

Projektarbeit: Erstellen einer Datenbank

Oliver Stählin
Dezember 2018

Inhalt

1. Zielsetzung.....	2
2. Eingesetzte Hard- und Software.....	2
3. Erstellen eines neuen Schemas	3
4. Download und Aufbereitung der Daten	5
5. Entwurf des Datenbankmodells	7
6. Erstellen der Tabellen im SQL Developer	8
7. Import der Daten aus den Excel-Tabellen	11
8. Aufbereitung der Hilfstabellen	14
9. Befüllen der Übergangstabelle.....	15
10. Befüllen der endgültigen Tabellen	19
11. Löschen der nicht mehr benötigten Objekte und Setzen der Constraints.....	21
12. Das fertige Entity-Relationship-Diagramm.....	22
13. Diskussion.....	25
Verletzung der Normalform durch redundante Daten und Beispiel-Export von einem Abfrage- Ergebnis	25
Weitere Verletzung der Normalform durch 1:1-Beziehungen.....	26
Abfrage-Beispiel: Bücher, die sowohl von ZEIT wie auch von Hellmuth Karasek vorgeschlagen wurden:	27
Weiteres Abfrage-Beispiel: Autoren sortiert nach Anzahl der empfohlenen Bücher.....	27

1. Zielsetzung

Erstellung eines neuen Datenbank-Schemas zur Wiederholung und Vertiefung des Stoffes.

2. Eingesetzte Hard- und Software

Hardware

Prozessor: Intel Core i7-5700HQ mit 2 x 2,70 GHz
Arbeitsspeicher: 16,0 GB

Software

Tabelle 1: verwendete Software

Typ	Software	Version
Betriebssystem	Windows 10 Education	64-Bit-Betriebssystem
Virtual Machine Monitor (auch Hypervisor genannt)	Oracle VirtualBox	5.2.8
Virtuelles Betriebssystem	Database App Development VM basierend auf Oracle Linux 7	DeveloperDaysVM2018-10-16_09.ova
SQL Datenbank	Oracle Database 18.3	Oracle Database 18c - Enterprise Edition Release 18.0.0.0.0 - Production Version 18.3.0.0.0
SQL-Entwicklungsumgebung	SQL Developer	18.3.0.277.2354
Tabellenkalkulationsprogram	Microsoft Excel	Microsoft Office Professional Plus 2016
Datenbankmanagementsystem	Microsoft Access	Microsoft Office Professional Plus 2016

3. Erstellen eines neuen Schemas

Zuerst muss im SQL Developer eine Verbindung als Datenbankadministrator eingerichtet werden, mit der man von der Entwicklungsumgebung aus als ‚SYSDBA‘ auf die Datenbank im Server zugreifen kann. Siehe Abbildungen 1, 2 und 3 für die notwendigen Schritte.

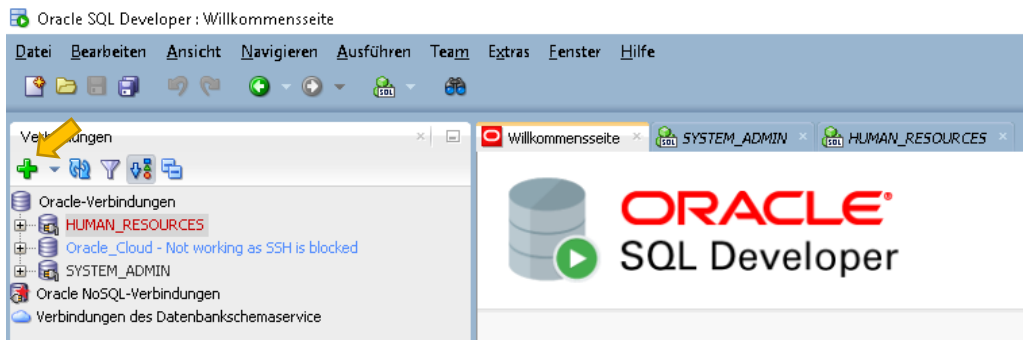


Abbildung 1. Öffnen des Datenverbindungs-Werkzeugs

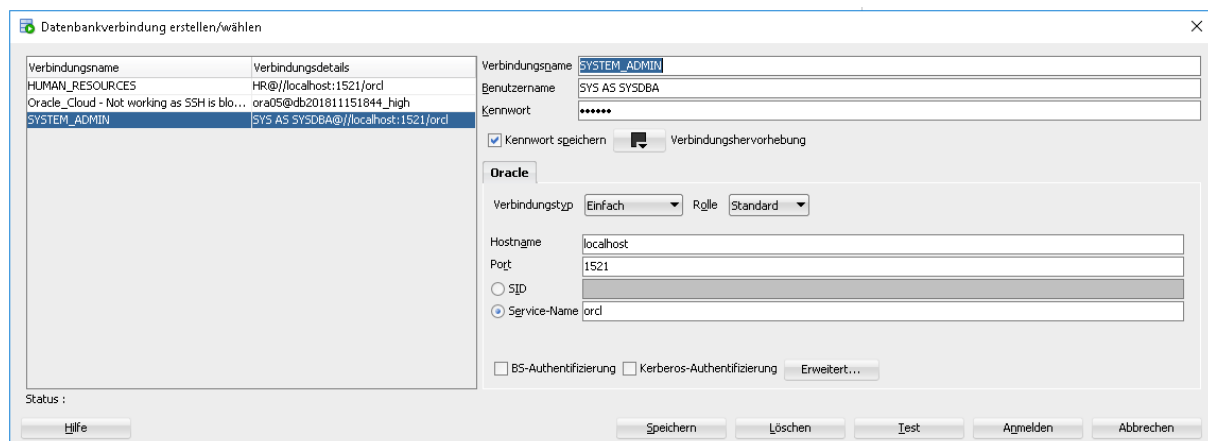


Abbildung2: Erstellen einer neuen Verbindung mit dem Benutzernamen SYS AS SYSDBA. Das Passwort ist wie bei allen Oracle VirtualBox-Anwendungen „oracle“.

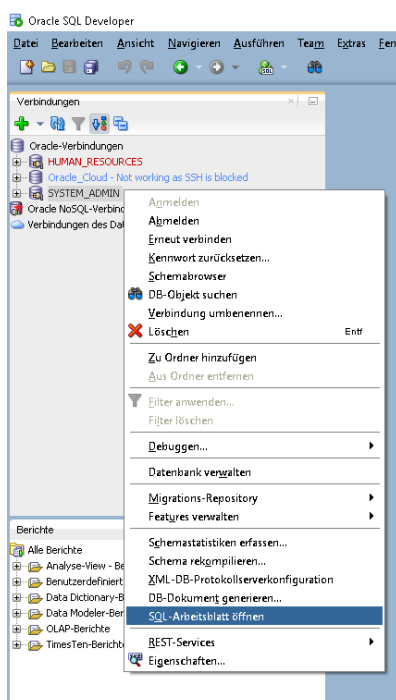


Abbildung3. Öffnen eines neuen SQL-Arbeitsblatts für die SYSTEM-ADMIN-Verbindung.

Anschließend kann in der Rolle des SYSDBA ein neuer Benutzer angelegt werden und diesem die nötigen Rechte für den Aufbau der Tabellen zugewiesen werden (siehe auch Skript-Datei 001_Erstellung_eines_neuen_Benutzers.sql).

Skript 1. Erstellung eines neuen Benutzers und Schemas

```
/* Creation of a new schema
based on information from:
https://stackoverflow.com/questions/33527917/i-am-trying-to-create-new-schema-
in-oracle-sql-developer
(Accessed: 2018-12-12)

Script has to be implemented as SYSDBA as you need Database-Administrator (DBA)
Privileges !
*/

-- Create role with specific privileges:
CREATE ROLE schema_user;
-- Grant user system privileges to this role:
GRANT
    CREATE SESSION,
    CREATE TABLE,
    CREATE SEQUENCE,
    CREATE VIEW,
    CREATE PROCEDURE,
TO schema_user;

-- Create user:
CREATE USER BK IDENTIFIED BY oracle;
-- Apply privileges from role "schema_user" to user "BK":
GRANT schema_user TO BK;
GRANT UNLIMITED TABLESPACE TO BK;

-- Verify schema creation:
SELECT username, account_status FROM dba_users WHERE username = 'BK';
```

Nach Ausführung des Skripts kann nun eine neue Verbindung im SQL Developer für den Benutzer BK erstellt werden (siehe Abbildung 4).

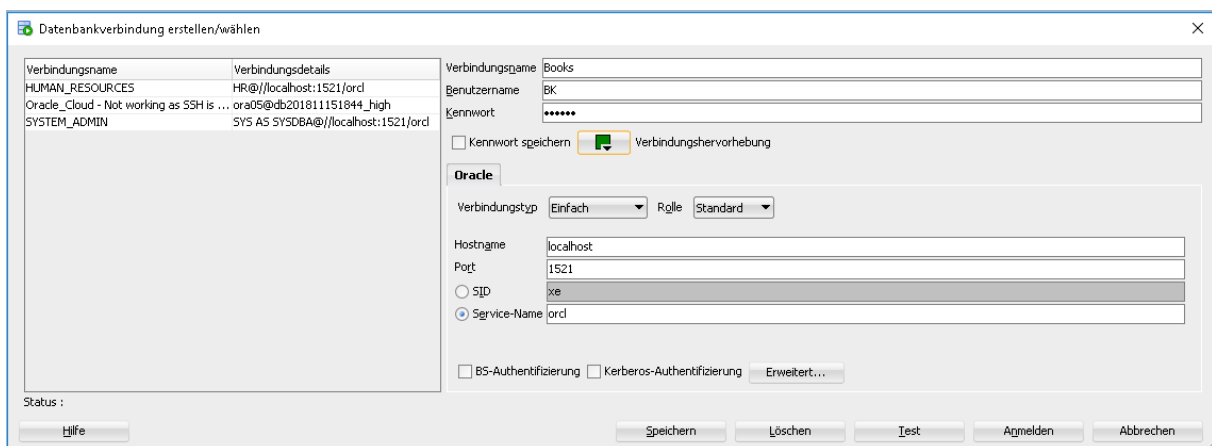


Abbildung4. Errichtung einer Verbindung für das Schema „Books“

4. Download und Aufbereitung der Daten

Als Datenquellen dienen vier Bücherlisten, die im Internet frei verfügbar sind. Alle Daten wurden am 13.12.2018 abgerufen (siehe Tabelle 1 bis 4).

Tabelle 2: Liste mit Leseempfehlungen von Hellmuth Karasek. Der Literaturkritiker hatte kurz vor seinem Tod für die Bild-Zeitung eine Liste von 25 deutschen Bücher zusammengestellt, die er für besonders Lesenwert hält

Titel	25 Bücher, die jeder gelesen haben sollte
Herausgeber	Hellmuth Karasek (1934 – 2015)
Beschreibung	Daten im Textformat als Teil eines Zeitungsartikels
Quelle	https://www.bild.de/unterhaltung/leute/hellmuth-karasek/auf-deutsch-die-jeder-gelesen-haben-sollte-42700230.bild.html

Tabelle 3: Liste mit Leseempfehlungen der deutschen Wochenzeitung ZEIT. Die Liste enthält Bücher aller Jahrhunderte und aus verschiedenen Ländern

Titel	ZEIT-Bibliothek der 100 Bücher
Herausgeber	Die Zeit
Beschreibung	Daten als Teil einer Wikipedia-Tabelle
Quelle	https://de.wikipedia.org/wiki/ZEIT-Bibliothek_der_100_B%C3%BCher

Tabelle 4: Liste mit Leseempfehlungen der französischen Tageszeitung Le Monde. Die Liste enthält nur Bücher die im 20. Jahrhundert veröffentlicht wurden.

Titel	Die 100 Bücher des Jahrhunderts von Le Monde
Herausgeber	Le Monde
Beschreibung	Daten als Teil einer Wikipedia-Tabelle
Quelle	https://de.wikipedia.org/wiki/Die_100_B%C3%BCher_des_Jahrhunderts_von_Le_Monde https://fr.wikipedia.org/wiki/Les_cent_livres_du_si%C3%A8cle

Tabelle 5: Liste mit Lesempfehlungen der britischen Rundfunkanstalt BBC. Die Liste enthält nur britische Bücher.

Titel	BBC-Auswahl der 100 bedeutendsten britischen Romane
Herausgeber	BBC
Beschreibung	Daten als Teil einer Wikipedia-Tabelle
Quelle	https://de.wikipedia.org/wiki/BBC-Auswahl_der_100_bedeutendsten_britischen_Romane

Die Listen aus Tabelle 3, 4 und 5 wurden mittels der Daten-Importfunktion von Excel heruntergeladen (siehe Abbildung 5). Für die Liste von Hellmuth Karasek (Tabelle 2) war das nicht möglich, da Excel aufgrund der Java-Inhalte auf der Webseite eine Fehlermeldung erzeugte. Die Daten wurden daher von Hand mittels Kopieren & Einfügen in ein Tabellenblatt übertragen.



Abbildung 5. Importieren von Daten von einer Webseite in ein Excel-Tabellenblatt.

Bei allen Listen wurde der zusätzliche Text von den Webseiten aus den Excel-Tabellenblättern gelöscht und die Spalten zum Teil neu geordnet und die Spaltennamen vereinheitlicht. Bei allen Tabellen wurde die Sprache des Autors hinzugefügt. Für mehrbändige Romane wurde neben dem Erscheinungsjahr des ersten Buches (Jahr1) noch ein zweites Datum für die Veröffentlichung des letzten Bandes (Jahr2) eingefügt. Siehe Abbildung 6 für ein Beispiel einer bereinigten Tabelle. Die Daten befinden sich in der Datei Rohdaten.xlsx.

	A	B	C	D	E	F	G
1	Nummer_import	Autor	Titel	Jahr1_import	Jahr2_import	Rezensent	Sprache
2	1	diverse	Die Bibel			Rudolf Augstein	verschiedene
3	2	Homer	Odyssee	-800		Herbert Bannert	Altgriechisch
4	3	Platon	Apologie	-399		Urs Jaeggi	Altgriechisch
5	4	Vergil	Aeneis	-19		Bernhard Kytzler	Lateinisch
6	5	Tacitus	Germania	98		Heinrich Böll	Lateinisch
7	6	Longos	Daphnis und Chloe	200		Bernhard Kytzler	Altgriechisch
8	7	Augustinus	Bekenntnisse	400		Golo Mann	Lateinisch
9	8	(Antoine Galland, Übersetzer, Herausgeber)	Die Erzählungen aus den tausendundein Nächten			Iring Fetscher	Arabisch
10	9	Wolfram von Eschenbach	Parzival	1200	1210	Peter Wapnewski	Deutsch
11	10	Gottfried von Straßburg	Tristan	1210		Peter Wapnewski	Deutsch
12	11	unbekannt	Das Nibelungenlied	1200		Peter Wapnewski	Deutsch
13	12	Dante Alighieri	Die Göttliche Komödie	1320		Horst Rüdiger	Italienisch

Abbildung 6. Ausschnitt einer bereinigten Tabelle.

5. Entwurf des Datenbankmodells

Das Datenbankmodell wurde zuerst mit Microsoft Access skizziert, da es dort einfach möglich ist, sich ein Entity-Relationship-Diagramm anzeigen zu lassen. Der Import der Daten in SQL erfolgt zuerst in mehrere „Hilfstabellen“. Diese Hilfstabellen werden dann zu einer großen Übergangstabelle zusammengefasst (Abbildung 7). Anschließend werden die Daten dann in die endgültigen Tabellen übertragen (Abbildung 8).

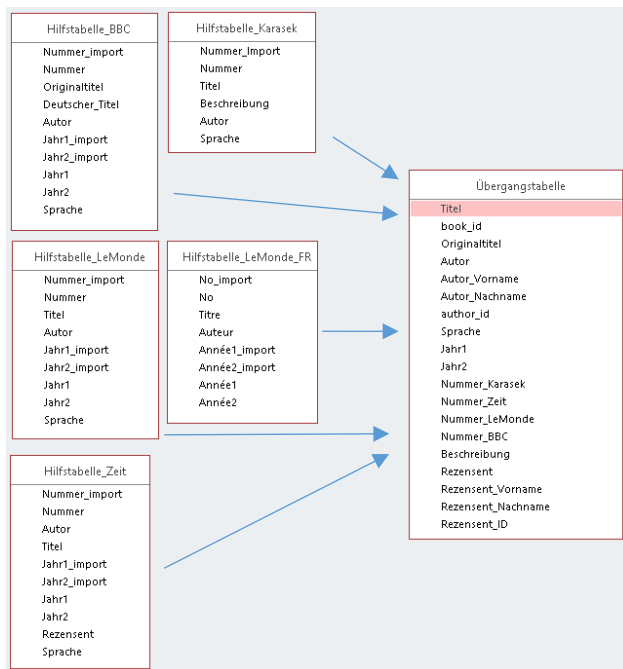


Abbildung 7. Schema der Hilfstabellen und der Übergangstabelle. Die Pfeile deuten den Weg der Datenübertragung an.

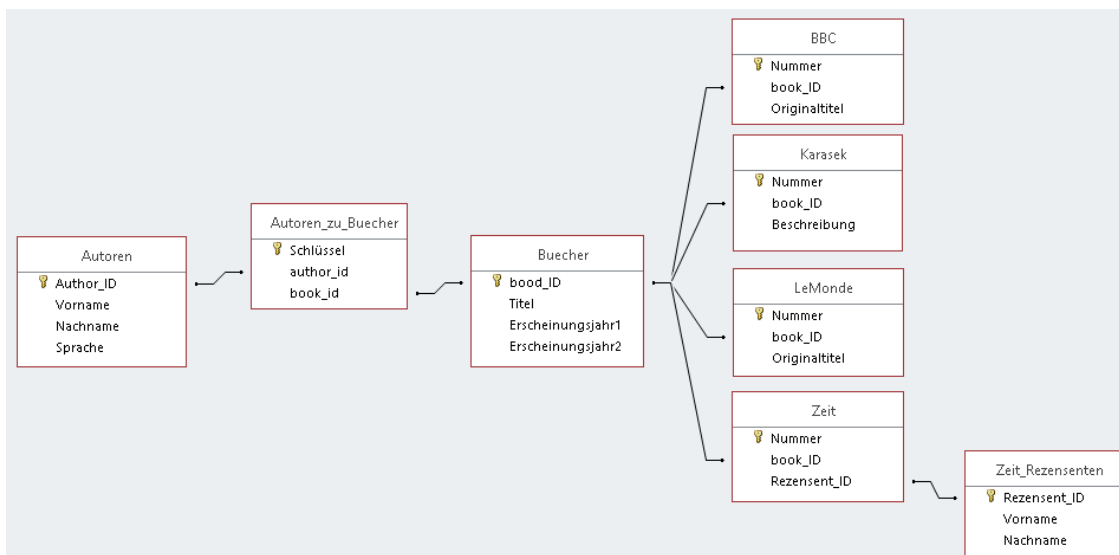


Abbildung 8. Entity-Relationship-Diagramm der endgültigen Tabellen.

6. Erstellen der Tabellen im SQL Developer

Als nächster Schritt werden die skizzierten Tabellen im SQL Developer erstellt. Die drei Skripte sind auch in der Datei 002_Erstellen_der_Tabellen.sql zusammengefasst.

Skript 2. Erstellung der Hilfstabellen

```
-- Erstellen der Hilfstabellen für den Datenimport

CREATE TABLE Hilfstabelle_BBC(
    Nummer_import CHAR(3),
    Nummer NUMBER(3),
    Originaltitel VARCHAR2(200),
    Deutscher_Titel VARCHAR2(200),
    Autor VARCHAR2(100),
    Jahr1_import CHAR(4),
    Jahr2_import CHAR(4),
    Jahr1 CHAR(4),
    Jahr2 CHAR(4),
    Sprache VARCHAR2(15));

CREATE TABLE Hilfstabelle_Karasek(
    Nummer_import CHAR(2),
    Nummer NUMBER(2),
    Titel VARCHAR2(200),
    Beschreibung VARCHAR2(1000),
    Autor VARCHAR2(100),
    Sprache VARCHAR2(15));

CREATE TABLE Hilfstabelle_LeMonde(
    Nummer_import CHAR(3),
    Nummer NUMBER(3),
    Titel VARCHAR2(200),
    Autor VARCHAR2(100),
    Jahr1_import CHAR(4),
    Jahr2_import CHAR(4),
    Jahr1 CHAR(4),
    Jahr2 CHAR(4),
    Sprache VARCHAR2(15));

CREATE TABLE Hilfstabelle_LeMonde_FR(
    No_import CHAR(3),
    No NUMBER(3),
    Titre VARCHAR2(200),
    Auteur VARCHAR2(100),
    Année1_import CHAR(4),
    Année2_import CHAR(4),
    Année1 NUMBER(4),
    Année2 NUMBER(4));

CREATE TABLE Hilfstabelle_Zeit(
    Nummer_import CHAR(3),
    Nummer NUMBER(3),
    Autor VARCHAR2(200),
    Titel VARCHAR2(1000),
    Jahr1_import CHAR(4),
    Jahr2_import CHAR(4),
    Jahr1 NUMBER(4),
    Jahr2 NUMBER(4),
    Rezensent VARCHAR(100),
    Sprache VARCHAR2(15));
```


Skript 3. Erstellung der Übergangstabelle

```
-- Erstellung der Übergangstabelle
CREATE TABLE ÜBERGANGSTABELLE(
  Titel          VARCHAR2(200),
  book_id        NUMBER(13),
  Originaltitel   VARCHAR2(200),
  Autor          VARCHAR2(100),
  Autor_Vorname   VARCHAR2(50),
  Autor_Nachname  VARCHAR2(25),
  author_id      char(12),
  Sprache        VARCHAR2(15),
  Jahr1          NUMBER(4),
  Jahr2          NUMBER(4),
  Nummer_Karasek  NUMBER(2),
  Nummer_Zeit     NUMBER(3),
  Nummer_LeMonde  NUMBER(3),
  Nummer_BBC      NUMBER(3),
  Beschreibung    VARCHAR2(1000),
  Rezensent       VARCHAR2(100),
  Rezensent_Vorname VARCHAR2(50),
  Rezensent_Nachname VARCHAR2(25),
  rezensent_ID    char(12));

-- Überprüfen, ob alle Tabellen vorhanden sind:
SELECT table_name from all_tables where owner = 'BK';

-- Code zum Erstellen der Löschbefehle, falls Tabellen neu aufgesetzt
-- werden müssen:
-- select 'drop table '||table_name||' cascade constraints;' from user_tables;
```

Skript 4. Erstellung der Haupttabellen

```
-- Erstellung der Haupttabellen

CREATE TABLE Autoren(
    author_ID    CHAR(12),
    Vorname      VARCHAR2(50),
    Nachname     VARCHAR2(25) NOT NULL,
    Sprache      VARCHAR2(15));

CREATE TABLE Autoren_zu_Buecher(
    Schlüssel    NUMBER(3),
    author_id    CHAR(12),
    book_ID      NUMBER(13));

CREATE TABLE Buecher(
    book_ID      NUMBER(13),
    Titel        VARCHAR2(200),
    Erscheinungsjahr1  NUMBER(4),
    Erscheinungsjahr2  NUMBER(4));

CREATE TABLE BBC(
    Nummer       NUMBER(3),
    book_ID      NUMBER(13),
    Originaltitel VARCHAR(200));

CREATE TABLE Karasek(
    Nummer       NUMBER(2),
    book_ID      NUMBER(13),
    Beschreibung  VARCHAR2(1000));

CREATE TABLE LeMonde(
    Nummer       NUMBER(3),
    book_ID      NUMBER(13),
    Originaltitel VARCHAR(200));

CREATE TABLE Zeit(
    Nummer       NUMBER(3),
    book_ID      NUMBER(13),
    rezensent_ID CHAR(12));

CREATE TABLE Zeit_Rezensenten(
    rezensent_ID CHAR(12),
    Vorname      VARCHAR2(25),
```

7. Import der Daten aus den Excel-Tabellen

Der nächste Schritt ist der Import der Daten in die Hilfstabellen. Der SQL Developer bietet hierfür ein eigenes Werkzeug. Dazu muss man im Fenster mit den Verbindungen mit der rechten Maustaste auf den Tabellennamen klicken und dann „Daten importieren ...“ auswählen. Siehe auch Abbildung 9.

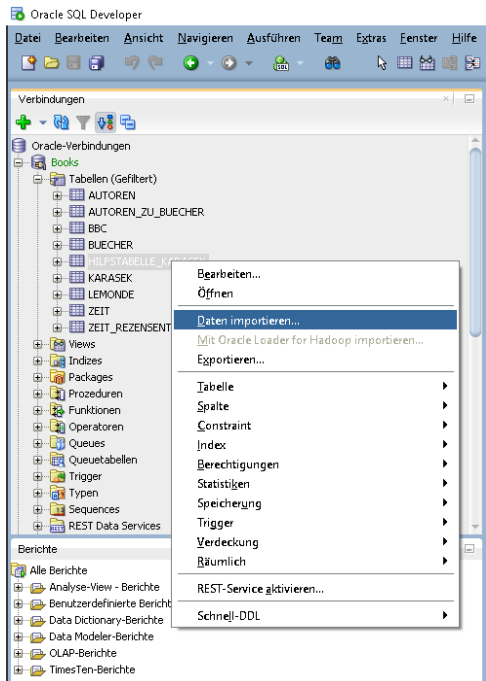


Abbildung 9. Aufrufen des Datenimportassistenten des SQL Developers.

Der weitere Import erfolgt dann über den Datenimportassistenten (Abbildung 10 bis 14). Da die Spaltennamen in SQL so gewählt wurden, dass sie exakt mit denen aus den Exceltabellen übereinstimmen, kann im Schritt 4 (Abbildung 13) die Option „Übereinstimmung nach Namen“ gewählt werden.

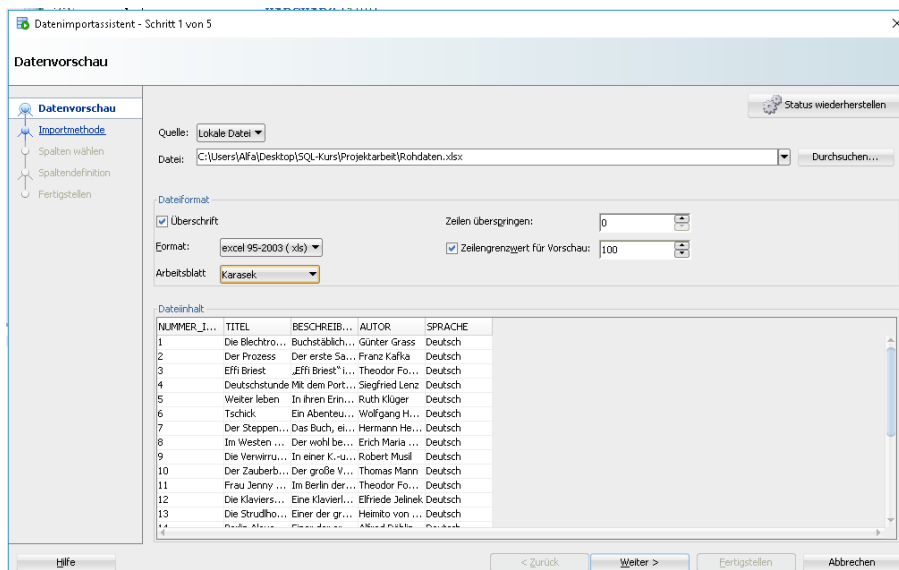


Abbildung 10. Startseite des Datenimportassistenten des SQL Developers.

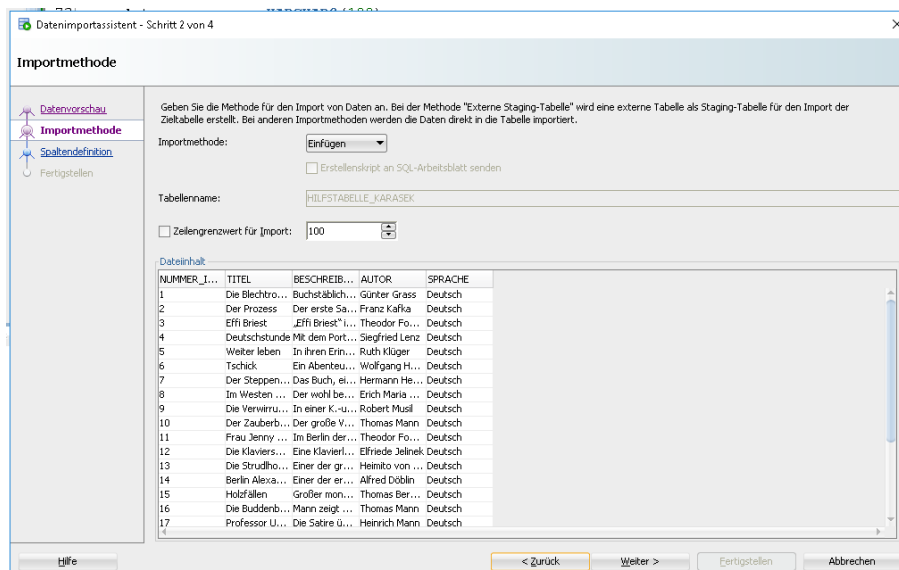


Abbildung 11. Schritt 2 des Datenimportassistenten des SQL Developers.

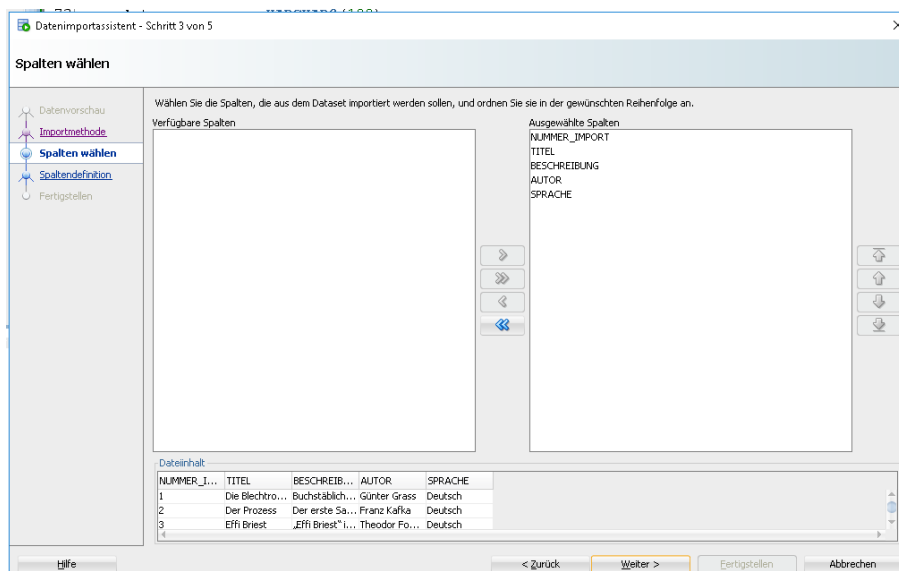


Abbildung 12. Schritt 3 des Datenimportassistenten des SQL Developers.

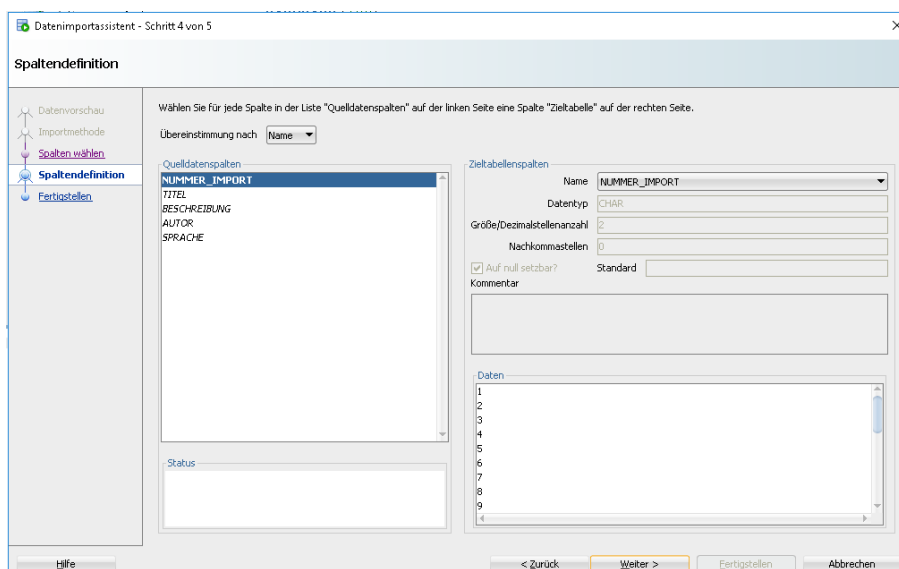


Abbildung 13. Schritt 4 des Datenimportassistenten des SQL Developers.

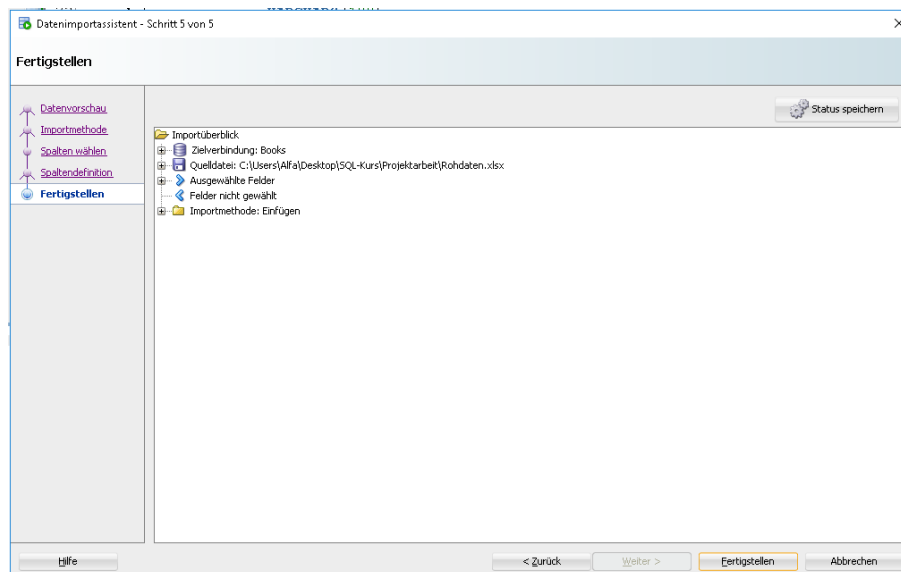


Abbildung 14. Schritt 5 des Datenimportassistenten des SQL Developers.

Zu beachten ist, dass die Jahreszahlen und die Nummer-Spalten der Exceltabellen nur in eine SQL-Spalte mit Datentyp CHAR oder VARCHAR2 importiert werden können, da der SQL Developer alle Excel-Felder als Text auffasst. Die Konvertierung in das richtige Format erfolgt später in SQL.

8. Aufbereitung der Hilfstabellen

Die Hilfstabellen müssen anschließend noch bearbeitet werden. Da der SQL Developer alle Spalten der Excel-Tabellen nur als CHAR oder VARCHAR2 importieren konnte, müssen die Jahreszahlen und Nummern-Werte noch in NUMBER konvertiert werden. Siehe auch die Datei

003_Aufbereitung_der_Hilfstabellen.sql.

Skript 5. Aufbereitung der Hilfstabellen

```
/* Aufbereitung der Hilfstabellen */

-- Hilfstabelle_BBC
-- Transfer Platz-Nummern in die neue Spalte
UPDATE hilfstabelle_bbc SET nummer = TO_NUMBER(nummer_import);
UPDATE hilfstabelle_bbc SET jahr1 = TO_NUMBER(jahr1_import);
UPDATE hilfstabelle_bbc SET jahr2 = TO_NUMBER(jahr2_import);
-- Löschen der nicht mehr benötigten Spalten
ALTER TABLE hilfstabelle_bbc DROP COLUMN nummer_import;
ALTER TABLE hilfstabelle_bbc DROP COLUMN jahr1_import;
ALTER TABLE hilfstabelle_bbc DROP COLUMN jahr2_import;
-- Ersetzen der '-' Einträge in der Titel-Spalte durch NULL
UPDATE hilfstabelle_bbc
SET deutscher_titel = NULL
WHERE deutscher_titel = '-';
-- Ergebnis ansehen:
SELECT * FROM hilfstabelle_bbc;

-- Hilfstabelle_Karasek
-- Transfer der Nummerierung in Zeile mit NUMBER-Format:
UPDATE hilfstabelle_karasek SET nummer = TO_NUMBER(nummer_import);
ALTER TABLE hilfstabelle_karasek DROP COLUMN nummer_import;
-- Ergebnis ansehen:
SELECT * FROM hilfstabelle_karasek;

-- Hilfstabelle_LeMonde
DESC hilfstabelle_LeMonde;
UPDATE hilfstabelle_lemonde SET nummer = TO_NUMBER(nummer_import);
UPDATE hilfstabelle_lemonde SET jahr1 = TO_NUMBER(jahr1_import);
UPDATE hilfstabelle_lemonde SET jahr2 = TO_NUMBER(jahr2_import);
ALTER TABLE hilfstabelle_lemonde DROP COLUMN nummer_import;
ALTER TABLE hilfstabelle_lemonde DROP COLUMN jahr1_import;
ALTER TABLE hilfstabelle_lemonde DROP COLUMN jahr2_import;
-- Ergebnis ansehen:
SELECT * FROM hilfstabelle_lemonde;

-- Hilfstabelle_LeMonde_FR
DESC hilfstabelle_lemonde_fr;
UPDATE hilfstabelle_lemonde_fr SET no = TO_NUMBER(no_import);
UPDATE hilfstabelle_lemonde_fr SET année1 = TO_NUMBER(année1_import);
UPDATE hilfstabelle_lemonde_fr SET année2 = TO_NUMBER(année2_import);
ALTER TABLE hilfstabelle_lemonde_fr DROP COLUMN no_import;
ALTER TABLE hilfstabelle_lemonde_fr DROP COLUMN année1_import;
ALTER TABLE hilfstabelle_lemonde_fr DROP COLUMN année2_import;
-- Ergebnis ansehen:
SELECT * FROM hilfstabelle_lemonde_fr;

-- Hilfstabelle_Zeit
DESC hilfstabelle_zeit;
UPDATE hilfstabelle_zeit SET nummer = TO_NUMBER(nummer_import);
UPDATE hilfstabelle_zeit SET jahr1 = TO_NUMBER(jahr1_import);
UPDATE hilfstabelle_zeit SET jahr2 = TO_NUMBER(jahr2_import);
ALTER TABLE hilfstabelle_zeit DROP COLUMN nummer_import;
ALTER TABLE hilfstabelle_zeit DROP COLUMN jahr1_import;
ALTER TABLE hilfstabelle_zeit DROP COLUMN jahr2_import;
-- Ergebnis ansehen:
SELECT * FROM hilfstabelle_zeit;
```

9. Befüllen der Übergangstabelle

Als nächstes wird die Übergangstabelle befüllt. Die einzelnen Skripte sind auch in der Datei 004_Befüllen_der_Übergangstabelle.sql zusammengefasst.

Skript 6. Befüllen der Übergangstabelle mit den Werten der Hilfstabelle_Karasek

```
-- Einfügen vom Inhalt von Hilfstabelle_Karasek
INSERT INTO übergangstabelle (titel, autor, sprache, nummer_karasek,
    beschreibung)
SELECT titel, autor, sprache, nummer, beschreibung FROM hilfstabelle karasek;
```

Skript 7. Befüllen der Übergangstabelle mit den Werten der Hilfstabelle_LeMonde

```
-- Hinzufügen vom Inhalt von Le Monde unter Vermeidung von Doppelungen
MERGE INTO übergangstabelle ü
USING (SELECT * FROM hilfstabelle_lemonde) lm
ON (ü.titel = lm.titel AND ü.autor = lm.autor AND ü.sprache = lm.sprache)
WHEN MATCHED THEN
UPDATE SET
    ü.jahr1 = lm.jahr1,
    ü.jahr2 = lm.jahr2,
    ü.nummer_lemonde = lm.nummer
WHEN NOT MATCHED THEN
INSERT VALUES (
    lm.titel,
    NULL,
    NULL,
    lm.autor,
    NULL,
    NULL,
    NULL,
    lm.sprache,
    lm.jahr1,
    lm.jahr2,
    NULL,
    NULL,
    lm.nummer,
    NULL,
    NULL,
    NULL,
    NULL,
    NULL,
    NULL);
-- Hinzufügen des Originaltitels für Werke, die von französisch-sprachigen
Autoren geschrieben wurden:
UPDATE übergangstabelle outer_table
SET originaltitel =
    ( SELECT titre
      FROM hilfstabelle_lemonde_fr inner_table
      WHERE inner_table.no = outer_table.nummer_lemonde
        AND inner_table.auteur = outer_table.autor);
-- Alle Originaltitel von nicht französisch-sprachigen Autoren werden wieder
NULL gesetzt:
UPDATE übergangstabelle
SET originaltitel = NULL
WHERE sprache != 'Französisch';
-- Test: Werke, die von Karasek und LeMonde empfohlen wurden:
SELECT * FROM übergangstabelle WHERE nummer_karasek IS NOT NULL AND
nummer_lemonde IS NOT NULL;
```

Skript 7. Befüllen der Übergangstabelle mit den Werten der Hilfstabelle_Zeit

```
-- Hinzufügen vom Inhalt von Zeit unter Vermeidung von Doppelungen
MERGE INTO übergangstabelle ü
USING (SELECT * FROM hilfstabelle_zeit) z
ON (ü.titel = z.titel AND ü.autor = z.autor AND ü.sprache = z.sprache)
WHEN MATCHED THEN
UPDATE SET
ü.jahr1 = z.jahr1,
ü.jahr2 = z.jahr2,
ü.nummer_zeit = z.nummer,
ü.rezensent = z.rezensent
WHEN NOT MATCHED THEN
INSERT VALUES (
    z.titel,
    NULL,
    NULL,
    z.autor,
    NULL,
    NULL,
    NULL,
    z.sprache,
    z.jahr1,
    z.jahr2,
    NULL,
    z.nummer,
    NULL,
    NULL,
    NULL,
    z.rezensent,
    NULL,
    NULL,
    NULL);
-- Test, welche Bücher auf sowohl von Karasek wie auch von der Zeit empfohlen
wurden:
SELECT * FROM übergangstabelle
WHERE nummer_karasek IS NOT NULL AND nummer_zeit IS NOT NULL;
-- Test, welche Bücher auf sowohl von Le Monde wie auch von der Zeit empfohlen
wurden:
SELECT * FROM übergangstabelle
WHERE nummer_lemonde IS NOT NULL AND nummer_zeit IS NOT NULL;
```


Skript 8. Befüllen der Übergangstabelle mit den Werten der Hilfstabelle_BBC

```
-- Hinzufügen vom Inhalt der BBC-Liste unter Vermeidung von Doppelungen
MERGE INTO übergangstabelle ü
USING (SELECT * FROM hilfstabelle_bbc) bbc
ON (ü.titel = bbc.deutscher_titel AND ü.autor = bbc.autor AND ü.sprache =
bbc.sprache)
WHEN MATCHED THEN
UPDATE SET
ü.originaltitel = bbc.originaltitel,
ü.jahr1 = bbc.jahr1,
ü.jahr2 = bbc.jahr2,
ü.nummer_bbc = bbc.nummer
WHEN NOT MATCHED THEN
INSERT VALUES(
    bbc.deutscher_titel,
    NULL,
    bbc.originaltitel,
    bbc.autor,
    NULL,
    NULL,
    NULL,
    bbc.sprache,
    bbc.jahr1,
    bbc.jahr2,
    NULL,
    NULL,
    NULL,
    bbc.nummer,
    NULL,
    NULL,
    NULL,
    NULL,
    NULL);
-- Test, welche Bücher auf sowohl von der BBC wie auch von der Zeit empfohlen
wurden:
SELECT * FROM übergangstabelle
WHERE nummer_bbc IS NOT NULL AND nummer_zeit IS NOT NULL;
```

Skript 9. Erstellen der author_ID und der book_ID

```
-- Definieren der author_ID
UPDATE übergangstabelle SET autor_vorname = SUBSTR(autor, 1, INSTR(autor, ' ', -1));
UPDATE übergangstabelle SET autor_nachname = SUBSTR(autor, INSTR(autor, ' ', -1)+1);
UPDATE übergangstabelle SET author_id = SUBSTR(autor_nachname, 1, 5) || SUBSTR(autor_vorname, 1, 3);

-- manuelle Überprüfung
SELECT COUNT(DISTINCT(autor)) FROM übergangstabelle;
SELECT COUNT(DISTINCT(author_id)) FROM übergangstabelle;
SELECT autor, autor_vorname, autor_nachname, author_id FROM übergangstabelle ORDER BY author_id;

-- Definieren der book_ID
-- Erstellen einer Sequenz für book_IDs
CREATE SEQUENCE book_ID_seq
START WITH 9780000000001
INCREMENT BY 1;
-- Einfügen der book_ID_numbers in die Übergangstabelle
-- da 'Asterix der Gallier' zwei Autoren hat, wird die Zeile für den Zweitautor Albert Uderzo ausgelassen
UPDATE übergangstabelle
SET book_ID = book_ID_seq.nextval
WHERE author_ID <> 'UderzAlb';
-- Einfügen der Book_ID für 'Asterix der Gallier' in die ausgelassene Zeile:
UPDATE übergangstabelle
SET book_ID = (SELECT book_id FROM übergangstabelle WHERE author_id = 'GosciRen')
WHERE author_id = 'UderzAlb';

-- Extraction der Rezensenten-Vornamen / Erstellen der Rezensenten-ID:
UPDATE übergangstabelle SET rezensent_vorname = SUBSTR(rezensent, 1, INSTR(rezensent, ' ', -1));
UPDATE übergangstabelle SET rezensent_nachname = SUBSTR(rezensent, INSTR(rezensent, ' ', -1)+1);
UPDATE übergangstabelle SET rezensent_id = SUBSTR(rezensent_nachname, 1, 5) || SUBSTR(rezensent_vorname, 1, 3);

-- visuelle Überprüfung der fertigen Tabelle:
SELECT * FROM übergangstabelle;
```

10. Befüllen der endgültigen Tabellen

Ausgehend von der Übergangstabelle werden jetzt die endgültigen Tabellen befüllt. Die einzelnen Skripte sind auch in der Datei 005_Befüllen_der_Haupttabellen.sql gespeichert.

Skript 10. Befüllen der Autor-Tabelle

```
-- Befüllen der Autoren-Tabelle
INSERT INTO autoren
SELECT
    DISTINCT author_id,
    autor_vorname,
    autor_nachname,
    sprache
FROM übergangstabelle;
-- Test:
SELECT * FROM autoren WHERE vorname IS NULL ORDER BY author_ID;
```

Skript 11. Befüllen der Autoren_zu_Buecher-Tabelle

```
-- Erzeugen einer Sequenz für den Primärschlüssel:
CREATE SEQUENCE autoren_zu_buecher_schlüssel_seq
START WITH 1
INCREMENT BY 1;
-- Einfügen der Werte:
INSERT INTO autoren_zu_buecher
SELECT
    autoren_zu_buecher_schlüssel_seq.nextval,
    author_id,
    book_id
FROM übergangstabelle;
-- Ansehen des Ergebnisses:
SELECT * FROM autoren_zu_buecher;
```

Skript 12. Befüllen der Buecher-Tabelle

```
INSERT INTO buecher
SELECT
    DISTINCT book_id,
    titel,
    jahr1,
    jahr2
FROM übergangstabelle ORDER BY book_id;
-- Test
SELECT * FROM buecher;
```

Skript 13. Befüllen der BBC-Tabelle

```
INSERT INTO BBC
SELECT
    nummer_bbc,
    book_id,
    originaltitel
FROM übergangstabelle WHERE nummer_BBC IS NOT NULL;
-- Ergebnis ansehen:
SELECT * FROM BBC;
```

Skript 14. Befüllen der Karasek-Tabelle

```
INSERT INTO karasek
SELECT
    nummer_karasek,
    book_id,
    beschreibung
FROM übergangstabelle WHERE nummer_karasek IS NOT NULL;
-- Ergebnis ansehen:
SELECT * FROM karasek;
```

Skript 15. Befüllen der LeMonde-Tabelle

```
INSERT INTO lemonde
SELECT
    DISTINCT nummer_lemonde,
    book_id,
    NULL
FROM übergangstabelle WHERE nummer_lemonde IS NOT NULL;
-- hinzufügen der Originaltitel für Bücher von französischsprachigen Autoren:
UPDATE lemonde outer_table
SET outer_table.originaltitel =
    ( SELECT inner_table.originaltitel
      FROM übergangstabelle inner_table
      WHERE inner_table.book_id = outer_table.book_id
        AND inner_table.sprache = 'Französisch'
        AND inner_table.autor != 'Albert Uderzo' );
-- Ergebnis ansehen:
SELECT * FROM lemonde;
```

Skript 16. Befüllen der ZEIT-Tabelle

```
INSERT INTO zeit
SELECT
    nummer_zeit,
    book_id,
    rezensent_ID
FROM übergangstabelle WHERE nummer_zeit IS NOT NULL;
-- Ergebnis ansehen:
SELECT * FROM zeit;
```

Skript 17. Befüllen der ZEIT-Rezensenten-Tabelle

```
INSERT INTO zeit_rezensenten
SELECT
    DISTINCT rezensent_id,
    rezensent_vorname,
    rezensent_nachname
FROM übergangstabelle WHERE rezensent IS NOT NULL;
-- Ergebnis ansehen:
SELECT * FROM zeit_rezensenten;
```

11. Löschen der nicht mehr benötigten Objekte und Setzen der Constraints

Die nicht mehr benötigten Tabellen und Sequenzen werden jetzt gelöscht. Auf den verbliebenen Tabellen die Constraints in Form von Primär- und Fremdschlüsseln definiert. Die Skripte sind auch in der Datei 006_Aufräumen_und_Constraints.sql zu finden.

Skript 18. Löschen der nicht mehr benötigten Objekte

```
-- Tabellen:
select 'drop table ' || table_name || ' cascade constraints;' from user_tables
ORDER BY table_name;
drop table HILFSTABELLE_BBC cascade constraints;
drop table HILFSTABELLE_KARASEK cascade constraints;
drop table HILFSTABELLE_LEMONDE cascade constraints;
drop table HILFSTABELLE_LEMONDE_FR cascade constraints;
drop table HILFSTABELLE_ZEIT cascade constraints;
drop table ÜBERGANGSTABELLE cascade constraints;

-- Sequenzen:
SELECT 'drop sequence ' || sequence_name || ';' FROM user_sequences;
drop sequence AUTOREN_ZU_BUECHER_SCHLÜSSEL_SEQ;
drop sequence BOOK_ID_SEQ;
```

Skript 19. Setzen der Primary Keys

```
ALTER TABLE autoren ADD CONSTRAINT autoren_pk PRIMARY KEY (author_id);
ALTER TABLE autoren_zu_buecher ADD CONSTRAINT autoren_zu_buecher_pk PRIMARY KEY
(schlüssel);
ALTER TABLE buecher ADD CONSTRAINT buecher_pk PRIMARY KEY (book_id);
ALTER TABLE bbc ADD CONSTRAINT bbc_pk PRIMARY KEY (nummer);
ALTER TABLE karasek ADD CONSTRAINT karasek_pk PRIMARY KEY (nummer);
ALTER TABLE lemonde ADD CONSTRAINT lemonde_pk PRIMARY KEY (nummer);
ALTER TABLE zeit ADD CONSTRAINT zeit_pk PRIMARY KEY (nummer);
ALTER TABLE zeit_rezensenten ADD CONSTRAINT zeit_rezensenten_pk PRIMARY KEY
(rezensent_id);
```

Skript 19. Setzen der Foreign Keys

```
ALTER TABLE autoren_zu_buecher ADD CONSTRAINT autoren_zu_buecher_author_id_fk
FOREIGN KEY (author_id) REFERENCES autoren(author_id);
ALTER TABLE autoren_zu_buecher ADD CONSTRAINT autoren_zu_buecher_book_id_fk
FOREIGN KEY (book_id) REFERENCES buecher(book_id);
ALTER TABLE bbc ADD CONSTRAINT bbc_book_id_fk FOREIGN KEY (book_id) REFERENCES
buecher(book_id);
ALTER TABLE karasek ADD CONSTRAINT karasek_book_id_fk FOREIGN KEY (book_id)
REFERENCES buecher(book_id);
ALTER TABLE lemonde ADD CONSTRAINT lemonde_book_id_fk FOREIGN KEY (book_id)
REFERENCES buecher(book_id);
ALTER TABLE zeit ADD CONSTRAINT zeit_book_id_fk FOREIGN KEY (book_id)
REFERENCES buecher(book_id);
ALTER TABLE zeit ADD CONSTRAINT zeit_rezensent_id_fk FOREIGN KEY (rezensent_id)
REFERENCES zeit_rezensenten(rezensent_id);
```

12. Das fertige Entity-Relationship-Diagramm

Über den Data Modeler lässt sich das fertig gestellte Entity-Relationship Diagramm anzeigen. Dies geht im SQL Developer über Datei -> Data Modeler -> Importieren -> Data Dictionary -> [Books auswählen] weiter -> [BK auswählen] weiter -> [Tabellen auswählen] weiter -> Fertigstellen.

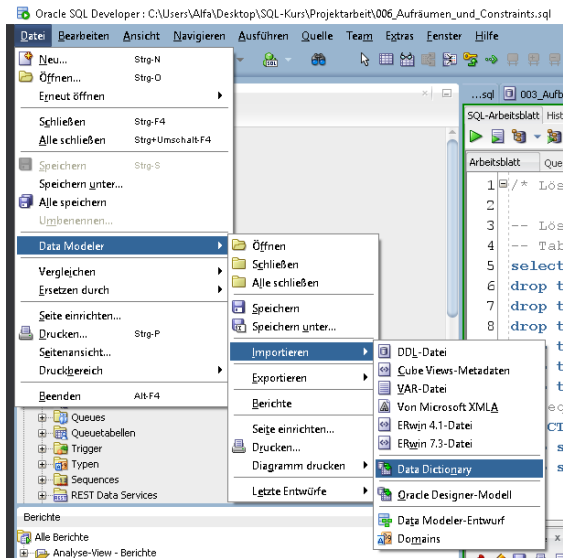


Abbildung 15. Aufruf des Data Modeler über die Data Dictionary Import-Funktion

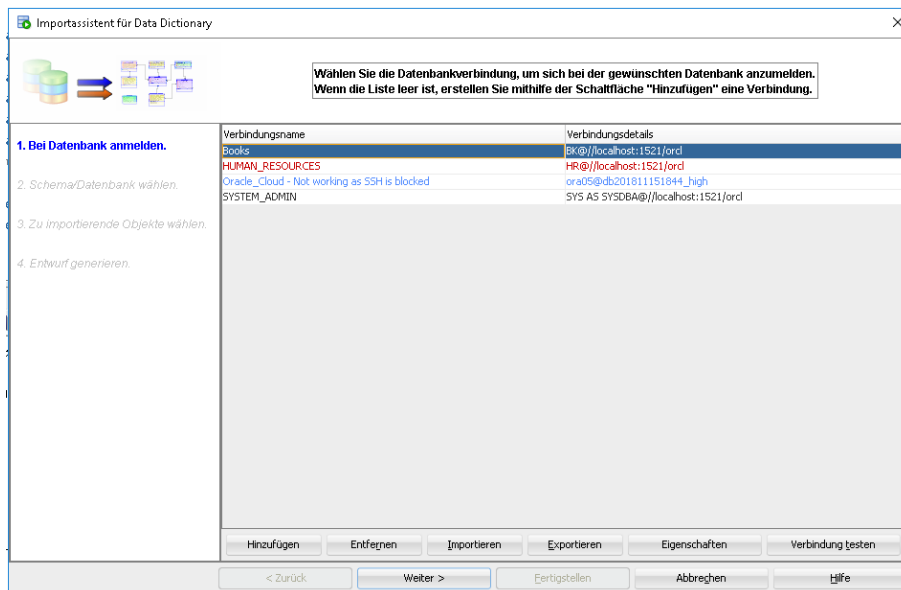


Abbildung 16. Schritt 1 des Data Modeler Importassistenten

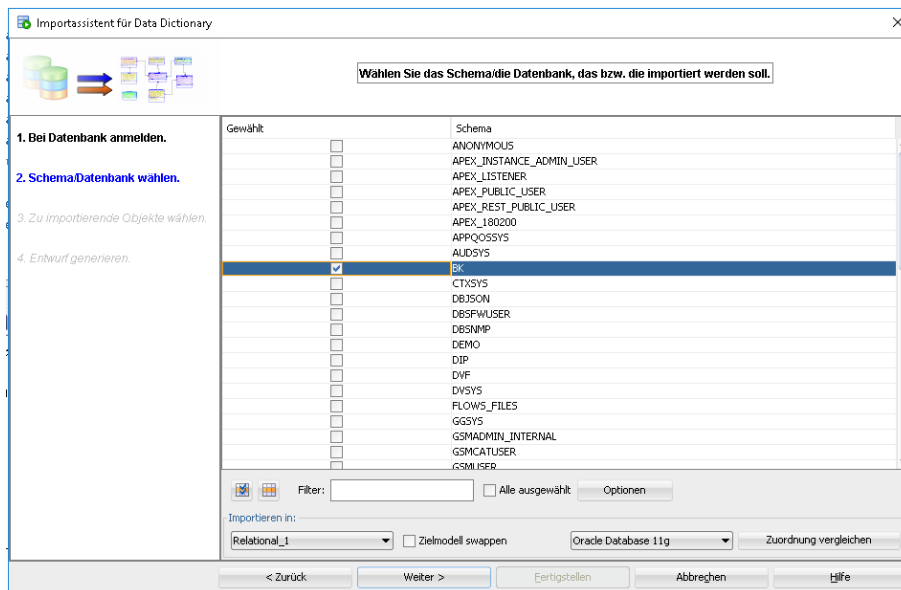


Abbildung 17. Schritt 2 des Data Modeler Importassistenten

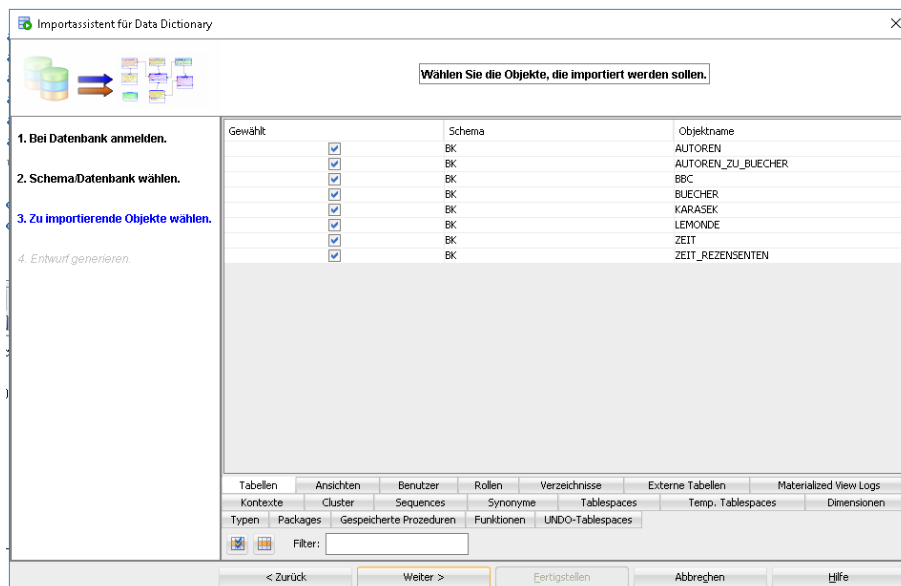


Abbildung 18. Schritt 3 des Data Modeler Importassistenten

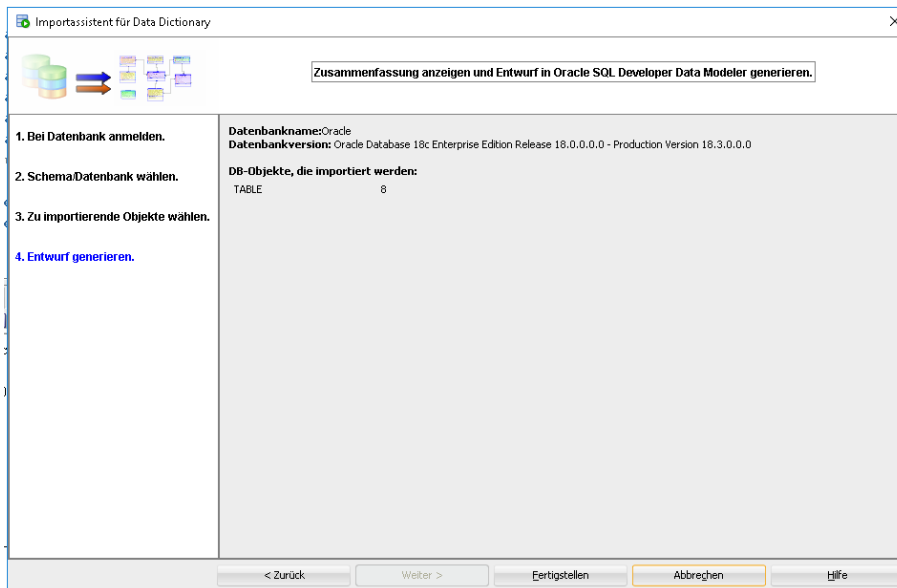


Abbildung 19. Schritt 4 des Data Modeler Importassistenten

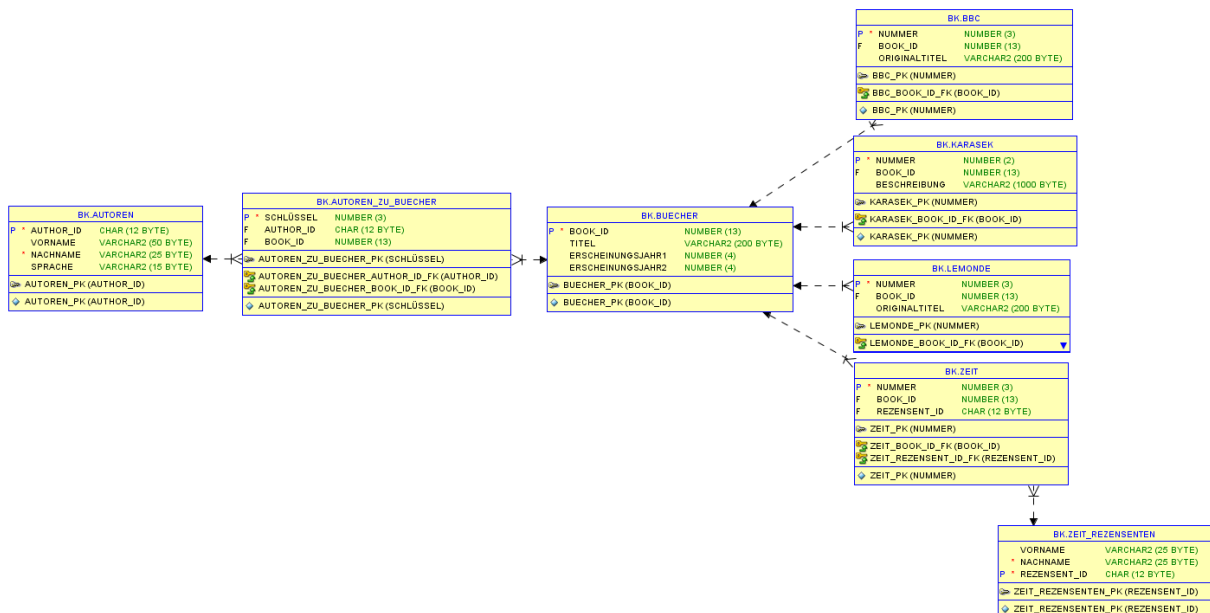


Abbildung 20. Das fertige Entity-Relationship-Modell so wie es vom Data Modeler erstellt wurde.

13. Diskussion

Die Datenbank ist nun bereit für verschiedene Abfragen.

Verletzung der Normalform durch redundante Daten und Beispiel-Export von einem Abfrage-Ergebnis

Durch eine einfache Abfrage lässt sich zeigen, dass das Entity-Relationship-Schema eine Redundanz enthält und damit gegen die Idee der Normalformen verstößt. Die folgende Abfrage sucht nach Personen, die sowohl in der Autoren-Tabelle als auch in der Rezensenten-Tabelle auftauchen:

Skript 20. Abfrage-Skript

```
SELECT au.vorname, au.nachname
FROM autoren au JOIN zeit_rezensenten ze
ON author_id = rezensent_id;
```

Es zeigt sich, dass zwei Autoren gleichzeitig auch Rezensenten waren. Das Ergebnis lässt sich über den SQL Developer als Excel-Datei exportieren.

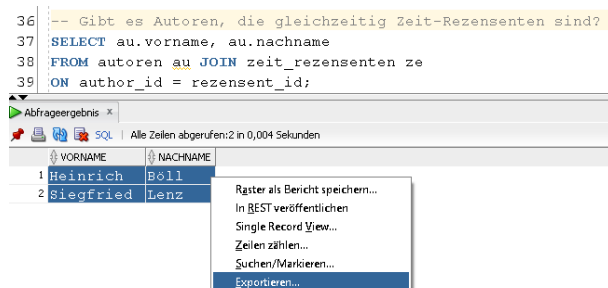


Abbildung 21. Export eines Abfrage-Ergebnisses. Im Schritt 1 wird per Rechtsklick auf die Ergebnistabelle im SQL Developer die Exportfunktion aufgerufen.

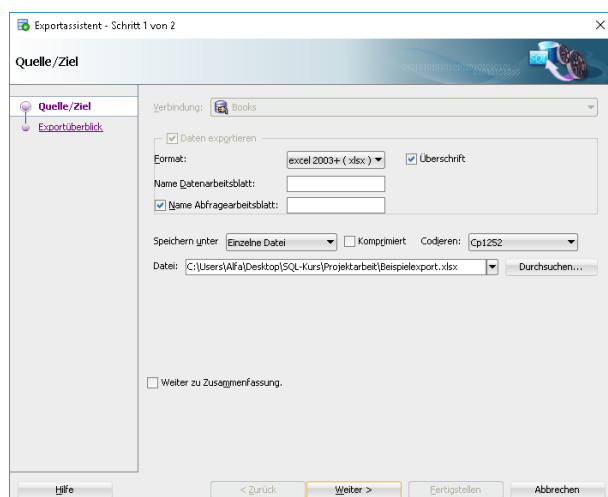
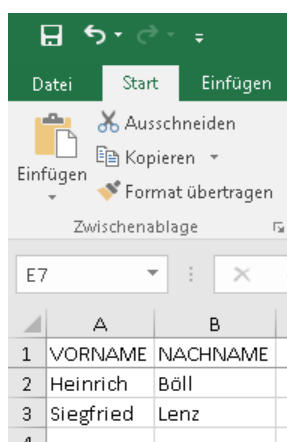
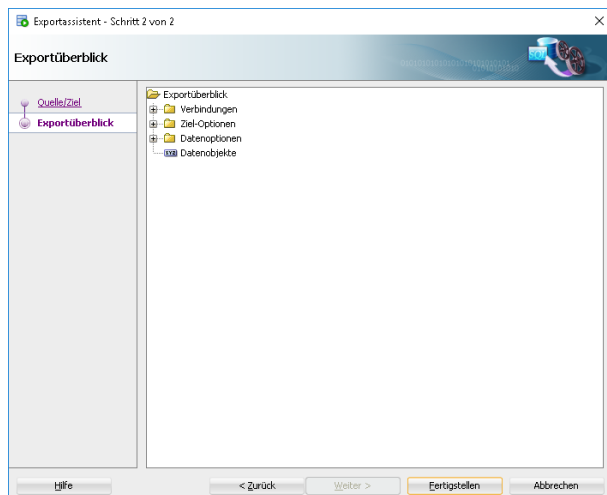


Abbildung 22. Export eines Abfrage-Ergebnisses – Schritt 2



Heinrich Böll und Siegfried Lenz tauchen also sowohl als Autoren als auch als Rezensenten auf. Das Wiederholungsverbot (1. Normalform) der Datenbank ist deshalb verletzt. Man könnte dies lösen, indem man statt den Tabellen AUTOREN und ZEIT-REZENSENTEN eine Tabelle PERSONEN erstellt und dort alle Personen mit einer `personen_ID` versieht. Sowohl die `author_id` aus der AUTOREN_ZU_BUECHER -Tabelle wie auch die `rezensent_id` der ZEIT-REZENSENTEN könnten dann als Fremdschlüssel zu dieser `personen_ID` verwendet werden.

Weitere Verletzung der Normalform durch 1:1-Beziehungen

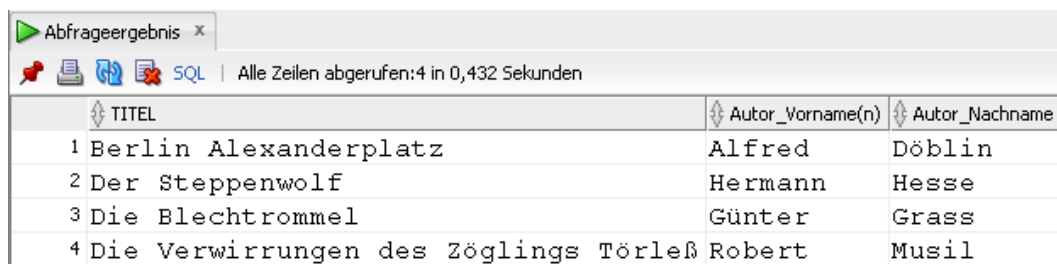
Streng genommen sind die Tabellen KARASEK, BBC und LEMONDE nicht nötig. Der Inhalt dieser Tabellen steht jeweils in einer 1:1 Relation zur Tabelle BUECHER. Man könnte folglich die Spalte mit den Originaltiteln und die Spalte mit dem Beschreibungstext auch in die BUECHER-Tabelle integrieren. Zur besseren Übersichtlichkeit wurde hier von der strikten Logik des relationalen Datenbanksystems abgewichen.

Abfrage-Beispiel: Bücher, die sowohl von ZEIT wie auch von Hellmuth Karasek vorgeschlagen wurden:

Die Datenbank kann jetzt für diverse Abfragen benutzt werden. Als Beispiel wird hier abgefragt, welche Bücher sowohl von Hellmuth Karasek wie auch von der ZEIT empfohlen wurden.

Skript 21. Abfrage-Skript

```
SELECT
    bu.titel,
    au.vorname AS "Autor_Vorname(n)",
    au.nachname AS "Autor_Nachname"
FROM buecher bu
INNER JOIN autoren_zu_buecher ab ON
ab.book_id = bu.book_id
INNER JOIN autoren au ON au.author_id =
ab.author_id
WHERE
    bu.book_id IN (SELECT book_ID FROM
karasek)
AND
    bu.book_id IN (SELECT book_ID FROM
zeit)
ORDER BY bu.titel;
```



	TITEL	Autor_Vorname(n)	Autor_Nachname
1	Berlin Alexanderplatz	Alfred	Döblin
2	Der Steppenwolf	Hermann	Hesse
3	Die Blechtrommel	Günter	Grass
4	Die Verwirrungen des Zöglings Törleß	Robert	Musil

Abbildung 25. Bücher die sowohl von Hellmuth Karasek wie auch von der ZEIT vorgeschlagen wurden.

Weiteres Abfrage-Beispiel: Autoren sortiert nach Anzahl der empfohlenen Bücher

Als weiteres Beispiel soll eine Liste der Autoren erstellt werden, die nach der Anzahl ihrer empfohlenen Bücher sortiert wird.

Skript 22. Abfrage-Skript

```
SELECT
    au.vorname,
    au.nachname,
    COUNT(bu.book_id) Anzahl
FROM buecher bu
INNER JOIN autoren_zu_buecher ab ON ab.book_id = bu.book_id
INNER JOIN autoren au ON au.author_id = ab.author_id
GROUP BY au.author_id, au.vorname, au.nachname
ORDER BY anzahl DESC, au.nachname;
```

Skriptausgabe x Abfrageergebnis x			
SQL 50 Zeilen abgerufen in 0,06 Sekunden			
	VORNAME	NACHNAME	ANZAHL
1	Charles	Dickens	5
2	Virginia	Woolf	5
3	Jane	Austen	4
4	Joseph	Conrad	3
5	Theodor	Fontane	3
6	Edward Morgan	Forster	3
7	Graham	Greene	3
8	Thomas	Hardy	3
9	Franz	Kafka	3
10	D. H.	Lawrence	3
11	Thomas	Mann	3
12	Samuel	Beckett	2
13	Heinrich	Böll	2

Abbildung 26. Autoren, sortiert nach Anzahl ihrer Bücher, die eine Empfehlung erhalten haben. Von Charles Dickens und Virginia Woolf wurden jeweils fünf Bücher empfohlen.

Über das folgende Skript erhält man alle Bücher von Charles Dickens, die empfohlen werden.

Skript 23. Abfrage-Skript

```
SELECT
    au.vorname,
    au.nachname,
    bu.titel
FROM buecher bu
INNER JOIN autoren_zu_buecher ab ON ab.book_id = bu.book_id
INNER JOIN autoren au ON au.author_id = ab.author_id
WHERE nachname = 'Dickens'
ORDER BY titel;
```

Skriptausgabe x Abfrageergebnis x			
SQL Alle Zeilen abgerufen:5 in 0,005 Sekunden			
	VORNAME	NACHNAME	TITEL
1	Charles	Dickens	Bleak House
2	Charles	Dickens	David Copperfield
3	Charles	Dickens	Dombey und Sohn
4	Charles	Dickens	Große Erwartungen
5	Charles	Dickens	Oliver Twist

Abbildung 27. Empfohlene Bücher vom Autor Charles Dickens.