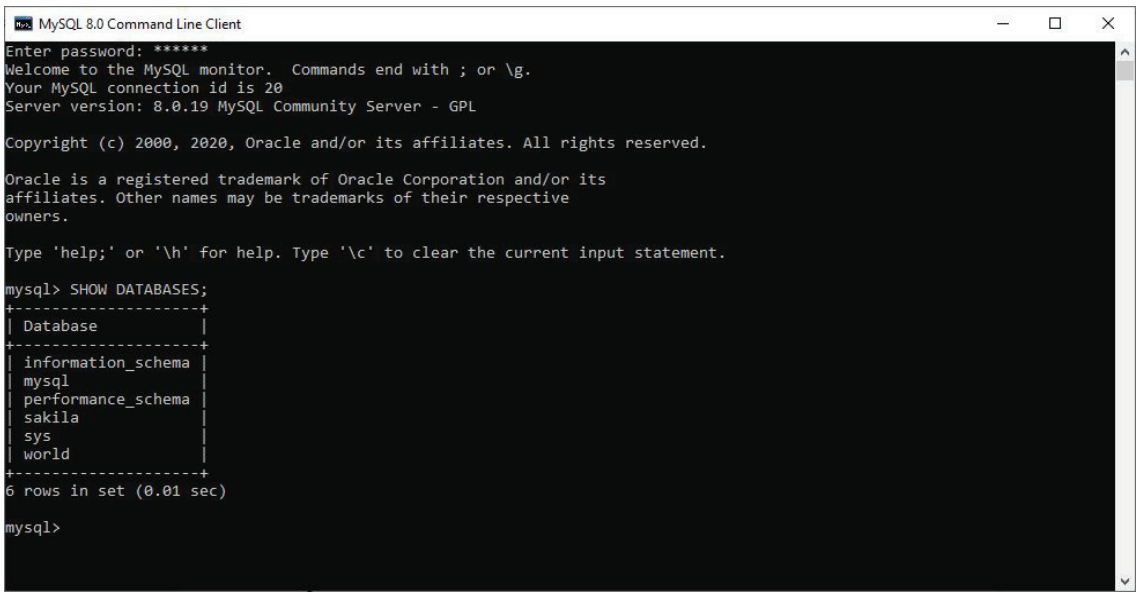

Abb. 3.7 Die Anzeige der vorhandenen Datenbanken

Nun geben wir den Befehl `CREATE DATABASE testDB1;` ein. Dieser erstellt eine neue Datenbank mit dem Namen `testDB1`. Namen von Datenbanken, Tabellen und andere Bezeichner werden normalerweise klein geschrieben. Das führt dazu, dass bereits auf den ersten Blick der Unterschied zu SQL-Befehlen deutlich wird. Daraufhin fügen wir nochmals den Befehl `SHOW DATABASES;` ein. Auf diese Weise erkennen wir, dass nun die neue Datenbank in der Auflistung erscheint, so wie dies in Abbildung 3.8 zu sehen ist. Wenn wir diese dann verwenden möchten – beispielsweise um Tabellen anzulegen und darin Daten einzufügen – müssen wir den Befehl `USE testDB1;` verwenden.

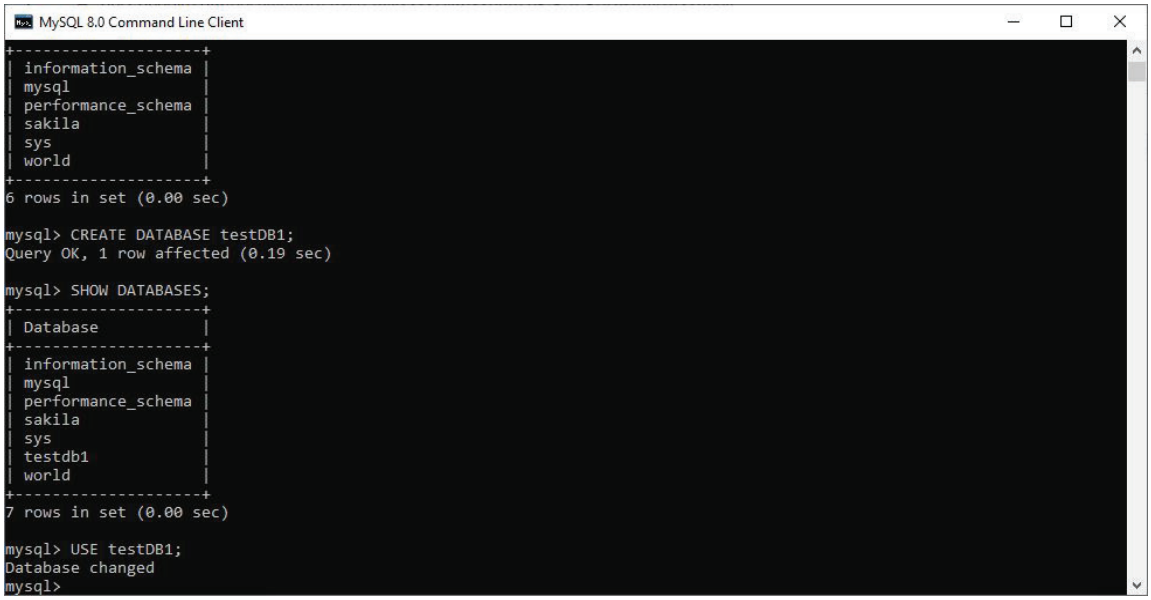

Abb. 3.8 Die Anzeige der neuen Datenbank

Eine weitere Möglichkeit stellt die MySQL Shell dar. Diese soll an dieser Stelle ebenfalls kurz vorgestellt werden – auch wenn wir sie später nicht weiter benutzen werden. Die MySQL Shell wirkt auf den ersten Blick recht ähnlich wie der eben vorgestellte Command Line Client. Tatsächlich gibt es dabei einige Gemeinsamkeiten. Allerdings bietet die MySQL Shell einige weitere Funktionen.

Dieses Werkzeug bietet sich in erster Linie für Programmierer an, die SQL innerhalb eines Computerprogramms verwenden und auf diese Weise die Datenbankabfragen automatisch durchführen lassen. Die MySQL Shell bietet verschiedene Modi an, die es ermöglichen, Programmcode in den Programmiersprachen Python oder JavaScript auszuführen. Gleichzeitig bietet die Shell selbstverständlich einen Zugriff auf die Datenbank. Das macht es ganz einfach, die entsprechenden Programme mit Datenbankanbindung auszuprobieren, weshalb die MySQL Shell hierfür ein sehr beliebtes Entwicklungswerkzeug darstellt.

An dieser kurzen Ausführung wird jedoch bereits deutlich, weshalb die MySQL Shell für dieses Lehrbuch nicht das passende Werkzeug darstellt. Da hier nur reines SQL vermittelt wird, ohne Bezug zu einer Programmiersprache, sind diese Zusatzfunktionen nicht erforderlich. Dennoch soll auch diese Oberfläche kurz vorgestellt werden.

Wenn wir das Programm öffnen, erscheint das Fenster, das in Abbildung 3.9 zu sehen ist.

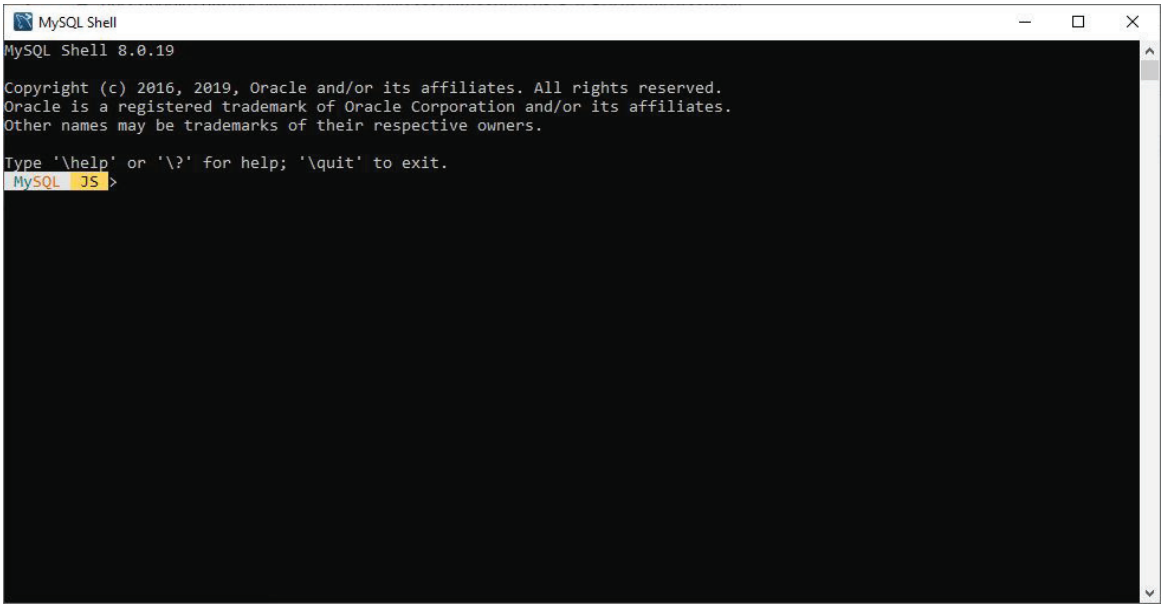

Abb. 3.9 Die MySQL Shell

Um mit der Arbeit zu beginnen, müssen wir uns hierbei zunächst mit dem Server verbinden. Dazu kommt folgender Befehl zum Einsatz: `\connect root@localhost`. Hierbei ist im Gegensatz zu den eigentlichen SQL-Kommandos kein Semikolon notwendig. Nachdem wir diesen mit der Eingabetaste bestätigt haben, müssen wir wieder das Passwort eingeben. Im Unterschied zum Command Line Client können wir nun jedoch nicht gleich damit beginnen, SQL-Befehle einzugeben. Zunächst ist es notwendig, den passenden Modus auszuwählen. Wie bereits erwähnt gibt es hierbei Modi für verschiedene Programmiersprachen und für reines SQL. Um in den SQL-Modus zu wechseln, kommt folgender Befehl zum Einsatz: `\sql`. Nun können wir mit der Shell genau auf die gleiche Weise arbeiten, wie mit dem Command Line Client. Abbildung 3.10 zeigt beispielsweise, dass wir uns wieder mit dem Befehl `SHOW DATABASES;` die vorhandenen Datenbanken anzeigen lassen können.

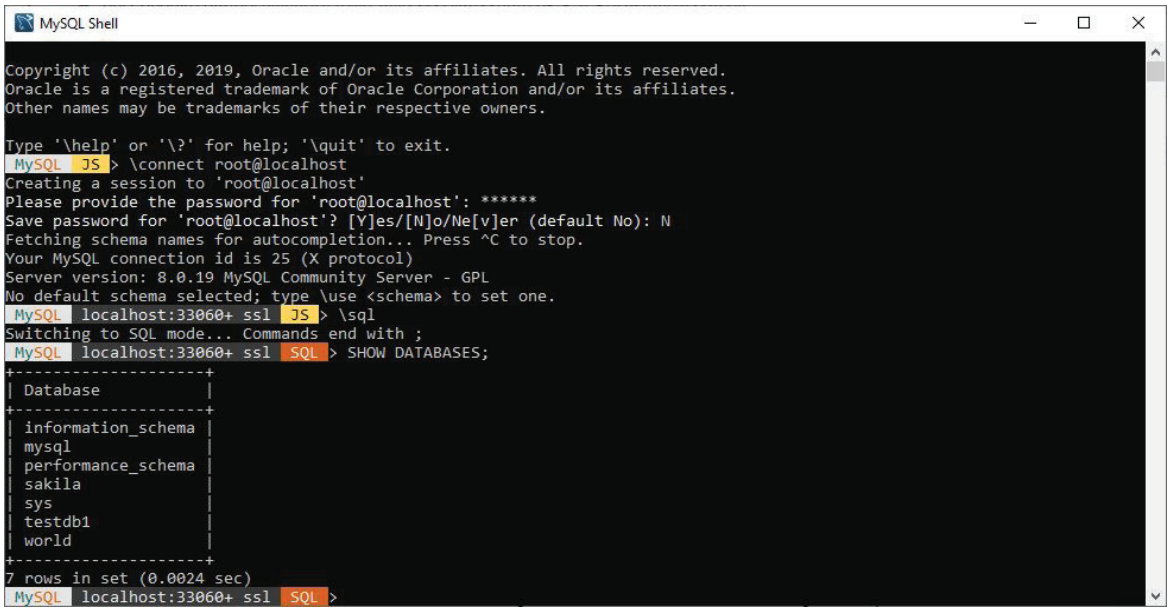

Abb. 3.10 Verbindung, Wechseln in den SQL-Modus und Anzeige der Datenbanken

Eine weitere Möglichkeit besteht darin, die SQL Workbench zu verwenden. Hierbei handelt es sich um eine grafische Benutzeroberfläche. Dabei ist es möglich, SQL-Befehle direkt einzugeben. Doch lassen sich damit Datenbanken und die darin enthaltenen Tabellen auch ganz ohne SQL-Kenntnisse gestalten. Durch einen entsprechenden Mausklick erzeugt die Software die zugehörigen SQL-Befehle und führt sie daraufhin automatisch aus.

Das Ziel dieses Buchs besteht jedoch gerade darin, die Verwendung von SQL zu vermitteln. Zu diesem Zweck ist es nicht zielführend, die Datenbanken automatisch erstellen zu lassen. In diesem Buch arbeiten wir zwar mit der MySQL Workbench, jedoch stets auf die Weise, dass wir die Befehle direkt eingeben. Dennoch ist es interessant, zu Beginn einmal die grafische Benutzeroberfläche kennenzulernen und damit innerhalb weniger Minuten die ersten eigenen Tabellen zu erzeugen.

Wenn wir diese Software aufrufen, erscheint zunächst die Ansicht, die in Abbildung 3.11 zu sehen ist.

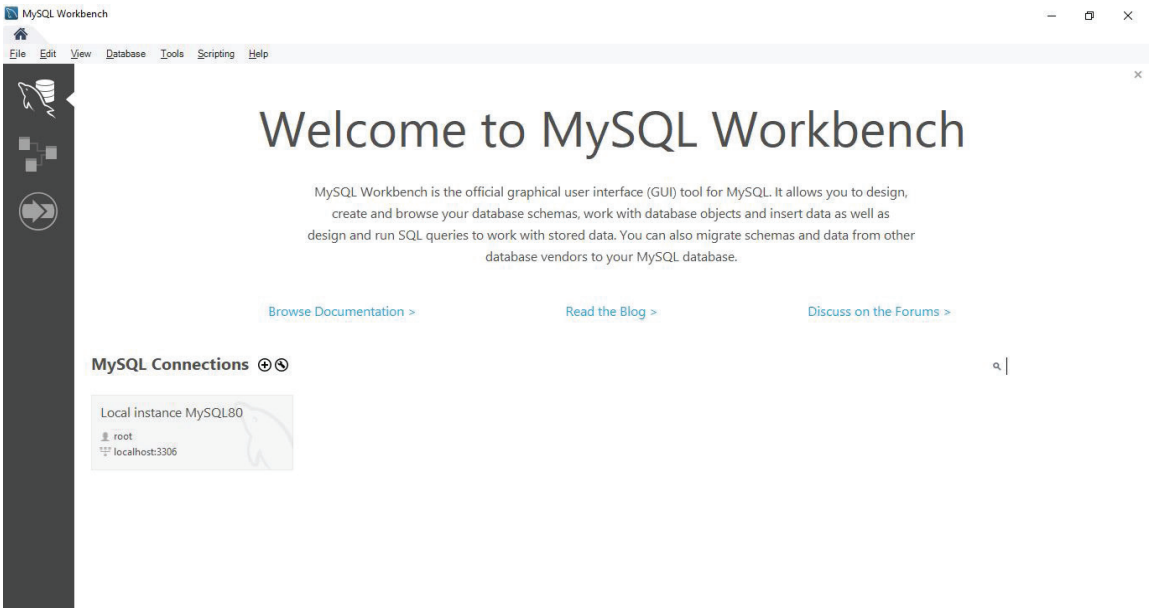

Abb. 3.11 Die Startseite der MySQL Workbench

Um die grafische Benutzeroberfläche kennenzulernen, klicken wir in der Menüleiste auf „File" und daraufhin auf „New Model". Im Fenster, das sich daraufhin öffnet, führen wir einen Doppelklick auf „Add Table" durch. Auf diese Weise ist es möglich, eine neue Tabelle zu erstellen – ohne dafür SQL-Befehle zu verwenden. Dennoch sind hierfür einige Kenntnisse über den Aufbau einer derartigen Tabelle notwendig. Hierbei ist es notwendig, die Spalten mit einer konkreten Bezeichnung und einem passenden Datentyp vorzugeben. Diese Details werden später noch mit allen Einzelheiten erklärt. Vorerst reicht es aus, die Einträge, die in Abbildung 3.12 zu sehen sind, zu übernehmen. Auf diese Weise ist es möglich, die Struktur für eine Datenbanktabelle vorzugeben.

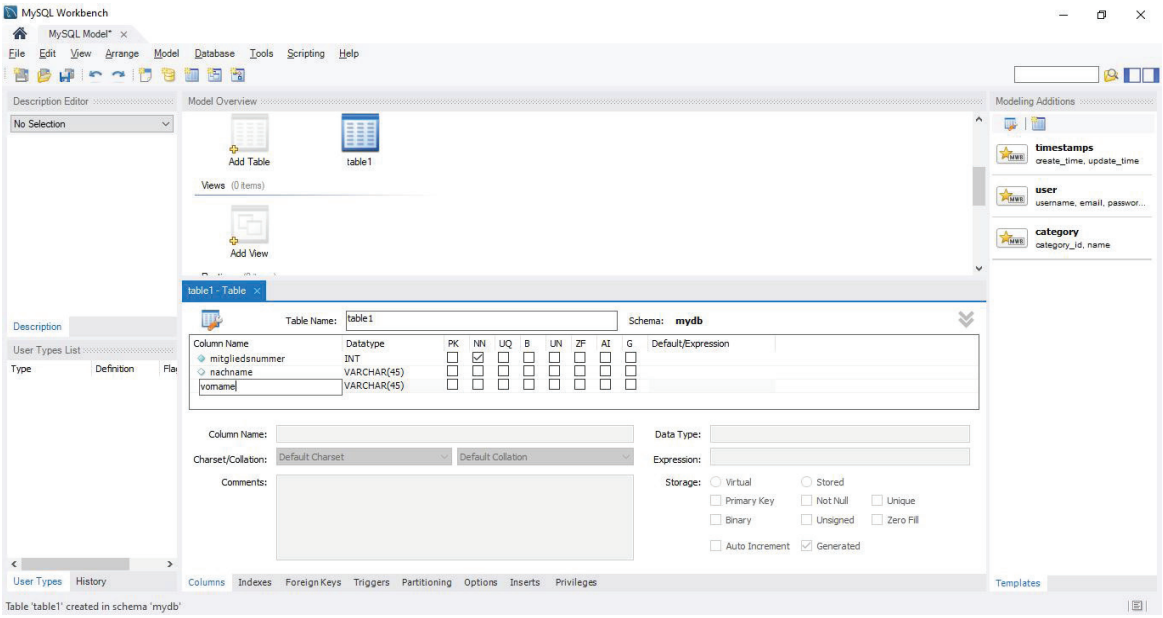

Abb. 3.12 Eine Tabelle über die grafische Benutzeroberfläche erstellen

Nun müssen wir die Tabelle noch erstellen und in eine Datenbank einfügen. Dazu klicken wir in der Menüleiste auf „Database" und anschließend auf „Forward Engineer", so wie das in Abbildung 3.13 dargestellt ist.

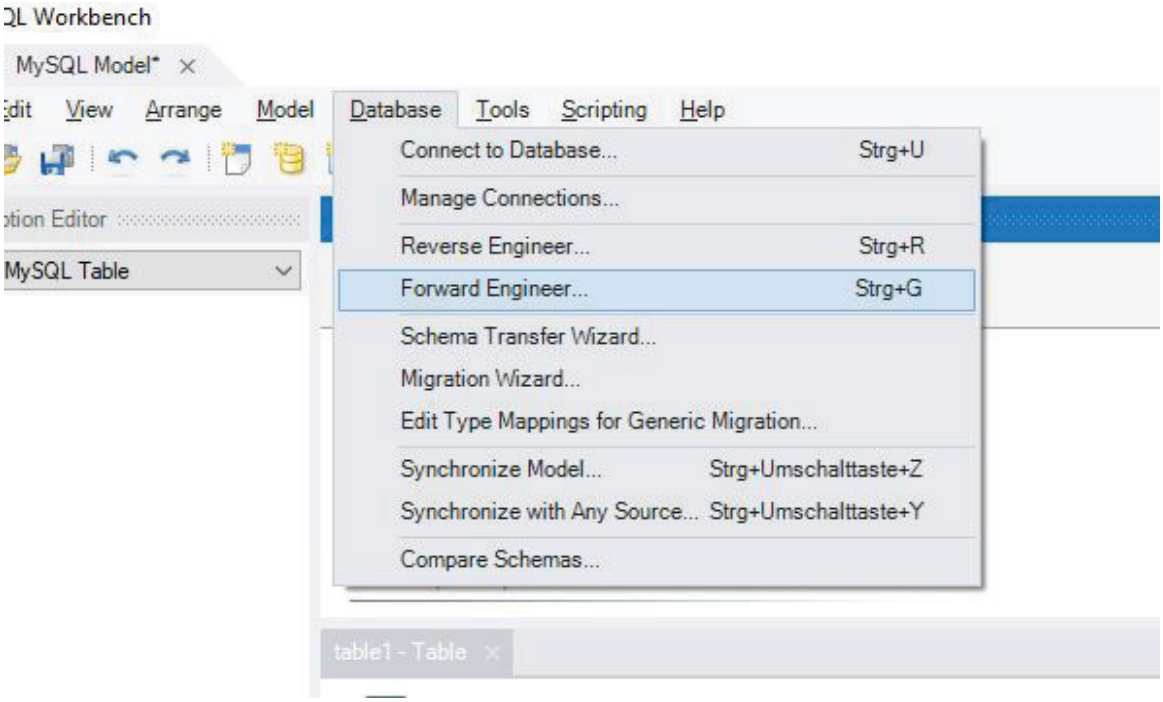

Abb. 3.13 Der erste Schritt für die Verbindung mit der Datenbank

Daraufhin erscheint ein weiteres Fenster, in dem wir die Angaben für die Verbindung zum DBMS machen müssen. Hier können wir jedoch die Standard-Einstellungen übernehmen und das Fenster einfach mit einem Klick auf „Next" bestätigen. Auch das nächste Fenster, in dem wir Optionen für die neue Datenbank vorgeben können, lassen wir unverändert. Im nächsten Schritt sehen wir dann die Objekte, die exportiert werden sollen. Abbildung 3.14 zeigt, dass hier nun eine Tabelle vorhanden ist. Dabei handelt es sich genau um die Tabelle, die wir gerade erstellt haben.

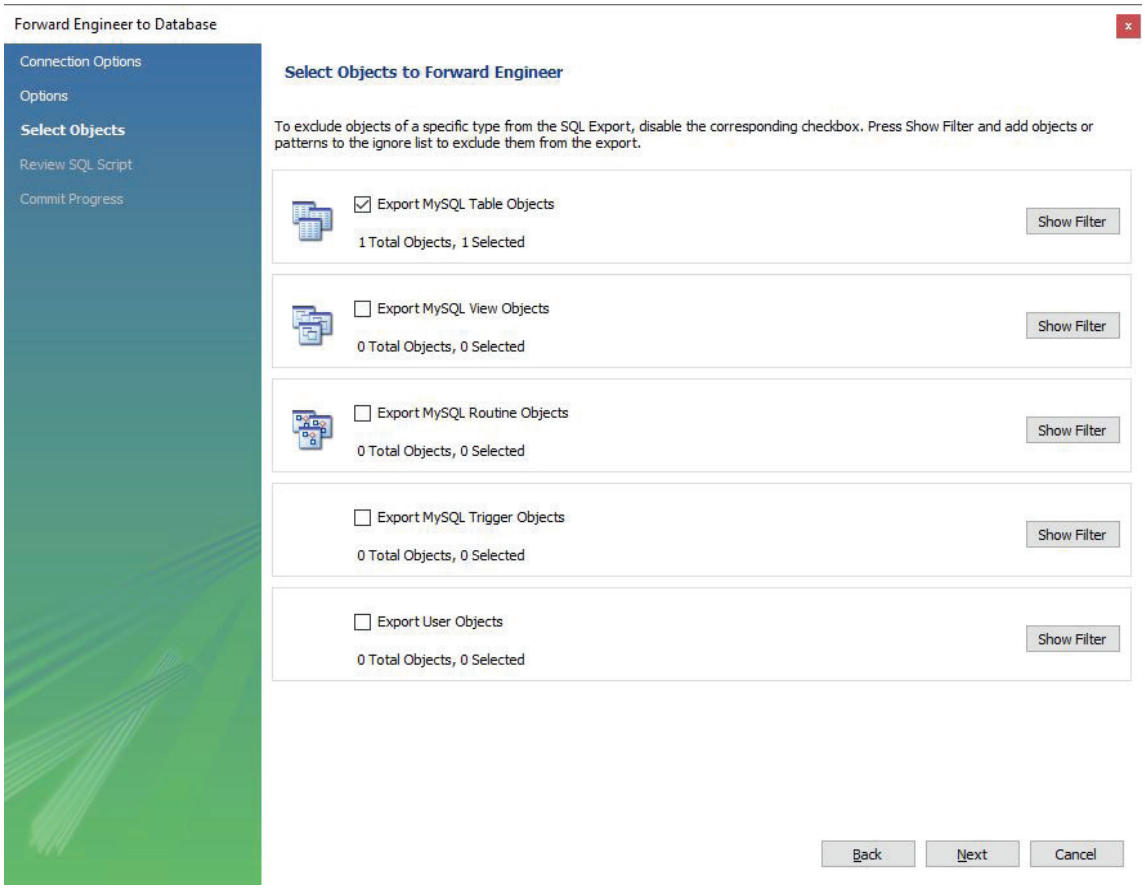

Abb. 3.14 Der Export der Tabelle

Im nächsten Fenster erscheinen nun viele SQL-Befehle, deren Bedeutung wir noch nicht kennen. Abbildung 3.15 zeigt, dass hier unter anderem der Befehl `CREATE SCHEMA IF NOT EXISTS 'mydb'` auftaucht – allerdings mit einigen weiteren Zusätzen. Dieser erzeugt ein neues Schema mit der Bezeichnung mydb. In MySQL sind die Begriffe Schema

und Database weitestgehend gleichbedeutend. Mit dem Schema können wir daher auf die gleiche Weise arbeiten wie mit einer Datenbank. Der Name des Schemas wurde von der Workbench automatisch zu Beginn vorgegeben. Wir hätten den Namen allerdings auch ändern können. Danach folgt der Befehl `USE 'mydb' ;`. Diesen haben wir bereits verwendet. Auf diese Weise wechseln wir zum gerade erstellten Schema, um dieses zu bearbeiten. Das nächste Kommando erstellt dann die Tabelle. Dessen Funktionsweise ist noch nicht bekannt – aber hierbei geht es ja gerade darum, eine Tabelle ohne SQL-Kenntnisse zu erstellen. Selbstverständlich wird es später noch detailliert erklärt.

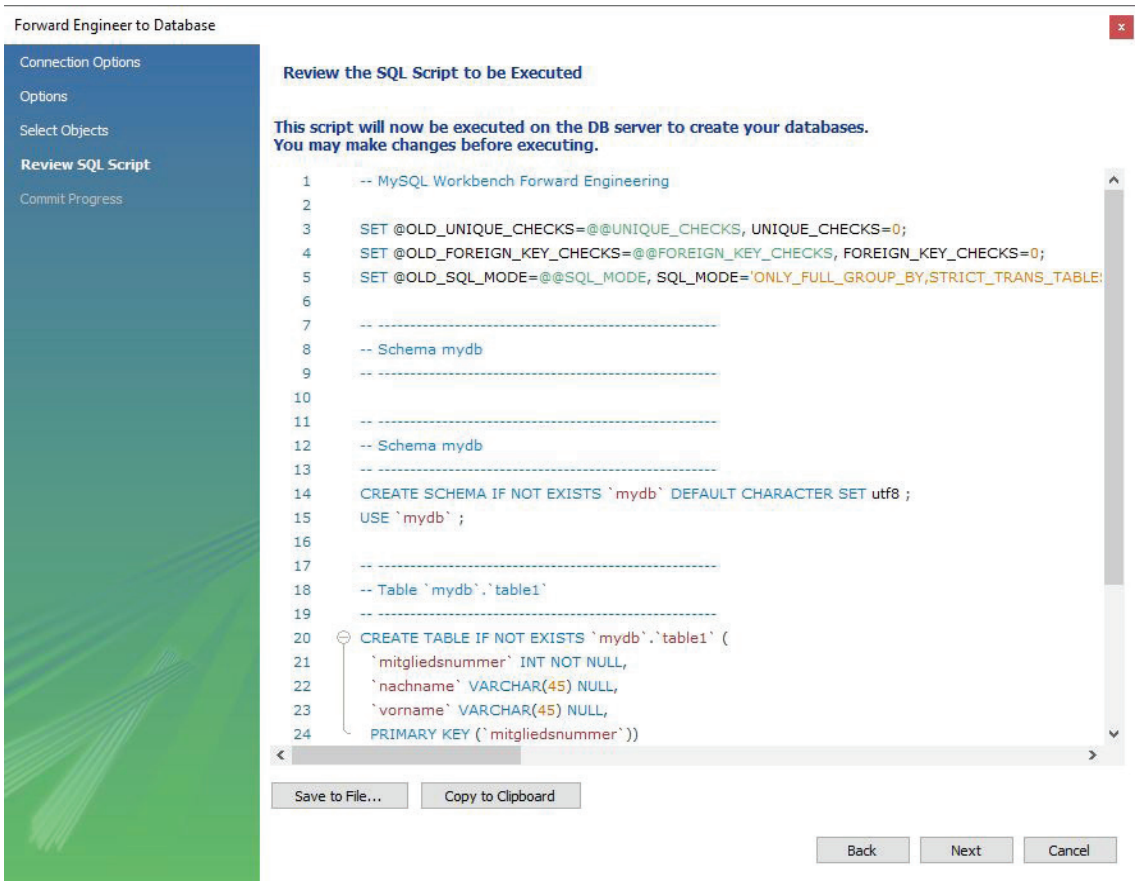

Abb. 3.15 Die SQL-Befehle für die Erzeugung des Schemas und das Einfügen der Tabelle

Abschließend erscheint noch ein weiteres Fenster, das die Ausführung der Befehle bestätigt. Damit ist die neue Tabelle erstellt. Leser, die sich für diese Funktionsweise interessieren, können an dieser Stelle gerne

noch einige Zeit verweilen und die weiteren Funktionen der Software ausprobieren. Dennoch geht dieses Lehrbuch nun nicht weiter auf diese Oberfläche ein. Teilweise sind die Befehle zwar intuitiv. Doch an vielen Stellen ist es auch notwendig, über gute Kenntnisse über die Struktur und die Funktionsweise einer Datenbank zu verfügen. Deshalb ist es sinnvoll, diese Fähigkeiten von Grund auf zu erwerben, indem man die *Datenbanksprache* SQL erlernt.

Um SQL innerhalb der Workbench zu verwenden, ist eine andere Benutzeroberfläche vorhanden, die wir fortan für dieses Buch verwenden. Um diese zu erreichen, ist es sinnvoll, das Programm erst einmal zu schließen und daraufhin erneut aufzurufen. Nun klicken wir wieder in der Menüleiste auf „File", wählen diesen Mal aber „New Query Tab" aus. Alternativ dazu ist es sinnvoll, sich gleich den Shortcut für diese Aufgabe zu merken: Strg + T. Daraufhin erscheint das Fenster, das in Abbildung 3.13 zu sehen ist.

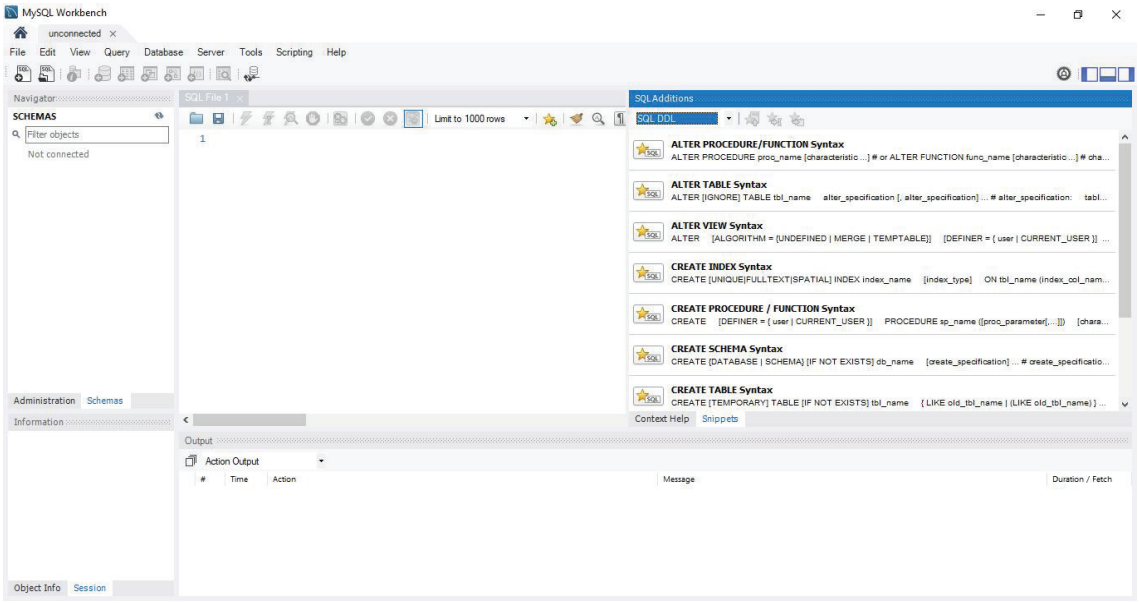

Abb. 3.13 Die Oberfläche für die Eingabe von SQL-Befehlen

Bevor wir uns der Eingabe der Befehle widmen, ist es auch hier notwendig, sich mit der Datenbank zu verbinden. Dazu wählen wir in der Menüleiste „Database" und anschließend „Connect to Database" aus – oder wir verwenden den Shortcut Strg + U. Im Fenster, das sich daraufhin öffnet, bestätigen wir einfach die gegebenen Einstellungen. Im

nächsten Fenster müssen wir dann wieder unser Passwort eingeben – so wie in Abbildung 3.14 zu sehen. Es ist wichtig, darauf zu achten, dass wir diese Verbindung jedes Mal aufs Neue herstellen müssen, wenn wir das Programm neu öffnen.

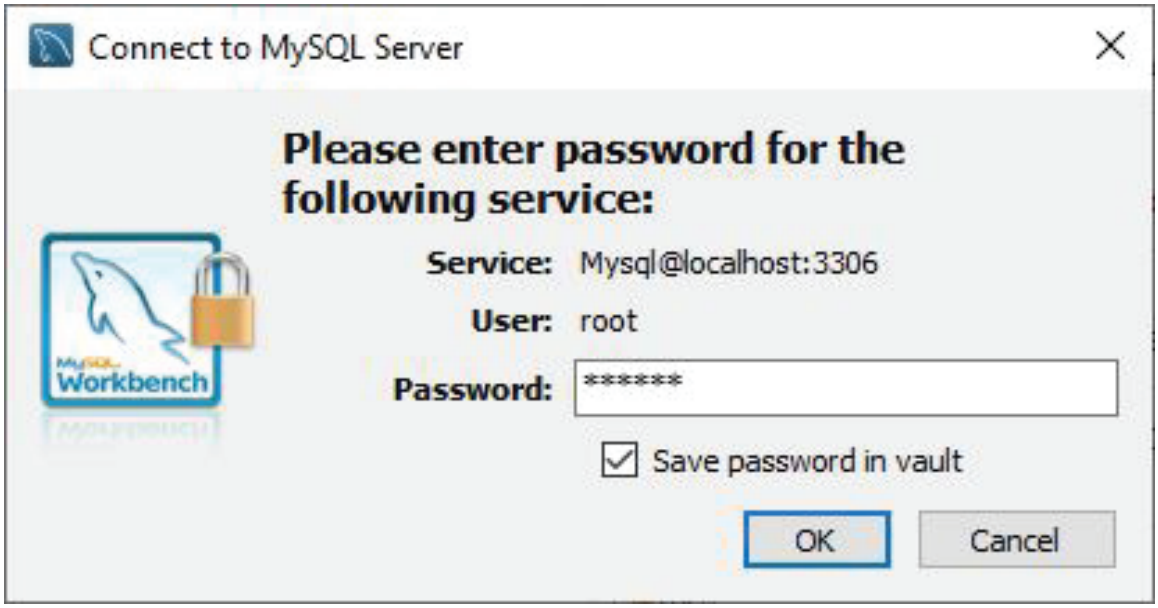

Abb. 3.14 Die Eingabe des Passworts

Im großen Feld in der Mitte der Oberfläche ist es nun möglich, die SQL-Befehle einzugeben. Allerdings erhalten wir auch hierbei eine Hilfestellung. Auf der rechten Seite sind verschiedene SQL-Befehle aufgeführt. Wenn wir uns nicht mehr genau daran erinnern, wie diese anzuwenden sind, ist es möglich, hier nachzuschauen. Das soll nun an einem Beispiel verdeutlicht werden. Bei der Vorstellung des Command Line Clients und der MySQL Shell haben wir bereits einen Befehl verwendet, um die vorhandenen Datenbanken aufzulisten. Wenn wir uns nun nicht mehr genau daran erinnern, wie dieser aufgebaut ist, können wir auf der rechten Seite nachschauen. Im oberen Bereich befindet sich ein Drop-Down-Menü – so wie dies in Abbildung 3.15 zu sehen ist.

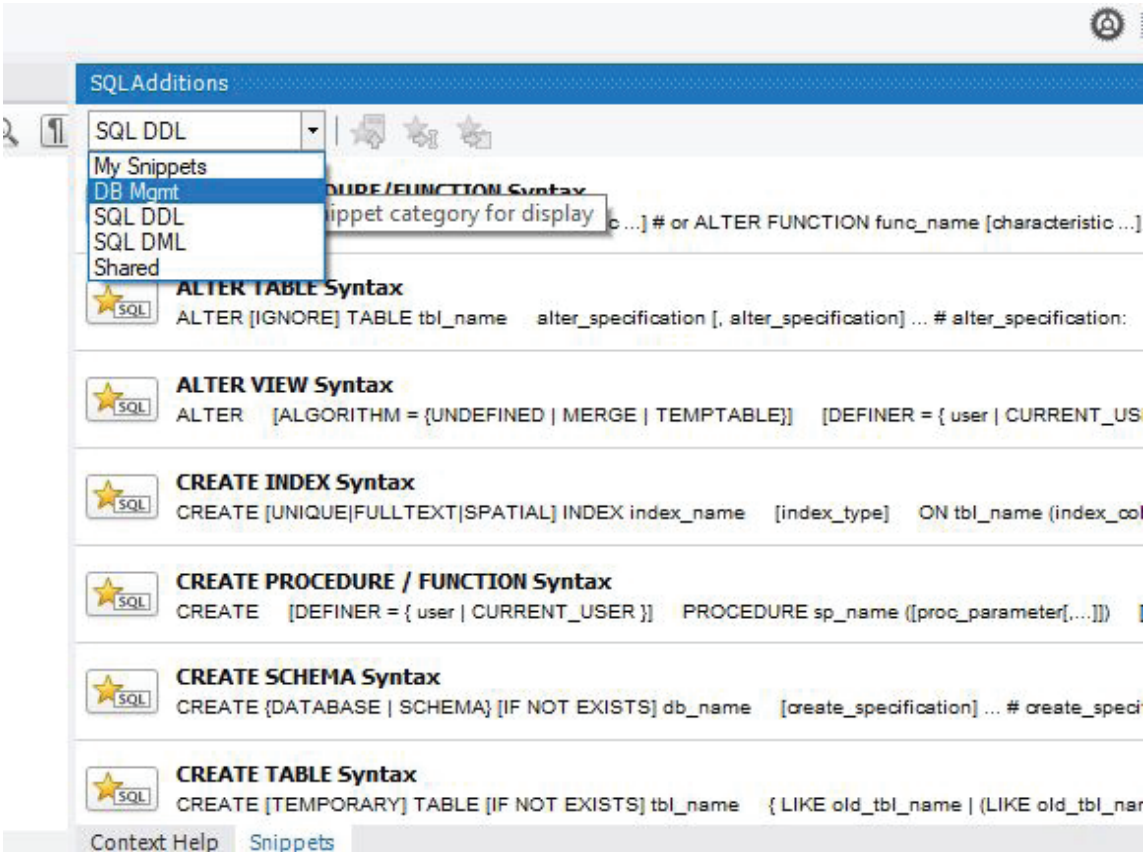

Abb. 3.15 Die Auswahl der Befehle für das Datenbank-Management

Die Befehle sind hierbei in mehrere Gruppen eingeteilt. Die Bedeutung der entsprechenden Begriffe wird im weiteren Verlauf des Buchs noch erklärt. Für unser erstes Beispiel wählen wir den Begriff DB Mgmt aus. Dieser steht für Database Management und enthält die Befehle für die Verwaltung der Datenbank.

Wenn wir hier etwas nach unten scrollen, erscheint der Eintrag SHOW DATABASE Syntax. Dieser Befehl sollte uns bereits bekannt vorkommen. Der Ausdruck Syntax steht für die Regeln für den Aufbau der Befehle. Wenn wir diesen Ausdruck doppelt anklicken, erscheint die Information, die in Abbildung 3.16 zu sehen ist.

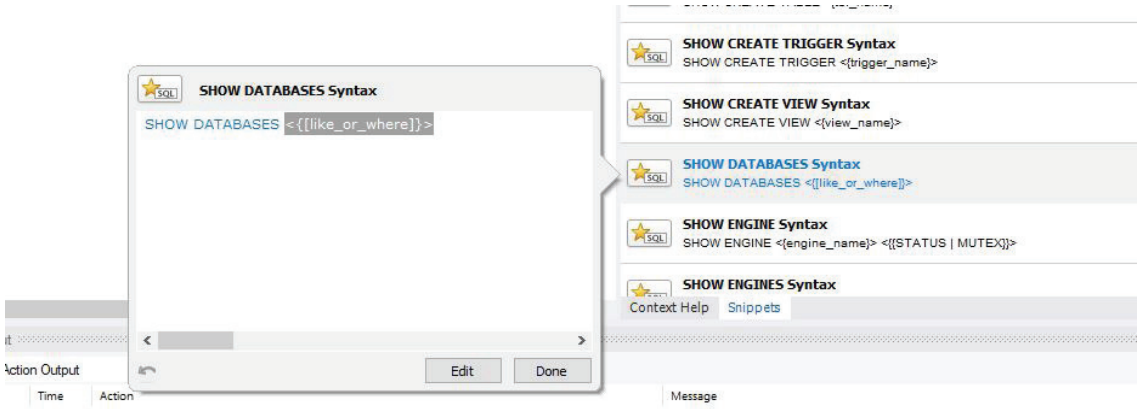

Abb. 3.16 Die Hilfestellung der MySQL Workbench

Hier sehen wir, dass der Befehl `SHOW DATABASES` lautet. Danach erscheinen in einer spitzen Klammer optionale Zusätze. Darum müssen wir uns vorerst jedoch nicht kümmern. Stattdessen kopieren wir den Befehl, schließen daraufhin das Hilfefenster und fügen ihn in das Fenster in der Mitte ein. Anschließend ist es nur noch notwendig, das Semikolon einzufügen.

Um den Befehl auszuführen, kommen mehrere Möglichkeiten infrage. Zum einen kann man auf die Blitz-Symbole über dem Eingabefenster klicken. Das erste von ihnen führt alle im Fenster vorhandenen Befehle aus. In der Workbench ist es nicht nur möglich, die Befehle einzeln auszuführen. Darüber hinaus kann man mehrere Befehle nacheinander in das Fenster einfügen. Der Klick auf das entsprechende Symbol führt dazu, dass diese dann gemeinsam ausgeführt werden. Alternativ dazu ist es auch möglich, einen bestimmten Bereich zu markieren. In diesem Fall wird nur der markierte Bereich ausgeführt.

Der zweite Blitz ist mit dem Cursor-Symbol versehen. Dieser führt dazu, dass nur ein einzelner Befehl ausgeführt wird – derjenige, in dem sich gerade der Cursor befindet. Das ist ideal, um die Befehle nacheinander abzuarbeiten. Daher kommt diese Schaltfläche besonders häufig zum Einsatz.

Zu beiden Schaltflächen bestehen noch alternative Wege für die Ausführung. Beispielsweise ist es möglich, die Menüleiste zu verwenden – über den Menüpunkt „Query" und anschließend über „Execute (All or Selection)" oder „Execute Current Statement". Da es sehr häufig not-

wendig ist, die SQL-Befehle auszuführen, ist es jedoch empfehlenswert, sich hierfür gleich die passenden Shortcuts zu merken. Das beschleunigt die Arbeit mit SQL deutlich. Diese sind Strg + Umschalttaste + Eingabetaste für die Ausführung aller Befehle und Strg + Eingabetaste für den aktuellen Befehl.

Wenn wir unseren SQL-Befehl auf eine dieser Weisen ausgeführt haben, erscheint wieder eine Liste mit den vorhandenen Datenbanken. Diese ist in Abbildung 3.17 zu sehen. Ganz unten erscheint außerdem eine kurze Information zum Ergebnis der Abfragen. Das ist insbesondere dann wichtig, wenn es zu einer fehlerhaften Eingabe kommt. Anhand der entsprechenden Nachricht ist es möglich, nachzuvollziehen, welcher Fehler aufgetreten ist.

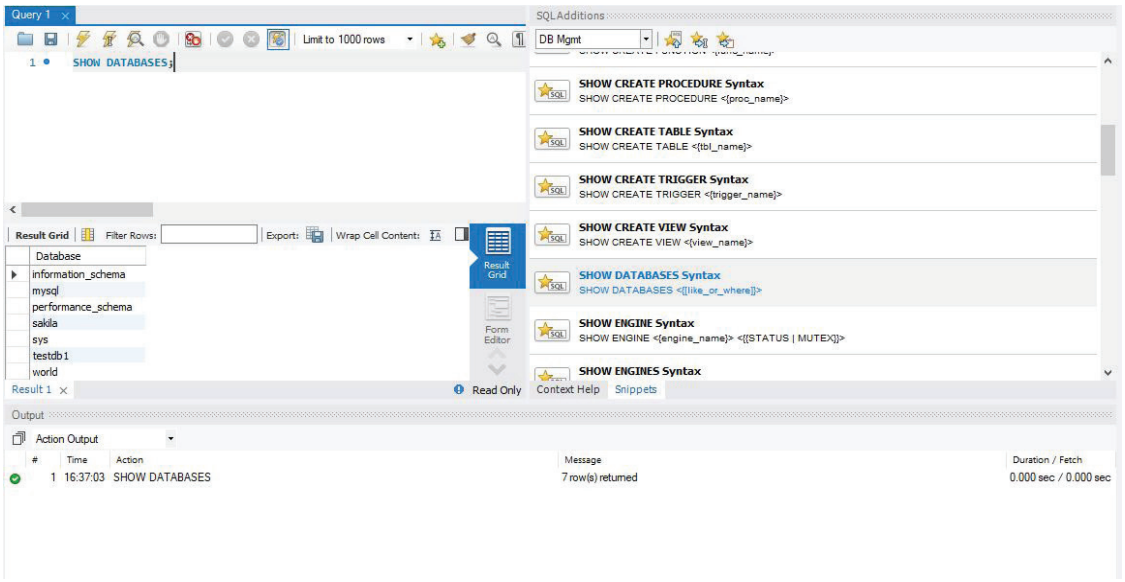

Abb. 3.17 Das Ergebnis der Ausführung des Befehls