2015

JDBC-Rueckwaertssalto

INSY-TGM
THOMAS STEDRONSKY

Inhalt

Angabe JDBC-Rueckwaertssalto

Erstelle ein Java-Programm, das Connection-Parameter und einen Datenbanknamen auf der Kommandozeile entgegennimmt und die Struktur der Datenbank als EER-Diagramm und Relationen Modell ausgibt (in Dateien geeigneten Formats, also z.B. PNG für das EER und TXT für das RM)

Verwende dazu u.A. das ResultSetMetaData-Interface, das Methoden zur Bestimmung von Metadaten zur Verfügung stellt.

Zum Zeichnen des EER-Diagramms kann eine beliebige Technik eingesetzt werden für die Java-Bibliotheken zur Verfügung stehen: Swing, HTML5, eine WebAPI, ... Externe Programme dürfen nur soweit verwendet werden, als sich diese plattformunabhängig auf gleiche Weise ohne Aufwand (sowohl technisch als auch lizenzrechtlich!) einfach nutzen lassen. (also z.B. ein Visio-File generieren ist nicht ok, SVG ist ok, da für alle Plattformen geeignete Werkzeuge zur Verfügung stehen)

Recherchiere dafür im Internet nach geeigneten Werkzeugen.

Die Extraktion der Metadaten aus der DB muss mit Java und JDBC erfolgen.

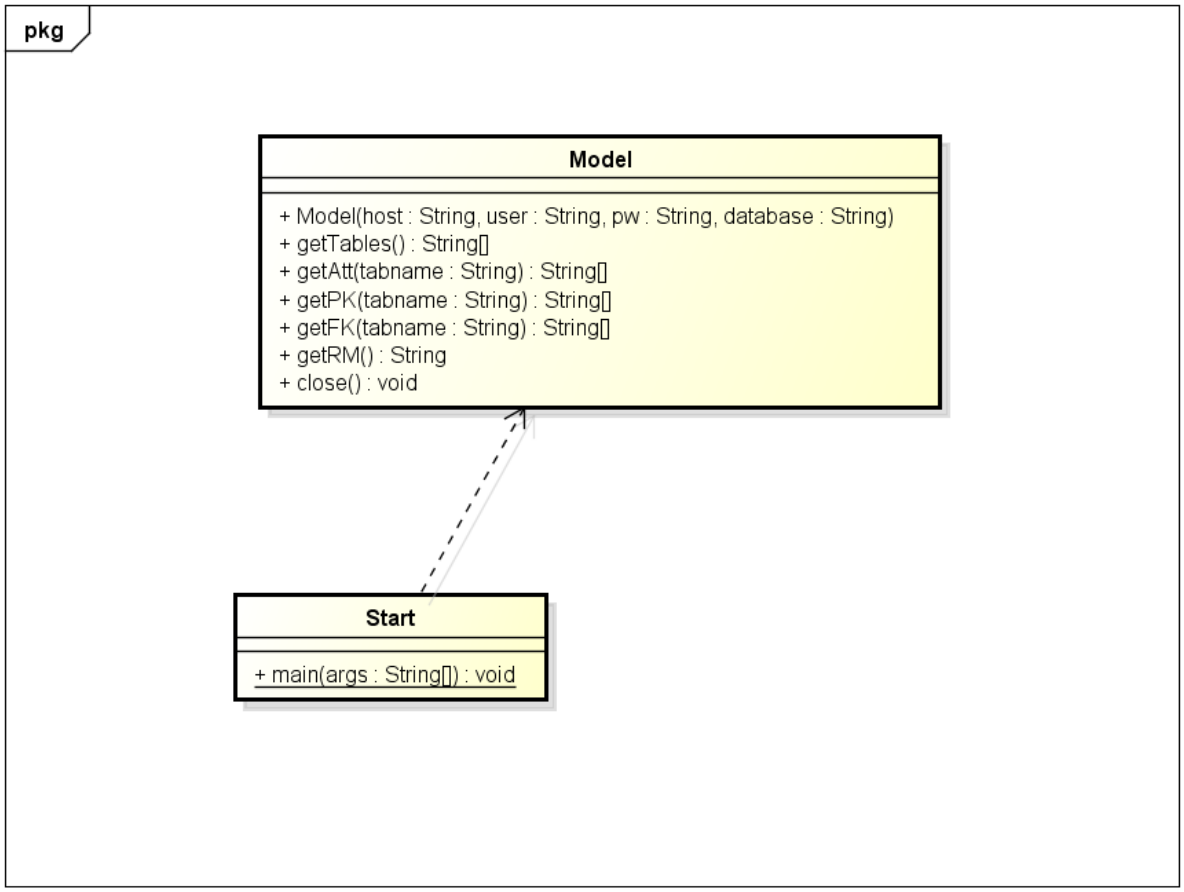
Im EER müssen zumindest vorhanden sein:

- korrekte Syntax nach Chen, MinMax oder IDEFIX
- alle Tabellen der Datenbank als Entitäten
- alle Datenfelder der Tabellen als Attribute
- Primärschlüssel der Datenbanken entsprechend gekennzeichnet
- Beziehungen zwischen den Tabellen inklusive Kardinalitäten soweit durch Fremdschlüssel nachvollziehbar. Sind mehrere Interpretationen möglich, so ist nur ein (beliebiger) Fall umzusetzen: 1:n, 1:n schwach, 1:1
- Kardinalitäten

Fortgeschritten (auch einzelne Punkte davon für Bonuspunkte umsetzbar)

- Zusatzattribute wie UNIQUE oder NOT NULL werden beim Attributnamen dazugeschrieben, sofern diese nicht schon durch eine andere Darstellung ableitbar sind (1:1 resultiert ja in einem UNIQUE)
- optimierte Beziehungen z.B. zwei schwache Beziehungen zu einer m:n zusammenfassen (ev. mit Attributen)
- Erkennung von Sub/Supertyp-Beziehungen

Design



In Arbeit!

Probleme

- In dem test Skript wird
  country CHAR(2) NOT NULL default '' REFERENCES countries(code),
  verwendet allerdings lassen sich die REFERENCES mit MetaDaten nicht auslesen.

Die Meta Daten wurden mit DatabaseMetaData ausgelesen. Hiermit lassen sich die Foreign Keys und deren Herkunft auslesen.

ERD

Das Toll GraphViz wurde bereits gefunden und ist momentan die Gewählte Lösung Änderungen vorbehalten.

http://www.graphviz.org/

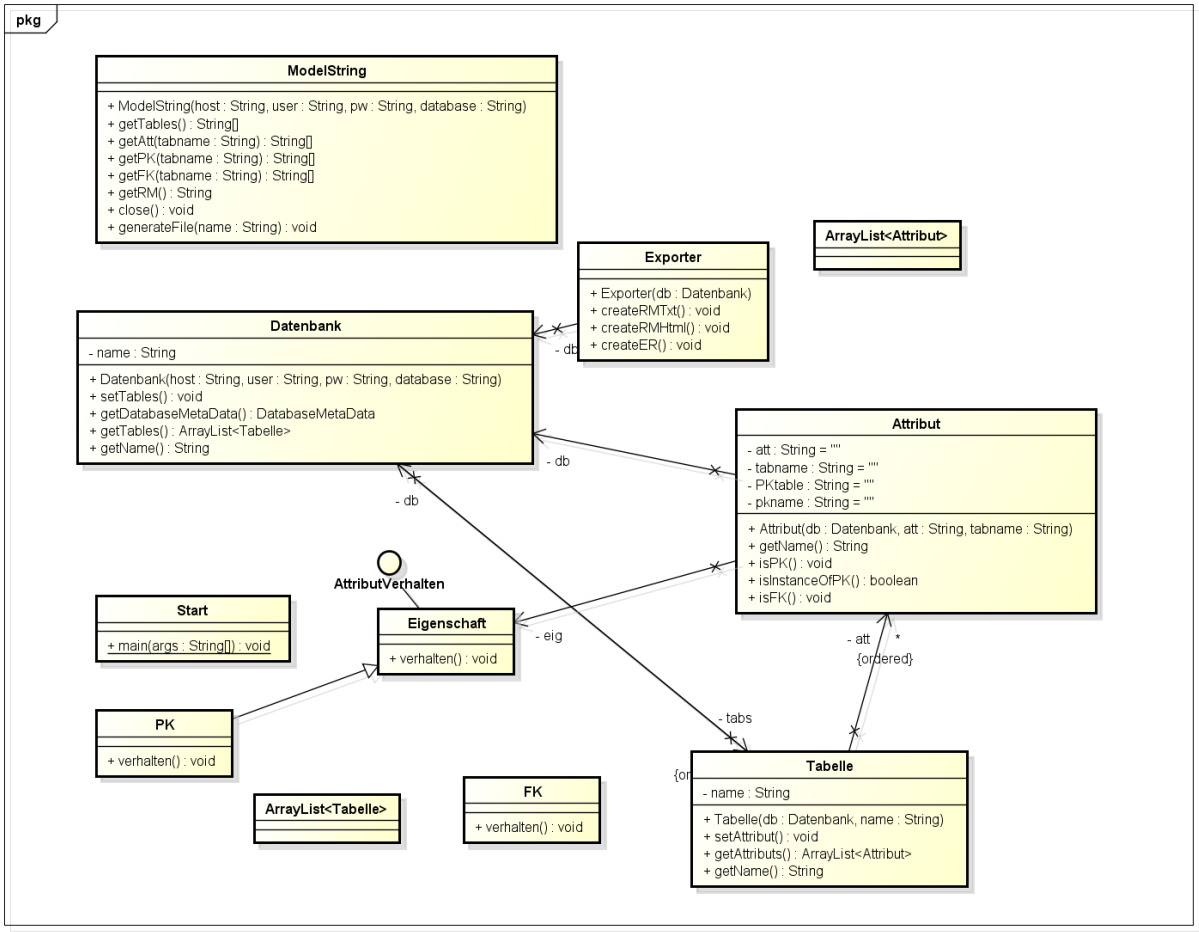
Download Link für das Tool

http://www.graphviz.org/Download_windows.php

Um das Programm zu starten muss es in die Umgebungsvariable eingetragen werden um es in der Konsole zu starten.

Design 2.0

Das Design wurde überarbeitet. Es wird mittels Objekten gearbeitet um offen für Erweiterungen zu sein.



In Arbeit

RM

Das RM wurde anfangs mittels String gelöst aber dann wurde eine Designänderung vorgenommen und dann mittels Objekten realisiert.

Die 1. Hürde die zur bewältigen war ist die PK und FK zu kennzeichnen.

airlines(id, name, country)
airports(airportcode, name, country, city)
countries(code, name)
flights(airline, flightnr, departure_time, departure_airport, destination_time, destination_airport, planetype)
freightplanes(id, maxcargo)
passengerplanes(id, maxseats, seatsperrow)
passengers(id, firstname, lastname, airline, flightnr)
planefleet(airline, plane, nr, bought, price)
planes(id, manufacturer, type, lengthoverall, span, maxspeed)
tickets(id, passenger, issued, rownr, seatposition, specialmenu)

Im Programm gibt es 2 Varianten sich das RM anzeigen zu lassen.

- Im .txt Format
- Im .html Format

Vorteil des HTML Formats: Es können PK unterstrichen werden und sind somit besser ersichtlich.

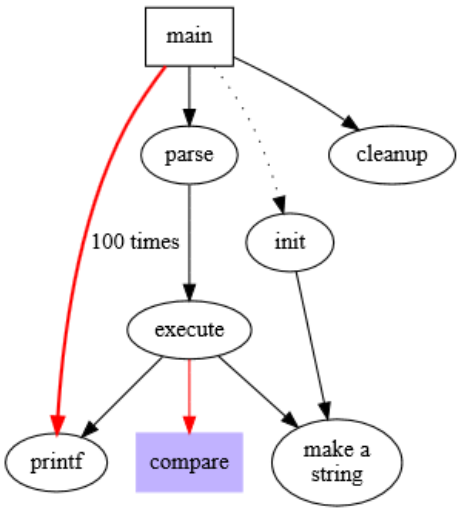
ERD

Das ERD wurde mittels GraphViz im .dot Format gelöst.

Hierzu: http://www.graphviz.org/pdf/dotguide.pdf

```
1:  digraph G {
2:       size ="4,4";
3:       main [shape=box];   /* this is a comment */
4:       main -> parse [weight=8];
5:       parse -> execute;
6:       main -> init [style=dotted];
7:       main -> cleanup;
8:       execute -> { make_string; printf}
9:       init -> make_string;
10:      edge [color=red];   // so is this
11:      main -> printf [style=bold,label="100 times"];
12:      make_string [label="make a\nstring"];
13:      node [shape=box,style=filled,color=".7 .3 1.0"];
14:      execute -> compare;
15: }
```

Hier wird der Grundlegende Aufbau eines dot-Files gezeigt. Wobei ein -> im File auch ein -> in der Grafik bedeutet.



Hier sieht man das Ergebnis des oben aufgezeigten Example.

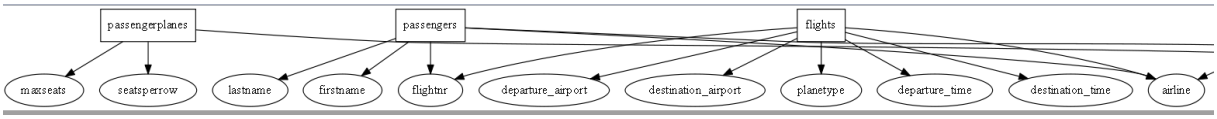
Anfänge des ERD

Es wurden anfangs nur Objekttypen und Attribute gezeichnet.

```
digraph G {
size="40,40"
airlines [shape=box];
airlines -> id;
airlines -> name;
airlines -> country;
airports [shape=box];
airports -> airportcode;
airports -> name;
airports -> country;
airports -> city;
countries [shape=box];
countries -> code;
countries -> name;
flights [shape=box];
flights -> airline;
flights -> flightnr;
flights -> departure_time;
flights -> departure_airport;
flights -> destination_time;
flights -> destination_airport;
flights -> planetype;
freightplanes [shape=box];
freightplanes -> id;
freightplanes -> maxcargo;
passengerplanes [shape=box];
passengerplanes -> id;
passengerplanes -> maxseats;
passengerplanes -> seatsperrow;
passengers [shape=box];
passengers -> id;
passengers -> firstname;
passengers -> lastname;
passengers -> airline;
passengers -> flightnr;
planefleet [shape=box];
planefleet -> airline;
planefleet -> plane;
planefleet -> nr;
planefleet -> bought;
planefleet -> price;
planes [shape=box];
planes -> id;
planes -> manufacturer;
planes -> type;
planes -> lengthoverall;
planes -> span;
planes -> maxspeed;
tickets [shape=box];
tickets -> id;
tickets -> passenger;
tickets -> issued;
tickets -> rownr;
tickets -> seatposition;
tickets -> specialmenu;
}rdsortiment -> stand;
}aschine [shape=box];
maschine -> mnr;
maschine -> mbez;
mitarbeiter [shape=box];
mitarbeiter -> einstdatum;
mitarbeiter -> pnr;
mitarbeiter -> kuend;
mitarbeiter -> pnr:person.pnr;
mitarbeiter -> kuend:kuendigung.kdatum;
```

So sieht das .dot File aus.

Hier sieht man das Wort vor dem -> ist hierbei der Objekttyp und das Wort danach das Attribut.



Ausschnitt des ERDs zu dem .dot File.

Hier werden wie bekannt in einen ERD die Objekttypen als Rechteck und die Attribute mittels Ellipse dargestellt. Der Pfeil zeigt hierbei immer auf die Attribute die der Objekttyp verwendet.

Problem: Objekttypen zeigen manchmal auf dasselbe Attribut(weil gleich Name)

Zeitaufzeichnung

Dauer in Std.	Leistung	Datum	Arbeitsstunden
1,00	DatabaseMetaData	19.01.2015	15
2,00	PK und FK	20.01.2015	Stedronsky
2,00	Design Neuüberlegungen	28.01.2015	
2,00	Design Vorlage	07.02.2015	
3,00	Desgin Umsetzung und RM	09.02.2015	
1,00	Fehler überarbeitung FK	10.02.2015	
4,00	ERD	17.02.2015	

GitHub Graphs