

Rückwärtssalto

Lukas Zainzinger, Gerald Blühberger

Protokoll: AU05 - Rückwärtssalto

Inhaltsverzeichnis

1. AUFGABENSTELLUNG:	3
2. ARBEITSAUFTEILUNG	3
3. GESCHÄTZTER AUFWAND/TATSÄCHLICHER AUFWAND	4
4. ZEITAUFGZEICHNUNG	4
4.1. ZEITAUFGZEICHNUNG ZAINZINGER:	4
4.2. ZEITAUFGZEICHNUNG BLÜHBERGER:	4
5. DESIGN	4
5.1. UML	4
6. VERWENDETE TOOLS	5
6.1. GRAPHVIZ – NEATO	5
6.1.1. <i>Installation:</i>	5
6.1.2. <i>Dot-File:</i>	5
6.1.3. <i>Ausführen/Zeichnen:</i>	5
7. TEST	5
8. QUELLEN	6

Rückwärtssalto

1. Aufgabenstellung:

Aus einer Datenbank ein RM und EER generieren.

Anzeige als RM:

- Tabellennamen
- Attributnamen
- (Kardinalität anzeigen)

Anzeige als EER:

- korrekte Syntax nach Chen
- 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

Git-Repository: <https://github.com/bluehbergerSYT1/rueckwertssalto>

2. Arbeitsaufteilung

Arbeit	Bearbeitet von	Fertig	Gesamtstunden
RM:			
Tabellennamen	LZ	+	3h
Attributnamen	LZ	+	4h (parallel mit TabNamen)
Kardinalität anzeigen	LZ, GB	~	3h, 4h
EER			
korrekte Syntax nach Chen	LZ, GB	-	
alle Tabellen der Datenbank als Entitäten	LZ	+	2,5h
alle Datenfelder der Tabellen als Attribute	LZ	+	3h
PK/FK der Datenbanken entsprechend gekennzeichnet	LZ	+	1,5h
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	GB	-	
Kardinalitäten	GB	-	

3. Geschätzter Aufwand/Tatsächlicher Aufwand

RM: 8h / 11h

ER: 10h / 7

4. Zeitaufzeichnung

4.1. Zeitaufzeichnung Zainzinger:

07.01.2015: 4h Verbindungs Aufbau, Tables Klasse
 10.01.2015: 3h RM generieren
 14.01.2015: 3h RM verbessert, RM wird nun Richtig angezeigt, Anzeige von PK und FK noch nicht möglich..
 15.02.2015: 1h RM verbessert, PK und FK werden nun richtig angezeigt, allerdings ohne verknüpfter Tabelle
 16.02.2015: 3h Installation Graphviz + Recherche + Beginn der Zeichnen Klasse
 17.02.2015: 6h Zeichnen Klasse: Methode makeDotFile erweitert
 Dokumentation Code: 0,5h
 Dokumentation Protokoll: 1h
 UML: 0,5h

Gesamt: 22h

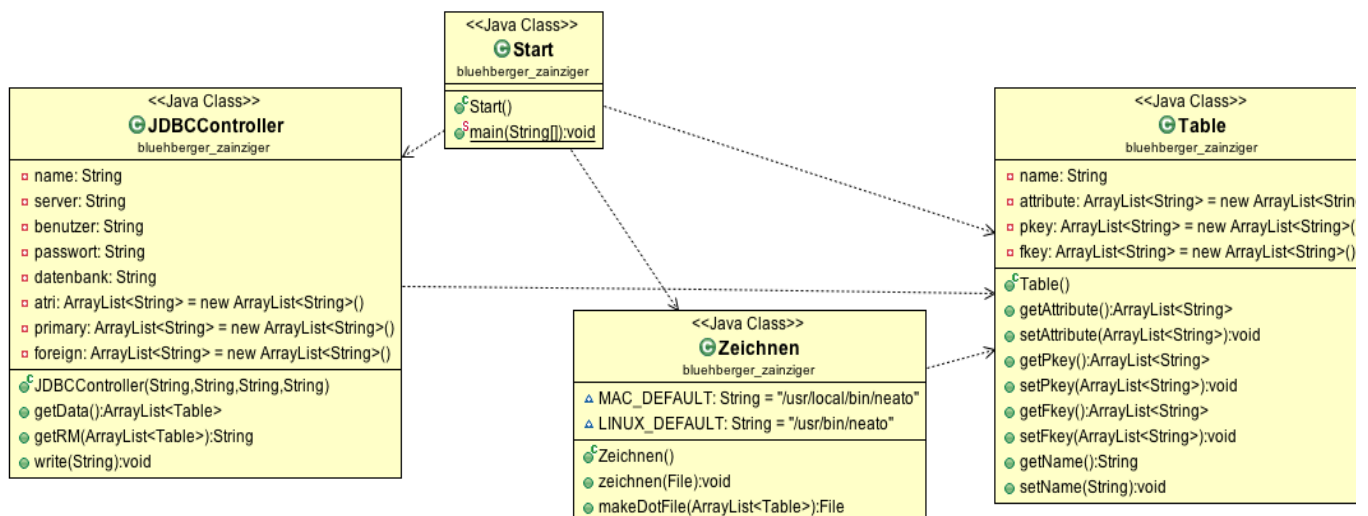
4.2. Zeitaufzeichnung Blühberger:

07.01.2015: 1h GIT Repository
 28.01.2015: 4h RM debugging

Gesamt: 5h

5. Design

5.1. UML



6. Verwendete Tools

6.1. Graphviz – Neato

6.1.1. Installation:

Installation über den Installer zu finden auf: <http://www.graphviz.org/Download.php>

Danach den Installer ausführen.

6.1.2. Dot-File:

Um ein Diagramm zu Zeichnen, muss ein .dot File übergeben werden. Dieses File enthält alle Informationen über den zu zeichnenden Inhalt.

Die Struktur sieht folgender Maßen aus:

```
diagram Diagrammname{
    node[shape=box, style=filled]; //Die Entitäten sind befüllte Boxen
    A -- B [style=bold]; //Dicker Strich bei Verbindung
    B -- C; //Normale Verbindung
    ....
}
```

6.1.3. Ausführen/Zeichnen:

Ausgeführt wird das .dot File mithilfe des Programmes neato.

Aufruf durch: /usr/local/bin/neato -Tpng tables.dot -o EER.png

Der Pfad gibt an, wo das Programm installiert ist.

-Tpng gibt an, dass das Diagramm als PNG ausgegeben werden soll

-o name.png gibt den Namen des Files an

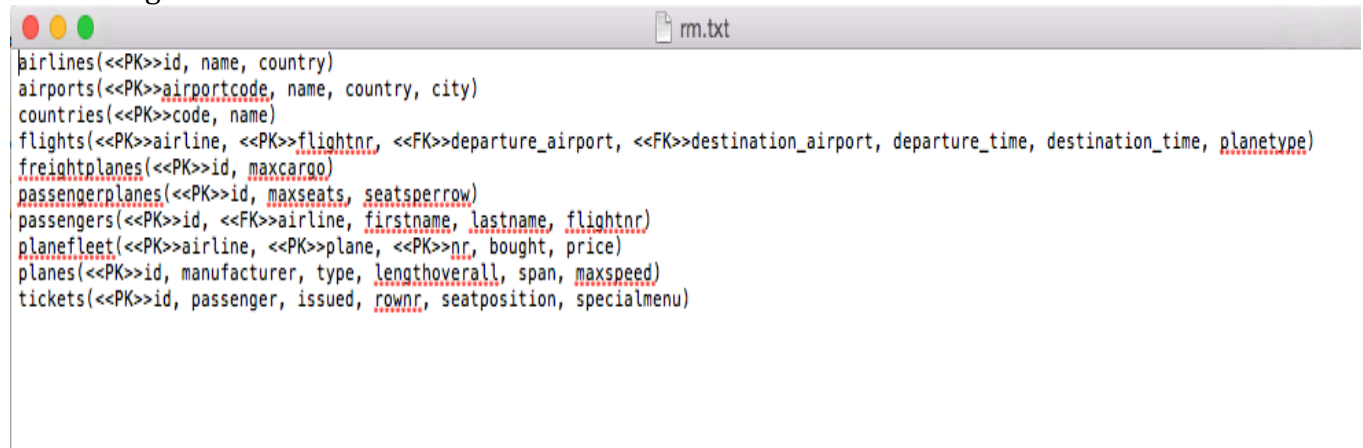
tables.dot steht für das Verwendete .dot File zur Erstellung des Diagramms

Dokumentation: <http://www.graphviz.org/Documentation.php>

7. Test

Zum Testen wird die Datenbank „agentspiel“ verwendet.

Das erzeugte RM:

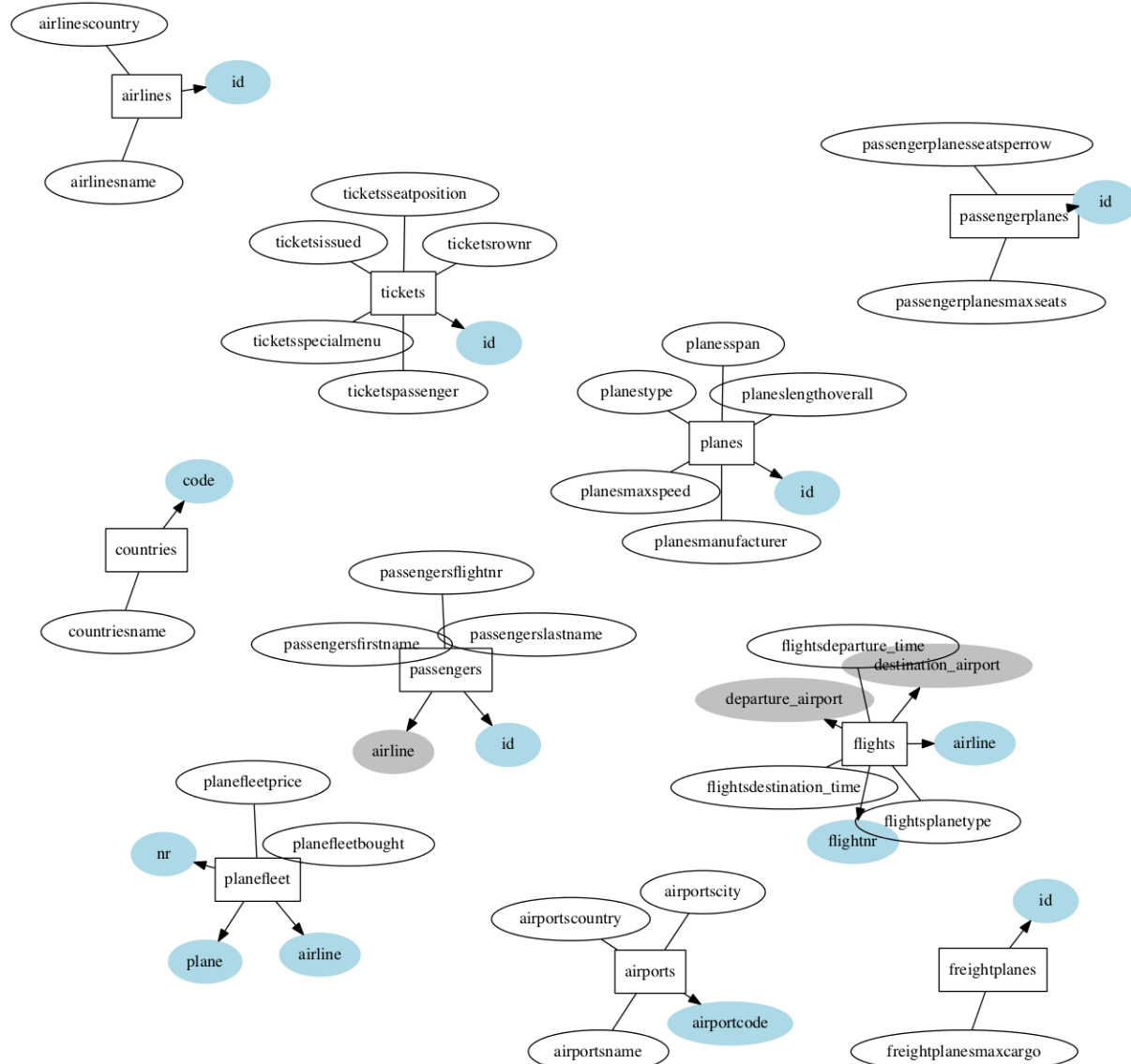


```
airlines(<<PK>>id, name, country)
airports(<<PK>>airportcode, name, country, city)
countries(<<PK>>code, name)
flights(<<PK>>airline, <<PK>>flightnr, <<FK>>departure_airport, <<FK>>destination_airport, departure_time, destination_time, planetype)
freightplanes(<<PK>>id, maxcargo)
passengerplanes(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

Primary Keys werden mit <<PK>> vorm Attributnamen gekennzeichnet.

Foreign Keys werden mit <<FK>> vorm Attributname gekennzeichnet.

Das erzeugte ERD:



Primary Keys werden Hellblau gekennzeichnet und Foreign Keys Grau.
 Alle Keys werden durch einen Pfeil statt einer normalen Verbindung gekennzeichnet.
 Die Verbindungen zwischen den Tabellen werden nicht realisiert.

8. Quellen

<http://docs.oracle.com/>

<http://www.java2s.com/Code/Java/Database-SQL-JDBC/CatalogDatabase-SQL-JDBC.htm>

<http://www.graphviz.org/About.php>

<http://www.graphviz.org/Documentation.php>