06.06.2002

DBS-Projekt: Ein internetgestütztes Musikarchiv

von Anja Jentzsch und Richard Cyganiak

DBS Projekt-"Ausschreibung"

Ein internetgestütztes Musikarchiv

Die Datenbank speichert Informationen über Musiker, Bands und ihre veröffentlichten CDs (Alben, Singles etc).

Ein veröffentlichter Tonträger (Es wird nicht zwischen CD, MC usw. unterschieden) wird mit Titel, Name des Künstlers bzw. der Band, Trackliste und Erscheinungsdatum gespeichert. Bei den Künstlern müssen wir zwischen Solisten und Bands unterscheiden. Bei Bands wollen wir wissen, wie die Mitglieder heißen, welches Instrument sie spielen, und gegebenenfalls von wann bis wann sie Mitglied waren. Zu jeder Person, ob Bandmitglied oder Solist, wollen wir auch das Geburtsdatum speichern.

Weiterhin soll es die Möglichkeit geben, neben dem eigentlichen Künstler auch weitere Mitwirkende zu speichern, z.B. den Produzent, den Songschreiber oder Gastmusiker.

Weitere Punkte, die wir berücksichtigt wissen wollen:

- Bandmitglieder können Soloalben veröffentlichen.
- manche Produzenten veröffentlichen eigene Alben
- Songs können Covers oder Remixe anderer Songs sein.
- Compilations: Jeder einzelne Song kann von einem anderen Künstler sein. Möglicherweise gibt es einen für die Auswahl verantwortlichen Künstler, z.B. bei DJ-Mix-Alben.

Außerdem wäre es gut, wenn wir zu jedem Song ein Musikgenres speichern könnten. Genres sind Hierarchien. Beispielsweise sind "Alternative", "Gothic", "Glam Rock" und "Punk" Unterkategorien von "Rock".

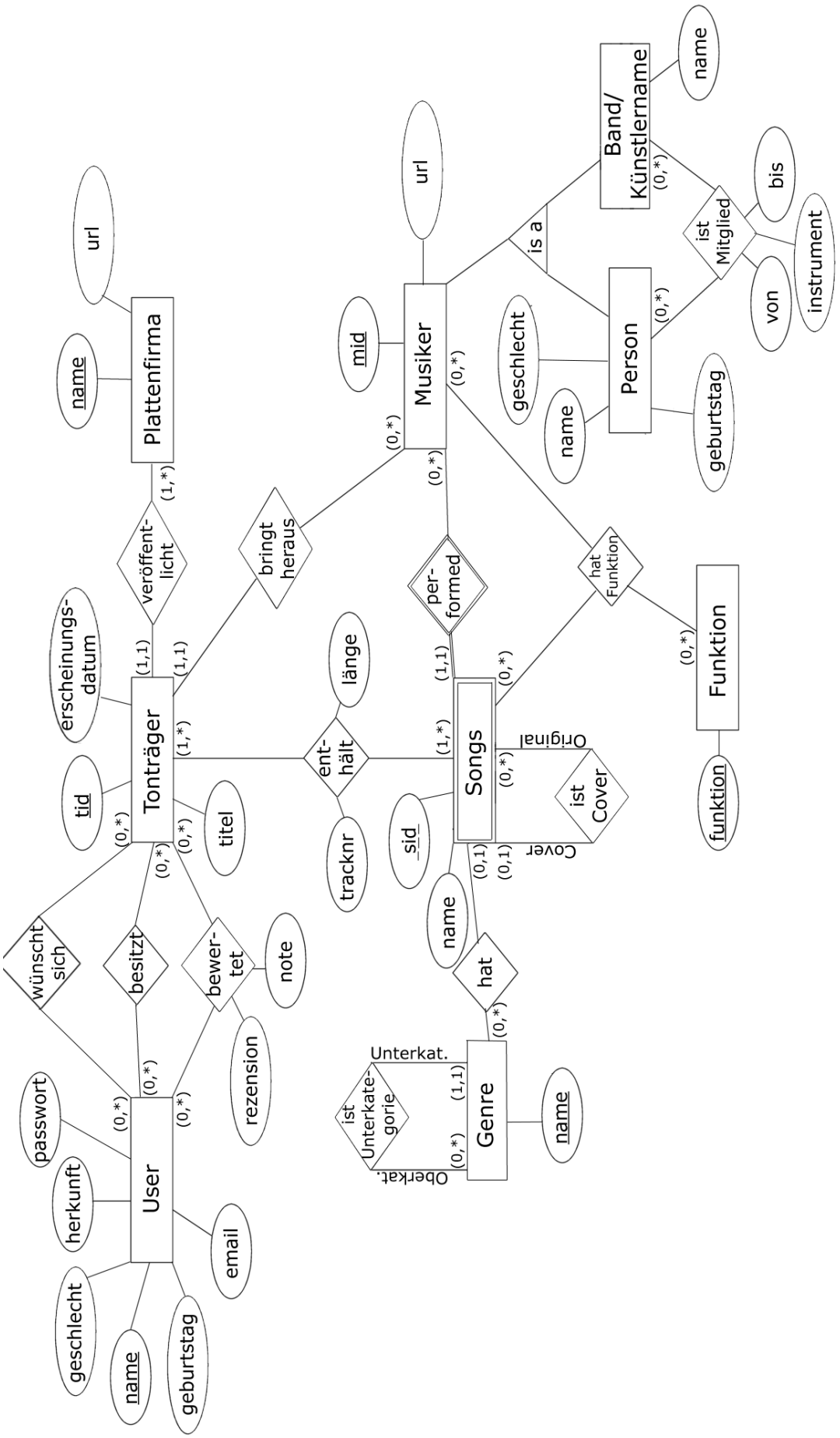
Eventuell könnte man zu jedem Tonträger auch noch die Plattenfirma speichern.

Prinzipiell sollte davon ausgegangen werden, dass alle Informationen optional sind, da sie evtl. zum Zeitpunkt der Eingabe nicht verfügbar oder zu unwichtig sind.

Benutzer der Datenbank können über ein Webinterface alle diese Informationen abfragen. Um aus dem Projekt eine vollwertige Webanwendung zu machen, müsste es den Benutzern gestattet sein, sich einen Account anzulegen. Dabei werden Geschlecht und Alter erfragt. Registrierte Benutzer können folgendes tun:

- selbst Informationen in die Datenbank eingeben
- Tonträger mit Noten bewerten
- Tonträger zu ihrer persönlichen "Sammlung" hinzufügen
- Tonträger zu ihrer "Wishlist" hinzufügen.

ER Modell


Plattenfirma
url
name
veröffent-licht
(1,*)
(1,1)
Tonträger
erscheinungs-datum
tid
titel
(0,*)
wünscht sich
besitzt
bewer-tet
note
rezension
User
passwort
herkunft
geschlecht
name
geburtstag
email
bringt heraus
(0,*)
Musiker
mid
url
per-formed
(1,1)
(1,*)
ent-hält
länge
tracknr
Songs
sid
name
(0,1)
hat
Genre
name
Unterkat.
ist Unterkate-gorie
Oberkat.
Original
ist Cover
Cover
hat Funktion
Funktion
funktion
Band/ Künstlername
name
is a
Person
geschlecht
name
geburtstag
ist Mitglied
bis
von
instrument

Relationales Modell

Vor der Zusammenfassung

Entitäten:
Tontraeger (tid , erscheinungsdatum , titel)
User (name , email , geburtstag , geschlecht , herkunft , passwort)
Plattenfirma (firmen_name , url)
Song (mid , sid , name)
Genre (genre_name)
Musiker (mid , url)
Person (mid , vorname , nachname , geburtstag , geschlecht)
Band_Kuenstlername (mid , name)
Funktion (funktion)

Relationen:
Besitzt (name , tid)
WuenschtSich (name , tid)
Bewertet (name , tid , rezension, note)
Enthaelt (tid , mid , sid , tracknr , laenge)
HatFunktion (mid , funktion , song_mid , sid)
IstMitglied (mid1 , mid2 , instrument , von , bis)
Veroeffentlicht (tid , firmen_name) wird mit Tontraeger zusammengefasst
Hat (mid , sid , genre_name) wird mit Song zusammengefasst
IstUnterkategorie (genre_name , obergenre_name) wird mit Genre zusammengefasst
IstCover (cover_mid , cover_sid , original_mid , original_sid) wird mit Song zusammengefasst
BringtHeraus (tid , mid) wird mit Tontraeger zusammengefasst

Nach der Zusammenfassung

Tontraeger (tid , titel , mid , erscheinungsdatum , plattenfirma)
User (name , email , geburtstag , geschlecht , herkunft , passwort)
Plattenfirma (firmen_name , url)
Song (mid , sid , name , genre_name , original_mid , original_sid)
Genre (genre_name , obergenre_name)
Musiker (mid , url)
Person (mid , vorname , nachname , geburtstag , geschlecht)
Band_Kuenstlername (mid , name)
Funktion (funktion)
Besitzt (name , tid)
WuenschtSich (name , tid)
Bewertet (name , tid , rezension, note)
Enthaelt (tid , tracknr , laenge , mid , sid)
HatFunktion (mid , funktion , song_mid , sid)
IstMitglied (mid1 , mid2 , instrument , von , bis)

Erzeugung der Relationen in Oracle

Das vollständige SQL-File zur Erstellung der Tabellen inkl. Primär- und Fremdschlüssel und Check-Bedingungen:

create.sql

```
CREATE TABLE Users (
        name VARCHAR2(30),
        email VARCHAR2(60) NOT NULL UNIQUE,
        geburtstag DATE,
        geschlecht char NOT NULL,
        herkunft VARCHAR2(50),
        passwort VARCHAR2(20) NOT NULL,
        CONSTRAINT pk_users PRIMARY KEY(name),
        CONSTRAINT sicheres_passwort CHECK (name <> passwort AND LENGTH(passwort)>5),
        CONSTRAINT geschlecht_m_oder_w CHECK (geschlecht in ('M','W'))
);

CREATE TABLE Plattenfirma (
        firmen_name VARCHAR2(40),
        url VARCHAR2(30),
        CONSTRAINT pk_plattenfirma PRIMARY KEY(firmen_name)
);

CREATE TABLE Genre (
        genre_name VARCHAR2(40),
        obergenre_name VARCHAR2(40),
        CONSTRAINT pk_genre PRIMARY KEY(genre_name),
        CONSTRAINT fk_genre_genre FOREIGN KEY(obergenre_name) REFERENCES Genre(genre_name)
                ON DELETE SET NULL
);

CREATE TABLE Musiker (
        mid NUMBER(10),
        url VARCHAR2(40),
        CONSTRAINT pk_musiker PRIMARY KEY(mid)
);

CREATE TABLE Person (
        mid NUMBER(10),
        vorname VARCHAR2(20) NOT NULL,
        nachname VARCHAR2(20) NOT NULL,
        geburtstag DATE,
        geschlecht char NOT NULL,
        CONSTRAINT pk_person PRIMARY KEY(mid),
        CONSTRAINT fk_person_musiker FOREIGN KEY(mid) REFERENCES Musiker(mid)
                ON DELETE CASCADE,
        CONSTRAINT person_geschlecht CHECK (geschlecht in ('M','W'))
);

CREATE TABLE Band_Kuenstlername (
        mid NUMBER(10),
        name VARCHAR2(50) NOT NULL,
        CONSTRAINT pk_band PRIMARY KEY(mid),
        CONSTRAINT fk_band_musiker FOREIGN KEY(mid) REFERENCES Musiker(mid) ON DELETE CASCADE
);

CREATE TABLE Funktion (
        funktion VARCHAR2(30),
        CONSTRAINT pk_funktion PRIMARY KEY(funktion)
);

CREATE TABLE Tontraeger (
        tid NUMBER(8),
        titel VARCHAR2(50) NOT NULL,
        mid NUMBER(10),
        erscheinungsdatum DATE,
        plattenfirma VARCHAR2(40),
        CONSTRAINT pk_tontraeger PRIMARY KEY(tid),
        CONSTRAINT fk_tontraeger_musiker FOREIGN KEY(mid) REFERENCES Musiker(mid)
                ON DELETE CASCADE,
        CONSTRAINT id_groesser_0 CHECK (tid > 0),
        CONSTRAINT fk_tontraeger_firma FOREIGN KEY(plattenfirma)
                REFERENCES Plattenfirma(firmen_name) ON DELETE CASCADE
);
```

```
CREATE TABLE Bewertet (
	name VARCHAR2(30),
	tid NUMBER(8),
	rezension CLOB,
	note NUMBER(2) NOT NULL,
	CONSTRAINT pk_bewertet PRIMARY KEY(name, tid),
	CONSTRAINT fk_bewertet_users FOREIGN KEY(name) REFERENCES Users(name)
		ON DELETE CASCADE,
	CONSTRAINT fk_bewertet_tontraeger FOREIGN KEY(tid) REFERENCES Tontraeger(tid)
		ON DELETE CASCADE,
	CONSTRAINT bewertet_korrekte_note CHECK (note BETWEEN 0 AND 10)
);

CREATE TABLE Besitzt (
	name VARCHAR2(30),
	tid NUMBER(8),
	CONSTRAINT pk_besitzt PRIMARY KEY(name, tid),
	CONSTRAINT fk_besitzt_users FOREIGN KEY(name) REFERENCES Users(name)
		ON DELETE CASCADE,
	CONSTRAINT fk_besitzt_tontraeger FOREIGN KEY(tid) REFERENCES Tontraeger(tid)
		ON DELETE CASCADE
);

CREATE TABLE WuenschtSich (
	name VARCHAR2(30),
	tid NUMBER(8),
	CONSTRAINT pk_wuenschtsich PRIMARY KEY(name, tid),
	CONSTRAINT fk_wuenschtsich_users FOREIGN KEY(name) REFERENCES Users(name)
		ON DELETE CASCADE,
	CONSTRAINT fk_wuenschtsich_tontraeger FOREIGN KEY(tid) REFERENCES Tontraeger(tid)
		ON DELETE CASCADE
);

CREATE TABLE Song (
	mid NUMBER(10) NOT NULL,
	sid NUMBER(10),
	name VARCHAR2(100) NOT NULL,
	genre_name VARCHAR2(40),
	original_mid NUMBER(10),
	original_sid NUMBER(10),
	CONSTRAINT pk_song PRIMARY KEY(sid, mid),
	CONSTRAINT fk_song_musiker FOREIGN KEY(mid) REFERENCES Musiker(mid)
		ON DELETE CASCADE,
	CONSTRAINT fk_song_original FOREIGN KEY (original_sid, original_mid)
		REFERENCES Song(sid, mid) ON DELETE SET NULL,
	CONSTRAINT fk_song_genre FOREIGN KEY(genre_name) REFERENCES Genre(genre_name)
);

CREATE TABLE Enthaelt (
	tid NUMBER(8),
	tracknr NUMBER(3),
	laenge NUMBER(6),
	mid NUMBER(10) NOT NULL,
	sid NUMBER(10) NOT NULL,
	CONSTRAINT pk_enthaelt PRIMARY KEY(tid, tracknr),
	CONSTRAINT fk_enthaelt_tontraeger FOREIGN KEY(tid) REFERENCES Tontraeger(tid)
		ON DELETE CASCADE,
	CONSTRAINT fk_enthaelt_song FOREIGN KEY (sid, mid) REFERENCES Song(sid, mid)
		ON DELETE CASCADE,
	CONSTRAINT tracknr_groesser_0 CHECK (tracknr > 0),
	CONSTRAINT laenge_groesser_0 CHECK (laenge > 0)
);

CREATE TABLE HatFunktion (
	mid NUMBER(10),
	funktion VARCHAR2(30),
	song_mid NUMBER(10),
	sid NUMBER(10),
	CONSTRAINT pk_hatfunktion PRIMARY KEY(funktion, mid, song_mid, sid),
	CONSTRAINT fk_hatfunktion_funktion FOREIGN KEY(funktion)
		REFERENCES Funktion(funktion) ON DELETE CASCADE,
	CONSTRAINT fk_hatfunktion_musiker FOREIGN KEY(mid) REFERENCES Musiker(mid)
		ON DELETE CASCADE,
	CONSTRAINT fk_hatfunktion_song FOREIGN KEY (sid, song_mid) REFERENCES Song(sid, mid)
		ON DELETE CASCADE
);

CREATE TABLE IstMitglied (
	mid1 NUMBER(10),
```

```
        mid2 NUMBER(10),
        instrument VARCHAR2(15),
        von DATE,
        bis DATE,
        CONSTRAINT pk_istmitglied PRIMARY KEY(mid1, mid2),
        CONSTRAINT fk_istmitglied_person FOREIGN KEY(mid1) REFERENCES Person(mid)
                ON DELETE CASCADE,
        CONSTRAINT fk_istmitglied_band FOREIGN KEY(mid2) REFERENCES Band_Kuenstlername(mid)
                ON DELETE CASCADE,
        CONSTRAINT istmitglied_korrektes_datum CHECK (TO_CHAR(bis) > TO_CHAR(von))
);
```

Es folgt die Übersicht der erstellten Tabellen. Der Output von SQLPlus wurde gegebenenfalls so verändert, dass übermäßig breite Spalten auf ein druckfreundlicheres Format gebracht wurden.

describe.lst

```
SQL> SELECT * FROM TAB;

TNAME                          TABTYPE  CLUSTERID
------------------------------ ------- ----------
BAND_KUENSTLERNAME             TABLE
BESITZT                        TABLE
BEWERTET                       TABLE
ENTHAELT                       TABLE
FUNKTION                       TABLE
GENRE                          TABLE
HATFUNKTION                    TABLE
ISTMITGLIED                    TABLE
MUSIKER                        TABLE
PERSON                         TABLE
PLATTENFIRMA                   TABLE
SONG                           TABLE
TONTRAEGER                     TABLE
TONTRAEGER_UEBERSICHT          VIEW
USERS                          TABLE
USER_UEBERSICHT                VIEW
WUENSCHTSICH                   TABLE

17 rows selected.

SQL> DESCRIBE Band_Kuenstlername
 Name                                                  Null?    Type
 ----------------------------------------------------- -------- ----------------------------
 MID                                                   NOT NULL NUMBER(10)
 NAME                                                  NOT NULL VARCHAR2(50)

SQL> DESCRIBE Besitzt
 Name                                                  Null?    Type
 ----------------------------------------------------- -------- ----------------------------
 NAME                                                  NOT NULL VARCHAR2(30)
 TID                                                   NOT NULL NUMBER(8)

SQL> DESCRIBE Bewertet
 Name                                                  Null?    Type
 ----------------------------------------------------- -------- ----------------------------
 NAME                                                  NOT NULL VARCHAR2(30)
 TID                                                   NOT NULL NUMBER(8)
 REZENSION                                                      CLOB
 NOTE                                                  NOT NULL NUMBER(2)

SQL> DESCRIBE Enthaelt
 Name                                                  Null?    Type
 ----------------------------------------------------- -------- ----------------------------
 TID                                                   NOT NULL NUMBER(8)
 TRACKNR                                               NOT NULL NUMBER(3)
 LAENGE                                                         NUMBER(6)
 MID                                                   NOT NULL NUMBER(10)
 SID                                                   NOT NULL NUMBER(10)

SQL> DESCRIBE Funktion
 Name                                                  Null?    Type
 ----------------------------------------------------- -------- ----------------------------
 FUNKTION                                              NOT NULL VARCHAR2(30)

SQL> DESCRIBE Genre
 Name                                                  Null?    Type
 ----------------------------------------------------- -------- ----------------------------
```

```
 GENRE_NAME                                        NOT NULL VARCHAR2(40)
 OBERGENRE_NAME                                             VARCHAR2(40)

SQL> DESCRIBE HatFunktion
 Name                                              Null?    Type
 ------------------------------------------------- -------- ----------------------------
 MID                                               NOT NULL NUMBER(10)
 FUNKTION                                          NOT NULL VARCHAR2(30)
 SONG_MID                                          NOT NULL NUMBER(10)
 SID                                               NOT NULL NUMBER(10)

SQL> DESCRIBE IstMitglied
 Name                                              Null?    Type
 ------------------------------------------------- -------- ----------------------------
 MID1                                              NOT NULL NUMBER(10)
 MID2                                              NOT NULL NUMBER(10)
 INSTRUMENT                                                 VARCHAR2(15)
 VON                                                        DATE
 BIS                                                        DATE

SQL> DESCRIBE Musiker
 Name                                              Null?    Type
 ------------------------------------------------- -------- ----------------------------
 MID                                               NOT NULL NUMBER(10)
 URL                                                        VARCHAR2(40)

SQL> DESCRIBE Person
 Name                                              Null?    Type
 ------------------------------------------------- -------- ----------------------------
 MID                                               NOT NULL NUMBER(10)
 VORNAME                                           NOT NULL VARCHAR2(20)
 NACHNAME                                          NOT NULL VARCHAR2(20)
 GEBURTSTAG                                                 DATE
 GESCHLECHT                                        NOT NULL CHAR(1)

SQL> DESCRIBE Plattenfirma
 Name                                              Null?    Type
 ------------------------------------------------- -------- ----------------------------
 FIRMEN_NAME                                       NOT NULL VARCHAR2(40)
 URL                                                        VARCHAR2(30)

SQL> DESCRIBE Song
 Name                                              Null?    Type
 ------------------------------------------------- -------- ----------------------------
 MID                                               NOT NULL NUMBER(10)
 SID                                               NOT NULL NUMBER(10)
 NAME                                              NOT NULL VARCHAR2(100)
 GENRE_NAME                                                 VARCHAR2(40)
 ORIGINAL_MID                                               NUMBER(10)
 ORIGINAL_SID                                               NUMBER(10)

SQL> DESCRIBE Tontraeger
 Name                                              Null?    Type
 ------------------------------------------------- -------- ----------------------------
 TID                                               NOT NULL NUMBER(8)
 TITEL                                             NOT NULL VARCHAR2(50)
 MID                                                        NUMBER(10)
 ERSCHEINUNGSDATUM                                          DATE
 PLATTENFIRMA                                               VARCHAR2(40)

SQL> DESCRIBE Users
 Name                                              Null?    Type
 ------------------------------------------------- -------- ----------------------------
 NAME                                              NOT NULL VARCHAR2(30)
 EMAIL                                             NOT NULL VARCHAR2(60)
 GEBURTSTAG                                                 DATE
 GESCHLECHT                                        NOT NULL CHAR(1)
 HERKUNFT                                                   VARCHAR2(50)
 PASSWORT                                          NOT NULL VARCHAR2(20)

SQL> DESCRIBE WuenschtSich
 Name                                              Null?    Type
 ------------------------------------------------- -------- ----------------------------
 NAME                                              NOT NULL VARCHAR2(30)
 TID                                               NOT NULL NUMBER(8)
```

Referentielle Integrität und Check-Bedingungen

Das Projekt enthält 15 Relationen, zwischen denen 21 referentielle Integritäts-Bedingungen bestehen. Wir haben die folgenden Check-Bedingungen festgelegt:

```
(a) Tontraeger:      CHECK (tid > 0)
(b) Enthaelt:        CHECK (tracknr > 0)
(c) Enthaelt:        CHECK (laenge > 0)
(d) Bewertet:        CHECK (note BETWEEN 0 AND 10)
(e) Users, Person:   CHECK (geschlecht in ('M','W'))
(f) Users:           CHECK (name <> passwort AND LENGTH(passwort)>5)
(g) IstMitglied:     CHECK (TO_CHAR(bis) > TO_CHAR(von))
```

(a) bis (e) sind Attribut-basierte Bedingungen. (d) testet, ob die Bewertungsnote im gültigen Bereich von 0 bis 10 liegt. (e) beschränkt die erlaubten Werte für das Geschlecht einer Person auf 'M' und 'W'. (f) und (g) sind Tupel-basierte Bedingungen. (f) stellt sicher, dass eine Person nicht ihren eigenen Usernamen als Passwort verwendet (eine "Minimalanforderung" an ein sicheres Passwort). Die letzte Bedingung (g) vergleicht Anfangs- und Enddatum in einer temporalen Beziehung, damit Mitglieder einer Band diese nicht verlassen, bevor sie hineingekommen sind.

Datenmodifikations-Kommandos

Das folgende SQL-File illustriert die Wirkung der verschiedenen Schlüssel und Checks.

constraints.sql

```
-- Diese Datensätze werden problemlos eingefügt
INSERT INTO Users VALUES ('user1', 'email1@host.de', to_date('1975-05-01', 'yyyy-mm-dd'),
       'W', NULL, 'passwd');
INSERT INTO Users VALUES ('user2', 'email2@host.de', to_date('1975-05-01', 'yyyy-mm-dd'),
       'W', NULL, 'passwd');
INSERT INTO Musiker (mid) VALUES (1);
INSERT INTO Tontraeger (tid, titel, mid) VALUES (1, 'Kuschelrock #152', 1);
INSERT INTO Bewertet (name, tid, note) VALUES ('user1', 1, 0);

-- 1. INSERT-Kommando erzeugt Schlüsselverletzung (doppelter Username)
INSERT INTO Users VALUES ('user1', 'email3@host.de', to_date('1975-05-01', 'yyyy-mm-dd'),
       'W', NULL, 'passwd');

-- 2. UPDATE-Kommando erzeugt Schlüsselverletzung (doppelte email-Adresse)
UPDATE Users SET email='email1@host.de' WHERE name='user2' and geschlecht='M';

-- 3. INSERT-Kommando erzeugt Verletzung der referentiellen Integrität (User existiert nicht)
INSERT INTO Bewertet (name, tid, note) VALUES ('user99', 1, 0);

-- 4. UPDATE-Kommando verletzt referentielle Integrität, da ein anderer Datensatz 'user1'
--    als Fremdschlüssel hat
UPDATE Users SET name='user99' WHERE name='user1';

-- 5. DELETE-Kommando, das eine Verletzung der referentiellen Integrität erzeugt
--    Gibt es bei uns nicht, da DELETE-Kommandos mit ON DELETE CASCADE bzw.
--    ON DELETE SET NULL an die referenzierenden Datensätze weitergegeben werden

-- 6. INSERT-Kommando erzeugt CHECK-Bedingungs-Verletzung (Passwort = Username)
INSERT INTO Users VALUES ('user5', 'email5@host.de', to_date('1980-08-10', 'yyyy-mm-dd'),
       'W', NULL, 'user5');

-- 7. UPDATE-Kommando erzeugt CHECK-Bedingungs-Verletzung (Geschlecht muss 'W' oder 'M' sein)
UPDATE Users SET geschlecht='Z' WHERE name='user1';
```

constraints.lst

```
SQL> @constraints

1 row created.


1 row created.


1 row created.


1 row created.


1 row created.

INSERT INTO Users VALUES ('user1', 'email3@host.de', to_date('1975-05-01', 'yyyy-mm-dd'),
'W', NULL, 'passwd')
*
ERROR at line 1:
ORA-00001: unique constraint (TZ21.SYS_C0026384) violated



0 rows updated.

INSERT INTO Bewertet (name, tid, note) VALUES ('user99', 1, 0)
*
ERROR at line 1:
ORA-02291: integrity constraint (TZ21.SYS_C0026411) violated - parent key not found


UPDATE Users SET name='user99' WHERE name='user1'
*
ERROR at line 1:
ORA-02292: integrity constraint (TZ21.SYS_C0026411) violated - child record found


INSERT INTO Users VALUES ('user5', 'email5@host.de', to_date('1980-08-10', 'yyyy-mm-dd'),
'W', NULL, 'user5')
*
ERROR at line 1:
ORA-02290: check constraint (TZ21.SICHERES_PASSWORT) violated


UPDATE Users SET geschlecht='Z' WHERE name='user1'
*
ERROR at line 1:
ORA-02290: check constraint (TZ21.GESCHLECHT_M_ODER_W) violated
```

Datenbasen

Die "kleine" Datenbank

Unsere "kleine" Datenbank enthält in einigen Relationen deutlich mehr als 5-10 Datensätze. Wenn man z.B. die Tontraeger-Relation mit realistischen 5 Datensätzen füllen will, ergibt das automatisch ca. die zehnfache Anzahl an Datensätzen für die Song- und Enthaelt-Relation. An einigen Stellen wurde das Listing daher gekürzt.

Wir verwenden nicht den Bulk Loader, sondern fügen die Daten per SQL-File ein.

data1.sql

```
INSERT INTO Genre VALUES ('Modern Rock', NULL);
INSERT INTO Genre VALUES ('Rock', NULL);
INSERT INTO Genre VALUES ('Electronica', NULL);
INSERT INTO Genre VALUES ('Alternative Rock', 'Modern Rock');
INSERT INTO Genre VALUES ('Brit Pop', 'Alternative Rock');
INSERT INTO Genre VALUES ('New Wave', 'Modern Rock');
INSERT INTO Genre VALUES ('Classic Rock', 'Rock');
INSERT INTO Genre VALUES ('Psychedelic Rock', 'Classic Rock');
INSERT INTO Genre VALUES ('Glam Rock', 'Classic Rock');
INSERT INTO Genre VALUES ('Ambient', 'Electronica');
INSERT INTO Genre VALUES ('Downbeat', 'Electronica');
INSERT INTO Genre VALUES ('Trip Hop', 'Downbeat');
INSERT INTO Genre VALUES ('House', 'Electronica');
INSERT INTO Genre VALUES ('Organic House', 'House');
INSERT INTO Genre VALUES ('Techno', 'Electronica');
INSERT INTO Genre VALUES ('Trance', 'Techno');

INSERT INTO Musiker VALUES (1, 'www.massiveattack.co.uk');
INSERT INTO Band_Kuenstlername VALUES (1, 'Massive Attack');
INSERT INTO Musiker VALUES (2, NULL);
INSERT INTO Band_Kuenstlername VALUES (2, 'The Doors');
INSERT INTO Musiker VALUES (3, 'www.chemicalbrothers.co.uk');
INSERT INTO Band_Kuenstlername VALUES (3, 'The Chemical Brothers');
INSERT INTO Musiker VALUES (4, NULL);
INSERT INTO Band_Kuenstlername VALUES (4, 'Iggy Pop');
INSERT INTO Musiker VALUES (5, NULL);
INSERT INTO Person VALUES (5, 'Brian', 'Eno', NULL, 'M');
INSERT INTO Musiker VALUES (6, NULL);
INSERT INTO Band_Kuenstlername VALUES (6, 'Primal Scream');
INSERT INTO Musiker VALUES (7, NULL);
INSERT INTO Band_Kuenstlername VALUES (7, 'Sleeper');
INSERT INTO Musiker VALUES (8, NULL);
INSERT INTO Band_Kuenstlername VALUES (8, 'New Order');
INSERT INTO Musiker VALUES (9, NULL);
INSERT INTO Band_Kuenstlername VALUES (9, 'Blur');
INSERT INTO Musiker VALUES (10, NULL);
INSERT INTO Person VALUES (10, 'Lou', 'Reed', NULL, 'M');
INSERT INTO Musiker VALUES (11, NULL);
INSERT INTO Band_Kuenstlername VALUES (11, 'Pulp');
INSERT INTO Musiker VALUES (12, NULL);
INSERT INTO Band_Kuenstlername VALUES (12, 'Bedrock');
INSERT INTO Musiker VALUES (13, NULL);
INSERT INTO Band_Kuenstlername VALUES (13, 'KYO');
INSERT INTO Musiker VALUES (14, NULL);
INSERT INTO Band_Kuenstlername VALUES (14, 'Elastica');
INSERT INTO Musiker VALUES (15, NULL);
INSERT INTO Band_Kuenstlername VALUES (15, 'Leftfield');
INSERT INTO Musiker VALUES (16, NULL);
INSERT INTO Band_Kuenstlername VALUES (16, 'Underworld');
INSERT INTO Musiker VALUES (17, NULL);
INSERT INTO Person VALUES (17, 'Damon', 'Albarn', NULL, 'M');

INSERT INTO Musiker VALUES (18, NULL);
INSERT INTO Band_Kuenstlername VALUES (18, 'Soul II Soul');
INSERT INTO Musiker VALUES (19, NULL);
INSERT INTO Person VALUES (19, 'Nellee', 'Hooper', NULL, 'M');
INSERT INTO Musiker VALUES (20, NULL);
INSERT INTO Person VALUES (20, 'Pete', 'Heller', NULL, 'M');

INSERT INTO Musiker VALUES (21, NULL);
INSERT INTO Person VALUES (21, 'Beth', 'Orton', NULL, 'W');
INSERT INTO Musiker VALUES (22, NULL);
```

```
INSERT INTO Person VALUES (22, 'Richard', 'Ashcroft', NULL, 'M');

INSERT INTO Musiker VALUES (23, NULL);
INSERT INTO Person VALUES (23, 'Jim', 'Morrison', NULL, 'M');
INSERT INTO Musiker VALUES (24, NULL);
INSERT INTO Person VALUES (24, 'John', 'Densmore', NULL, 'M');
INSERT INTO Musiker VALUES (25, NULL);
INSERT INTO Person VALUES (25, 'Robby', 'Krieger', NULL, 'M');
INSERT INTO Musiker VALUES (26, NULL);
INSERT INTO Person VALUES (26, 'Ray', 'Manzarek', NULL, 'M');
INSERT INTO Musiker VALUES (29, NULL);
INSERT INTO Person VALUES (29, 'Paul', 'Rothchild', NULL, 'M');

INSERT INTO Musiker VALUES (27, NULL);
INSERT INTO Person VALUES (27, 'Ed', 'Simons', NULL, 'M');
INSERT INTO Musiker VALUES (28, NULL);
INSERT INTO Person VALUES (28, 'Tom', 'Rowlands', NULL, 'M');

INSERT INTO Musiker VALUES (30, NULL);
INSERT INTO Person VALUES (30, 'Grant', 'Marshall', NULL, 'M');
INSERT INTO Musiker VALUES (31, NULL);
INSERT INTO Person VALUES (31, 'Robert', 'del Naja', NULL, 'M');
INSERT INTO Musiker VALUES (32, NULL);
INSERT INTO Person VALUES (32, 'Andrew', 'Vowles', NULL, 'M');
INSERT INTO Musiker VALUES (33, NULL);
INSERT INTO Band_Kuenstlername VALUES (33, 'Daddy Gee');
INSERT INTO Musiker VALUES (34, NULL);
INSERT INTO Band_Kuenstlername VALUES (34, '3D');
INSERT INTO Musiker VALUES (35, NULL);
INSERT INTO Band_Kuenstlername VALUES (35, 'Mushroom');
INSERT INTO Musiker VALUES (36, NULL);
INSERT INTO Person VALUES (36, 'Tracey', 'Thorn', NULL, 'W');
INSERT INTO Musiker VALUES (37, NULL);
INSERT INTO Band_Kuenstlername VALUES (37, 'Everything But The Girl');
INSERT INTO Musiker VALUES (38, NULL);
INSERT INTO Band_Kuenstlername VALUES (38, 'Tricky');
INSERT INTO Musiker VALUES (39, NULL);
INSERT INTO Person VALUES (39, 'Horace', 'Andy', NULL, 'M');
INSERT INTO Musiker VALUES (40, NULL);
INSERT INTO Band_Kuenstlername VALUES (40, 'Nicolette');

INSERT INTO IstMitglied VALUES (23, 2, 'voc', NULL, NULL);
INSERT INTO IstMitglied VALUES (24, 2, 'd', NULL, NULL);
INSERT INTO IstMitglied VALUES (25, 2, 'git', NULL, NULL);
INSERT INTO IstMitglied VALUES (26, 2, 'keyb', NULL, NULL);
INSERT INTO IstMitglied VALUES (19, 18, NULL, NULL, NULL);
INSERT INTO IstMitglied VALUES (27, 3, NULL, NULL, NULL);
INSERT INTO IstMitglied VALUES (28, 3, NULL, NULL, NULL);
INSERT INTO IstMitglied VALUES (30, 1, NULL, NULL, NULL);
INSERT INTO IstMitglied VALUES (31, 1, NULL, NULL, NULL);
INSERT INTO IstMitglied VALUES (32, 1, NULL, NULL, NULL);
INSERT INTO IstMitglied VALUES (30, 33, NULL, NULL, NULL);
INSERT INTO IstMitglied VALUES (31, 34, NULL, NULL, NULL);
INSERT INTO IstMitglied VALUES (32, 35, NULL, NULL, NULL);
INSERT INTO IstMitglied VALUES (36, 37, 'voc', NULL, NULL);

INSERT INTO Funktion VALUES ('Produzent');
INSERT INTO Funktion VALUES ('Remix');
INSERT INTO Funktion VALUES ('Featuring');
INSERT INTO Funktion VALUES ('Vocals');

INSERT INTO Plattenfirma VALUES ('Circa', NULL);
INSERT INTO Plattenfirma VALUES ('Elektra/Asylum', 'www.electra.com');
INSERT INTO Plattenfirma VALUES ('EMI', 'www.emigroup.com');
INSERT INTO Plattenfirma VALUES ('Virgin', 'www.virginrecords.co.uk');

INSERT INTO Tontraeger VALUES (1, 'Protection', 1, to_date('1995-01-19', 'yyyy-mm-dd'),
'Circa');
INSERT INTO Tontraeger VALUES (2, 'The Doors', 2, to_date('1967-01-22', 'yyyy-mm-dd'),
'Elektra/Asylum');
INSERT INTO Tontraeger VALUES (3, 'Trainspotting', NULL, to_date('1996-07-25', 'yyyy-mm-dd'),
'EMI');
INSERT INTO Tontraeger VALUES (4, 'Come With Us', 3, to_date('2002-01-28', 'yyyy-mm-dd'),
'Virgin');
INSERT INTO Tontraeger VALUES (5, 'Star Guitar', 3, to_date('2002-01-14', 'yyyy-mm-dd'),
'Virgin');

INSERT INTO Song VALUES (2, 1, 'Break On Through (To The Other Side)', 'Psychedelic Rock',
NULL, NULL);
INSERT INTO Song VALUES (2, 2, 'Soul Kitchen', 'Psychedelic Rock', NULL, NULL);
```

```
INSERT INTO Song VALUES (2, 3, 'The Crystal Ship', 'Psychedelic Rock', NULL, NULL);
INSERT INTO Song VALUES (2, 4, 'Twentieth Century Fox', 'Psychedelic Rock', NULL, NULL);
INSERT INTO Song VALUES (2, 5, 'Alabama Song (Whisky Bar)', 'Psychedelic Rock', NULL, NULL);
INSERT INTO Song VALUES (2, 6, 'Light My Fire', 'Psychedelic Rock', NULL, NULL);
INSERT INTO Song VALUES (2, 7, 'Back Door Man', 'Psychedelic Rock', NULL, NULL);
INSERT INTO Song VALUES (2, 8, 'I Looked At You', 'Psychedelic Rock', NULL, NULL);
INSERT INTO Song VALUES (2, 9, 'End Of The Night', 'Psychedelic Rock', NULL, NULL);
INSERT INTO Song VALUES (2, 10, 'Take It As It Comes', 'Psychedelic Rock', NULL, NULL);
INSERT INTO Song VALUES (2, 11, 'The End', 'Psychedelic Rock', NULL, NULL);

INSERT INTO Song VALUES (1, 1, 'Protection', 'Trip Hop', NULL, NULL);
INSERT INTO Song VALUES (1, 2, 'Karmacoma', 'Trip Hop', NULL, NULL);
INSERT INTO Song VALUES (1, 3, 'Three', 'Trip Hop', NULL, NULL);
INSERT INTO Song VALUES (1, 4, 'Weather Storm', 'Trip Hop', NULL, NULL);
INSERT INTO Song VALUES (1, 5, 'Spying Glass', 'Trip Hop', NULL, NULL);
INSERT INTO Song VALUES (1, 6, 'Better Things', 'Trip Hop', NULL, NULL);
INSERT INTO Song VALUES (1, 7, 'Euro Child', 'Trip Hop', NULL, NULL);
INSERT INTO Song VALUES (1, 8, 'Sly', 'Trip Hop', NULL, NULL);
INSERT INTO Song VALUES (1, 9, 'Heat Miser', 'Trip Hop', NULL, NULL);
INSERT INTO Song VALUES (1, 10, 'Light My Fire', 'Trip Hop', 2, 6);

INSERT INTO Song VALUES (4, 1, 'Lust For Life', 'Rock', NULL, NULL);
INSERT INTO Song VALUES (5, 1, 'Deep Blue Day', 'Ambient', NULL, NULL);
INSERT INTO Song VALUES (6, 1, 'Trainspotting', 'Alternative Rock', NULL, NULL);
INSERT INTO Song VALUES (7, 1, 'Atomic', 'Brit Pop', NULL, NULL);
INSERT INTO Song VALUES (8, 1, 'Temptation', 'New Wave', NULL, NULL);
INSERT INTO Song VALUES (4, 2, 'Nightclubbing', 'Rock', NULL, NULL);
INSERT INTO Song VALUES (9, 1, 'Sing', 'Brit Pop', NULL, NULL);
INSERT INTO Song VALUES (10, 1, 'Perfect Day', 'Glam Rock', NULL, NULL);
INSERT INTO Song VALUES (11, 1, 'Mile End', 'Brit Pop', NULL, NULL);
INSERT INTO Song VALUES (12, 1, 'For What You Dream Of', 'Trance', NULL, NULL);
INSERT INTO Song VALUES (14, 1, '2:1', 'Brit Pop', NULL, NULL);
INSERT INTO Song VALUES (15, 1, 'A Final Hit', 'Organic House', NULL, NULL);
INSERT INTO Song VALUES (16, 1, 'Born Slippy', 'House', NULL, NULL);
INSERT INTO Song VALUES (17, 1, 'Closet Romantic', 'Alternative Rock', NULL, NULL);

INSERT INTO Song VALUES (3, 1, 'Come With Us', 'Electronica', NULL, NULL);
INSERT INTO Song VALUES (3, 2, 'It Began In Africa', 'Electronica', NULL, NULL);
INSERT INTO Song VALUES (3, 3, 'Galaxy Bounce', 'Electronica', NULL, NULL);
INSERT INTO Song VALUES (3, 4, 'Star Guitar', 'Electronica', NULL, NULL);
INSERT INTO Song VALUES (3, 5, 'Hoops', 'Electronica', NULL, NULL);
INSERT INTO Song VALUES (3, 6, 'My Elastic Eye', 'Electronica', NULL, NULL);
INSERT INTO Song VALUES (3, 7, 'The State We''re In', 'Electronica', NULL, NULL);
INSERT INTO Song VALUES (3, 8, 'Denmark', 'Electronica', NULL, NULL);
INSERT INTO Song VALUES (3, 9, 'Pioneer Skies', 'Electronica', NULL, NULL);
INSERT INTO Song VALUES (3, 10, 'The Test', 'Electronica', NULL, NULL);

INSERT INTO Song VALUES (3, 11, 'Star Guitar (Edit)', 'Electronica', NULL, NULL);
INSERT INTO Song VALUES (3, 12, 'Base 6', 'Electronica', NULL, NULL);
INSERT INTO Song VALUES (3, 13, 'Star Guitar (Pete Heller''s Expanded Mix)', 'Electronica',
NULL, NULL);

INSERT INTO HatFunktion VALUES (19, 'Produzent', 1, 1);
INSERT INTO HatFunktion VALUES (19, 'Produzent', 1, 2);
INSERT INTO HatFunktion VALUES (19, 'Produzent', 1, 3);
INSERT INTO HatFunktion VALUES (19, 'Produzent', 1, 4);
INSERT INTO HatFunktion VALUES (19, 'Produzent', 1, 5);
INSERT INTO HatFunktion VALUES (19, 'Produzent', 1, 6);
INSERT INTO HatFunktion VALUES (19, 'Produzent', 1, 7);
INSERT INTO HatFunktion VALUES (19, 'Produzent', 1, 8);
INSERT INTO HatFunktion VALUES (19, 'Produzent', 1, 9);
INSERT INTO HatFunktion VALUES (19, 'Produzent', 1, 10);

INSERT INTO HatFunktion VALUES (29, 'Produzent', 2, 1);
INSERT INTO HatFunktion VALUES (29, 'Produzent', 2, 2);
INSERT INTO HatFunktion VALUES (29, 'Produzent', 2, 3);
INSERT INTO HatFunktion VALUES (29, 'Produzent', 2, 4);
INSERT INTO HatFunktion VALUES (29, 'Produzent', 2, 5);
INSERT INTO HatFunktion VALUES (29, 'Produzent', 2, 6);
INSERT INTO HatFunktion VALUES (29, 'Produzent', 2, 7);
INSERT INTO HatFunktion VALUES (29, 'Produzent', 2, 8);
INSERT INTO HatFunktion VALUES (29, 'Produzent', 2, 9);
INSERT INTO HatFunktion VALUES (29, 'Produzent', 2, 10);
INSERT INTO HatFunktion VALUES (29, 'Produzent', 2, 11);

INSERT INTO HatFunktion VALUES (13, 'Featuring', 12, 1);
INSERT INTO HatFunktion VALUES (20, 'Remix', 3, 13);
INSERT INTO HatFunktion VALUES (21, 'Vocals', 3, 7);
INSERT INTO HatFunktion VALUES (22, 'Vocals', 3, 10);
```

```
INSERT INTO HatFunktion VALUES (36, 'Vocals', 1, 1);
INSERT INTO HatFunktion VALUES (38, 'Vocals', 1, 2);
INSERT INTO HatFunktion VALUES (40, 'Vocals', 1, 3);
INSERT INTO HatFunktion VALUES (39, 'Vocals', 1, 8);
INSERT INTO HatFunktion VALUES (40, 'Vocals', 1, 10);

INSERT INTO Enthaelt VALUES (1, 1, 322, 1, 1);
INSERT INTO Enthaelt VALUES (1, 2, 316, 1, 2);
INSERT INTO Enthaelt VALUES (1, 3, 228, 1, 3);
INSERT INTO Enthaelt VALUES (1, 4, 299, 1, 4);
INSERT INTO Enthaelt VALUES (1, 5, 321, 1, 5);
INSERT INTO Enthaelt VALUES (1, 6, 234, 1, 6);
INSERT INTO Enthaelt VALUES (1, 7, 311, 1, 7);
INSERT INTO Enthaelt VALUES (1, 8, 325, 1, 8);
INSERT INTO Enthaelt VALUES (1, 9, 219, 1, 9);
INSERT INTO Enthaelt VALUES (1, 10, 195, 1, 10);

INSERT INTO Enthaelt VALUES (2, 1, 148, 2, 1);
INSERT INTO Enthaelt VALUES (2, 2, 213, 2, 2);
INSERT INTO Enthaelt VALUES (2, 3, 154, 2, 3);
INSERT INTO Enthaelt VALUES (2, 4, 153, 2, 4);
INSERT INTO Enthaelt VALUES (2, 5, 199, 2, 5);
INSERT INTO Enthaelt VALUES (2, 6, 427, 2, 6);
INSERT INTO Enthaelt VALUES (2, 7, 213, 2, 7);
INSERT INTO Enthaelt VALUES (2, 8, 141, 2, 8);
INSERT INTO Enthaelt VALUES (2, 9, 172, 2, 9);
INSERT INTO Enthaelt VALUES (2, 10, 136, 2, 10);
INSERT INTO Enthaelt VALUES (2, 11, 699, 2, 11);

INSERT INTO Enthaelt VALUES (3, 1, 311, 4, 1);
INSERT INTO Enthaelt VALUES (3, 2, 236, 5, 1);
INSERT INTO Enthaelt VALUES (3, 3, 633, 6, 1);
INSERT INTO Enthaelt VALUES (3, 4, 308, 7, 1);
INSERT INTO Enthaelt VALUES (3, 5, 419, 8, 1);
INSERT INTO Enthaelt VALUES (3, 6, 252, 4, 2);
INSERT INTO Enthaelt VALUES (3, 7, 360, 9, 1);
INSERT INTO Enthaelt VALUES (3, 8, 223, 10, 1);
INSERT INTO Enthaelt VALUES (3, 9, 270, 11, 1);
INSERT INTO Enthaelt VALUES (3, 10, 384, 12, 1);
INSERT INTO Enthaelt VALUES (3, 11, 152, 14, 1);
INSERT INTO Enthaelt VALUES (3, 12, 195, 15, 1);
INSERT INTO Enthaelt VALUES (3, 13, 583, 16, 1);
INSERT INTO Enthaelt VALUES (3, 14, 188, 17, 1);

INSERT INTO Enthaelt VALUES (4, 1, 276, 3, 1);
INSERT INTO Enthaelt VALUES (4, 2, 315, 3, 2);
INSERT INTO Enthaelt VALUES (4, 3, 207, 3, 3);
INSERT INTO Enthaelt VALUES (4, 4, 386, 3, 4);
INSERT INTO Enthaelt VALUES (4, 5, 390, 3, 5);
INSERT INTO Enthaelt VALUES (4, 6, 221, 3, 6);
INSERT INTO Enthaelt VALUES (4, 7, 385, 3, 7);
INSERT INTO Enthaelt VALUES (4, 8, 306, 3, 8);
INSERT INTO Enthaelt VALUES (4, 9, 243, 3, 9);
INSERT INTO Enthaelt VALUES (4, 10, 465, 3, 10);

INSERT INTO Enthaelt VALUES (5, 1, 240, 3, 11);
INSERT INTO Enthaelt VALUES (5, 2, 394, 3, 12);
INSERT INTO Enthaelt VALUES (5, 3, 508, 3, 13);

INSERT INTO Users VALUES ('paul', 'paul@gmx.de', to_date('1974-07-15', 'yyyy-mm-dd'), 'M',
'München', '123abcdefg');
INSERT INTO Users VALUES ('fritz', 'fritz@web.de', to_date('1982-01-22', 'yyyy-mm-dd'), 'M',
'Berlin', '123abcdefg');
INSERT INTO Users VALUES ('sandra', 'sandra@freenet.de', to_date('1977-11-01', 'yyyy-mm-dd'),
'W', 'Frankfurt', '123abcdefg');

INSERT INTO Bewertet VALUES ('paul', 1, NULL, 8);
INSERT INTO Bewertet VALUES ('paul', 2, NULL, 9);
INSERT INTO Bewertet VALUES ('paul', 3, NULL, 6);
INSERT INTO Bewertet VALUES ('paul', 4, NULL, 10);
INSERT INTO Bewertet VALUES ('paul', 5, NULL, 5);
INSERT INTO Bewertet VALUES ('fritz', 2, NULL, 8);
INSERT INTO Bewertet VALUES ('fritz', 3, NULL, 10);
INSERT INTO Bewertet VALUES ('fritz', 4, NULL, 4);
INSERT INTO Bewertet VALUES ('sandra', 1, NULL, 10);
INSERT INTO Bewertet VALUES ('sandra', 2, NULL, 10);

INSERT INTO WuenschtSich VALUES ('paul', 1);
INSERT INTO Besitzt VALUES ('paul', 2);
INSERT INTO Besitzt VALUES ('paul', 3);
```

```
INSERT INTO Besitzt VALUES ('paul', 4);
INSERT INTO Besitzt VALUES ('paul', 5);

INSERT INTO Besitzt VALUES ('fritz', 2);
INSERT INTO Besitzt VALUES ('fritz', 3);
INSERT INTO Besitzt VALUES ('fritz', 4);

INSERT INTO Besitzt VALUES ('sandra', 2);
INSERT INTO WuenschtSich VALUES ('sandra', 1);
INSERT INTO WuenschtSich VALUES ('sandra', 4);
```

data1.lst

```
SQL> SELECT * FROM Band_Kuenstlername;

       MID NAME
---------- ------------------------------
         1 Massive Attack
         2 The Doors
         3 The Chemical Brothers
         4 Iggy Pop
         6 Primal Scream
         7 Sleeper
         8 New Order
         9 Blur
        11 Pulp
        12 Bedrock
        13 KYO
        14 Elastica
        15 Leftfield
        16 Underworld
        18 Soul II Soul
        33 Daddy Gee
        34 3D
        35 Mushroom
        37 Everything But The Girl
        38 Tricky
        40 Nicolette

21 rows selected.

SQL> SELECT * FROM Besitzt;

NAME                                    TID
------------------------------ ----------
paul                                     2
paul                                     3
paul                                     4
paul                                     5
fritz                                    2
fritz                                    3
fritz                                    4
sandra                                   2

8 rows selected.

SQL> SELECT * FROM Bewertet;

NAME                          TID REZENSION            NOTE
--------------------- ---------- -------------- ----------
paul                           1                         8
paul                           2                         9
paul                           3                         6
paul                           4                        10
paul                           5                         5
fritz                          2                         8
fritz                          3                        10
fritz                          4                         4
sandra                         1                        10
sandra                         2                        10

10 rows selected.
```

```
SQL> SELECT * FROM Enthaelt;

       TID    TRACKNR     LAENGE        MID        SID
---------- ---------- ---------- ---------- ----------
         1          1        322          1          1
         1          2        316          1          2
         1          3        228          1          3
         1          4        299          1          4
         1          5        321          1          5
         1          6        234          1          6
                    .
                    .
                    .
         4          8        306          3          8
         4          9        243          3          9
         4         10        465          3         10
         5          1        240          3         11
         5          2        394          3         12
         5          3        508          3         13

48 rows selected.

SQL> SELECT * FROM Funktion;

FUNKTION
------------------------------
Produzent
Remix
Featuring
Vocals

SQL> SELECT * FROM Genre;

GENRE_NAME                             OBERGENRE_NAME
-------------------------------------- --------------------------------------
Modern Rock
Rock
Electronica
Alternative Rock                       Modern Rock
Brit Pop                               Alternative Rock
New Wave                               Modern Rock
Classic Rock                           Rock
Psychedelic Rock                       Classic Rock
Glam Rock                              Classic Rock
Ambient                                Electronica
Downbeat                               Electronica
Trip Hop                               Downbeat
House                                  Electronica
Organic House                          House
Techno                                 Electronica
Trance                                 Techno

16 rows selected.

SQL> SELECT * FROM HatFunktion;

       MID FUNKTION                       SONG_MID        SID
---------- ------------------------------ ---------- ----------
        19 Produzent                               1          1
        19 Produzent                               1          2
        19 Produzent                               1          3
        19 Produzent                               1          4
        19 Produzent                               1          5
                           .
                           .
                           .
        22 Vocals                                  3         10
        36 Vocals                                  1          1
        38 Vocals                                  1          2
        40 Vocals                                  1          3
        39 Vocals                                  1          8
        40 Vocals                                  1         10

30 rows selected.
```

```
SQL> SELECT * FROM IstMitglied;

      MID1       MID2 INSTRUMENT      VON       BIS
---------- ---------- --------------- --------- ---------
        23          2 voc
        24          2 d
        25          2 git
        26          2 keyb
        19         18
        27          3
        28          3
        30          1
        31          1
        32          1
        30         33
        31         34
        32         35
        36         37 voc

14 rows selected.

SQL> SELECT * FROM Musiker;

       MID URL
---------- ----------------------------------------
         1 www.massiveattack.co.uk
         2
         3 www.chemicalbrothers.co.uk
         4
         5
                     .
                     .
                     .
        40

40 rows selected.

SQL> SELECT * FROM Person;

       MID VORNAME              NACHNAME             GEBURTSTA G
---------- -------------------- -------------------- --------- -
         5 Brian                Eno                            M
        10 Lou                  Reed                           M
        17 Damon                Albarn                         M
        19 Nellee               Hooper                         M
        20 Pete                 Heller                         M
        21 Beth                 Orton                          W
        22 Richard              Ashcroft                       M
        23 Jim                  Morrison                       M
        24 John                 Densmore                       M
        25 Robby                Krieger                        M
        26 Ray                  Manzarek                       M
        29 Paul                 Rothchild                      M
        27 Ed                   Simons                         M
        28 Tom                  Rowlands                       M
        30 Grant                Marshall                       M
        31 Robert               del Naja                       M
        32 Andrew               Vowles                         M
        36 Tracey               Thorn                          W
        39 Horace               Andy                           M

19 rows selected.

SQL> SELECT * FROM Plattenfirma;

FIRMEN_NAME                            URL
-------------------------------------- ------------------------------
Circa
Elektra/Asylum                         www.electra.com
EMI                                    www.emigroup.com
Virgin                                 www.virginrecords.co.uk
```

```
SQL> SELECT * FROM Song;

MID SID NAME                                       GENRE_NAME        ORIGINAL_MID ORIGINAL_SID
--- --- ---------------------------------------- ----------------- ------------ ------------
  2   1 Break On Through (To The Other Side)      Psychedelic Rock
  2   2 Soul Kitchen                              Psychedelic Rock
  2   3 The Crystal Ship                          Psychedelic Rock
  2   4 Twentieth Century Fox                     Psychedelic Rock
  2   5 Alabama Song (Whisky Bar)                 Psychedelic Rock
                      .
                      .
                      .
  3   7 The State We're In                        Electronica
  3   8 Denmark                                   Electronica
  3   9 Pioneer Skies                             Electronica
  3  10 The Test                                  Electronica
  3  11 Star Guitar (Edit)                        Electronica
  3  12 Base 6                                    Electronica
  3  13 Star Guitar (Pete Heller's Expanded Mix)  Electronica

48 rows selected.

SQL> SELECT * FROM Tontraeger;

       TID TITEL               MID ERSCHEINU PLATTENFIRMA
---------- ----------------- ----- --------- --------------------
         1 Protection            1 19-JAN-95 Circa
         2 The Doors             2 22-JAN-67 Elektra/Asylum
         3 Trainspotting           25-JUL-96 EMI
         4 Come With Us          3 28-JAN-02 Virgin
         5 Star Guitar           3 14-JAN-02 Virgin

SQL> SELECT * FROM Users;

NAME           EMAIL                GEBURTSTA G HERKUNFT        PASSWORT
-------------- -------------------- --------- - --------------- --------------------
paul           paul@gmx.de          15-JUL-74 M M|nchen         123abcdefg
fritz          fritz@web.de         22-JAN-82 M Berlin          123abcdefg
sandra         sandra@freenet.de    01-NOV-77 W Frankfurt       123abcdefg

SQL> SELECT * FROM WuenschtSich;

NAME                                TID
------------------------------ ----------
paul                                 1
sandra                               1
sandra                               4
```

Die "große" Datenbank

Wir verwenden nicht völlig zufällig erzeugte Daten, damit unsere SELECT-Anfragen halbwegs nachvollziehbare Ergebnisse produzieren. Die Hierarchie der Genres, die Liste der Plattenfirmen, eine Liste von Vornamen und Ländern für die Users stammen aus verschiedenen Quellen im Internet und werden mit einem Java-Programm weiterverarbeitet. Die Beziehungen zwischen Tonträgern, Songs und Musikern gewinnen wir mit einem Script in der Sprache PHP aus Winamp-M3U-Playlisten. Andere Informationen (z.B. Zuordnung von Songs zu Genres) sind dann wieder völlig zufällig erzeugt.

m3u.php

```
<pre>
<?php

$file = file('m3ulist.txt');

$artists = array();
$songs = array();
$albums = array();
$albumnames = array();

$recordcompanies = array(
"Alias", "Alternative Tentacles", "American Recordings", "Amphetamine Reptile", "Artemis",
"Asian Man Records", "Asphodel", "Astralwerks", "Atlantic", "B.A. Records", "B.Y.O.",
"Bad Afro", "Bad Taste Records", "Beggars Banquet", "Beyond Music", "Birdnest", "Bitzcore",
"Black Out Records", "BluNoise", "BMG", "Bomp", "Bong Load", "Broken Rekids", "Bungalow",
"Burning Heart", "Cargo Deutschland", "Century Media", "Chemikal Underground", "City Slang",
"Clean Up", "Cold Front", "Columbia", "Compost", "Cooking Vinyl", "Core Tex",
"Crippled Dick Hot Wax", "Damaged Goods", "Deceptive", "Def Jam", "Def Jam (Deutschland)",
"Delicious Vinyl", "Digital Hardcore", "Dischord", "Divine Recordings", "Doghouse Records",
"Dr. Strange", "DreamWorks", "Drive Thru Records", "Earache", "eastwest", "Edel", "EFA",
"El Pocho Loco", "Elementree", "EMI", "Epic", "Epitaph", "Equal Vision", "Estrus",
"Fanboy Records", "Fat Possum", "Fat Wreck", "Fearless Records", "fiction.friction",
"Flip Records", "Food Records", "Four Music", "Fueled By Ramen", "Fused And Bruised",
"G7 Welcoming Committee", "Good Ink Records", "Grand Royal", "Grita!", "Groove Attack",
"Hard On Records", "Heartcore Records", "Hell Cat", "Hollywood Records", "Honest Dons",
"Hopeless Records", "Hulk Rackorz", "Immortal", "Interscope", "Ipecac", "Island Records",
"Jade Tree", "Jeepster", "K Records", "Kill Rock Stars", "Koch Records", "Kool Arrow",
"Kung Fu Records", "Lookout", "Loosegroove", "Loud Records", "LAge DOr", "Mammoth",
"Mans Ruin", "Marina Records", "Mascot Records", "Matador Europe", "Maverick", "Mercury",
"Merge", "Metal Blade", "Meteor City", "Mojo Records", "Mole Listening Pearls",
"Monika Enterprise", "Motor", "Moving Shadow", "Mo Wax", "Mr. Lady", "Music For Nations",
"Mute", "My Records", "Ninja Tune", "Nitro", "No Limit", "Nois-O-Lution", "Nothing",
"Nuclear Blast", "Nude", "On-U Sound", "One Little Indian", "Onefoot Records",
"People Like U", "Plattenmeister", "Polydor", "Poptones", "Posthuman Records",
"Priority Records", "Quarterstick", "Radioactive", "Rawkus", "Recess", "Relapse",
"Revelation", "Roadrunner", "Sanctuary Records Group", "Shimmy Disc", "Skin Graft",
"Skint", "Skunk", "Smells Like Records", "Soda Records", "Some Records", "Sony Music",
"Southern", "SPV", "Stickman Records", "Sub Pop", "Supermusic", "Supersonic", "Surf Dog",
"Sympathy For The Record Industry", "Taang!", "The Music Company", "Theologian Records",
"Thrill Jockey", "Time Bomb", "Tommy Boy", "tontrager and Tontraeger TONTRager",
"Too Pure", "Triggerfish", "TVT Records", "Ubiquity", "V2", "Vaccination", "Vagrant",
"Vinyl Japan", "Virgin", "Virtual Volume", "Vitaminepillen", "Warp", "WEA", "Wolverine",
"WordSound", "XL Recordings", "Yo Mama", "Zomba");

$genres = array(
'Modern Rock', 'Rock', 'Heavy Metal', 'Punk Rock', 'Experimental', 'Electronica', 'Jazz',
'Blues', 'Country', 'Folk', 'Oldies', 'Pop', 'Hip Hop', 'R and B', 'Reggae', 'World',
'Classical', 'New Age', 'Comedy', 'Spoken Word', 'Novelty', 'Contemporary Christian',
'Seasonal', 'Show Tunes', 'Childrens', 'Alternative Rock', 'Experimental Rock', 'Indie Rock',
'Jam Rock', 'New Wave', 'Post Punk', 'Power Pop', 'Classic Rock', 'Hard Rock', 'Funk Metal',
'Hair Metal', 'Industrial Metal', 'Rap Core', 'Nu Metal', 'Thrash', '77 Style Punk',
'Cow Punk', 'Hardcore Punk', 'Pop Punk', 'Proto-punk', 'Psychobilly', 'Riot Grrrl',
'Ska Punk', 'Electroacoustic', 'Environments', 'Experimental Improvisation',
'Experimental Noise', 'Acid Jazz', 'Ambient', 'Big Beat', 'Breakbeat', 'Downbeat',
'Drum n  Bass', 'Electro Funk', 'House', 'Industrial', 'Intelligent Dance Music', 'Techno',
'Be Bop', 'Big Band', 'Crossover Jazz', 'Lounge', 'Vocal Jazz', 'Chicago Blues',
'Country Blues', 'Female Vocal Blues', 'Zydeco Blues', 'Alt Country', 'Bluegrass',
'Contemporary Country', 'Country Rock', 'Traditional Country', '60s Revival', 'Anti-Folk',
'British Folk', 'Contemporary Folk', 'Singer-Songwriter', 'Traditional Folk', 'Doo Wop',
'Early Rock and Roll', 'Rockabilly', 'Surf', 'Dance Pop', 'Easy Listening', 'Euro Pop',
'Soft Rock', 'Teen Pop', 'Vocalists', 'Abstract Hip Hop', 'Bass', 'Gangsta Rap',
'Independent Hip Hop', 'Old School Hip Hop', 'Pop Rap', 'Southern Hip Hop', 'Funk', 'G-Funk',
'Gospel', 'Soul', 'Roots Reggae', 'Ska', 'African', 'Asian', 'Eastern European',
```

```
'Indigenous Music', 'Latin', 'Western European', 'Avant Classical', 'Chamber Music',
'Classical Guitar', 'Composers', 'Opera', 'Solo Instrumental', 'Symphony'
);

$functions = array(
'Produzent', 'Vocals', 'Remix', 'Gastmusiker', 'Co-Writer', 'Video-Regisseur', 'Inspiration',
'Buehnenarbeiter', 'Studioreinigungskraft'
);

foreach($file as $line) {
    $line = trim($line);
    if (!ereg("^\.\\\\([^\]+)\\\\[^\]+\.m3u:(.*)$", $line, $match)) {
        echo "strange line: $line\n";
        continue;
    }
    $record = $match[1];
    $info = $match[2];
    if ($info == "#EXTM3U") continue;
    if (substr($info, 0, 1) != '#') continue;
    if (!ereg("^#EXTINF:([0-9]+),(.*)$", $info, $match)) continue;
    $length = $match[1];
    $song = $match[2];
    if (ereg("^(.*) - (.*)$", $record, $match)) {
        $artist = removeThe(convert($match[1]));
        $album = convert($match[2]);
    } else {
        $artist = null;
        $album = convert($record);
    }
    if (ereg("(.*) - (.*)$", $song, $match)) {
        $songartist = removeThe(convert($match[1]));
        $songtitle = convert($match[2]);
       if (ereg("(.*) - (.*)", $songartist, $match)) {
              $songartist = $match[1];
              $songtitle = "$match[2] - $songtitle";
       }
    } else {
        echo "strange song: $song\n";
    }
              if ($artist) {
           if (!in_array($artist, $artists)) {
               $artists[] = $artist;
           }
    }
    if ($songartist) {
           if (!in_array($songartist, $artists)) {
               $artists[] = $songartist;
           }
         }
    $songs[] = array(
        'artist' => $songartist,
        'title' => $songtitle,
        'length' => $length,
        'album' => "$artist - $album"
    );
    if (!in_array("$artist - $album", $albumnames)) {
        $albums[] = array('artist' => $artist, 'album' => $album);
        $albumnames[] = "$artist - $album";
    }
}

foreach ($functions as $function) {
    echo "INSERT INTO Funktion VALUES ('$function');\n";
}
echo "\n";

$count_mid = 0;
foreach ($artists as $artist) {
    $count_mid++;
    echo "INSERT INTO Musiker VALUES ($count_mid, NULL);\n";
    echo "INSERT INTO Band_Kuenstlername VALUES ($count_mid, '$artist');\n";
}
echo "\n";

$count_tid = 0;
foreach ($albums as $album) {
    $count_tid++;
    $artist = $album['artist'];
    $title = $album['album'];
    $company = $recordcompanies[rand(0, count($recordcompanies)-1)];
```

```
    if ($artist) {
        $mid = array_search($artist, $artists) + 1;
    } else {
        $mid = 'NULL';
    }
    echo "INSERT INTO Tontraeger VALUES ($count_tid, '$title', $mid, NULL, '$company');\n";
}
echo "\n";

$songcounts = array();
$trackcounts = array();
$written_mids = array();
$written_sids = array();

for ($i = 1; $i <= $count_mid; $i++) {
    $songcounts[$i] = 0;
}
for ($i = 1; $i <= $count_tid; $i++) {
    $trackcounts[$i] = 0;
}
foreach ($songs as $song) {
    $artist = $song['artist'];
    $title = $song['title'];
    $length = $song['length'];
    $album = $song['album'];
    $mid = array_search($artist, $artists) + 1;
    $sid = (++$songcounts[$mid]);
    $tid = array_search($album, $albumnames) + 1;
    $trackcounts[$tid]++;
    $genre = $genres[rand(0, count($genres)-1)];
    $written_mids[] = $mid;
    $written_sids[] = $sid;

    if (rand(0, 100) <= 2) {
        $index = rand(1, count($written_mids));
        $original_mid = $written_mids[$index];
        $original_sid = $written_sids[$index];
        if ($original_sid == 0) {
            $original_mid = 'NULL';
            $original_sid = 'NULL';
        }
    } else {
        $original_mid = 'NULL';
        $original_sid = 'NULL';
    }
    echo "INSERT INTO Song VALUES ($mid, $sid, '$title', '$genre', $original_mid,
$original_sid);\n";
    echo "INSERT INTO Enthaelt VALUES ($tid, $trackcounts[$tid], $length, $mid, $sid);\n";
    while (rand(0, 100) > 50) {
        $function = $functions[rand(0, count($functions) - 1)];
        $who = rand(1, $count_mid);
        echo "INSERT INTO HatFunktion VALUES ($who, '$function', $mid, $sid);\n";
    }
}

echo "\n-- Anzahl Tontraeger: $count_tid\n-- Anzahl Musiker: $count_mid\n";

function convert($s) {
        $s = strtolower($s);
        $s = str_replace('ü', 'ue', $s);
        $s = str_replace('ä', 'ae', $s);
        $s = str_replace('ö', 'oe', $s);
        $s = str_replace('ß', 'ss', $s);
        $s = str_replace("'", "''", $s);
        $s = str_replace(' & ', ' and ', $s);
        $s = str_replace('&', ' and ', $s);
        return $s;
}

function removeThe($s) {
        return (substr($s, 0, 4) == 'the ') ? substr($s, 4) : $s;
}

?>
</pre>
```

SQLScriptGenerator.java

```
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Random;
import java.util.Vector;

public class SQLScriptGenerator {
	Random generator = new Random();

	String[] usernames = {"Adinda", "Adrienne", "Aila", "Aimée", "Aina", "Aisha", "Aja",
"Alea", "Alessia", "Aletta", "Alena", "Alessa", "Alexa", "Alina", "Alissa", "Amelie", "Ana",
"Anabelle", "Andrea", "Angeline", "Apollonia", "Apollina", "Arabella", "Ariane", "Arieta",
"Arielle", "Audrey", "Berit", "Bianca", "Blanca", "Blanche", "Bonita", "Carine", "Catja",
"Catrin", "Celie", "Chiara", "Chloe", "Cinja", "Cirstin", "Clarissa", "Claire", "Dana",
"Dania", "Danja", "Dany", "Darina", "Darja", "Desire", "Dina", "Diane", "Elfi", "Elfe",
"Eliane", "Ena", "Enja", "Fenja", "Floriane", "Florina", "Felina", "Gila", "Gina", "Hannah",
"Hanja", "Heide", "Ina", "Insa", "Ira", "Jade", "Jana", "Jean", "Jill", "Katja", "Kim",
"Kirstin", "Kyra", "Lara", "Larissa", "Laura", "Laurine", "Leonie", "Macie", "Madita",
"Maja", "Majella", "Malinda", "Mara", "Mariam", "Marina", "Marja", "Marita", "Melanie",
"Meret", "Miriam", "Mirinda", "Mirja", "Moana", "Mona", "Nathalie", "Nadja", "Nina",
"Noelani", "Noemi", "Ornella", "Sandrina", "Saskia", "Serena", "Silja", "Silvie", "Simone",
"Sina", "Sinja", "Sonia", "Sophie", "Stephanie", "Svenja", "Tina", "Twyla", "Una", "Undine",
"Uta", "Valea", "Valerie", "Vanessa", "Yasmine", "Alain", "Andree", "Axel", "Bjoern",
"Boleslaw", "Cai", "Carl", "Curd", "Christian", "Claude", "Enno", "Falk", "Floris", "Gabor",
"Gero", "Golo", "Hilger", "Hilmar", "Hinrich", "Ian", "Ibo", "Ingvar", "Jan", "Janik",
"Jano", "Jaro", "Jason", "Jean", "Jens", "Jendrik", "Jeremy", "Jero", "Jirko", "Jiri",
"Jonah", "Juan", "Jul", "Jules", "Julian", "Joerg", "Kay", "Kim", "Kolja", "Lars", "Laszlo",
"Laurin", "Leander", "Lennart", "Lasse", "Luc", "Marian", "Marlin", "Marjan", "Mika",
"Mikkel", "Mitja", "Nando", "Nic", "Nicky", "Niels", "Niki", "Raffael", "Ralph", "Rico",
"Rodja", "Sando", "Sandro", "Sean", "Serge", "Simon", "Stephane", "Till", "Timo", "Tyll",
"Ugo", "Uli", "Ulf", "Vadim", "Valerian", "Vincent", "Wanja", "Wasja", "Welf", "Yanik",
"Yannic", "Yves"};
	String[] countrynames = {"Aegypten", "Albanien", "Andorra", "Armenien", "Azoren",
"Belgien", "Bosnien", "Bulgarien", "Danemark", "Deutschland", "England", "Estland",
"Faroeer", "Finnland", "Frankreich", "Georgien", "Griechenland", "Irland", "Island",
"Israel", "Italien", "Jugoslawien", "Kanada", "Kanarische Inseln", "Kroatien", "Kurdistan",
"Lettland", "Liechtenstein", "Litauen", "Luxemburg", "Malta", "Marokko", "Mazedonien",
"Moldawien", "Monaco", "Niederlande", "Norwegen", "Oesterreich", "Palastina", "Polen",
"Portugal", "Rumanien", "Russland", "San Marino", "Schottland", "Schweden", "Schweiz",
"Slovakei", "Slovenien", "Spanien", "Tschechien", "Tunesien", "Tuerkei", "Ukraine", "Ungarn",
"U.S.A.", "Vatikan", "Weissrussland", "Zypern"};
	String[] plattenfirmen = {"Alias", "Alternative Tentacles", "American Recordings",
"Amphetamine Reptile", "Artemis", "Asian Man Records", "Asphodel", "Astralwerks", "Atlantic",
"B.A. Records", "B.Y.O.", "Bad Afro", "Bad Taste Records", "Beggars Banquet", "Beyond Music",
"Birdnest", "Bitzcore", "Black Out Records", "BluNoise", "BMG", "Bomp", "Bong Load", "Broken
Rekids", "Bungalow", "Burning Heart", "Cargo Deutschland", "Century Media", "Chemikal
Underground", "City Slang", "Clean Up", "Cold Front", "Columbia", "Compost", "Cooking Vinyl",
"Core Tex", "Crippled Dick Hot Wax", "Damaged Goods", "Deceptive", "Def Jam", "Def Jam
(Deutschland)", "Delicious Vinyl", "Digital Hardcore", "Dischord", "Divine Recordings",
"Doghouse Records", "Dr. Strange", "DreamWorks", "Drive Thru Records", "Earache", "eastwest",
"Edel", "EFA", "El Pocho Loco", "Elementree", "EMI", "Epic", "Epitaph", "Equal Vision",
"Estrus", "Fanboy Records", "Fat Possum", "Fat Wreck", "Fearless Records",
"fiction.friction", "Flip Records", "Food Records", "Four Music", "Fueled By Ramen", "Fused
And Bruised", "G7 Welcoming Committee", "Good Ink Records", "Grand Royal", "Grita!", "Groove
Attack", "Hard On Records", "Heartcore Records", "Hell Cat", "Hollywood Records", "Honest
Dons", "Hopeless Records", "Hulk Rackorz", "Immortal", "Interscope", "Ipecac", "Island
Records", "Jade Tree", "Jeepster", "K Records", "Kill Rock Stars", "Koch Records", "Kool
Arrow", "Kung Fu Records", "Lookout", "Loosegroove", "Loud Records", "LAge DOr", "Mammoth",
"Mans Ruin", "Marina Records", "Mascot Records", "Matador Europe", "Maverick", "Mercury",
"Merge", "Metal Blade", "Meteor City", "Mojo Records", "Mole Listening Pearls", "Monika
Enterprise", "Motor", "Moving Shadow", "Mo Wax", "Mr. Lady", "Music For Nations", "Mute", "My
Records", "Ninja Tune", "Nitro", "No Limit", "Nois-O-Lution", "Nothing", "Nuclear Blast",
"Nude", "On-U Sound", "One Little Indian", "Onefoot Records", "People Like U",
"Plattenmeister", "Polydor", "Poptones", "Posthuman Records", "Priority Records",
"Quarterstick", "Radioactive", "Rawkus", "Recess", "Relapse", "Revelation", "Roadrunner",
"Sanctuary Records Group", "Shimmy Disc", "Skin Graft", "Skint", "Skunk", "Smells Like
Records", "Soda Records", "Some Records", "Sony Music", "Southern", "SPV", "Stickman
Records", "Sub Pop", "Supermusic", "Supersonic", "Surf Dog", "Sympathy For The Record
Industry", "Taang!", "The Music Company", "Theologian Records", "Thrill Jockey", "Time Bomb",
"Tommy Boy", "tontrager and Tontraeger TONTRager", "Too Pure", "Triggerfish", "TVT Records",
"Ubiquity", "V2", "Vaccination", "Vagrant", "Vinyl Japan", "Virgin", "Virtual Volume",
"Vitaminepillen", "Warp", "WEA", "Wolverine", "WordSound", "XL Recordings", "Yo Mama",
"Zomba"};

	String[] obergenres = {"Modern Rock", "Rock", "Heavy Metal", "Punk Rock",
"Experimental", "Electronica", "Jazz", "Blues", "Country", "Folk", "Oldies", "Pop", "Hip
```

```
Hop", "R and B", "Reggae", "World", "Classical", "New Age", "Comedy / Spoken Word",
"Novelty", "Contemporary Christian", "Seasonal", "Show Tunes", "Childrens"};

	String[] genres0 = {"Alternative Rock", "Experimental Rock", "Indie Rock", "Jam
Rock", "New Wave", "Post Punk", "Power Pop"};
	String[] genres1 = {"Classic Rock", "Hard Rock"};
	String[] genres2 = {"Funk Metal", "Hair Metal", "Industrial Metal", "Rap Core", "Nu
Metal", "Thrash"};
	String[] genres3 = {"77 Style Punk", "Cow Punk", "Hardcore Punk", "Pop Punk", "Proto-
punk", "Psychobilly", "Riot Grrrl", "Ska Punk"};
	String[] genres4 = {"Electroacoustic", "Environments", "Experimental Improvisation",
"Experimental Noise"};
	String[] genres5 = {"Acid Jazz", "Ambient", "Big Beat", "Breakbeat", "Downbeat",
"Drum n Bass", "Electro Funk", "House", "Industrial", "Intelligent Dance Music", "Techno"};
	String[] genres6 = {"Be Bop", "Big Band", "Crossover Jazz", "Lounge", "Vocal Jazz"};
	String[] genres7 = {"Chicago Blues", "Country Blues", "Female Vocal Blues", "Zydeco
Blues"};
	String[] genres8 = {"Alt Country", "Bluegrass", "Contemporary Country", "Country
Rock", "Traditional Country"};
	String[] genres9 = {"60s Revival", "Anti-Folk", "British Folk", "Contemporary Folk",
"Singer-Songwriter", "Traditional Folk"};
	String[] genres10 = {"Doo Wop", "Early Rock and Roll", "Rockabilly", "Surf"};

	String[] genres11 = {"Dance Pop", "Easy Listening", "Euro Pop", "Soft Rock", "Teen
Pop", "Vocalists"};
	String[] genres12 = {"Abstract Hip Hop", "Bass", "Gangsta Rap", "Independent Hip
Hop", "Old School Hip Hop", "Pop Rap", "Southern Hip Hop"};
	String[] genres13 = {"Funk", "G-Funk", "Gospel", "Soul"};
	String[] genres14 = {"Roots Reggae", "Ska"};
	String[] genres15 = {"African", "Asian", "Eastern European", "Indigenous Music",
"Latin", "Western European"};
	String[] genres16 = {"Avant Classical", "Chamber Music", "Classical Guitar",
"Composers", "Opera", "Solo Instrumental", "Symphony"};

	String songFile = "";

	Vector fileLines = new Vector();

	Users[] users = new Users[550];

	public SQLScriptGenerator() {
		//Users-Tabelle
		String[] emails = new String[users.length];
		String[] namen = new String[users.length];
		for(int i=0; i<users.length; i++) {

			boolean emailIsUnique;
			boolean nameIsUnique;
			do {
				String randomUsername =
					usernames[generateNum(usernames.length)];
				namen[i] = generateString(randomUsername,
					generateNum(14-randomUsername.length()));
				emails[i] = generateEmail(namen[i]);

				emailIsUnique = true;
				nameIsUnique = true;
				for (int j=0; j<i; j++) {
					if (emails[i].equals(emails[j])) {
						emailIsUnique = false;
					}
				}
				for (int j=0; j<i; j++) {
					if (namen[i].equals(namen[j])) {
						nameIsUnique = false;
					}
				}
			} while (!emailIsUnique || !nameIsUnique);
			String geburtstag = generateDate();
			int geschlechtNr = generateNum(2);
			char geschlecht ='W';
			if (geschlechtNr==0) {
				geschlecht ='M';
			}
			String herkunft = countrynames[generateNum(countrynames.length)];
			String passwort;
			do {
				passwort = generateString(generateNum(9));
			} while (passwort.length()<6);
```

```
        users[i] = new Users(namen[i], emails[i], geburtstag, geschlecht,
                herkunft, passwort);
}


//Plattenfirma-Tabelle
Plattenfirma[] plattenfirma = new Plattenfirma[plattenfirmen.length];

for(int i=0; i<plattenfirmen.length; i++) {
        String firmenname = plattenfirmen[i];
        String url = generateUrl(firmenname);
        plattenfirma[i] = new Plattenfirma(firmenname, url);
}

// Bewertet-Tabelle
Bewertet[] bewertet = new Bewertet[200];

for(int i=0; i<bewertet.length; i++) {
        int bewerteterTontraeger = generateNum(121) + 1;
        int bewertenderUser;
        do {
                bewertenderUser = generateNum(users.length);
        } while (!isUniqueCombination(bewertenderUser, bewerteterTontraeger,
                (UniqueTestIf[])bewertet));
        int note = generateNum(11);
        String rezension;
        if (generateNum(100)>90) {
                rezension = "'" + generateRezension(generateNum(1000)) + "'";
        } else {
                rezension = "NULL";
        }
        bewertet[i] = new Bewertet(users[bewertenderUser].name,
                bewerteterTontraeger, rezension, note);
}

// Besitzt-Tabelle
Besitzt[] besitzt = new Besitzt[200];

for(int i=0; i<besitzt.length; i++) {
        int besitzer = generateNum(users.length);
        int besitztTontraeger;
        do {
                besitztTontraeger = generateNum(121) + 1;
        } while (!isUniqueCombination(besitzer, besitztTontraeger,
                (UniqueTestIf[])besitzt));
        besitzt[i] = new Besitzt(users[besitzer].name, besitztTontraeger);
}

// WuenschtSich-Tabelle
WuenschtSich[] wuenschtSich = new WuenschtSich[200];

for(int i=0; i<wuenschtSich.length; i++) {
        int wuenschender = generateNum(users.length);
        int gewuenschterTontraeger;
        do {
                gewuenschterTontraeger = generateNum(121) + 1;
        } while (!isUniqueCombination(wuenschender, gewuenschterTontraeger,
                wuenschtSich));
        wuenschtSich[i] = new WuenschtSich(users[wuenschender].name,
                gewuenschterTontraeger);
}

// alles ausgeben lassen
String script = "";

// Users
for(int i=0; i<users.length; i++) {
        script = new String("INSERT INTO Users VALUES ('"+users[i].name+"',
                '"+users[i].email+"', to date('"+users[i].geburtstag+"',
                'yyyy-mm-dd'), '"+users[i].geschlecht+"',
                '"+users[i].herkunft+"', '"+users[i].passwort+"');" );
        fileLines.addElement(script);
}

// Plattenfirma
for(int i=0; i<plattenfirma.length; i++) {
        script = new String("INSERT INTO Plattenfirma VALUES
                ('"+plattenfirma[i].firmenname+"', '"+plattenfirma[i].url+"');"
                );
```

```
        fileLines.addElement(script);
}

// Bewertet
for(int i=0; i<bewertet.length; i++) {
        script = new String("INSERT INTO Bewertet VALUES
                ('"+bewertet[i].user+"', "+bewertet[i].tid+",
                "+bewertet[i].rezension+", "+bewertet[i].note+");" );
        fileLines.addElement(script);
}

// Besitzt
for(int i=0; i<besitzt.length; i++) {
        script = new String("INSERT INTO Besitzt VALUES
                ('"+besitzt[i].user+"', "+besitzt[i].besitztTontraeger+");" );
        fileLines.addElement(script);
}

// WuenschtSich
for(int i=0; i<wuenschtSich.length; i++) {
        script = new String("INSERT INTO WuenschtSich VALUES
                ('"+wuenschtSich[i].user+"',
                "+wuenschtSich[i].gewuenschterTontraeger+");" );
        fileLines.addElement(script);
}

for(int i=0; i<obergenres.length; i++) {
        script = new String("INSERT INTO Genre VALUES ('"+obergenres[i]+"',
                NULL);" );
        fileLines.addElement(script);
}
for(int i=0; i<genres0.length; i++) {
        script = new String("INSERT INTO Genre VALUES ('"+genres0[i]+"',
                '"+obergenres[0]+"');");
        fileLines.addElement(script);
}
for(int i=0; i<genres1.length; i++) {
        script = new String("INSERT INTO Genre VALUES ('"+genres1[i]+"',
                '"+obergenres[1]+"');");
        fileLines.addElement(script);
}
for(int i=0; i<genres2.length; i++) {
        script = new String("INSERT INTO Genre VALUES ('"+genres2[i]+"',
                '"+obergenres[2]+"');");
        fileLines.addElement(script);
}
for(int i=0; i<genres3.length; i++) {
        script = new String("INSERT INTO Genre VALUES ('"+genres3[i]+"',
                '"+obergenres[3]+"');");
        fileLines.addElement(script);
}
for(int i=0; i<genres4.length; i++) {
        script = new String("INSERT INTO Genre VALUES ('"+genres4[i]+"',
                '"+obergenres[4]+"');");
        fileLines.addElement(script);
}
for(int i=0; i<genres5.length; i++) {
        script = new String("INSERT INTO Genre VALUES ('"+genres5[i]+"',
                '"+obergenres[5]+"');");
        fileLines.addElement(script);
}
for(int i=0; i<genres6.length; i++) {
        script = new String("INSERT INTO Genre VALUES ('"+genres6[i]+"',
                '"+obergenres[6]+"');");
        fileLines.addElement(script);
}
for(int i=0; i<genres7.length; i++) {
        script = new String("INSERT INTO Genre VALUES ('"+genres7[i]+"',
                '"+obergenres[7]+"');");
        fileLines.addElement(script);
}
for(int i=0; i<genres8.length; i++) {
        script = new String("INSERT INTO Genre VALUES ('"+genres8[i]+"',
                '"+obergenres[8]+"');");
        fileLines.addElement(script);
}
for(int i=0; i<genres9.length; i++) {
        script = new String("INSERT INTO Genre VALUES ('"+genres9[i]+"',
                '"+obergenres[9]+"');");
        fileLines.addElement(script);
```

```
        }
        for(int i=0; i<genres10.length; i++) {
                script = new String("INSERT INTO Genre VALUES ('"+genres10[i]+"',
                        '"+obergenres[10]+"');");
                fileLines.addElement(script);
        }
        for(int i=0; i<genres11.length; i++) {
                script = new String("INSERT INTO Genre VALUES ('"+genres11[i]+"',
                        '"+obergenres[11]+"');");
                fileLines.addElement(script);
        }
        for(int i=0; i<genres12.length; i++) {
                script = new String("INSERT INTO Genre VALUES ('"+genres12[i]+"',
                        '"+obergenres[12]+"');");
                fileLines.addElement(script);
        }
        for(int i=0; i<genres13.length; i++) {
                script = new String("INSERT INTO Genre VALUES ('"+genres13[i]+"',
                        '"+obergenres[13]+"');");
                fileLines.addElement(script);
        }
        for(int i=0; i<genres14.length; i++) {
                script = new String("INSERT INTO Genre VALUES ('"+genres14[i]+"',
                        '"+obergenres[14]+"');");
                fileLines.addElement(script);
        }
        for(int i=0; i<genres15.length; i++) {
                script = new String("INSERT INTO Genre VALUES ('"+genres15[i]+"',
                        '"+obergenres[15]+"');");
                fileLines.addElement(script);
        }
        for(int i=0; i<genres16.length; i++) {
                script = new String("INSERT INTO Genre VALUES ('"+genres16[i]+"',
                        '"+obergenres[16]+"');");
                fileLines.addElement(script);
        }
        writeToFile(fileLines);
}

public boolean isUniqueCombination(int user, int record, UniqueTestIf[] array) {
        for (int i = 0; i < array.length; i++) {
                if (array[i] == null) continue;
                if (array[i].isSame(users[user].name, record)) return false;
        }
        return true;
}

public void writeToFile(Vector lines) {
        String scriptFile = "script.txt";
        try {
                BufferedWriter endfile = new BufferedWriter(new
                        FileWriter(scriptFile));
                for (int i=0; i<lines.size(); i++) {
                        String line = lines.elementAt(i).toString();
                        endfile.write(line);
                        endfile.newLine();
                }
                endfile.close();
                System.out.println("SQL*Plus-Script erstellt");
        }
        catch (IOException e) {
                System.out.println("Fehler beim Erstellen der Datei");
        }
}

public static void main(String[]args) {
        new SQLScriptGenerator();

}

public String generateEmail(String name) {
//String name = generateString(1+generateNum(7));
String host = generateString(1+generateNum(7));
        String country = generateString(2);
        return (name+"@"+host+"."+country);
}

public String generateUrl(String name) {
//String name = generateString(1+generateNum(12));
        String country = generateString(2);
```

```
        name = name.replace( ' ', '_' );
        if (name.length()>13) {
                name = name.substring(0,13);
        }
        return ("http://www."+name+"."+country);
}


public String generateDate() {
        int month = 1 + generateNum(11);
int year = 1940 + generateNum(61);
        int day = 1 + generateNum(30);
        if (month==2 && day>28) {
                day = 28;
        }
        return (year+"-"+month+"-"+day);
}

public int generateNum(int number) {
int randomNum = generator.nextInt(number);
        return randomNum;
}

public String generateString(int number) {
        String returnString = new String();
        for(int j=0; j<number; j++) {
                int g =(int)(Math.random()*36);
                char c = toChar(g);
                returnString += c;
        }
        return returnString;
}

public String generateRezension(int number) {
        String returnString = new String();
        for(int j=0; j<number; j++) {
                int g =(int)(Math.random()*80);
                char c;
                if (g>62) {
                        c = ' ';
                } else {
                        c = toChar(g);
                }
                returnString += c;
        }
        return returnString;
}

public String generateString(String name, int number) {
        String returnString = new String(name);
        for(int j=0; j<(name.length()-number); j++) {
                int g =(int)(Math.random()*36);
                char c = toChar(g);
                returnString += c;
        }
        return returnString;
}

public char toChar(int i)     {
        char d='a';
        switch(i) {
                case  0: {d='z';break;}
                case  1: {d='a';break;}
                case  2: {d='b';break;}
                case  3: {d='c';break;}
                case  4: {d='d';break;}
                case  5: {d='e';break;}
                case  6: {d='f';break;}
                case  7: {d='g';break;}
                case  8: {d='h';break;}
                case  9: {d='i';break;}
                case 10: {d='j';break;}
                case 11: {d='k';break;}
                case 12: {d='l';break;}
                case 13: {d='m';break;}
                case 14: {d='n';break;}
                case 15: {d='o';break;}
                case 16: {d='p';break;}
                case 17: {d='q';break;}
                case 18: {d='r';break;}
```

```
                        case 19: {d='s';break;}
                        case 20: {d='t';break;}
                        case 21: {d='u';break;}
                        case 22: {d='v';break;}
                        case 23: {d='w';break;}
                        case 24: {d='x';break;}
                        case 25: {d='y';break;}
                        case 26: {d='z';break;}
                        case 27: {d='0';break;}
                        case 28: {d='1';break;}
                        case 29: {d='2';break;}
                        case 30: {d='3';break;}
                        case 31: {d='4';break;}
                        case 32: {d='5';break;}
                        case 33: {d='6';break;}
                        case 34: {d='7';break;}
                        case 35: {d='8';break;}
                        case 36: {d='9';break;}
                        case 37: {d='A';break;}
                        case 38: {d='B';break;}
                        case 39: {d='C';break;}
                        case 40: {d='D';break;}
                        case 41: {d='E';break;}
                        case 42: {d='F';break;}
                        case 43: {d='G';break;}
                        case 44: {d='H';break;}
                        case 45: {d='I';break;}
                        case 46: {d='J';break;}
                        case 47: {d='K';break;}
                        case 48: {d='L';break;}
                        case 49: {d='M';break;}
                        case 50: {d='N';break;}
                        case 51: {d='O';break;}
                        case 52: {d='P';break;}
                        case 53: {d='Q';break;}
                        case 54: {d='R';break;}
                        case 55: {d='S';break;}
                        case 56: {d='T';break;}
                        case 57: {d='U';break;}
                        case 58: {d='V';break;}
                        case 59: {d='W';break;}
                        case 60: {d='X';break;}
                        case 61: {d='Y';break;}
                        case 62: {d='Z';break;}
                }
                return d;
        }
}
```

Users.java

```
public class Users {
        String email;
        String name;
        String geburtstag;
        char geschlecht;
        String herkunft;
        String passwort;

        public Users(String name, String email, String geburtstag, char geschlecht, String
herkunft, String passwort) {
                this.email = email;
                this.name = name;
                this.geburtstag = geburtstag;
                this.geschlecht = geschlecht;
                this.herkunft = herkunft;
                this.passwort = passwort;
        }
}
```

Plattenfirma.java

```
public class Plattenfirma {
	String firmenname;
	String url;

	public Plattenfirma(String firmenname, String url) {
		this.firmenname = firmenname;
		this.url = url;
	}
}
```

Genre.java

```
public class Genre {
	String genre name;
	String obergenre_name;

	public Genre(String genre_name, String obergenre_name) {
		this.genre name = genre name;
		this.obergenre_name = obergenre_name;
	}
}
```

UniqueTestIf.java

```
public interface UniqueTestIf {
	public boolean isSame(String user, int item);
}
```

Besitzt.java

```
public class Besitzt implements UniqueTestIf {
	String user;
	int besitztTontraeger;

	public Besitzt(String user, int besitztTontraeger) {
		this.user = user;
		this.besitztTontraeger = besitztTontraeger;
	}

	public boolean isSame(String user, int item) {
		return (user == this.user && item == besitztTontraeger);
	}
}
```

Bewertet.java

```
public class Bewertet implements UniqueTestIf {
	String user;
	int tid;
	String rezension ="";
	int note;

	public Bewertet(String user, int tid, String rezension, int note) {
		this.user = user;
		this.tid = tid;
		this.rezension = rezension;
		this.note = note;
	}

	public boolean isSame(String user, int item) {
		return (user == this.user && item == tid);
	}
}
```

WuenschtSich.java

```
public class WuenschtSich implements UniqueTestIf {
	String user;
	int gewuenschterTontraeger;

	public WuenschtSich(String user, int gewuenschterTontraeger) {
		this.user = user;
		this.gewuenschterTontraeger = gewuenschterTontraeger;
	}

	public boolean isSame(String user, int item) {
		return (user == this.user && item == gewuenschterTontraeger);
	}
}
```

results.lst

```
SQL> SELECT 'Band_Kuenstlername' AS relation, COUNT(*) AS datensaetze FROM Band_Kuenstlername
  2  UNION
  3  SELECT 'Besitzt' AS relation, COUNT(*) AS datensaetze FROM Besitzt
  4  UNION
  5  SELECT 'Bewertet' AS relation, COUNT(*) AS datensaetze FROM Bewertet
  6  UNION
  7  SELECT 'Enthaelt' AS relation, COUNT(*) AS datensaetze FROM Enthaelt
  8  UNION
  9  SELECT 'Funktion' AS relation, COUNT(*) AS datensaetze FROM Funktion
 10  UNION
 11  SELECT 'Genre' AS relation, COUNT(*) AS datensaetze FROM Genre
 12  UNION
 13  SELECT 'HatFunktion' AS relation, COUNT(*) AS datensaetze FROM HatFunktion
 14  UNION
 15  SELECT 'IstMitglied' AS relation, COUNT(*) AS datensaetze FROM IstMitglied
 16  UNION
 17  SELECT 'Musiker' AS relation, COUNT(*) AS datensaetze FROM Musiker
 18  UNION
 19  SELECT 'Person' AS relation, COUNT(*) AS datensaetze FROM Person
 20  UNION
 21  SELECT 'Plattenfirma' AS relation, COUNT(*) AS datensaetze FROM Plattenfirma
 22  UNION
 23  SELECT 'Song' AS relation, COUNT(*) AS datensaetze FROM Song
 24  UNION
 25  SELECT 'Tontraeger' AS relation, COUNT(*) AS datensaetze FROM Tontraeger
 26  UNION
 27  SELECT 'Users' AS relation, COUNT(*) AS datensaetze FROM Users
 28  UNION
 29  SELECT 'WuenschtSich' AS relation, COUNT(*) AS datensaetze FROM WuenschtSich;

RELATION           DATENSAETZE
------------------ -----------
Band_Kuenstlername         410
Besitzt                    200
Bewertet                   200
Enthaelt                  1735
Funktion                     9
Genre                      119
HatFunktion               1689
IstMitglied                  0
Musiker                    410
Person                       0
Plattenfirma               181
Song                      1735
Tontraeger                 122
Users                      549
WuenschtSich               200

15 rows selected.
```

SQL-Anfragen

1. Durchschnittsnote, mit der die User die Alben eines Künstlers (z.B. Bruce Springsteen) bewerten, aufgeteilt nach Geschlecht der User

```
SQL> SELECT AVG(b.note) AS durchschnittsnote, u.geschlecht
FROM Bewertet b, Tontraeger t, Band_Kuenstlername k, Users u
WHERE u.name = b.name
        AND b.tid = t.tid
        AND t.mid = k.mid
        AND k.name LIKE '%springsteen%'
GROUP BY u.geschlecht;
  2    3    4    5    6    7
DURCHSCHNITTSNOTE G
----------------- -
              3.3 M
              5.2 W
```

2. Alle Songs, die ein Künstler (z.B. Marilyn Manson) selbst veröffentlicht hat, oder an denen er mitgewirkt (Produzent, Gastsänger...) hat.

```
SQL> (SELECT s.name, k.name, 'Eigener Song' AS funktion
  2  FROM Song s, Band_Kuenstlername k
  3  WHERE s.mid = k.mid
  4     AND k.name LIKE '%marilyn manson%')
  5  UNION
  6  (SELECT s.name, k1.name, f.funktion
  7  FROM Song s, Band_Kuenstlername k1, HatFunktion f, Band_Kuenstlername k2
  8  WHERE k1.mid = s.mid
  9     AND s.mid = f.song_mid AND s.sid = f.sid
 10     AND f.mid = k2.mid
 11     AND k2.name LIKE '%marilyn manson%');

NAME                           NAME                      FUNKTION
------------------------------ ------------------------- -----------------
my best was never good enough  bruce springsteen         Remix
reason to believe              bruce springsteen         Buehnenarbeiter
rock is dead                   marilyn manson            Eigener Song
valentine's day                bruce springsteen         Produzent
```

3. Welche Genres hat die Plattenfirma 'Bad Taste Records' im Programm? Wie viele Songs von jedem dieser Genres?

```
SQL> SELECT s.genre_name, COUNT(*) AS song_anzahl
FROM Song s, Enthaelt e, Tontraeger t
WHERE s.sid = e.sid AND s.mid = e.mid
        AND e.tid = t.tid
        AND t.plattenfirma = 'Bad Taste Records'
GROUP BY s.genre_name
ORDER BY COUNT(*) DESC;
  2    3    4    5    6    7
GENRE_NAME                                SONG_ANZAHL
----------------------------------------- -----------
Electroacoustic                                     2
Solo Instrumental                                   2
60s Revival                                         1
Ambient                                             1
Bass                                                1
House                                               1
Proto-punk                                          1
World                                               1
Rap Core                                            1
Nu Metal                                            1
Chamber Music                                       1
Classical                                           1
Contemporary Country                                1

13 rows selected.
```

4. Liste aller Plattenfirmen, sortiert nach dem prozentualen Anteil von Songs mit dem Genre 'Teen Pop' in ihrem Programm

```
SQL> SELECT (teenpop_songs/all_songs*100) AS teenpop_anteil, query1.plattenfirma
  2  FROM
  3     (SELECT COUNT(*) AS all_songs, t.plattenfirma
  4     FROM Song s, Enthaelt e, Tontraeger t
  5     WHERE s.sid = e.sid AND s.mid = e.mid
  6             AND e.tid = t.tid
  7     GROUP BY t.plattenfirma) query1,
  8     (SELECT COUNT(*) AS teenpop_songs, t.plattenfirma
  9     FROM Song s, Enthaelt e, Tontraeger t
 10     WHERE s.sid = e.sid AND s.mid = e.mid
 11             AND e.tid = t.tid
 12             AND s.genre_name = 'Teen Pop'
 13     GROUP BY t.plattenfirma) query2
 14  WHERE query1.plattenfirma = query2.plattenfirma
 15  ORDER BY teenpop_anteil DESC;

TEENPOP_ANTEIL PLATTENFIRMA
-------------- ----------------------------------------
    5.88235294 Time Bomb
             5 Hulk Rackorz
    4.87804878 Epitaph
    4.54545455 Crippled Dick Hot Wax
    3.22580645 Triggerfish
    3.03030303 Birdnest
    3.03030303 WEA
     2.3255814 BluNoise
    1.63934426 Shimmy Disc

9 rows selected.
```

5. Welche Produzenten arbeiten am erfolgreichsten? Dazu ziehen wir die User-Bewertungen der von ihm produzierten Songs heran.

```
SQL> SELECT k.name, b.bewertung
  2  FROM
  3     Band_Kuenstlername k,
  4     (SELECT AVG(b.note) AS bewertung, f.mid
  5     FROM Bewertet b, Enthaelt e, HatFunktion f
  6     WHERE b.tid = e.tid
  7             AND e.mid = f.song_mid AND e.sid = f.sid
  8             AND f.funktion = 'Produzent'
  9     GROUP BY f.mid) b
 10  WHERE k.mid = b.mid AND b.bewertung IN
 11     (SELECT MAX(AVG(b.note))
 12     FROM Bewertet b, Enthaelt e, HatFunktion f
 13     WHERE b.tid = e.tid
 14             AND e.mid = f.song_mid AND e.sid = f.sid
 15             AND f.funktion = 'Produzent'
 16     GROUP BY f.mid);


NAME                           BEWERTUNG
------------------------------ ----------
bruce springsteen                      10
kmc                                    10
shidapu                                10
vectrolab                              10
ukw                                    10
```

6. Welche Cover-Songs erzielen bei den Usern höhere Bewertungen als das Original?

```
SQL> SELECT s1.name AS cover_titel, k1.name cover_kuenstler,
  2     s2.name AS original_titel, k2.name AS original_kuenstler
  3  FROM Song s1, Band_Kuenstlername k1, Song s2, Band_Kuenstlername k2,
  4     (SELECT e.mid, e.sid, AVG(b.note) as bewertung
  5     FROM Enthaelt e, Bewertet b
  6     WHERE b.tid = e.tid
  7     GROUP BY e.mid, e.sid) b1,
  8     (SELECT e.mid, e.sid, AVG(b.note) as bewertung
  9     FROM Enthaelt e, Bewertet b
 10     WHERE b.tid = e.tid
 11     GROUP BY e.mid, e.sid) b2
 12  WHERE s1.mid = k1.mid
 13     AND s2.mid = k2.mid
 14     AND s1.original_mid = s2.mid AND s1.original_sid = s2.sid
 15     AND s1.mid = b1.mid AND s1.sid = b1.sid
 16     AND s2.mid = b2.mid AND s2.sid = b2.sid
 17     AND b1.bewertung > b2.bewertung;

COVER_TITEL                                 COVER_KUENSTLER
------------------------------------------- ------------------------------
ORIGINAL_TITEL                              ORIGINAL_KUENSTLER
------------------------------------------- ------------------------------
wonderland                                  nils lofgren
better days                                 bruce springsteen

education                                   inchtabokatables
darkness on the edge of town                bruce springsteen

retox                                       fatboy slim
my beautiful reward                         bruce springsteen

krieg                                       gundermann
highway 29                                  bruce springsteen

shopping                                    guy farley
sinaloa cowboys                             bruce springsteen

col                                         morcheeba
seven angels                                bruce springsteen

exchange                                    massive attack
so much more                                beth orton

gotcha - starsky and hutch                  butlers
blood red river                             beth orton

human touch                                 bruce springsteen
central reservation (the then again version)  beth orton

demons                                      fatboy slim
strong                                      blank and jones

got glint?                                  chemical brothers
b-boy-style                                 blank and jones

universal nation '99 (ferry corsten mix)    push
crossroads                                  butlers

little wing                                 jimi hendrix
after dark                                  tito and tarantula

i'm through with love                       keith jarrett
synaesthesia (en motion mix)                thrillseekers

funky shit                                  prodigy
think about                                 dan dillon

get up                                      messer banzani
all out                                     nils lofgren

star guitar (edit)                          chemical brothers
jack rabbit slim twist contest              jarome patrick hoban

deltabeat                                   adam blade
6 underground                               sneaker pimps


18 rows selected.
```

7. Wo stammen die User her, die Tonträger mit dem Genre "Jazz" besitzen?

```
SQL> SELECT u.herkunft, COUNT(*)
FROM Users u, Besitzt b, Enthaelt e, Song s
WHERE s.genre_name = 'Jazz'
        AND s.mid = e.mid AND s.sid = e.sid
        AND e.tid = b.tid
        AND b.name = u.name
GROUP BY u.herkunft
ORDER BY COUNT(*) DESC;
  2    3    4    5    6    7    8
HERKUNFT                                           COUNT(*)
-------------------------------------------------- ----------
Norwegen                                                    2
Aegypten                                                    1
Andorra                                                     1
Belgien                                                     1
Bulgarien                                                   1
England                                                     1
Faroeer                                                     1
Weissrussland                                               1
U.S.A.                                                      1
Kanarische Inseln                                           1
Kurdistan                                                   1
Lettland                                                    1
Marokko                                                     1
Monaco                                                      1

14 rows selected.
```

8. Wie lang sind die Songs verschiedener Genres im Durchschnitt (in Sekunden)?

```
SQL> SELECT AVG(e.laenge) AS durchschnittslaenge, s.genre_name
FROM Song s, Enthaelt e
WHERE s.mid = e.mid AND s.sid = e.sid
GROUP BY s.genre_name
ORDER BY AVG(e.laenge) DESC;
  2    3    4    5
DURCHSCHNITTSLAENGE GENRE_NAME
------------------- ----------------------------------------
             396.75 Funk Metal
                373 Country Rock
          346.647059 Roots Reggae
             339.16 Spoken Word
             337.75 Oldies

                    ...

            227.9375 Drum n Bass
          217.083333 Bluegrass
          213.578947 Rockabilly
          208.454545 Teen Pop
          203.666667 Childrens

119 rows selected.
```

Sichten

Unser erster View liefert eine nützliche Sicht auf die gespeicherten Tonträger. Sie fasst verwandte Informationen aus verschiedenen anderen Relationen mit der Tontraeger-Relation zusammen. Dazu gehören der Name des Künstlers, die Anzahl der Titel, die Gesamtspielzeit und die durchschnittliche Bewertung des Tonträgers durch die User.

```
SQL> CREATE VIEW Tontraeger_uebersicht (titel, musiker_name, plattenfirma,
  2     durchschnittsnote, track_anzahl, track_gesamtlaenge) AS
  3  SELECT t.titel, k.name, t.plattenfirma,
  4     AVG(b.note), COUNT(e.sid), SUM(e.laenge)
  5  FROM Tontraeger t, Enthaelt e, Band_Kuenstlername k, Bewertet b
  6  WHERE t.tid = e.tid
  7     AND t.mid = k.mid
  8     AND t.tid = b.tid
  9  GROUP BY t.titel, t.plattenfirma, k.name;

View created.

SQL> SELECT * FROM Tontraeger_Uebersicht;

TITEL              MUSIKER_NAME    PLATTENFIRMA         DURCHSCHNITTSNOTE TRACK_ANZAHL TRACK_GESAMTLAENGE
------------------ --------------- -------------------- ----------------- ------------ ------------------
acoustic live      nils lofgren    Time Bomb                          8.5           34               8840
am piano i         rio reiser      Hopeless Records                   2.5           36               5302
am piano ii        rio reiser      Skin Graft                         7.5           42               7036
autobahn           kraftwerk       Fat Possum                           4            5               2560
                                    .
                                    .
                                    .
vienna concert     keith jarrett   American Recordings                  9            2               4079
wander this world  jonny lang      Supersonic                 5.66666667           36               9495
wanja's choice     butlers         Bad Taste Records                  5.5           30               7272
who can you trust  morcheeba       BluNoise                             6           12               3345

77 rows selected.
```

Unser zweiter View ist eine alternative Sicht auf die Users-Relation. Dabei wurde absichtlich das Passwort-Attribut nicht in die Relation aufgenommen. Die Passwörter sind natürlich streng vertraulich und dürfen nicht in die Hände Unbefugter gelangen. Mit diesem View kann man anderen Personen trotzdem Zugriff auf die anderen Daten der Users-Relation gestatten.

```
SQL> CREATE VIEW User_Uebersicht (username, email, geburtstag, geschlecht, herkunft, besitzt_anzahl, wuenschtsich_anzahl, bewertet_anzahl) AS
  2  SELECT u.name, u.email, u. geburtstag, u.geschlecht, u.herkunft, COUNT(b1.tid),
COUNT(b2.tid), COUNT(w.tid)
  3  FROM Users u
  4     LEFT OUTER JOIN Besitzt b1 ON u.name = b1.name
  5     LEFT OUTER JOIN Bewertet b2 ON u.name = b2.name
  6     LEFT OUTER JOIN WuenschtSich w ON u.name = w.name
  7  GROUP BY u.name, u.email, u. geburtstag, u.geschlecht, u.herkunft;

View created.

SQL> SELECT * FROM User_Uebersicht;

USERNAME               EMAIL                     GEBURTSTA G HERKUNFT        BESITZT_ANZAHL WUENSCHTSICH_ANZAHL BEWERTET_ANZAHL
---------------------- ------------------------- --------- - --------------- -------------- ------------------- ---------------
Adinda5hv              Adinda5hv@k7xm3j4.b2      27-SEP-80 M Deutschland                1                   0               0
Adindaeufo2c           Adindaeufo2c@z4.8t        19-MAR-47 M Kroatien                   1                   0               1
Adrienne3hm            Adrienne3hm@r7.lg         15-JAN-51 W Weissrussland              0                   0               1
Adriennev27v           Adriennev27v@a.zs         01-FEB-46 M Estland                    0                   0               0
                                                           .
                                                           .
                                                           .
Yasminecc6tl6          Yasminecc6tl6@5s.or       23-JAN-44 W Griechenland               0                   0               0
Yves                   Yves@qvce.yx              21-JAN-78 W Italien                    0                   1               0
Yveslemm               Yveslemm@8jsi1hu.2e       28-OCT-97 W Tunesien                   0                   0               0
Yvesnp                 Yvesnp@1.c1               12-OCT-81 M Zypern                     1                   0               0

549 rows selected.
```