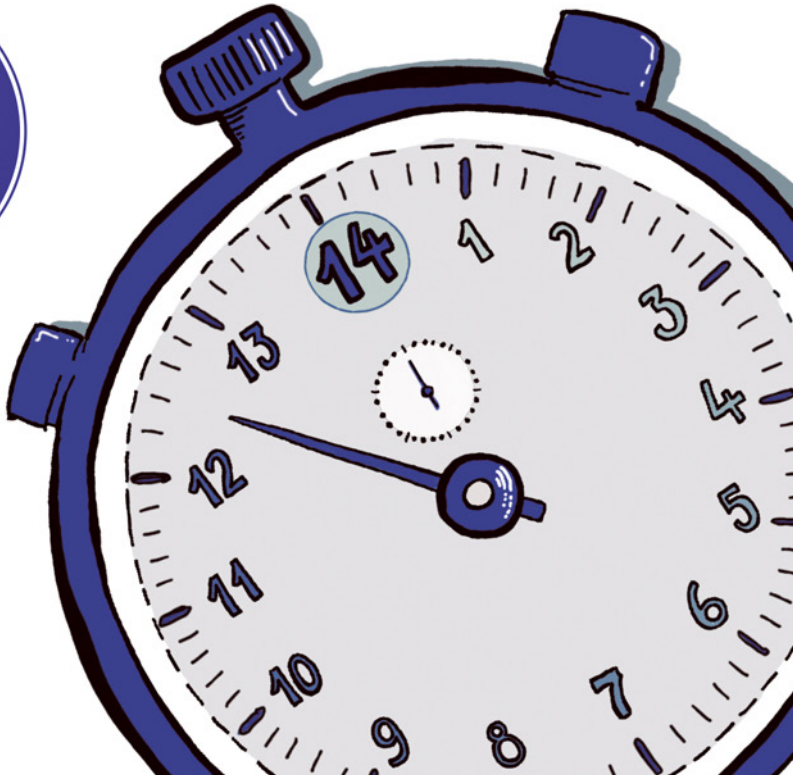
Erik Schönwälder

SQL Schnelleinstieg

Datenbanken abfragen und verwalten in 14 Tagen

Einfach und ohne Vorkenntnisse

Zahlreiche Praxisbeispiele



mitp

Hinweis des Verlages zum Urheberrecht und Digitalen Rechtemanagement (DRM)

Liebe Leserinnen und Leser,

dieses E-Book, einschließlich aller seiner Teile, ist urheberrechtlich geschützt. Mit dem Kauf räumen wir Ihnen das Recht ein, die Inhalte im Rahmen des geltenden Urheberrechts zu nutzen. Jede Verwertung außerhalb dieser Grenzen ist ohne unsere Zustimmung unzulässig und strafbar. Das gilt besonders für Vervielfältigungen, Übersetzungen sowie Einspeicherung und Verarbeitung in elektronischen Systemen.

Je nachdem wo Sie Ihr E-Book gekauft haben, kann dieser Shop das E-Book vor Missbrauch durch ein digitales Rechtemanagement schützen. Häufig erfolgt dies in Form eines nicht sichtbaren digitalen Wasserzeichens, das dann individuell pro Nutzer signiert ist. Angaben zu diesem DRM finden Sie auf den Seiten der jeweiligen Anbieter.

Beim Kauf des E-Books in unserem Verlagsshop ist Ihr E-Book DRM-frei.

Viele Grüße und viel Spaß beim Lesen,

Ihr mitp-Verlagsteam

mitp

Erik Schönwälder

SQL

Schnelleinstieg

Datenbanken abfragen
und verwalten in 14 Tagen

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über https://portal.dnb.de/opac.htm abrufbar.

ISBN 978-3-7475-0869-5
1. Auflage 2025

www.mitp.de
E-Mail: mitp-verlag@lila-logistik.com
Telefon: +49 7953 / 7189 - 079
Telefax: +49 7953 / 7189 - 082

© 2025 mitp Verlags GmbH & Co. KG, Augustinusstr. 9a, DE 50226 Frechen

Dieses Werk, einschließlich aller seiner Teile, ist urheberrechtlich geschützt. Jede Verwertung außerhalb der engen Grenzen des Urheberrechtsgesetzes ist ohne Zustimmung des Verlages unzulässig und strafbar. Dies gilt insbesondere für Vervielfältigungen, Übersetzungen, Mikroverfilmungen und die Einspeicherung und Verarbeitung in elektronischen Systemen.
Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.

Lektorat: Janina Vervost
Sprachkorrektorat: Jürgen Benvenuti
Covergestaltung: Janina Vervost, Christian Kalkert
Satz: Petra Kleinwegen

Inhalt

Die ersten Schritte – das Fundament einer jeden Datenbankabfrage

Die Datenflut bezwingen – Daten mit dem WHERE-Befehl filtern

Gruppierung von Daten und mit Daten rechnen

Verknüpfen von mehreren Tabellen

Verschachteln von Datenbankabfragen und Datenmanipulation

Verknüpfen von mehreren Ergebnissen

Erstellen der ersten eigenen Datenbank

Was nicht in eine Datenbank gehört – Einschränkungen definieren

Eine bestehende Datenbank mit ihrem Inhalt anpassen

Wer darf eigentlich was? – Rechteverwaltung und Zugriffskontrolle

SQL für Fortgeschrittene – Aktion und Reaktion

Ausblick – Gibt es noch mehr?

Einleitung

Programmieren lernen in 14 Tagen

Mit diesem Buch haben Sie sich für einen einfachen, praktischen und fundierten Einstieg in die Welt der Programmierung entschieden. Sie lernen ohne unnötigen Ballast alle Grundlagen, um SQL effektiv in Ihrem Berufs- und Interessensgebiet einzusetzen.

Wenn Sie Zeit genug haben, können Sie jeden Tag ein neues Kapitel durcharbeiten und so innerhalb von zwei Wochen den Umgang mit SQL erlernen. Alle Erklärungen sind leicht verständlich formuliert und setzen keine Vorkenntnisse voraus. Am besten lesen Sie das Buch neben der Computertastatur und probieren die Codebeispiele und Übungen gleich aus.

Der Aufbau des Buchs

Das Buch ist in zwei Hauptteile gegliedert: Datenbankabfragen und Datenbankerstellung. Bevor es allerdings mit dem ersten Teil startet, werden in den einführenden Kapiteln alle wichtigen Grundlagen erklärt: die Funktionsweise von Datenbanken, die Integration von SQL sowie die Installation und Einführung der notwendigen Software. Ausgerüstet mit dem gelernten Fundament, legt das erste Kapitel den Fokus auf das Abfragen von Daten aus einer Datenbank. Schritt für Schritt lernen Sie hier, wie Abfragen formuliert werden, indem Sie Daten filtern, gruppieren oder auch verknüpfen.

Der zweite Teil des Buchs widmet sich der praktischen Erstellung einer eigenen Datenbank samt ihrer Tabellen und Daten. Neben dem gesamten Erstellungsprozess wird hierbei auch ein besonderes Augenmerk auf Sicherheitskonzepte gelegt, wie etwa ein gelungenes Benutzer- und Rechtemanagement sowie die Wartung einer Datenbank durch das Aktualisieren der zugrunde liegenden Strukturen und Daten. Abschließend bieten die letzten beiden Kapitel wertvolle Hinweise und weiterführende Ansätze, wie Sie Ihre SQL-Kenntnisse nach dem Schnelleinstieg weiter vertiefen können.

Auch wenn im Buch ein besonderer Fokus auf die beiden Datenbanksysteme MySQL und PostgreSQL gelegt wird, kann das erlernte Wissen ebenfalls auf andere Systeme übertragen werden.

Ihr Tagespensum schließt mit praktischen Übungen, in denen Sie Ihr neu gewonnenes Wissen vertiefen können. Die Lösungen zu diesen Übungen finden Sie am Ende eines jeden Kapitels.

Am Ende des Buchs finden Sie ein Stichwortverzeichnis, das Ihnen hilft, bestimmte Inhalte schneller zu finden.

Beispieldatenbank

Um stets einen Praxisbezug zu gewährleisten, arbeitet das Buch mit einer Beispieldatenbank, die während der einzelnen Kapitel und auch in den Übungen verwendet wird.

Die Beispieldatenbank sowie die Lösungen zu den Übungen stehen Ihnen auf der Webseite des Verlags unter *www.mitp.de/0868* zum Download zur Verfügung.

Eine nähere Beschreibung der Beispieldatenbank und wie sie in MySQL und PostgreSQL eingepflegt werden kann, finden Sie im Kapitel 2.

Fragen und Feedback

Unsere Verlagsprodukte werden mit großer Sorgfalt erstellt. Sollten Sie trotzdem einen Fehler bemerken oder eine andere Anmerkung zum Buch haben, freuen wir uns über eine direkte Rückmeldung an *lektorat@mitp.de*.

Falls es zu diesem Buch bereits eine Errata-Liste gibt, finden Sie diese unter *www.mitp.de/0868* im Reiter Downloads.

Wir wünschen Ihnen viel Erfolg und Spaß bei der Programmierung mit SQL!

1 Einstieg in die Welt der Daten

»Daten« – kaum ein anderer Begriff erhält so viel Aufmerksamkeit in den Medien. Ob im Zusammenhang mit Datenschutz, Datensicherheit oder Big Data, Daten sind allgegenwärtig. Nicht zu Unrecht kommen infolgedessen zahlreiche Fragen auf: Warum sind Daten so wichtig? Wie können sie abgespeichert werden? Und wie werden sie eigentlich verwaltet?

In diesem Kapitel erhalten Sie eine Einführung in die Welt der Daten. Dabei werden nicht nur die oben angesprochenen Fragen beantwortet, sondern Sie bekommen auch elementare Grundlagen an die Hand, welche für den weiteren Verlauf des Buchs essenziell sind.

1.1 Wozu benötigen wir Datenbanken?

Personenbezogene Daten sind das Gold der heutigen Zeit. Einmal analysiert, geben sie Unternehmen wie Google, Meta oder Amazon die Möglichkeit, personalisierte Werbung oder Produktempfehlungen zu schalten. Auch kleinere Unternehmen setzen Daten gewinnbringend ein, um beispielsweise ihre Arbeitsabläufe zu optimieren oder gezielter Kunden anzuwerben. Unabhängig vom betrachteten Beispiel zeigt sich stets eine zentrale Gemeinsamkeit: die Verwendung von Datenbanken. Mithilfe von Datenbanken können die relevanten Daten gesammelt und gemeinsam abgespeichert werden. Weiterhin bieten sie die Option, abgelegte Daten je nach Art und Umfang der Analyse zu verwalten und abzurufen.

Umgangssprachlich formuliert ist eine Datenbank also ein Ort, an welchem Daten aller Art zentral gespeichert, verwaltet und wieder abgerufen werden können. Die Herkunft der Daten spielt dabei keine Rolle. Während beispielsweise eine Trockenbaufirma Datenbanken für ihre Kunden-, Mitarbeiter- und

Lieferantendaten nutzt, legt eine Shopping-Webseite hier ihre Produkt- und Bestelldaten ab.

1.2 Bestandteile einer Datenbank

Tatsächlich ist Datenbank nicht gleich Datenbank. Eine Datenbank besteht aus mehreren Teilen, deren Bezeichnungen oft durcheinandergeworfen werden. Aber beginnen wir von vorn. Ein *Datenbanksystem*, was fälschlicherweise oft als Datenbank bezeichnet wird, setzt sich aus genau zwei Teilen zusammen:

- der *Datenbank*, auch als *Datenbasis* bezeichnet
- dem *Datenbank-Management-System* (oft abgekürzt mit DBMS)

In der Datenbank werden die Daten physisch abgelegt. Ein direktes Arbeiten auf diesen Daten ist mithilfe der Datenbank jedoch nicht möglich. Hierfür ist eine spezielle Software notwendig, das Datenbank-Management-System. In der Abbildung 1.1 ist dieser Zusammenhang genauer veranschaulicht.

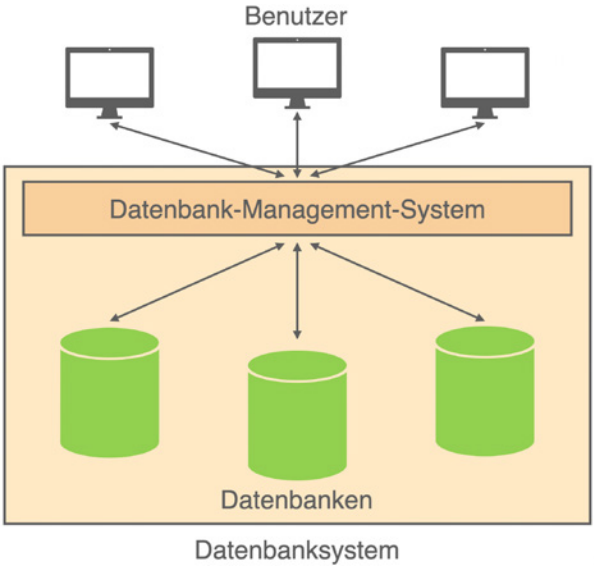


Abb. 1.1: Bestandteile eines Datenbanksystems

Möchte ein Benutzer seine Daten also mithilfe eines Datenbanksystems ablegen, so kann er nicht direkt auf die Datenbank zugreifen. Diese ist lediglich für das Speichern der Daten zuständig und eine Interaktion ist nur über eine

Schnittstelle möglich. Das Datenbank-Management-System repräsentiert diese Schnittstelle und ist für die Verwaltung der Daten verantwortlich. Sollen Daten eingepflegt, modifiziert oder gegebenenfalls gelöscht werden, so interagiert der Benutzer über das Datenbank-Management-System (kurz DBMS) mit der Datenbank.

Neben dem Abspeichern, Verwalten und Abrufen von Daten ist ein DBMS noch für weitere Aufgaben zuständig. Beispielsweise muss es sich auch um die Konsistenz einer Datenbank kümmern. Angenommen eine Firma hat all ihre Daten in einem Datenbanksystem abgelegt. Weiterhin wurde die Regel aufgestellt, dass jeder Mitarbeiter am Ende des Monats mindestens 1.000 Euro Gehalt ausgezahlt bekommt. Die Aufgabe des DBMS ist es nun sicherzustellen, dass diese Regel niemals verletzt wird. Weder das Hinzufügen eines neuen Mitarbeiters mit einem Monatslohn von weniger als 1.000 Euro noch die Reduzierung des Monatslohns unter die Mindestgrenze eines bestehenden Mitarbeiters ist also gestattet.

1.3 Einblick in die verschiedenen Datenbanktechnologien

So wie es etliche Anwendungsfälle gibt, in welchen Datenbanksysteme eingesetzt werden können, so gibt es auch etliche Arten bzw. Technologien, um ein Datenbanksystem aufzubauen. Das wohl bekannteste und in diesem Buch auch relevante Datenbanksystem ist das *relationale Datenbanksystem*. Wie der Name schon vermuten lässt, werden in einem relationalen Datenbanksystem relationale Daten, also Daten in Tabellenform, abgespeichert und alle Daten durch Relationen bzw. Tabellen beschrieben. Die Begriffe *Tabelle* und *Relation* werden in diesem Kontext oft synonym verwendet.

Eine Tabelle besteht aus mehreren Spalten, oftmals auch als *Attribute* bezeichnet, die die Daten beschreiben. Die Zeilen einer Tabelle, sogenannte *Entitäten* oder *Tupel*, stellen dann die Ausprägungen bzw. die einzelnen Bausteine der Daten dar. In der Abbildung 1.2 ist dieses abstrakte Konzept anhand eines Beispiels präsentiert.

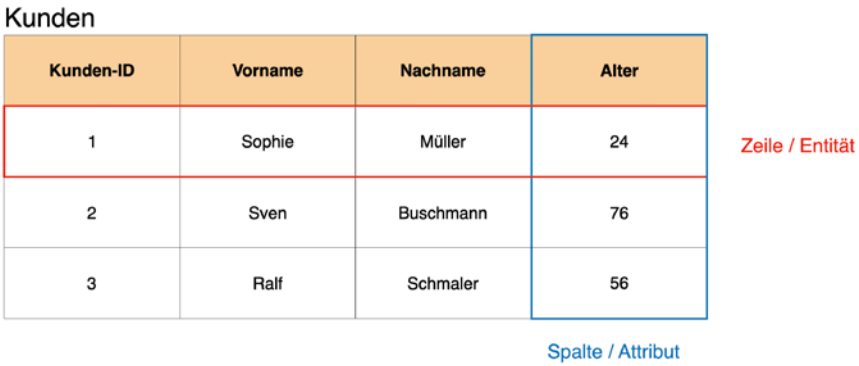


Abb. 1.2: Beispiel einer relationalen Datenbank – Kunden-Relation

Die Tabelle bzw. Relation namens »Kunden« beinhaltet die Kundeninformationen eines Onlineshops. Zu einem Kunden werden die Attribute »Vorname«, »Nachname« und »Alter« abgespeichert. In der Tabelle sind diese Eigenschaften durch Spalten repräsentiert. Weiterhin ist jedem Kunden eine eindeutige »Kunden-ID« zugeordnet. Das Konzept hinter den IDs wird im Abschnitt 1.5 detailliert erläutert und ist an dieser Stelle noch unbedeutend. Die spezifischen Kunden, folglich die einzelnen Individuen, werden über die Zeilen ausgedrückt. Beispielsweise beschreibt die erste Zeile die 24-jährige Sophie Müller mit der ID 1 oder die dritte Zeile den 56-jährigen Ralf Schmaler mit der ID 3.

Zusammengefasst werden die Merkmale eines Kunden, oder allgemeiner gesprochen einer Entität, durch die einzelnen Spalten einer Tabelle beschrieben. Die individuellen Ausprägungen der Daten, im gezeigten Beispiel die Kunden, werden dann über die Zeilen einer Tabelle dargestellt.

Ein relationales Datenbanksystem besitzt allerdings in den seltensten Fällen nur eine Tabelle. Meistens besteht ein solches System aus Dutzenden Tabellen, die untereinander durch Beziehungen verknüpft sind. So kann es zum Beispiel sein, dass der Onlineshop aus der letzten Abbildung neben der Kunden-Relation noch eine weitere Bestellungen-Relation mit den Attributen »Bestell-ID«, »Kunden-ID«, »Produktname« und »Menge« beinhaltet.

Bestellungen

Bestell-ID	Kunden-ID	Produktname	Menge
1	3	T-Shirt	4
2	6	Hose	1
3	4	Socken	10

Abb. 1.3: Bestellungen-Relation

Die in Abbildung 1.3 dargestellte Bestellungen-Relation dokumentiert alle getätigten Bestellungen. Dabei gibt die Kunden-ID Auskunft über den Kunden, der die Bestellung aufgegeben hat. Ebenso erfasst die Relation den Produktnamen, welches Produkt bestellt wurde, sowie die Menge, in der das Produkt geordert wurde. Durch die Kunden-ID stehen die Kunden- und Bestellungen-Relationen nun miteinander in Beziehung. Durch den Zusammenschluss gleicher IDs können Tupel bzw. Entitäten beider Relationen miteinander verknüpft werden, wodurch zusätzliche Informationen abgespeichert werden.

Laut der ersten Zeile der Bestellungen-Relation hat der Kunde mit der ID 3 vier T-Shirts bestellt. Mithilfe der ID 3 können Sie nun leicht in der Kunden-Relation nach der entsprechenden Entität suchen. Resultierend stellen Sie fest, dass der 56-jährige Ralf Schmaler der Absender der entsprechenden Bestellung ist.

Neben relationalen Datenbanksystemen existiert noch eine Vielzahl weiterer Systeme. Typischerweise werden diese unter dem Begriff *NoSQL* zusammengefasst. Diese Systeme grenzen sich von der relationalen Welt ab und beschäftigen sich folglich mit Daten, die ohne eine tabellarische Struktur auskommen.

Namentlich zu erwähnen sind hier sogenannte *Document-Stores*, die schemafreie Dokumente abspeichern. Schemafrei bedeutet, dass Kunden beispielsweise nicht immer mit den gleichen Merkmalen wie in einer Tabelle ausgestattet sein müssen. So kann ein Kunde auch gern mal keinen Vor- und Nachnamen, aber dafür einen Künstlernamen besitzen oder anders als die anderen Kunden eine zusätzliche Präferenzliste mitführen, in welcher die Favoritenprodukte abgelegt sind.

In *Key-Value-Stores* dagegen werden Datenpunkte immer in Verbindung mit einem Schlüssel abgespeichert. Der Schlüssel ist zu einem späteren Zeitpunkt wichtig, um die Datenpunkte wieder abzurufen. Nicht nur ist dieses System aufgrund seiner Einfachheit sehr beliebt, es glänzt zusätzlich noch mit einem hohen Effizienz- und Flexibilitätsgrad.

Auch *Graphdatenbanken* sollten in diesem Kontext erwähnt werden. Hier werden die einzelnen Entitäten durch Knoten in einem Netzwerk aus Beziehungen repräsentiert. Insbesondere soziale Strukturen lassen sich hiermit hervorragend abbilden.

1.4 Datenbanken und SQL, wie hängt das zusammen?

Ein SQL-Buch beinhaltet neben den grundlegenden Konzepten wie der Unterscheidung verschiedener Datenbankarten oder den Bestandteilen einer Datenbank natürlich auch den essenziellsten Schwerpunkt: SQL selbst. Wie kann SQL also in den gerade erläuterten Datenbankkontext gebracht werden, und was ist SQL eigentlich?

SQL steht für »Structured Query Language« und ist eine Abfragesprache. Abfragesprachen sind ähnlich wie Programmiersprachen, nur dass sie einen geringeren Funktionsumfang besitzen und für einen spezifischen Anwendungsfall bestimmt sind. Während Programmiersprachen wie C++ oder Java quasi uneingeschränkt in verschiedensten Szenarien Anwendung finden, widmet sich eine Abfragesprache nur einer konkreten Problematik, nämlich dem Verwalten einer Datenbank. Wie bereits im Abschnitt 1.2 dargestellt, wird eine Datenbank über das Datenbank-Management-System verwaltet. Ein Datenbank-Management-System wiederum nutzt hierfür eine Abfragesprache. In relationalen Datenbanksystemen ist diese Abfragesprache standardmäßig SQL.

Zusammengefasst ist SQL also eine Abfragesprache, die zum Einfügen, Verändern, Löschen und Abrufen von Daten in einem relationalen Datenbanksystem verwendet wird.

1.5 Identifikation von Daten mit Primär- und Fremdschlüsseln

Im Abschnitt 1.3 wurden zur Erklärung eines relationalen Datenbanksystems folgende Beispielrelationen präsentiert:

Kunden

Kunden-ID	Vorname	Nachname	Alter
1	Sophie	Müller	24
2	Sven	Buschmann	76
3	Ralf	Schmaler	56

Bestellungen

Bestell-ID	Kunden-ID	Produktname	Menge
1	3	T-Shirt	4
2	6	Hose	1
3	4	Socken	10

Abb. 1.4: Kunden- und Bestellungen-Relation

1.5.1 Primärschlüssel

Die Spalte »Kunden-ID« spielt eine bedeutende Rolle, da sie als Primärschlüssel in der Kunden-Relation fungiert. Als Primärschlüssel wird in einer Relation ein Attribut bzw. eine Gruppe von Attributen bezeichnet, die die Entitäten der Relation, folglich die Tabellenzeilen, eindeutig kennzeichnen. In der gegebenen Kunden-Relation repräsentiert beispielsweise die Kunden-ID einen solchen Primärschlüssel. Mithilfe dieses Attributs kann unkompliziert auf die einzelnen Zeilen der Kunden-Relation zugegriffen werden. Wenn beispielsweise die Kunden-ID 3 gegeben ist, so kann mit dieser eindeutig auf den 56-jährigen Ralf Schmaler verwiesen werden.

Aber warum überhaupt eine separate ID? Könnte nicht einfach der Vor- und Nachname eines Kunden als Primärschlüssel dienen? Nun ja, es ist natürlich denkbar, dass mehrere Kunden denselben Namen besitzen. Wenn folglich mehrere »Ralf Schmaler« in der Relation abgespeichert werden, kann mit Vor- und Nachnamen nicht mehr eindeutig auf eine Entität bzw. eine Zeile gezeigt werden. Welcher Ralf Schmaler ist denn gemeint? Um diesem Problem aus dem Weg zu gehen, wird üblicherweise ein künstlicher Primärschlüssel wie eine ID oder eine Nummer als zusätzliches Attribut herangezogen. Wird schlussfolgernd eine neue Entität in die entsprechende Relation eingepflegt, so wird dieser Entität eine eindeutige und noch nicht verwendete ID zugeordnet.

Somit kann sichergestellt werden, dass jede Entität ohne Doppeldeutigkeiten über den Primärschlüssel identifiziert wird.

1.5.2 Zusammengesetzte Primärschlüssel

Wie bereits in der Definition eines Primärschlüssels erwähnt, kann ein Primärschlüssel auch aus mehr als einem Attribut bestehen. Dies ist insbesondere dann erforderlich, wenn ein einzelnes Attribut nicht zur Beschreibung der gesamten Relation ausreicht.

Bibliothek

Ausleih-ID	Benutzer-ID	Datum	Titel
100	19	12.10.2024	Sofort zum SQL-Profi werden
100	19	12.10.2024	Python für Anfänger
201	6	14.10.2024	Wie funktioniert das Internet

Abb. 1.5: Bibliothek-Relation

In der in Abbildung 1.5 dargestellten »Bibliothek-Relation« werden Ausleihvorgänge gespeichert. Dabei besitzt jeder Vorgang die Attribute:

- »Ausleih-ID«, um den Vorgang eindeutig zu identifizieren
- »Benutzer-ID«, um den Benutzer, der Bücher ausleiht, zu kennzeichnen
- »Datum«, um die Zeit, wann der Vorgang stattgefunden hat, zu beschreiben
- »Titel«, um die geliehene Literatur zu definieren

Eine Zeile der Bibliothek-Relation beschreibt den Ausleihprozess von genau einem Buch. In der ersten Zeile leiht sich beispielsweise der Benutzer mit der ID 19 das Buch »Sofort zum SQL-Profi werden« aus. Dieser Ausleihvorgang wird dabei mit der Ausleih-ID 100 identifiziert. Unglücklicherweise reicht hier die alleinige Ausleih-ID nicht als Primärschlüssel aus. Während eines Ausleihvorgangs kann ein Benutzer natürlich mehrere Bücher gleichzeitig ausleihen, wodurch die Ausleih-ID nicht eindeutig auf eine Entität der Relation zeigt. Ist mit der ID 100 nun die erste oder die zweite Zeile gemeint?

Diese Problematik kann über verschiedene Lösungsansätze adressiert werden. Eine Möglichkeit besteht darin, eine weitere ID hinzuzufügen, die jedes

einzelne ausgeliehene Buch identifiziert. Dadurch können verschiedene IDs für ein und denselben Ausleihvorgang vergeben werden und es treten keine Duplikate auf.

Alternativ kann die Relation in ihrer aktuellen Form beibehalten werden. Statt den Primärschlüssel jedoch nur mit einem Attribut zu bilden, werden mehrere Attribute als Primärschlüssel gewählt. Resultierend ist der neue Primärschlüssel ein zusammengesetzter und besteht aus der Ausleih-ID und dem Titel. Mit einem Paar (Ausleih-ID, Titel) kann nun eindeutig auf eine Entität geschlossen werden. Während die Ausleih-ID 100 und der Titel »Sofort zum SQL-Profi werden« auf die erste Zeile verweist, beschreibt das Paar 100 und »Python für Anfänger« die zweite Zeile.

Der zusammengesetzte Primärschlüssel (Ausleih-ID, Titel) funktioniert nur unter der Annahme, dass ein Benutzer kein Buch doppelt in einem Ausleihvorgang ausleihen kann. Ansonsten könnte das Paar (100, »Sofort zum SQL-Profi werden«) nicht eindeutig auf eine Entität verweisen. Um diesen Fall zu lösen, müsste eine weitere ID hinzugefügt werden.

1.5.3 Fremdschlüssel

Neben der Idee des Primärschlüssels existiert das Konzept des Fremdschlüssels. Ein Fremdschlüssel ist ein Attribut einer Relation, welches gleichzeitig als Primärschlüssel einer anderen Relation fungiert. In den in der Abbildung 1.4 dargestellten Relationen ist die Kunden-ID in der Bestellungen-Relation folglich ein Fremdschlüssel, während die Bestell-ID den Primärschlüssel der Relation repräsentiert. Auf diese Weise werden mithilfe von Fremdschlüsseln Beziehungen zwischen verschiedenen Relationen etabliert.

Eine Relation kann beliebig viele Fremdschlüssel beinhalten, ohne Limits. Das bedeutet, dass eine Relation sowohl keinen als auch fünf oder sogar zehn Fremdschlüssel führen kann.

2 Einrichten der Arbeitsfläche

Relationale Datenbanksysteme werden über die Abfragesprache SQL verwaltet. Sei es für das Einfügen neuer Daten oder das Aktualisieren, Löschen oder Abrufen bestehender Daten, das Arbeiten mit SQL ist unumgänglich.

Aufgrund der Vielzahl an möglichen relationalen Datenbanksystemen und auch wegen der Komplexität von SQL gestaltet sich der SQL-Einstieg meist schwierig. Um dieses Problem zu adressieren, bietet das vorliegende Kapitel Ihnen eine Übersicht zu relationalen Datenbanksystemen und ihrem Einfluss auf SQL. Weiterhin erhalten Sie eine Schritt-für-Schritt-Anleitung zur Installation und Einrichtung der Arbeitsumgebung für die beiden im Buch behandelten Datenbanksysteme PostgreSQL und MySQL.

2.1 Der SQL-Dialekt – Ist SQL immer gleich SQL?

Bislang existiert noch kein Buch, welches SQL in seiner ganzheitlichen Form erklärt und zusammenfasst. Tatsächlich hat dies auch einen klaren Grund. SQL ist nicht einheitlich. Obwohl die Abfragesprache SQL standardisiert ist, gibt es bei der Implementierung der Sprache keine Richtlinien. Verschiedene DBMS implementieren SQL auf unterschiedliche Weise und entwickeln so ein SQL mit einer eigenen Syntax, einem eigenen Funktionsumfang oder mit eigenen Performance-Optimierungen.

Auch wenn das zunächst abschreckend wirkt, sind die Unterschiede der einzelnen SQLs meist nur marginal. Üblicherweise wird deswegen auch von einem »SQL-Dialekt« gesprochen. Die einzelnen SQLs stellen keine eigene neue Sprache dar, sie sind vielmehr eine leicht abgewandelte Ausprägung des Standards.

2.1.1 Übersicht verschiedener SQLs

Da die SQL-Dialekte aus den verschiedenen Datenbanksystemen resultieren, ist es sinnvoll, einen genaueren Blick auf diese zu werfen. Im Folgenden wird eine Auswahl der populärsten relationalen Datenbanksysteme präsentiert.

- *Oracle Database* ist ein *Closed-Source*-Datenbanksystem der Firma Oracle. Closed Source bedeutet in diesem Zusammenhang, dass der Code hinter dem Datenbanksystems nicht öffentlich ist. Dieses Vorgehen ist gerade bei kommerziellen Produkten üblich und hat seine Vor- und Nachteile. Auf der einen Seite fallen bei der Nutzung von Oracle Database Lizenzgebühren an, die insbesondere für kleinere Unternehmen oft nicht tragbar sind. Weiterhin kann durch die Closed-Source-Eigenschaft nicht auf den Code zugegriffen werden, wodurch auftretende Fehler im System nicht selbst behoben oder gewünschte Features nicht selbst implementiert werden können. Auf der anderen Seite ist der Tech-Support bei Oracle sehr stark und zu jedem erdenklichen Problem wird häufig schnell und gezielt eine Lösung gefunden. Darüber hinaus ist die Wahrscheinlichkeit eines Ausfalls bei kommerziell vertriebenen Closed-Source-Produkten meist geringer und sie werden als stabiler laufend betrachtet. Neben den Closed Source bedingten Eigenschaften von Oracle Database gilt die Anwendung als relativ schwierig zu erlernen. Das Ausnutzen der Mächtigkeit des Datenbanksystems ist folglich meist Experten vorbehalten.
- *Microsoft SQL Server*, entwickelt von Microsoft, ist ebenso wie Oracle Database ein Closed-Source-Datenbanksystem und teilt deswegen ähnliche Eigenschaften. Besonders wenn bereits andere Microsoft-Produkte im Unternehmen verwendet werden, profitiert Microsoft SQL Server vom Microsoft-Ökosystem, welches eine einfache Integration der einzelnen Produkte ermöglicht. Nichtsdestotrotz fallen auch hier hohe Lizenzgebühren für die Nutzung an und die Komplexität der Anwendung erschwert das Erlernen.
- *MySQL* ist ein *Open-Source*-Datenbanksystem, das inzwischen auch zu Oracle gehört. Open Source bedeutet, dass der Code von MySQL frei zugänglich ist, zumindest fast. Oracle hat mittlerweile mehrere Lizenzmodelle für MySQL vorgestellt. Wenn Sie beispielsweise neue Features für MySQL entwickeln und diese kommerziell vertreiben möchten, ist eine kostenpflichtige MySQL-Lizenz notwendig. Neben der Open-Source-Variante von MySQL können durch Lizenzen auch Versionen mit erweitertem Funktionsumfang erworben werden. Im Vergleich zu den bisher vorgestell-

ten Datenbanksystemen ist die Komplexität von MySQL deutlich geringer, weshalb es auch wesentlich einfacher zu erlernen ist.

- *MariaDB* ist genauso wie MySQL ein Open-Source-Datenbanksystem, welches ursprünglich aus MySQL entstanden ist. Nach der Übernahme von MySQL durch Oracle haben sich der Gründer und weitere Mitarbeiter von MySQL abgespalten und gründeten MariaDB. MariaDB ist vollständig offen und ähnelt MySQL in Struktur und Aufbau sehr. Auch MariaDB gilt als relativ einfach zu erlernen.
- *PostgreSQL* ist ein Open-Source-Datenbanksystem, welches ähnlich wie MariaDB vollständig offen ist. Nicht zuletzt wegen dieser Eigenschaft und des starken Community-Supports erfreut sich PostgreSQL insbesondere in der datenbankbezogenen Forschung großer Beliebtheit. Im Vergleich zu MySQL verfügt das Datenbanksystem über einen größeren und fortgeschritteneren Funktionsumfang. Aufgrund dieser Komplexität ist das Datenbanksystem allerdings schwerer zu verstehen und zu erlernen.

Die obige Liste von Datenbanksystemen ist nur eine Auswahl. Alle Datenbanksysteme haben spezifische Vor- und Nachteile und je nach Anwendungsfall können verschiedene Systeme geeignet sein.

2.1.2 Warum eigentlich PostgreSQL und MySQL?

Dieses Buch behandelt die Datenbanksysteme PostgreSQL und MySQL. Nicht zu Unrecht kann da die Frage entstehen, wieso genau diese beiden Systeme? Insbesondere nach der eben präsentierten Übersicht sind die möglichen Alternativen schier überwältigend. Alle Datenbanksysteme haben ihre Vor- und Nachteile und vielmehr sollte eine Systemauswahl anwendungsspezifisch erfolgen. Beim Erlernen von SQL sind allerdings andere Faktoren entscheidend.

Vergleichbar ist dies mit dem Erlernen einer Programmiersprache. Welche Programmiersprachen kommen hier infrage? Sollen Sie sich lieber mit Java, C++ oder Python befassen? Die Antwort ist, es ist ziemlich egal. Wenn Sie Java erlernt haben, können Sie auch C++ und andersherum. Entscheidend ist hier das Verständnis der grundlegenden Konzepte und die sind bei jeder Sprache ähnlich. Natürlich variieren einzelne Implementierungen oder Strukturen, aber mit ein paar Google-Anfragen lassen sich diese Hürden auch überwinden.

Ähnlich verhält sich dies mit PostgreSQL und MySQL. Wenn Sie ein System beherrschen, so können Sie Ihr Wissen auch auf andere Systeme übertragen. Die Inkompatibilitäten zwischen ihnen stellen dann nur noch kleinere Hür-

den dar. Aufgrund der Open-Source-Natur von PostgreSQL und MySQL eignen sich diese Systeme besonders für den Einstieg.

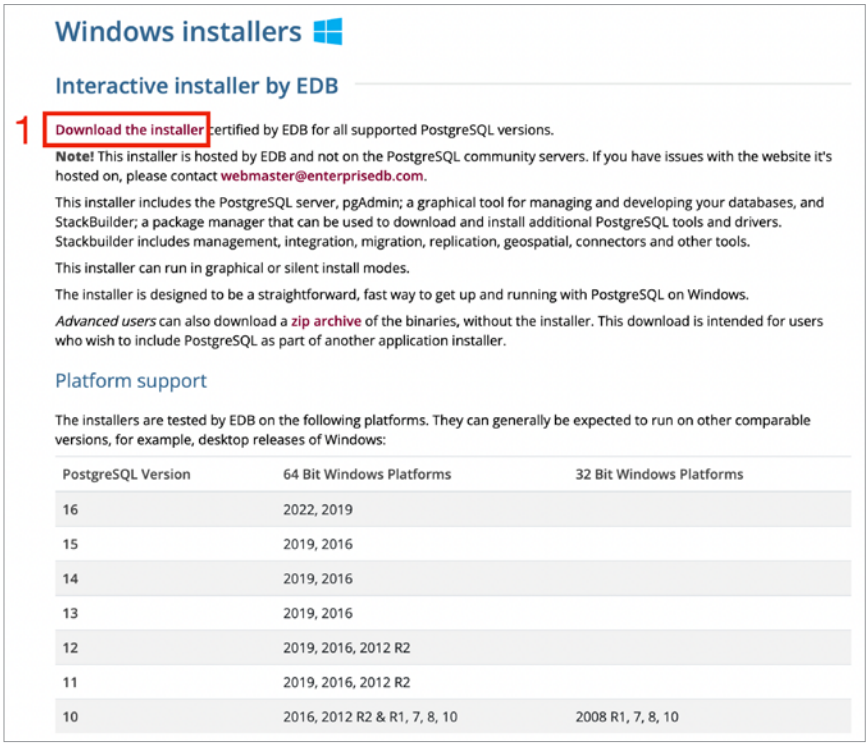
Tatsächlich empfehle ich, dass Sie sich beim Durcharbeiten des Buchs für eines der beiden Systeme entscheiden, entweder PostgreSQL oder MySQL. Besonders beim Erlernen von SQL ist ein paralleles Betrachten der Systeme eher hinderlich.

2.2 Installation von PostgreSQL

Wenn Sie sich für PostgreSQL entschieden haben, ist es notwendig, das Datenbanksystem mitsamt seiner zusätzlichen Anwendungen herunterzuladen und anschließend auf Ihrem Computer zu installieren. Der Installationsprozess unterscheidet sich zwischen den einzelnen Betriebssystemen nur marginal, weshalb Sie die folgenden Schritte unabhängig davon, ob Sie mit Windows, macOS oder Linux arbeiten, einfach durchführen können.

Über den Link *https://www.postgresql.org/download/* gelangen Sie zum PostgreSQL-Download-Portal. Auf dieser Seite können Sie Ihr entsprechendes Betriebssystem auswählen und auf der nächsten Seite ganz oben unter Download the installer, wie in **1** in Abbildung 2.1 gezeigt, die zugehörige Download-Seite erreichen. Je nach Betriebssystem steht nun der Installer mit der aktuellsten PostgreSQL-Version zum Download bereit.

Ist der Installer einmal heruntergeladen, so können Sie ihn ausführen und damit die Installation starten. Für dieses Buch sind keine speziellen Einstellungen notwendig, weshalb die einzelnen Menüpunkte durch ein einfaches Klicken auf Next bzw. Weiter übersprungen und in ihrem Standard belassen werden können. Nur zwei Menüpunkte sind hier entscheidend. Der erste, der in Abbildung 2.2 zu sehen ist, bestimmt, welche Anwendungen installiert werden sollen. Entweder wählen Sie hier jede Komponente aus, um auf den vollen Funktionsumfang von PostgreSQL zuzugreifen, oder Sie nehmen den Haken für die Stack-Builder-Komponente heraus. Der Stack Builder ist für das Herunterladen und Installieren von PostgreSQL-Erweiterung nützlich, was in diesem Buch jedoch keine weitere Rolle spielen soll.



Windows installers

Interactive installer by EDB

1 Download the installer certified by EDB for all supported PostgreSQL versions.

Note! This installer is hosted by EDB and not on the PostgreSQL community servers. If you have issues with the website it's hosted on, please contact webmaster@enterprisedb.com.

This installer includes the PostgreSQL server, pgAdmin; a graphical tool for managing and developing your databases, and StackBuilder; a package manager that can be used to download and install additional PostgreSQL tools and drivers. Stackbuilder includes management, integration, migration, replication, geospatial, connectors and other tools.

This installer can run in graphical or silent install modes.

The installer is designed to be a straightforward, fast way to get up and running with PostgreSQL on Windows.

Advanced users can also download a zip archive of the binaries, without the installer. This download is intended for users who wish to include PostgreSQL as part of another application installer.

Platform support

The installers are tested by EDB on the following platforms. They can generally be expected to run on other comparable versions, for example, desktop releases of Windows:

PostgreSQL Version	64 Bit Windows Platforms	32 Bit Windows Platforms
16	2022, 2019	
15	2019, 2016	
14	2019, 2016	
13	2019, 2016	
12	2019, 2016, 2012 R2	
11	2019, 2016, 2012 R2	
10	2016, 2012 R2 & R1, 7, 8, 10	2008 R1, 7, 8, 10

Abb. 2.1: Das Download-Portal von PostgreSQL

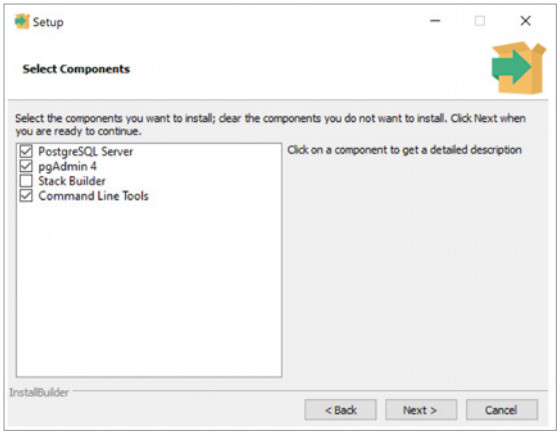


Abb. 2.2: Installation der einzelnen PostgreSQL-Komponenten