|  |
| --- |
| INSY |
| Rückwärtssalto |
| 4AHIT |

|  |
| --- |
| FOCK & WEINBERGER  07.01.2015 |

Inhalt

[Aufgabenstellung 2](#_Toc413782981)

[JDBC: Rückwärtssalto 2](#_Toc413782982)

[GitHub-Repolink 3](#_Toc413782983)

[Requirementsanalyse 4](#_Toc413782984)

[Design 5](#_Toc413782985)

[Umsetzung 6](#_Toc413782986)

[ Parameter entgegennehmen 6](#_Toc413782987)

[ Verbindung zur Datenbank herstellen 6](#_Toc413782988)

[ Metadaten aus der Datenbank auslesen 6](#_Toc413782989)

[ Metadaten aus der Datenbank abspeichern 6](#_Toc413782990)

[ Output 6](#_Toc413782991)

[Ausführung des Programmes 7](#_Toc413782992)

[Things we have learned 8](#_Toc413782993)

[Quellen 9](#_Toc413782994)

# Aufgabenstellung

## JDBC: Rückwärtssalto

Erstelle ein Java-Programm, dass Connection-Parameter und einen Datenbanknamen auf der Kommandozeile entgegennimmt und die Struktur der Datenbank als EER-Diagramm und Relationenmodell 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

# GitHub-Repolink

**https://github.com/hfock-tgm/Rueckwaertssalto.git**

# Requirementsanalyse + Zeitaufzeichnung

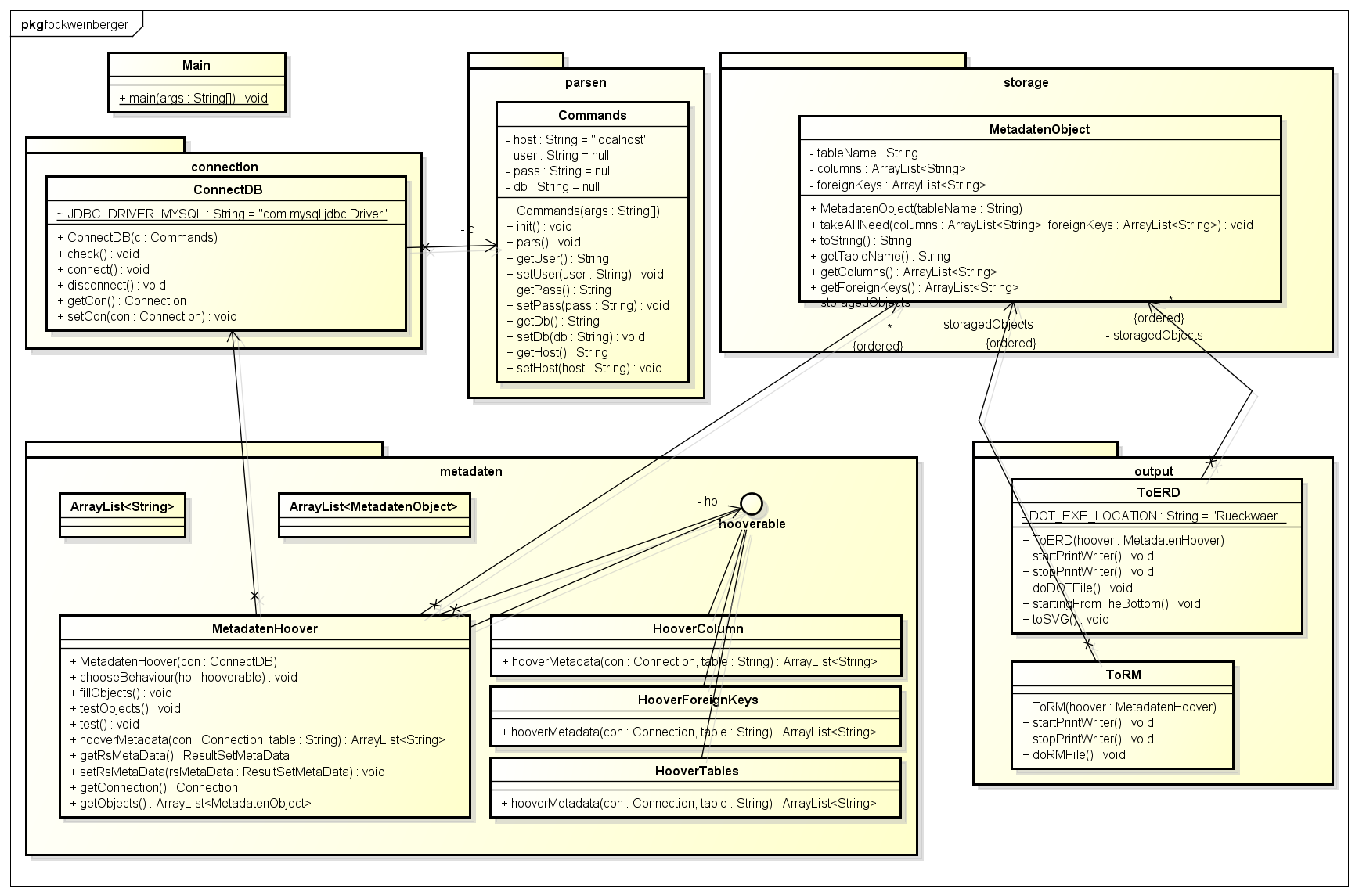
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Funktionale Anforderungen** | | | | | |
|  | **Name** | **Umgesetzt** | **Getestet** | **Gesch. Zeit** | **Tats. Zeit** |
| **CLI Argumente** | Weinberger | x | x | 30 min | 20 min |
| **Verbindung mit der DB** | Weiberger | x | - | 20 min | 20 min |
| **Verarbeitung der Metadaten** | Fock&Wein. | x | - | 60 min | 120 min |
| **Abspeicherung der Metadaten** | Fock | x | - | 20 min | 30 min |
| **Generieren des RMs** | Weinberger | x | - | 40 min | 40 min |
| **Generieren des ERDs** | Fock | x | - | 90 min | 150 min |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

# 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Organisatorische Anforderungen** | | | | | |
|  | **Name** | **Umgesetzt** | **Getestet** | **Gesch. Zeit** | **Tats. Zeit** |
| **ERD-Tool recherchiert** | Weinberger | x | - | 60 min | 70 min |
| **DOT-file** | Fock | x | - | 40 min | 60 min |
| **Dokumentation** | Fock & Wein. | X | - | 90 min | 90 min |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
| **Name** | **Gesch. Zeit** | **Gesch. Zeit** |
| **Fock** | 300 min | 450 min |
| **Weinberger** | 300 min | 360 min |

# Design



# Umsetzung

## Parameter entgegennehmen

* + Mithilfe der Apache CLI haben wir die Entgegennahme der Argumente gelöst.

**this**.**options**.addOption(OptionBuilder  
 .*withLongOpt*(**"host"**)  
 .*withDescription*(  
 **"-h ... Hostname des DBMS. Standard: localhost\n"**)  
 .*withValueSeparator*(**' '**).*hasArg*().*create*(**"h"**));

**if** (**cmd**.hasOption(**"h"**) || **cmd**.hasOption(**"host"**)) {  
 **this**.setHost(**cmd**.getOptionValue(**"h"**));

**public** String getHost() {  
 **return host**;  
}

## Verbindung zur Datenbank herstellen

* + Die Verbindung mit einer Datenbank wurde mit der JDBC gelöst.

*// JDBC driver name***static final** String ***JDBC\_DRIVER\_MYSQL*** = **"com.mysql.jdbc.Driver"**;

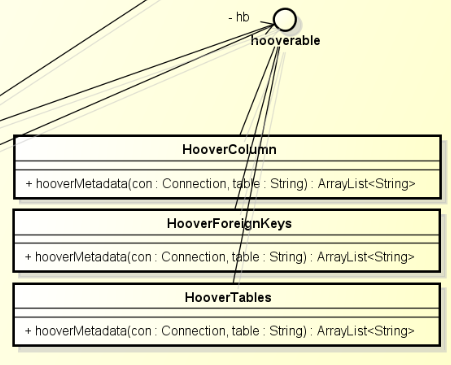
*// Register JDBC driver*Class.*forName*(***JDBC\_DRIVER\_MYSQL***);

*// Open a connection*System.***out***.println(**"Connecting to database..."**);  
setCon(DriverManager.*getConnection*(**"jdbc:mysql://"** + **c**.getHost()  
 + **"/"** + **c**.getDb(), **c**.getUser(), **c**.getPass()));

System.***out***.println(**"Disconnecting from database..."**);  
**if** (!(**con** == **null**))  
 **con**.close();

## Metadaten aus der Datenbank auslesen

* + Die Metadaten wurden mit der DatabaseMetData Klasse ausgelesen.
  + Und als Pattern wurde das Strategy Pattern verwendet.



* + - Tablenames

DatabaseMetaData md = con.getMetaData();

ResultSet rs = md.getTables(**null**, **null**, **"%"**, **null**);  
**while** (rs.next()) {  
 result.add(rs.getString(3));  
}

* + - Columns

Statement st = con.createStatement();  
 ResultSet rs = st.executeQuery(**"SELECT \* FROM "** + table);  
 ResultSetMetaData rsMetaData = rs.getMetaData();  
*// Hier wird der Column count herausgelesen* **int** numberOfColumns = rsMetaData.getColumnCount();

**for** (**int** i = 1; i < numberOfColumns; i++) {  
 String columnName = rsMetaData.getColumnName(i);

}

* + - ForeignKeys

DatabaseMetaData meta = con.getMetaData();

ResultSet rs = meta.getExportedKeys(con.getCatalog(), **null**, table);  
**while** (rs.next()) {  
 String fkTableName = rs.getString(**"FKTABLE\_NAME"**);

}

## Metadaten aus der Datenbank abspeichern

* + All die ausgelesenen Metadaten werden in der MetadatenObject Klasse mithilfe von ArraysList<String> abgespeichert.
  + Jedes MetadatenObject Objekt repräsentiert eine Tabelle.

## Output

### Relationen Modell

* + - Primary Keys werden mit \_PK gekennzeichnet
    - NotNull wird mit \_NN gekennzeichnet
    - Der Tabellename wird zu jedem dazugehoerigen Attribut hinzugeschrieben

airlines(airlines\_id\_PK\_NN, airlines\_name\_NN, planefleet\_airline)

airports(airports\_airportcode\_PK\_NN, airports\_name, airports\_country\_NN, flights\_departure\_airport, flights\_destination\_airport)

countries(countries\_code\_PK\_NN, airlines\_country, airports\_country)

flights(flights\_airline\_PK\_NN, flights\_flightnr\_PK\_NN, flights\_departure\_time\_NN, flights\_departure\_airport, flights\_destination\_time\_NN, flights\_destination\_airport, passengers\_airline, passengers\_flightnr)

freightplanes(freightplanes\_id\_PK\_NN)

passengerplanes(passengerplanes\_id\_PK\_NN, passengerplanes\_maxseats)

passengers(passengers\_id\_PK\_NN, passengers\_firstname, passengers\_lastname, passengers\_airline, tickets\_passenger)

planefleet(planefleet\_airline\_PK\_NN, planefleet\_plane\_PK\_NN, planefleet\_nr\_PK\_NN, planefleet\_bought, flights\_airline, flights\_planetype)

planes(planes\_id\_PK\_NN, planes\_manufacturer\_NN, planes\_type\_NN, planes\_lengthoverall, planes\_span, freightplanes\_id, passengerplanes\_id, planefleet\_plane)

tickets(tickets\_id\_PK\_NN, tickets\_passenger\_NN, tickets\_issued\_NN, tickets\_rownr, tickets\_seatposition)

### Entity-Relationship-Modell

* + Um das ERD zu generieren wurde mit Graphviz[6] gearbeitet.
    - Es wird mit mehreren for-Schleifen alle MetadatenObjecte durchiteriert und dann mit den erhaltenen Strings ein DOT-File erstellt.

graph ERD {

airlines [shape=box];

airlines -- airlines\_id\_PK\_NN

airlines -- airlines\_name\_NN

airlines -- planefleet\_airline0

airports [shape=box];

airports -- airports\_airportcode\_PK\_NN

airports -- airports\_name

airports -- airports\_country\_NN

airports -- flights\_departure\_airport1

airports -- flights\_destination\_airport1

}

* + - Das DOT-File kann dann mithilfe von Graphviz in eine Grafik generiert werden.
      * dot -Tsvg ERD.dot -o ERD.svg
    - Leider konnten nicht alle Kardinalitäten eingezeichnet werden und deswegen wurden sie weggelassen.

# Ausführung des Programmes

Um das Programm zu starten werden folgende Argumente benötigt.

**-h || --host ... Hostname des DBMS. Standard: localhost**

**-u || --user ...Benutzername. Standard: Benutzername des im Betriebssystem angemeldeten Benutzers**

**-p || --password ... Passwort. Alternativ kann ein Passwortprompt angezeigt werden.**

**-d || --database ... Name der Datenbank**

Durch die Ausführung des Programmes werden im selben Verzeichnis fünf Dateien generiert.

* Relationenmodell
  + rm.txt
* ER-Diagramm
  + ERD.dot
  + ColorERD.dot
  + ERD.svg
  + ColorERD.svg

Um aus dem Dot-File eine Grafik zu machen muss Graphviz installiert sein. Anderfalls wird kein ERD.svg aus dem ERD.dot generiert.

Graphviz ist eine kostenlose Open-Source-Software, welche auf jeder gängigen Plattform installiert werden kann.

Graphiz kann von der offiziellen Seite unter folgendem Link heruntergeladen werden:

* http://www.graphviz.org/Download..php

# Things we have learned

* JDBC [3]
* Graphviz [6]

# Quellen

* [1] Apache CLI;
  + Online: <http://commons.apache.org/proper/commons-cli/download_cli.cgi>
    - Zuletzt besucht am 07.01.2015
* [2] Github Repo vom ChangeVision;
  + Online: <https://github.com/ChangeVision/astah-db-reverse-plugin/blob/master/src/main/java/com/change_vision/astah/extension/plugin/dbreverse/reverser/finder/DatatypeFinder.java>
    - Zuletzt besucht am 07.01.2015
* [3] JDBC;
  + Online: <http://www.oracle.com/technetwork/java/javase/jdbc/index.html>
    - Zuletzt besucht am 14.01.2015
* [4] SchemaCrawler;
  + Online: <http://schemacrawler.sourceforge.net/>
    - Zuletzt besucht am 14.01.2015
* [5] ResultSetMetaData-Interface;
  + Online: <http://docs.oracle.com/javase/7/docs/api/java/sql/ResultSetMetaData.html>
    - Zuletzt besucht am 28.01.2015
* [6] Graphviz;
  + Online: <http://www.graphviz.org/>
    - Zuletzt besucht am 23.02.2015