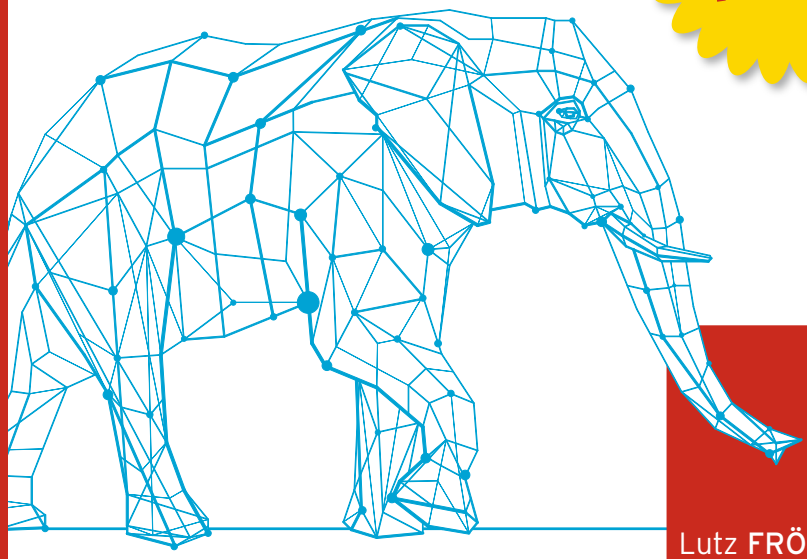
Update inside



Lutz FRÖHLICH

PostgreSQL

PRAXISBUCH FÜR ADMINISTRATOREN UND ENTWICKLER

Zu Version PostgreSQL 14

HANSER

Fröhlich

PostgreSQL

Ihr Plus – digitale Zusatzinhalte!

Auf unserem Download-Portal finden Sie zu diesem Titel kostenloses Zusatzmaterial.

Geben Sie auf **plus.hanser-fachbuch.de** einfach diesen Code ein:

plus-92Fgh-ci4R7

Bleiben Sie auf dem Laufenden!

Unser **Computerbuch-Newsletter** informiert Sie monatlich über neue Bücher und Termine. Profitieren Sie auch von Gewinnspielen und exklusiven Leseproben. Gleich anmelden unter:

www.hanser-fachbuch.de/newsletter

Update inside.

Mit unserem kostenlosen Update-Service zum Buch erhalten Sie aktuelle Infos zu den neuen Versionen von PostgreSQL.

Und so funktioniert es:

1. Registrieren Sie sich unter:
   www.hanser-fachbuch.de/postgresql-update
2. Geben Sie diesen Code ein:

2WHK-SeoTr-bF5x

Der Update-Service läuft bis Mai 2024.
Als registrierter Nutzer werden Sie in diesem Zeitraum persönlich per E-Mail informiert, sobald ein neues Buch-Update zum Download verfügbar ist.

Wenn Sie Fragen haben, wenden Sie sich gerne mit dem Betreff „PostgreSQL“ an:
update-inside@hanser.de

Lutz Fröhlich

PostgreSQL

Praxisbuch für Administratoren und Entwickler

HANSER

Der Autor:

Lutz Fröhlich
lutz@lutzfroehlich.de

Bibliografische Information der Deutschen Nationalbibliothek:

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

Lektorat: Sylvia Hasselbach
Copy editing: Petra Kienle, Fürstenfeldbruck
Umschlagdesign: Marc Müller-Bremer, München, www.rebranding.de
Umschlagrealisation: Max Kostopoulos
Titelmotiv: © shutterstock.com/yosart
Gesamtherstellung: Eberl & Koesel, Altusried-Krugzell
Ausstattung patentrechtlich geschützt. Kösel FD 351, Patent-Nr. 0748702
Printed in Germany

Print-ISBN 978-3-446-46929-7
E-Book-ISBN 978-3-446-47315-7
E-Pub-ISBN: 978-3-446-47510-6

Inhalt

1 Einführung und Geschichte ... 1

1.1 Die Geschichte von PostgreSQL ... 2
1.2 Verwendete Versionen ... 3
1.3 Konventionen ... 3
1.4 Software und Skripte ... 3
1.5 Update inside ... 4

2 Installation mit Paketen und aus dem Quellcode ... 5

2.1 Paketinstallation ... 5
2.1.1 Paketinstallation unter Linux ... 5
2.1.2 Paketinstallation unter Windows ... 7
2.1.3 Paketinstallation unter macOS ... 9
2.2 Installation aus dem Quellcode ... 11
2.2.1 Installation aus dem Quellcode unter Linux ... 11
2.2.2 Installation aus dem Quellcode unter Windows ... 13
2.2.3 Installation aus dem Quellcode unter macOS ... 15
2.3 Erste Schritte ... 17

3 Upgrades und Versionen ... 22

3.1 Upgrade mit pg_dumpall ... 23
3.2 Upgrade mit pg_upgrade ... 24
3.3 Upgrade mit logischer Replikation ... 27
3.4 Migration nach Native Partitioning ... 28
3.5 Regressionstests ... 30

4 Die Architektur von PostgreSQL ... 32

4.1 Überblick ... 32
4.2 Memory und Prozesse ... 33
4.2.1 Hintergrundprozesse ... 34
4.2.2 Shared Memory ... 36
4.3 VACUUM ... 44
4.4 Cluster, Datenbanken und Tabellen ... 47

5 Server und Datenbanken administrieren ... 52

5.1 Parametereinstellungen ... 52
5.1.1 Einstellungen im Betriebssystem ... 52
5.1.2 Cluster-Einstellungen ... 54
5.1.3 Gebietsschema und Zeichensatz ... 65
5.2 Datenbanken verwalten ... 68
5.3 Konkurrenz ... 71
5.4 Die WAL-Archivierung einschalten ... 74
5.5 Wartungsaufgaben ... 76
5.5.1 VACUUM ... 76
5.5.2 ANALYZE ... 78
5.6 Nützliche Skripte und Hinweise ... 79
5.6.1 Eine Passwortdatei verwenden ... 79
5.6.2 Welche Parameter sind Nicht-Standard? ... 80
5.6.3 Eine Session killen ... 80
5.6.4 Eine Tabelle nach Excel kopieren ... 81
5.6.5 Die Datei .psqlrc ... 82
5.6.6 Einen WAL-Switch manuell auslösen ... 82
5.6.7 Die PostgreSQL-Server-Logdatei in eine Tabelle laden ... 83
5.6.8 Automatisches Rotieren von Logdateien ... 84
5.6.9 Nicht verwendete Indexe identifizieren ... 84
5.6.10 Microsoft Excel als Datenbank-Client ... 84
5.6.11 Den Inhalt der Kontrolldatei ausgeben ... 86
5.6.12 Platzverbrauch von Tabellen ... 87
5.6.13 Die Anzahl von Verbindungen begrenzen ... 88
5.6.14 Tabellen und Indexe in einen anderen Tablespace legen ... 88
5.6.15 Temporäre Dateien verwalten ... 89
5.6.16 Lang laufende SQL-Anweisungen ... 90
5.7 Beispielschemata ... 90

6 Neue Features ... 92

6.1 JSONB Subscripting ... 93
6.2 B-Tree Bottom-up Deletion ... 94
6.3 Pipeline Queries ... 95
6.4 Große Transaktionen für logische Replikation ... 96
6.5 Query-ID für SQL-Anweisungen ... 96
6.6 Sonstige Verbesserungen und Erweiterungen ... 97

7 Sicherung und Wiederherstellung ... 99

7.1 Online-Sicherung mit Point-in-time-Recovery ... 100
7.2 Offline-Sicherung auf Dateisystemebene ... 105
7.3 SQL Dump ... 105

8 Sicherheit und Überwachung ... 110
8.1 Sicherheit ... 111
8.1.1 Rollen und Privilegien ... 111
8.1.2 Authentifizierung und Zugangskontrolle ... 118
8.1.3 Passwortmigration ... 121
8.1.4 Rechteverwaltung ... 122
8.1.5 Sichere Verbindungen ... 127
8.1.6 Out-of-the-box-Sicherheit ... 132
8.1.7 Hacker-Attacken abwehren ... 133
8.2 Überwachung ... 142
8.2.1 Auditing ... 142
8.2.2 Monitoring ... 145

9 Logical Decoding ... 151
9.1 Logical Decoding mit SQL-Funktionen ... 153
9.2 Logical Decoding mit Java als Consumer ... 154

10 Replikation und Streaming ... 158
10.1 Physische Replikation ... 159
10.1.1 Vorbereitung und Planung ... 159
10.1.2 Konfiguration und Aktivierung ... 160
10.1.3 Kaskadenförmige Replikation ... 164
10.1.4 Hot Standby ... 165
10.1.5 Synchrone Replikation ... 166
10.1.6 Die Replikation überwachen ... 167
10.1.7 Failover und Switchover ... 170
10.2 Logische Replikation ... 175

11 Indexe effektiv einsetzen ... 181
11.1 B-Tree-Index ... 184
11.2 Block-Range-Index (BRIN) ... 188
11.3 Hash-Index ... 192
11.4 Generalized Inverted Index (GIN) ... 196
11.5 Generalized Search Tree-Index (GiST) ... 199
11.6 Expression-Index ... 202
11.7 Partieller Index ... 203
11.8 Individueller Index ... 205
11.9 Indexe und Parallelität ... 209

12 Textverarbeitung ... 211
12.1 Funktionen, Operatoren und Konfigurationen ... 213
12.2 Praktische Textverarbeitung ... 217

13 Performance Tuning ... 222
13.1 Out-of-the-box-Tuning ... 222
13.1.1 Goldene Regeln für neue Server und Datenbanken ... 223
13.1.2 Das Utility pgTune ... 224
13.1.3 Optimierung von Memory-Parametern ... 225
13.2 Performance-Analyse ... 228
13.2.1 Analyse mit dem Statistics Collector ... 228
13.2.2 Der Background Writer ... 235
13.2.3 Analyse mit pgstatspack ... 236

14 Optimierung von SQL-Anweisungen ... 239
14.1 Ausführungsschritte ... 240
14.2 Der SQL-Optimizer ... 241
14.3 Statistiken und Histogramme ... 242
14.4 Zugriffsmethoden ... 245
14.5 Join-Methoden ... 247
14.6 SQL-Optimierung ... 250
14.6.1 Der EXPLAIN-Befehl ... 250
14.6.2 Ausführungspläne verstehen und optimieren ... 255
14.6.3 Parallelisierung von SQL-Ausführungen ... 264
14.6.4 Logging von SQL-Anweisungen ... 276

15 Einsatz großer Datenbanken ... 278
15.1 Skalierung großer Datenbanken ... 279
15.2 Partitionierung von Tabellen ... 280
15.3 Paralleles SQL ... 287
15.4 Materialized Views ... 289
15.5 Indexe für große Tabellen ... 290

16 PostGIS ... 291
16.1 PostGIS und PostgreSQL ... 291
16.2 PostGIS installieren ... 292
16.2.1 Paketorientierte Installation ... 292
16.2.2 Installation aus dem Quellcode ... 295
16.3 Erste Schritte mit PostGIS ... 296
16.4 PostGIS in der Praxis anwenden ... 301

17 Applikationen für PostgreSQL entwickeln ... 307
17.1 Applikationsdesign ... 307
17.2 Entwicklungswerkzeuge ... 311
17.3 PostgreSQL Extensions ... 312

18 Erweiterungen und Module ... 313
18.1 Datentypen ... 313
18.2 Funktionen und Sprachen ... 314
18.2.1 SQL-Funktionen ... 314
18.2.2 Funktionen mit prozeduralen Programmiersprachen ... 319
18.2.3 C-Funktionen ... 322
18.3 Operatoren ... 328
18.4 Das Extension-Netzwerk ... 330
18.4.1 Extensions entwickeln ... 330
18.4.2 Extensions publizieren ... 334

19 PL/pgSQL-Funktionen und Trigger ... 337
19.1 PL/pgSQL-Funktionen ... 337
19.1.1 Abfragen und Resultsets ... 341
19.1.2 Cursor verwenden ... 343
19.1.3 DML-Anweisungen ... 345
19.1.4 Dynamische SQL-Anweisungen ... 347
19.1.5 Fehlerbehandlung ... 348
19.2 Trigger ... 349

20 Embedded SQL (ECPG) ... 352

21 Java-Programmierung ... 362
21.1 Eine Entwicklungsumgebung einrichten ... 362
21.2 Verarbeitung von Resultsets ... 365
21.3 DML-Anweisungen und Transaktionen ... 368
21.4 Bindevariablen verwenden ... 370
21.5 Java und Stored Functions ... 371
21.6 Large Objects ... 374
21.7 JDBC-Tracing ... 378

22 Die C-Library libpq ... 381
22.1 Die Entwicklungsumgebung einrichten ... 381
22.2 Programme mit libpq erstellen ... 386

23 PHP-Applikationen ... 399
23.1 Installation und Konfiguration ... 400
23.2 Applikationen mit PHP entwickeln ... 402
23.3 Die PDO-API ... 409

24 REST API ... 412

25 Client-Programmierung mit Perl-DBI ... 417
25.1 SELECT-Anweisungen und Resultsets ... 420
25.2 DML-Anweisungen ... 424
25.3 Bindevariablen verwenden ... 425
25.4 Fehlerbehandlung und Tracing ... 427
25.5 Nützliche Skripte und Beispiele ... 429
25.5.1 Mehrere Server abfragen ... 429
25.5.2 Parallele Verbindungen ... 431
25.5.3 Large Objects verarbeiten ... 433
25.5.4 Asynchrone Abfragen ... 434
25.5.5 Datenbanken vergleichen ... 435

26 Programmierung mit Python ... 438
26.1 Client-Programmierung mit Python ... 438
26.2 Server-Programmierung mit PL/Python ... 445

27 Data Science und Machine Learning ... 456
27.1 Werkzeuge ... 457
27.2 Einführung in Data Science ... 459
27.3 Ein Beispiel ... 460
27.4 Daten verwalten ... 464
27.5 Data Cleaning ... 469
27.6 Daten analysieren ... 472
27.6.1 SQL-Funktionen ... 473
27.6.2 Analysen durchführen ... 478
27.7 Stimmungslagenanalysen ... 489

28 PostgreSQL in die IT-Landschaft einbinden ... 498
28.1 Features und Funktionen ... 498
28.2 Datensicherheit und Wiederherstellung ... 499
28.3 Desaster Recovery ... 500
28.4 Überwachung ... 501
28.5 Administrierbarkeit ... 501
28.6 Verfügbarkeit ... 502
28.7 Datensicherheit und Auditing ... 502
28.8 Performance und Skalierbarkeit ... 503
28.9 Schnittstellen und Kommunikation ... 504
28.10 Support ... 504
28.11 Fazit ... 505

29 Migration von MySQL-Datenbanken ... 506
29.1 Unterschiede zwischen MySQL und PostgreSQL ... 506
29.2 Eine Migration durchführen ... 508

30 Von Oracle nach PostgreSQL migrieren ... 513
30.1 Die Migration planen ... 513
30.2 Unterschiede zwischen Oracle und PostgreSQL ... 515
30.2.1 Unterschiede der Datentypen ... 515
30.2.2 Syntaktische und logische Unterschiede ... 516
30.2.3 Steigerung der Kompatibilität von PostgreSQL ... 519
30.3 Portierung von Oracle PL/SQL ... 520
30.4 Tools zur Unterstützung der Migration ... 523
30.5 Technisches Vorgehen ... 525
30.6 Ein Migrationsbeispiel ... 525
30.6.1 Manuelle Migration ... 526
30.6.2 Migration unter Verwendung von Ora2Pg ... 532
30.6.3 Große Tabellen laden ... 536

31 Replikation zwischen Oracle und PostgreSQL ... 538
31.1 Datenbanklink zwischen Oracle und PostgreSQL ... 538
31.2 Replikation mit Oracle XStream ... 544

32 PostgreSQL in der Cloud ... 556
32.1 Private Cloud ... 557
32.2 Public Cloud ... 559
32.3 Hybrid Cloud ... 562

Index ... 574

1 Einführung und Geschichte

Seit dem Erscheinen des Buchs „PostgreSQL 10 – Praxisbuch für Administratoren und Entwickler“ sind fast vier Jahre vergangen. In dieser Zeit hat sich bezüglich der Weiterentwicklung und des Einsatzes von PostgreSQL-Datenbanken viel getan. Ein guter Grund, die neue Version mit ihren neuen Features und Verbesserungen zu präsentieren.

So wie bei anderen Herstellern von Datenbanksystemen ist man dazu übergegangen, anstelle von großen Major-Release-Paketen neue Features kontinuierlicher zur Verfügung zu stellen. Das hat zur Folge, dass die Versionen schneller hochgezählt werden als in der Vergangenheit. Das Buch bezieht sich auf die Version 14. Es ist keine einfache Überarbeitung des Vorgängers, sondern wurde inhaltlich neu strukturiert und erweitert. Das Buch präsentiert sich weiterhin im bekannten Stil des Autors mit vielen Beispielen und Skripten.

Für die Themenwahl wurde auch die allgemeine Entwicklung im IT-Sektor berücksichtigt. Für die Installation gibt es nun auch eine Anleitung für macOS. Macbook-Besitzer können nun direkt einsteigen und wahlweise aus Paketen oder dem Quellcode installieren.

In Kapitel 4 finden Sie wiederum eine ausführliche Einführung in die Architektur, um das Verständnis für interne Prozesse und Abläufe zu fördern. Das Thema „Sicherheit und Überwachung“ wurde um interne Details zur Authentifizierung und Hacker-Abwehr erweitert.

PostgreSQL tummelt sich zunehmend im Bereich der großen Datenbanken. Die Themen „Performance- und SQL-Optimierung“ sowie der Umgang mit großen Datenbanken bilden deshalb einen Schwerpunkt. Dazu wurde das Kapitel 11 „Indexe effizient einsetzen“ neu aufgenommen. Es zeigt die Überlegenheit von PostgreSQL in diesem Umfeld gegenüber manch anderem Datenbanksystem. Für den Einsatz von großen Datenbanken zeigen die Möglichkeiten der Parallelisierung, wie PostgreSQL-Datenbanken und SQL-Anweisungen gegen rapides Wachstum skalieren können.

Ein weiterer Schwerpunkt ist der Bereich „Data Science und maschinelles Lernen“. Es werden die Möglichkeiten und Vorteile für den Einsatz von PostgreSQL als Datenbasis, aber auch der vorhandenen Features für Analyse und Auswertungen von Daten dargestellt. Hierbei spielt die Textverarbeitung, der ein eigenes Kapitel gewidmet wurde, eine wichtige Rolle. Die Datenbankprogrammierung mit Python, client- und serverseitig, wurde ebenfalls neu aufgenommen und ergänzt die Darstellung und Beispiele der anderen Programmiersprachen. Das Buch wird damit Administratoren und Entwicklern gleichermaßen gerecht.

Am Thema „Cloud“ führt längst kein Weg mehr vorbei. Die Cloud-Angebote haben sich vervielfacht. Aber auch die Einsatzmöglichkeiten und Features haben sich erweitert. Aktuelle Trends wie die Hybrid Cloud machen auch vor PostgreSQL nicht halt.

Das Buch ist als Einstieg und Nachschlagewerk für IT-Profis geschrieben und setzt Basiskenntnisse von relationalen Datenbanken voraus. Auf eine Erläuterung von gängigen Begriffen wird deshalb bewusst verzichtet, auch um den Umfang des Buchs überschaubar zu halten. Dennoch finden Sie viele Beispiele und Praxistipps, die auch Einsteigern die Möglichkeit bieten, sich in das Produkt einzuarbeiten.

PostgreSQL hat in den vergangenen Jahren an Verbreitung und Popularität erheblich zugenommen. Dazu haben in erheblichem Maß die permanente Erweiterung mit neuen Features und die Anpassung an die Belange der Anwender beigetragen. PostgreSQL ist der lebende Beweis, dass Open-Source-Software nicht nur mit kommerziellen Produkten mithalten kann, sondern in einigen Bereichen sogar überlegen ist. Der kommerzielle Druck steht nicht im Vordergrund und lässt die Entwickler-Community frei arbeiten und Innovationen umsetzen.

Neben einem robusten Transaktionskern sowie einer hohen Zuverlässigkeit bietet PostgreSQL viele Features eines modernen Datenbankbetriebssystems und kann problemlos in eine vorhandene IT-Infrastruktur integriert werden. Durch den hohen Kompatibilitätsgrad zu Oracle ist der Migrationsaufwand überschaubar und ein Mischbetrieb gut umzusetzen.

PostgreSQL kann auf allen populären Plattformen wie Linux, macOS, Solaris oder Windows eingesetzt werden. Obwohl es sich um ein Open-Source-Produkt handelt, kann kommerzieller Support zu einem vernünftigen Preis hinzugekauft werden. Einem professionellen Einsatz steht damit nichts im Wege.

Freuen Sie sich auf einen PostgreSQL-Server 14 mit spannenden neuen Features!

1.1 Die Geschichte von PostgreSQL

PostgreSQL geht zurück auf das POSTGRES-Projekt, das an der University of California at Berkeley in den 1980er-Jahren angesiedelt war. Die erste vorzeigbare Version erschien im Jahre 1987 als Postgres-Version 1. Als Reaktion auf die ersten Kritiken wurde das noch heute in PostgreSQL vorhandene Rule-System entwickelt. Version 3 erschien im Jahre 1991 mit einer Weiterentwicklung der Abfrageeinheit. 1993 beendete die University of California das Projekt mit der Version 4.2, um die rasant wachsenden Supportanforderungen nicht mehr tragen zu müssen.

Nach Hinzufügen eines SQL-Abfrageinterpreters im Jahr 1995 wurde die Software unter dem Begriff Postgres95 ins Web gestellt, mit dem Quellcode des originalen Berkeley-Postgres. Das Produkt war zu dieser Zeit komplett in ANSI C geschrieben. Durch Verbesserungen in den Bereichen Wartbarkeit und Performance lief es schließlich bis zu 50 % schneller als das originale Berkeley-Postgres.

Die Entscheidung, die Jahreszahl aus dem Produktnamen zu entfernen, fiel im Jahre 1996. Damit wurde Postgres95 zu PostgreSQL und es begann die ständige Weiterentwicklung von PostgreSQL als Open-Source-Produkt. Obwohl es über viele Jahre ein Schattendasein im Licht der großen kommerziellen Datenbanken, aber auch der durch den Internet-Boom schnell verbreiteten Open-Source-Datenbank MySQL führte, erfolgte seine konsequente Weiterentwicklung durch die Community.

Heute präsentiert sich PostgreSQL als ausgereift und stabil und erfüllt alle Anforderungen an ein modernes relationales Datenbanksystem. Für viele überraschend: Die Performance ist vergleichbar mit so manchem kommerziellen Produkt.

1.2 Verwendete Versionen

Das Buch bezieht sich auf die während der Manuskripterstellung vorliegende aktuelle Version 14. Schauen Sie regelmäßig nach weiteren Veröffentlichungen, insbesondere für neuere Features und Versionen, auf der Webseite des Verlags und der Autoren-Webseite vorbei. Alles rund um die PostgreSQL-Community finden Sie auf der Webseite *http://www.postgresql.org*.

1.3 Konventionen

Begriffe in spitzen Klammern bezeichnen eine zu ersetzende Variable (so ist zum Beispiel der Ausdruck <VERSION> in der Regel durch die aktuelle Version 14.1 zu ersetzen). Die meisten Darstellungen beziehen sich gleichermaßen auf UNIX- und Windows-Betriebssysteme. Die Darstellung der Umgebungsvariablen erfolgt im Wesentlichen im UNIX-Format, das heißt z. B. $BIN statt %BIN% für Windows. Sie können das Format einfach nach Windows übertragen. Das Gleiche gilt für das Trennzeichen der Pfade: „/“ unter Unix sowie „\“ unter Windows.

1.4 Software und Skripte

Sie können die aktuelle Version von PostgreSQL aus dem Internet herunterladen und installieren. Es wird die Installation aus dem Quellcode empfohlen, um alle Beispiele nachvollziehen zu können. Ideal ist es, wenn Sie auf einem Linux-, macOS- oder Windows-Betriebssystem arbeiten.

Alle nummerierten Listings im Buch können als Datei von der Webseite des Verlags sowie von der Autoren-Webseite *https://www.lutzfroehlich.de* heruntergeladen werden:

Geben Sie auf

https://plus.hanser-fachbuch.de

diesen Code ein:

```
plus-92Fgh-ci4R7
```

1.5 Update inside

PostgreSQL stellt jährlich neue Major-Release-Versionen mit neuen Features und Erweiterungen zur Verfügung. Davon sind alle Bereiche vom Kernsystem bis hin zur Applikationsentwicklung betroffen.

Damit Sie möglichst lange mit diesem Buch arbeiten können, haben Sie die Möglichkeit, sich für den kostenlosen Update inside-Service zu registrieren: Geben Sie unter

www.hanser-fachbuch.de/postgresql-update

diesen Code ein:

`2WHK-SeoTr-bF5x`

Dann erhalten Sie bis Mai 2024 Aktualisierungen in Form zusätzlicher Kapitel als PDF. Darin stelle ich Ihnen wichtige Neuerungen vor und gehe auf Änderungen ein, die die Inhalte dieses Buches betreffen.

Putbus und Oasis del Sol, im April 2022

Lutz Fröhlich

lutz@lutzfroehlich.de

2 Installation mit Paketen und aus dem Quellcode

Die Installation kann sowohl von vorgefertigten Paketen als auch aus dem Quellcode erfolgen. Wenn Sie eine schnelle Installation bevorzugen, ist die einfachere Paketinstallation zu empfehlen. Allerdings müssen Sie dann mit dem Setup leben, das das Paket zur Verfügung stellt. Mit der Installation aus dem Quellcode sind Anpassungen und Erweiterungen möglich. Für den produktiven Einsatz ist zu beachten, dass Supportleistungen von Anbietern sich ausschließlich auf die entsprechenden Pakete beziehen. Diese Pakete sind bis zu einem gewissen Grad getestet. Beachten Sie, dass damit eine Abhängigkeit vom Release-Zyklus des Drittanbieters besteht. Ein wichtiger Vorteil der Paketinstallation ist, dass einheitliche Installationsstände ausgerollt werden können.

In diesem Kapitel werden beide Varianten vorgestellt. Um alle in diesem Buch vorgestellten Features und Optionen nachvollziehen zu können, wird eine Installation aus dem Quellcode empfohlen.

2.1 Paketinstallation

Alle erforderlichen Informationen für die Installation befinden sich auf der Seite *https://postgresql.org/download* unter dem Abschnitt *Binary Packages*. Wir führen zuerst Paketinstallationen für Linux und danach für Windows und macOS durch.

2.1.1 Paketinstallation unter Linux

Pakete für Linux stehen für die gebräuchlichsten Derivate zur Verfügung. Wir führen eine Installation unter Oracle Linux 7 durch. Das Vorgehen ist identisch mit einer Installation unter Red Hat Linux und CentOS. Danach erfolgt die Installation unter Ubuntu.

2.1.1.1 Paketinstallation unter Oracle Linux, Red Hat oder CentOS

Am einfachsten ist die Installation mit Hilfe des yum-Repository. Die folgenden Schritte beschreiben die Installation:

1. Installieren Sie das RPM-Paket im Repository.

```
# yum install -y https://download.postgresql.org/pub/repos/yum/reporpms/EL-7-x86_64/
pgdg-redhat-repo-latest.noarch.rpm
```

2. Die Installation des Pakets erfolgt mit dem yum-Utility.

```
# yum install -y postgresql14-server
```

3. Zum Schluss wird die Datenbank initialisiert und für den automatischen Start konfiguriert.

```
# /usr/pgsql-13/bin/postgresql-14-setup initdb
# systemctl enable postgresql-14
# systemctl start postgresql-14
```

Falls keine Verbindung vom Datenbankserver zum Internet besteht, muss das Paket manuell heruntergeladen und übertragen werden. Gehen Sie dazu auf die Seite *https://yum.postgresql.org* und wählen Sie das passende Paket aus. Die Installation kann mit dem yum- oder dem rpm-Utility erfolgen.

2.1.1.2 Paketinstallation unter Ubuntu

Ubuntu ist ein weit verbreitetes Linux-Derivat und wird häufig als Betriebssystem für Desktops und Laptops verwendet. Mit den folgenden Schritten erfolgt die Installation der neuesten verfügbaren Version.

1. Erstellen Sie die Konfiguration für das Repository.

```
$ sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-
pgdg main" > /etc/apt/sources.list.d/pgdg.list'
```

2. Importieren Sie den Key für das Repository und führen Sie das Upgrade für die Paketliste durch.

```
$  wget --quiet -O – https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-
key add –
$ sudo apt-get update
```

3. Installieren Sie die letzte verfügbare Version von PostgreSQL.

```
$ sudo apt-get -y install postgresql
. . .
Success. You can now start the database server using:
    pg_ctlcluster 14 main start
```

Die Paketinstallation ist damit abgeschlossen.

2.1.2 Paketinstallation unter Windows

Für die Paketinstallation unter Windows erfolgt eine Weiterleitung von der Seite *https://postgresql.org/download* auf die Webseite von „EnterpriseDB“. Dort finden Sie das Paket für Windows in den letzten verfügbaren Versionen als Windows-Installer. EnterpriseDB ist ein Anbieter von vorgefertigten Paketen und Supportleistungen für diese Distributionen. Führen Sie die folgenden Schritte für die Paketinstallation aus:

1. Starten Sie den Windows-Installer.

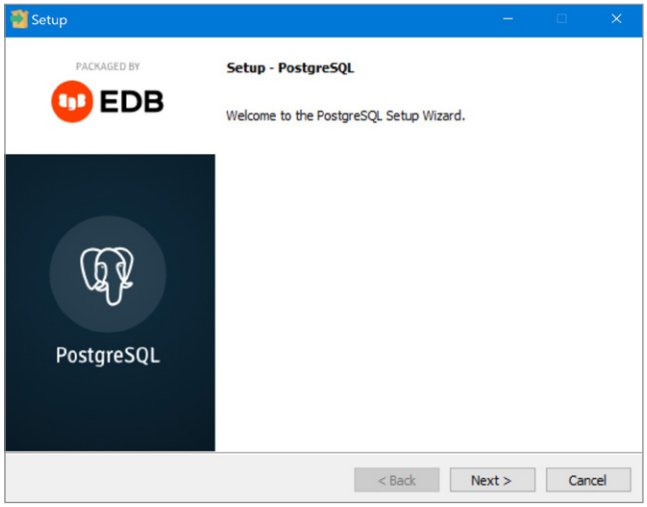


Bild 2.1 Die PostgreSQL-Paketinstallation für Windows starten

2. Wählen Sie das Installationsverzeichnis aus.

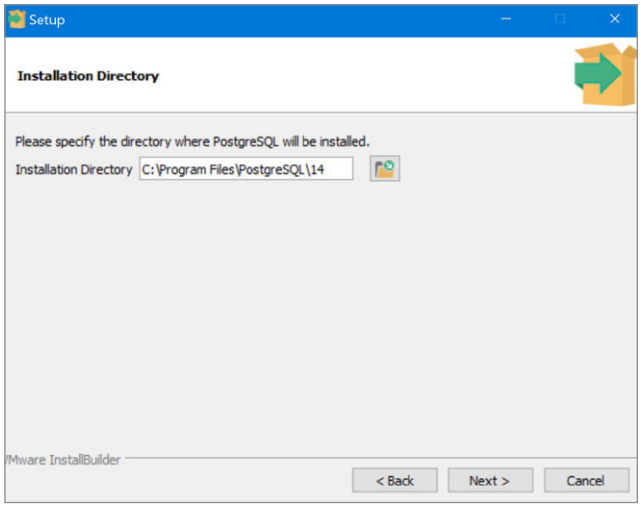


Bild 2.2 Das Installationsverzeichnis festlegen

3. Im nächsten Schritt können die Komponenten für die Installation ausgewählt werden.

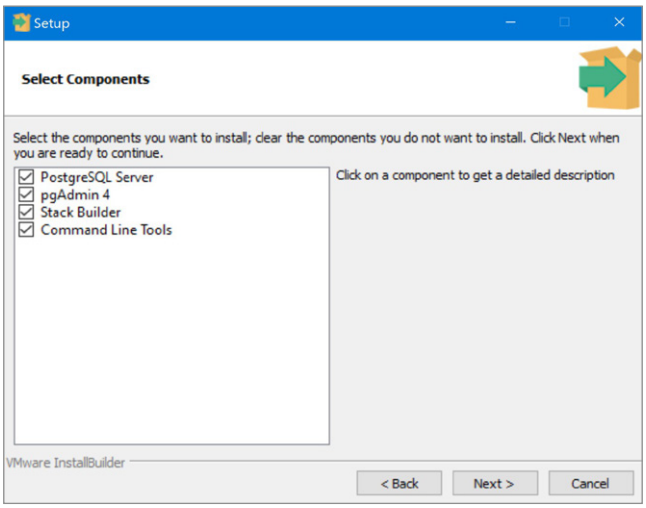


Bild 2.3 Auswahl der Komponenten für die Installation

4. Es erfolgt die Festlegung des Verzeichnisses für die Daten.
5. In den folgenden Schritten werden das Passwort für den Benutzer „postgres“, der Port sowie die Locale festgelegt.
6. Starten Sie die Installation.

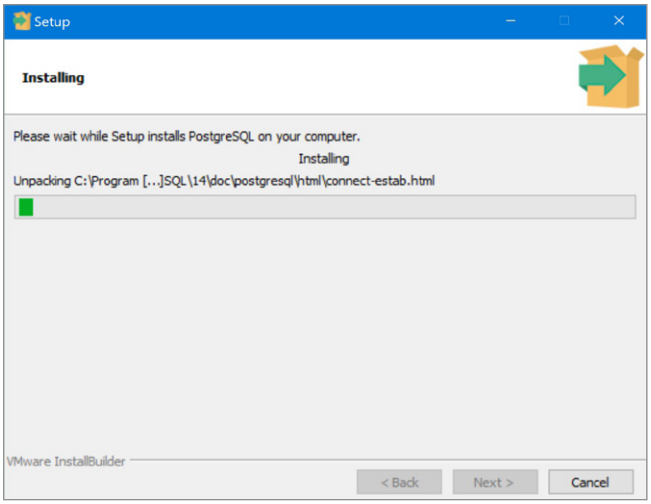


Bild 2.4 Die Paketinstallation ausführen

Der Installer legt einen Windows-Dienst zur Verwaltung des PostgreSQL-Servers an. Damit ist die Installation abgeschlossen.

2.1.3 Paketinstallation unter macOS

Vorgefertigte Pakete für macOS gibt es von verschiedenen Anbietern. An dieser Stelle verwenden wir ein Paket der Firma EnterpriseDB, analog zur Windows-Installation. Der Link zum Download lautet:

https://www.enterprisedb.com/downloads/postgres-postgresql-downloads

1. Öffnen Sie das Paket. Es erscheint das Startfenster des Setup Wizzard.

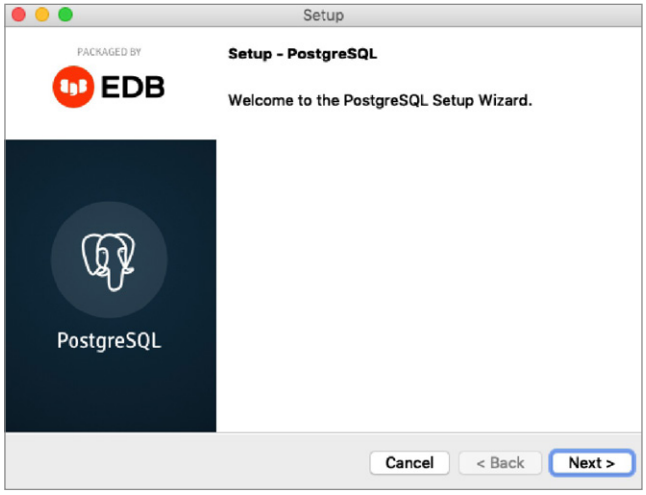


Bild 2.5 Der Setup Wizzard für macOS

2. Im nächsten Schritt kann das Installationsverzeichnis ausgewählt werden.

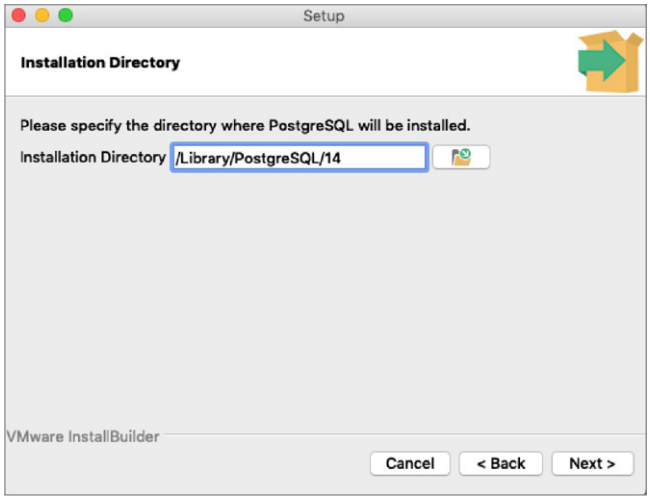


Bild 2.6 Das Installationsverzeichnis unter macOS auswählen

3. Es folgt die Auswahl der zu installierenden Komponenten. Neben dem PostgreSQL-Server können „pgAdmin“, der Stack Builder sowie die Kommandozeilen-Werkzeuge mit installiert werden.

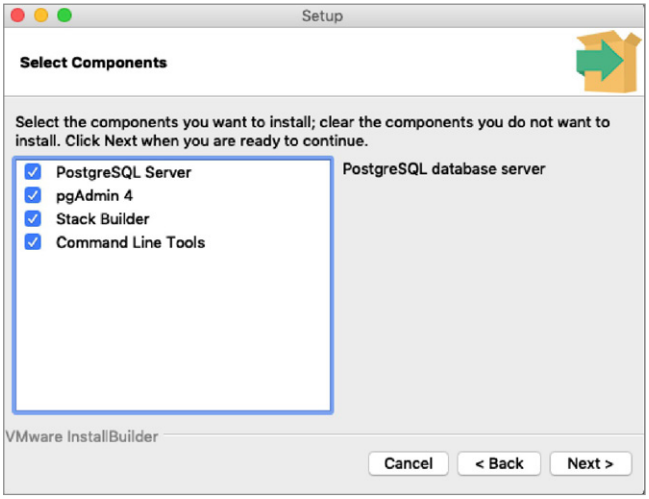


Bild 2.7 Auswahl der Komponenten für die Paket-Installation

4. Vergeben Sie Benutzername und Passwort für den Superuser (zum Beispiel „postgres“).
5. In den nächsten Schritten werden der Port sowie die Locale-Einstellungen abgefragt.
6. Verifizieren Sie die Eingaben in der Zusammenfassung und starten Sie die Installation.

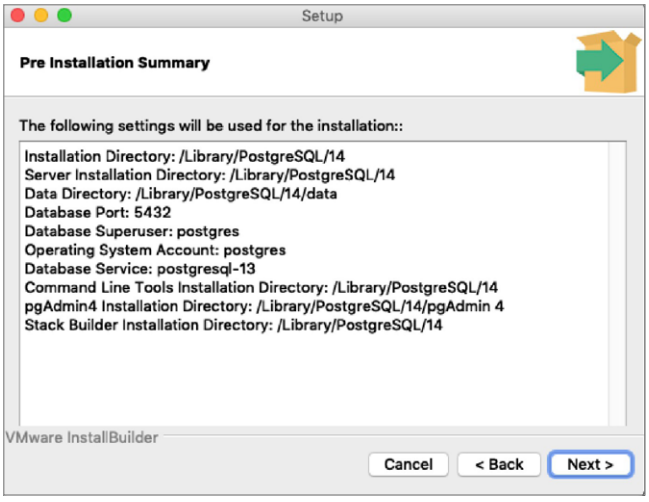


Bild 2.8 Zusammenfassung der Installation

Nach der Erfolgsmeldung ist die Installation abgeschlossen und der PostgreSQL-Server kann verwendet werden.

2.2 Installation aus dem Quellcode

Das Vorgehen bei der Installation aus dem Quellcode ist für alle Betriebssysteme ähnlich. Die Quellprogramme werden kompiliert, gelinkt und anschließend im Installationsverzeichnis bereitgestellt. In diesem Abschnitt werden die Schritte für Linux, Windows und macOS vorgestellt. Mit einer Installation aus dem Quellcode kann besser individualisiert und erweitert werden. Sie können sogar eigene Programmkomponenten einbinden.

2.2.1 Installation aus dem Quellcode unter Linux

Wir installieren an dieser Stelle zunächst den Standardumfang. Es wird im weiteren Verlauf des Buchs darauf hingewiesen, wenn zusätzliche Optionen eingebunden werden. Für PostgreSQL 14 bieten sich die Linux-Versionen 7 oder 8, zum Beispiel Red Hat, CentOS oder Oracle Linux an. Da die Programme komplett aus dem Quellcode übersetzt und mit den im Betriebssystem installierten Bibliotheken verbunden werden, sind wir an dieser Stelle sehr flexibel, was die Versionen und Derivate des Betriebssystems betrifft. Das folgende Vorgehen beschreibt die Installation:

1. Laden Sie den Quellcode von der Webseite *https://www.postgresql.org/ftp/source* herunter. Der Dateiname hat das Format *postgresql-<version>.tar.gz*.
2. Entpacken Sie den Quellcode und wechseln Sie in das Verzeichnis *postgresql-<version>*.

```
# tar -xvzf postgresql-14.1.tar.gz
# cd postgresql-14.1
```

3. Im ersten Schritt der Installation werden die Quellen für das Betriebssystem konfiguriert. Das Skript *configure* führt einige Tests aus, um die Werte einiger systemabhängiger Variablen zu bestimmen.

```
# ./configure
```

4. Starten Sie anschließend den Build-Prozess. Es werden die Libraries und Binaries erstellt. Verwenden Sie das GNU Make Utility und achten Sie auf die Erfolgsmeldung am Ende des Durchlaufs.

```
# make
. . .
All of PostgreSQL successfully made. Ready to install.
```

5. Im nächsten Schritt erfolgt die Installation der Binaries in die Zielverzeichnisse. Dieser Schritt muss ebenfalls unter dem Benutzer *root* ausgeführt werden, um die erforderlichen Berechtigungen zu erstellen. Standardmäßig erfolgt die Installation in das Verzeichnis */usr/local/pgsql*.

```
# make install
. . .
PostgreSQL installation complete.
```

6. Damit ist die Installation abgeschlossen. In der Nachbereitung sind folgende Schritte erforderlich:

   Einen Benutzer *postgres* als Eigentümer der Software und des Servers erstellen. Der Benutzername ist frei wählbar:

```
# adduser postgres
```

   Ein Verzeichnis zum Speichern der Datenbankdateien erstellen. Das Verzeichnis kann beliebig gewählt werden:

```
# mkdir /usr/local/pgsql/data
# chown postgres /usr/local/pgsql/data
```

   Den Datenbankserver initialisieren:

```
# su – postgres
$ /usr/local/pgsql/bin/initdb -D /usr/local/pgsql/data
. . .
initdb: warning: enabling "trust" authentication for local connections
You can change this by editing pg_hba.conf or using the option -A, or
--auth-local and --auth-host, the next time you run initdb.
Success. You can now start the database server using:
    /usr/local/pgsql/bin/pg_ctl -D /usr/local/pgsql/data -l logfile start
```

   Den Datenbankserver starten. Verifizieren Sie, dass die Prozesse laufen:

```
$ /usr/local/pgsql/bin/pg_ctl -D /usr/local/pgsql/data -l logfile start
waiting for server to start.... done
server started
[postgres@serv1 ~]$ ps -ef|grep postg
postgres 14543     1  0 10:40 ? 00:00:00 /usr/local/pgsql/bin/postgres -D /usr/local/
pgsql/data
postgres 14545 14543  0 10:40 ? 00:00:00 postgres: checkpointer
postgres 14546 14543  0 10:40 ? 00:00:00 postgres: background writer
postgres 14547 14543  0 10:40 ? 00:00:00 postgres: walwriter
postgres 14548 14543  0 10:40 ? 00:00:00 postgres: autovacuum launcher
postgres 14549 14543  0 10:40 ? 00:00:00 postgres: stats collector
postgres 14550 14543  0 10:40 ? 00:00:00 postgres: logical replication launcher
```

   Eine Testdatenbank mit dem Namen *hanser* anlegen:

```
$ /usr/local/pgsql/bin/createdb hanser
$ /usr/local/pgsql/bin/psql hanser
psql (14.1)
Type "help" for help.
hanser=# SELECT version();
                                version
------------------------------------------------------------------------
 PostgreSQL 14.1 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red
Hat 4.8.5-44.0.3), 64-bit
(1 row)
```

Damit ist die Installation aus dem Quellcode abgeschlossen und der PostgreSQL-Server ist einsatzbereit.

2.2.2 Installation aus dem Quellcode unter Windows

Die Installation aus dem Quellcode auf einem Windows-Betriebssystem umfasst dieselben Schritte wie unter Linux. Der Quellcode kann mit Hilfe des Visual C++-Compilers von Microsoft übersetzt und verbunden werden.

HINWEIS: Für das Erstellen der Software aus dem Quellcode wird mindestens das Microsoft Windows SDK benötigt. Alternativ kann Microsoft Visual Studio installiert werden, das ebenfalls die erforderlichen Programme für das Kompilieren und Linken der Quellen enthält.

In den folgenden Schritten erfolgt die Installation unter Windows 10 auf einer 64-Bit-Plattform.

1. Installieren Sie Microsoft Windows SDK oder Visual Studio.
2. Für die Installation ist ein Perl-Interpreter erforderlich. Installieren Sie im Bedarfsfall *ActiveState Perl*. Das Paket sowie die Installationsanleitung finden Sie auf der Webseite *https://www.activestate.com*.
3. Entpacken Sie den heruntergeladenen PostgreSQL-Quellcode. Auch wenn die Datei die Endung *.tar.gz* besitzt, kann sie mit den meisten Kompressionswerkzeugen direkt unter Windows ausgepackt werden. Im vorliegenden Beispiel werden die Quellen in das Verzeichnis *C:\PostgreSQL* gelegt. Beim Entpacken wird ein Unterverzeichnis im Format *postgresql-<version>* angelegt.
4. Öffnen Sie eine Windows-Eingabeaufforderung (DOS-Fenster) mit Administratorrechten und wechseln Sie in das Verzeichnis *C:\PostgreSQL\postgresql-<version>*.
5. Gegebenenfalls muss noch die Umgebung für das Windows SDK oder Visual Studio gesetzt werden. Compiler und Linker müssen sich im Pfad befinden und auf 64-Bit-Programme gestellt werden. Für das Visual Studio lautet der Aufruf:

```
C:\PostgreSQL\postgresql-14.1>"C:\Program Files (x86)\Microsoft Visual Studio\2019\
Community\VC\Auxiliary\Build\vcvars64.bat"
**********************************************************************
** Visual Studio 2019 Developer Command Prompt v16.8.4
** Copyright (c) 2020 Microsoft Corporation
**********************************************************************
[vcvarsall.bat] Environment initialized for: 'x64'
```

6. Wechseln Sie in das Unterverzeichnis *...\src\tools\msvc* und starten Sie Kompilierung und Verbinden des Quellcodes. Achten Sie auf die Erfolgsmeldung am Ende.

```
C:\PostgreSQL\postgresql-14.1>cd src\tools\msvc
C:\PostgreSQL\postgresql-14.1\src\tools\msvc>build
. . .
Der Buildvorgang wurde erfolgreich ausgeführt.
    0 Warnung(en)
    0 Fehler
Verstrichene Zeit 00:03:38.04
```

7. Im letzten Schritt erfolgt die Installation. Geben Sie das gewünschte Verzeichnis an, unter dem die Software installiert werden soll.

```
C:\PostgreSQL\postgresql-14.1\src\tools\msvc>install C:\PostgreSQL
Installing version 14 for release in C:\PostgreSQL
. . .
Installation complete.
```

8. Analog zu den Installationsschritten unter Linux lässt sich der Datenbankserver jetzt initialisieren. Das Verzeichnis für den Cluster kann frei gewählt werden. Legen Sie das Verzeichnis vorher an.

```
C:\PostgreSQL\bin>initdb -D C:\PostgreSQL\data
. . .
Success. You can now start the database server using:
    pg_ctl -D ^"C^:^\PostgreSQL^\data^" -l logfile start
```

9. Zum Schluss kann der Server gestartet und eine Datenbank mit dem Namen *hanser* angelegt werden.

```
C:\PostgreSQL\bin>pg_ctl -D C:\PostgreSQL\data -l logfile start
waiting for server to start.... done
server started
C:\PostgreSQL\bin>createdb hanser
```

10. Mit dem Kommandozeilen-Utility *psql* lässt sich die Verbindung zur Datenbank herstellen und SQL-Anweisungen können verarbeitet werden.

```
C:\PostgreSQL\bin>psql hanser
psql (14.1)
Type "help" for help.
hanser=# SELECT version();
                           version
-------------------------------------------------------------
 PostgreSQL 14.1, compiled by Visual C++ build 1928, 64-bit
(1 row)
```

Damit ist die Installation abgeschlossen und der PostgreSQL-Server kann verwendet werden.

HINWEIS: Die Syntax sowie die meisten Beispiele im Buch beziehen sich auf ein Linux-System. Wenn Sie vorzugsweise unter Windows arbeiten, dann können Sie dennoch alles nachvollziehen, wenn Sie die betriebssystemspezifischen Abweichungen beachten. Das Verhalten von PostgreSQL ist plattformübergreifend analog.

Optional kann ein Windows-Dienst eingerichtet werden, der das automatische Starten und Stoppen bei Start und Shutdown des Betriebssystems übernimmt.

```
C:\PostgreSQL\bin>pg_ctl register -D C:\PostgreSQL\data -N PostgreSQL14 -U Lutz -P
xxxxxxx
```

Der Dienst ist nun in der Dienste-Verwaltung von Windows sichtbar.

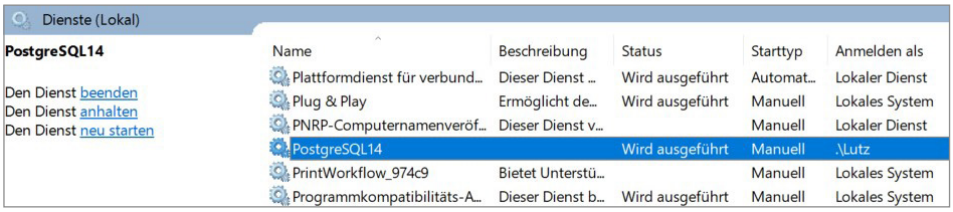


Bild 2.9 Einen Windows-Dienst für PostgreSQL anlegen

2.2.3 Installation aus dem Quellcode unter macOS

Die Installation verläuft ähnlich wie die Linux-Installation. Das folgende Beispiel beschreibt die Installation:

1. Laden Sie den Quellcode von der Webseite *https://www.postgresql.org/ftp/source* herunter. Der Dateiname hat das Format *postgresql-<version>.tar.gz*.
2. Entpacken Sie den Quellcode und wechseln Sie in das Verzeichnis *postgresql-<version>*.

```
$ tar -xvzf postgresql-14.1.tar.gz
$ cd postgresql-14.1
```

3. Für das Kompilieren und Linken des Quellcodes werden die Entwicklungstools für die Kommandozeile von Apple benötigt. Installieren Sie diese falls erforderlich.

```
$ xcode-select --install
```

4. Im ersten Schritt der Installation werden die Quellen für das Betriebssystem konfiguriert. Das Skript *configure* führt einige Tests aus, um die Werte einiger systemabhängiger Variablen zu bestimmen. Dafür wird der Pfad für „sysroot“ benötigt. Der Pfad variiert je nach installierter Version des Betriebssystems. Sie finden den Pfad mit dem Kommando „xrun“.

```
$ xrun –show-sdk-path
/Library/Developer/CommandLineTools/SDKs/MacOSX.sdk
$ ./configure PG_SYSROOT=/Library/Developer/CommandLineTools/SDKs/MacOSX.sdk
```

5. Starten Sie anschließend den Build-Prozess. Es werden die Libraries und Binaries erstellt. Verwenden Sie das GNU Make Utility und achten Sie auf die Erfolgsmeldung am Ende des Durchlaufs.

```
$ make PG_SYSROOT=/Library/Developer/CommandLineTools/SDKs/MacOSX.sdk all
. . .
All of PostgreSQL successfully made. Ready to install.
```

6. Im nächsten Schritt erfolgt die Installation der Binaries in die Zielverzeichnisse. Dieser Schritt muss ebenfalls unter dem Benutzer *root* ausgeführt werden, um die erforderlichen Berechtigungen zu erstellen. Standardmäßig erfolgt die Installation in das Verzeichnis */usr/local/pgsql*.

```
$ make install
. . .
PostgreSQL installation complete.
```

7. Damit ist die Installation abgeschlossen. In der Nachbereitung wird das Verzeichnis für den Server erzeugt.

```
$ mkdir /usr/local/pgsql/data
$ /usr/local/pgsql/bin/initdb -D /usr/local/pgsql/data
. . .
initdb: warning: enabling "trust" authentication for local connections
You can change this by editing pg_hba.conf or using the option -A, or
--auth-local and --auth-host, the next time you run initdb.
Success. You can now start the database server using:
    /usr/local/pgsql/bin/pg_ctl -D /usr/local/pgsql/data -l logfile start
```

8. Der Datenbankserver kann nun gestartet werden. Verifizieren Sie, ob alle Prozesse laufen.

```
$ /usr/local/pgsql/bin/pg_ctl -D /usr/local/pgsql/data -l logfile start
waiting for server to start.... done
server started
$ ps -ef|grep postg
501 14543     1  0 10:40 ? 00:00:00 /usr/local/pgsql/bin/postgres -D /usr/local/
pgsql/data
501 14545 14543  0 10:40 ? 00:00:00 postgres: checkpointer
501 14546 14543  0 10:40 ? 00:00:00 postgres: background writer
501 14547 14543  0 10:40 ? 00:00:00 postgres: walwriter
501 14548 14543  0 10:40 ? 00:00:00 postgres: autovacuum launcher
501 14549 14543  0 10:40 ? 00:00:00 postgres: stats collector
501 14550 14543  0 10:40 ? 00:00:00 postgres: logical replication launcher
```

9. Schließlich haben wir noch eine Testdatenbank mit dem Namen „hanser“ angelegt:

```
$ /usr/local/pgsql/bin/createdb hanser
$ /usr/local/pgsql/bin/psql hanser
psql (14.1)
Type "help" for help.
hanser=# SELECT version();
                                  version
-------------------------------------------------------------------------
 PostgreSQL 14.1 on x86_64-apple-darwin18.2.0, compiled by Apple LLVM version 10.0.1
(clang-1001.0.46.4), 64-bit
(1 row)
```

Damit ist die Installation aus dem Quellcode unter macOS abgeschlossen und der PostgreSQL-Server ist einsatzbereit.