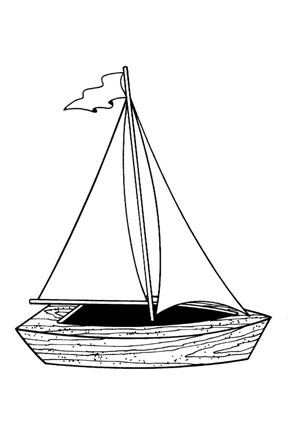
20.3.2015

Andreas Ernhofer

4BHIT

Segelverein

Protokoll zur Aufgabe „Segelverein“



Inhaltsverzeichnis

[Aufgabenstellung 2](#_Toc414578406)

[1. Beschreibung 2](#_Toc414578407)

[2. ER – Diagramm 3](#_Toc414578408)

[3. Das Relationenmodell 3](#_Toc414578409)

[4. Die SQL-Anfragen 4](#_Toc414578410)

[5. Java und JDBC 4](#_Toc414578411)

[6. Abgabe 5](#_Toc414578412)

[Aufwandsabschätzung & Zeitaufzeichnung 6](#_Toc414578413)

[Arbeitsdurchführung 7](#_Toc414578414)

[1. Erzeugen des Benutzers und der Datenbank 7](#_Toc414578415)

[2. Erstellen eines *git* Repositorys 7](#_Toc414578416)

[3. Erstellen einer *create.sql* Datei 7](#_Toc414578417)

[4. Erstellen eines *Generators* 7](#_Toc414578418)

[5. Erstellen von zwei .sh Scripts 8](#_Toc414578419)

[5.1. start.sh 8](#_Toc414578420)

[5.2. login.sh 8](#_Toc414578421)

[6. Java und JDBC 8](#_Toc414578422)

[7. UML-Erzeugen 8](#_Toc414578423)

[8. Git log 8](#_Toc414578424)

[SQL-Anfragen 9](#_Toc414578425)

[Quellen 11](#_Toc414578426)

# Aufgabenstellung

Sie erhalten den Auftrag für den Europäischen Dachverband der Segelvereine eine Datenbank zu implementieren.

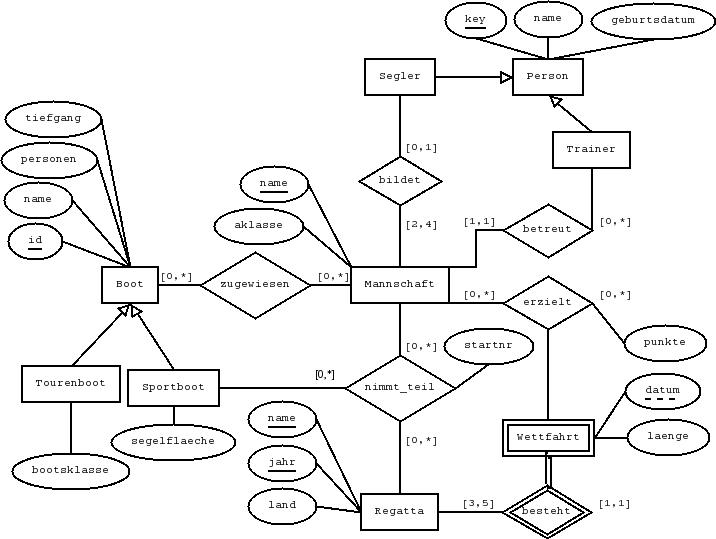
## Beschreibung

Für Segler und Trainer sind Name (NAME) und Geburtsdatum (GEBURTSDATUM) bekannt. Sie werden beide identifiziert durch eine eindeutige Nummer (KEY). Mindestens zwei Segler, maximal jedoch vier Segler bilden eine Mannschaft. Für jede Mannschaft wird ein eindeutiger Name (NAME) und eine Altersklasse (AKLASSE) gespeichert. Jede Mannschaft wird genau von einem Trainer betreut. Ein Trainer kann jedoch mehrere Mannschaften betreuen.

Jeder Mannschaft sind Boote zugewiesen. Ein Boot kann mehreren Mannschaften zugewiesen sein. Ein Boot wird eindeutig durch eine Nummer (ID) identifiziert. Weiters sind zu jedem Boot ein Name (NAME), die Anzahl der Personen (PERSONEN) und der Tiefgang (TIEFGANG) bekannt. Es gibt Tourenboote und Sportboote. Tourenboote haben zusätzlich eine Bootsklasse (BOOTSKLASSE) und Sportboote haben zusätzlich eine Segelfläche (SEGELFLAECHE) gespeichert. Es ist außerdem bekannt welche Mannschaften mit welchen Sportbooten an welchen Regatten mit welcher Startnummer (STARTNR) teilgenommen haben.

Eine Regatta wird eindeutig identifiziert durch ihren Namen (NAME) und durch das Jahr (JAHR), in dem sie stattgefunden hat. Das Land (LAND) ist außerdem noch bekannt. Jede Regatta besteht aus mindestens drei jedoch maximal fünf Wettfahrten. Wettfahrten werden durch die zugehörige Regatta und das Datum (DATUM) identifiziert, außerdem wird die Länge (LAENGE) der Strecke gespeichert. Mannschaften können bei jeder Wettfahrt Punkte (PUNKTE) erzielen.

## ER – Diagramm



## Das Relationenmodell

Person(key, name, geburtsdatum)   
Segler(key: Person.key)   
Trainer(key: Person.key)

Boot(id, name, personen, tiefgang)   
Tourenboot(id: Boot.id, bootsklasse)   
Sportboot(id: Boot.id, segelflaeche)

Mannschaft(name, aklasse, key: Trainer.key)

Regatta(name, jahr, land)   
Wettfahrt(name: Regatta.name, jahr: Regatta.jahr, datum, laenge)

bildet(key: Segler.key, name: Mannschaft.name)   
zugewiesen(id: Boot.id, name: Mannschaft.name)   
nimmt\_teil(mname: Mannschaft.name, rname: Regatta.name, rjahr: Regatta.jahr, sportboot: Sportboot.id, startnr)   
erzielt(mname: Mannschaft.name, wname: Wettfahrt.name, wjahr: Wettfahrt.jahr, wdatum: Wettfahrt.datum, punkte)

## Die SQL-Anfragen

1. Geben Sie alle Mannschaften aus, die bei der Bodenseeregatta im Jahr 2014 teilgenommen haben. Wenn eine Mannschaft mit einem Boot mit der Segelfläche kleiner als 20 m2 teilgenommen hat, soll auch die ID des Bootes ausgegeben werden.
2. Geben Sie den Namen und das Geburtsdatum der jüngsten Trainer aus (können auch mehrere sein).
3. Geben Sie alle Personen geordnet nach Geburtsdatum aus, die sowohl Segler als auch Trainer sind, allerdings in keiner Mannschaft dabei sind.
4. Geben Sie alle Personen geordnet nach Geburtsdatum aus, die entweder Segler oder Trainer sind, jedoch nicht beides und vermerken Sie in einer Spalte, ob es sich um einen Trainer oder einen Segler handelt.
5. Geben Sie die Regatten (Name und Jahr) mit den wenigsten Wettfahrten an und geben Sie auch die Anzahl aus.
6. Geben Sie die Namen jener Trainer aus, die zwei oder mehr Mannschaften betreuen.
7. Welche Altersklasse ist am aktivsten (hat an den meisten Wettfahrten Punkte erzielt)?
8. Um wieviel gehen Tourenboote durchschnittlich tiefer als Sportboote?
9. Geben Sie für alle Mannschaften aus, an wievielen Regatten sie bereits teilgenommen haben und wieviele Punkte sie dort erzielt haben.
10. Welches Land bietet die längste Wettfahrtsstrecke und hat zusätzlich nicht die kürzeste?
11. Wie heißt der Trainer, der die Manschaft mit den meisten Punkten trainiert hat?
12. Geben Sie für JEDE Mannschaft aus, wieviele Punkte Sie bei der 'Bodenseeregatta' in 'Oesterreich' erzielt haben.
13. Geben Sie die ID und den Namen jener Sportboote aus, die mindestens an zwei Regatten Teil genommen haben, aber keiner Mannschaft zugewiesen sind.
14. Geben Sie die Regatten (Name, Jahr und Land) aus, die über die kürzeste Distanz gehen.

## Java und JDBC

Schreiben Sie einen Java Client, der eine JDBC-Verbindung zur Datenbank herstellt und AUTOCOMMIT ausschaltet. Realisieren Sie eine GUI, die einfache CRUD-Befehle auf die Boote des Vereins implementiert (keine explizite SQL-Eingabe). Verwenden Sie dabei auf jeden Fall eine JTable, die auch eine grafische Veränderung der Datensätze erlauben soll.

Als Erweiterung (Bonuspunkte) soll bei der Anzeige der Boote die Möglichkeit der Sortierung und Filterung über ein neues SQL-Kommando bereitgestellt werden. Auch hier soll nicht der Benutzer die SQL-Befehle eingeben, sondern es muss die Funktionalität über entsprechende GUI-Elemente realisiert werden!

Ermöglichen Sie die gleichzeitige Verbindung von mehreren Clients auf die Datenbasis. Implementieren Sie dabei eine transaktionell, gesicherte Erstellung und Änderung von Wettfahrten. Beachten Sie dabei, dass der Punktestand der einzelnen Wettfahrten laufend und von mehreren Clients gleichzeitig aktualisiert werden könnte. Stellen Sie für die Eingabe der Wettfahrt und Mannschaft eine einfache grafische Möglichkeit zur Verfügung.

## Abgabe

Die Abgabe ist am 20. März 2015 um 08:00 per elearning zu tätigen. Es wird ein Protokoll (Metaregeln), die SQL-Files (drop.sql, create.sql, start.sql, insert.sql, insert-\*.sql, queries.sql) in einem eigenen Verzeichnis. Dies alles wird in einem ausführbaren JAR-Archiv erwartet. Die Abgabe wird mit einem Prüfungsgespräch validiert, wobei auf eine eigenständige Lösung geachtet wird - kopierte Lösungen führen zu einer negativen Benotung! Quellen sollen somit auf den theoretischen Background und auf die Manuals beschränkt sein. Teile von bestehendem JDBC-Code aus dem Internet und vorgefertigte SQL-Abfragen dürfen somit nicht verwendet werden. Im Zweifelsfall ist es notwendig die Lehrkräfte um Freigabe von Quellen zu bitten. Das Beispiel soll für eine Postgresql 9.4 Umgebung implementiert werden.

Die Inserts sollen mindestens 10.000 Einträge enthalten. Es bleibt Ihnen überlassen, ob Sie einen selbstgeschriebenen Generator oder ein externes Tool verwenden möchten. Auf jeden Fall muss Ihre Vorgehensweise gut dokumentiert und nachvollziehbar sein. Die Daten sollen so nahe wie möglich der Wirklichkeit entsprechen, um entsprechende Testfälle und Performancetests auf der Datenbank starten zu können (Person1..10000 sind somit keine erwünschten Datensätze).

Bei Problemen mit dem Create-Script und den Inserts kann ein Example-Set bei den Lehrenden angefordert werden. Dies muss aber für jeden Kandidaten einzeln geschehen! Anfragen bitte immer per eMail an BEIDE Lehrer.

(c) Markus Pichlmair; adaptiert bei Michael Borko und Erhard List

# Aufwandsabschätzung & Zeitaufzeichnung

# Arbeitsdurchführung

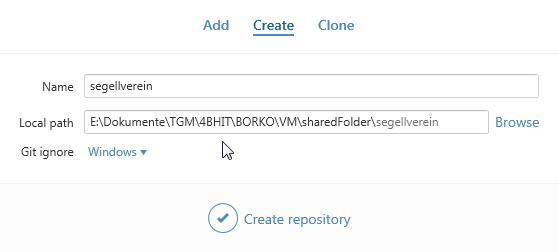
## Erzeugen des Benutzers und der Datenbank

Zu Beginn habe ich einen neuen Benutzer in der Datenbank angelegt.  


Danach habe ich eine neue Datenbank mit dem Namen *segelverein* erzeugt.  


Jetzt muss man dem Benutzer noch alle Rechte auf dieser Datenbank geben.  


## Erstellen eines *git* Repositorys



Hierbei hat mir der Herr Kopec ein private Repository zur Verfügung gestellt, da ich selbst keines habe.

## Erstellen einer *create.sql* Datei

## Erstellen eines *Generators*

Verwendete Programmiersprache: PHP  
Name der Datei: generator.php

Einzelne Funktionen zum Generieren der inserts für jede einzelne Tabelle.  
Eine Funktion, die dann alle Einzelfunktionen ausführt.

## Erstellen von zwei .sh Scripts

### start.sh

Ein script, das mir den Generator ausführt und mich anschließend fragt ob ich mich auf der Datenbank einloggen möchte (ohne Passworteingabe).[1]

#!/bin/sh

php generator.php

PGPASSWORD=segel psql -U segel -h localhost segelverein < start.sql

echo "Would you like to log in segelverein [y/n]?"

read ans

if [ $ans = y -o $ans = Y -o $ans = yes -o $ans = Yes -o $ans = YES ]

then

./login.sh

Fi

### login.sh

Ein Script, mit dem ich mich auf der Datenbank einloggen kann (ohne Passworteingabe).[1]

#!/bin/sh

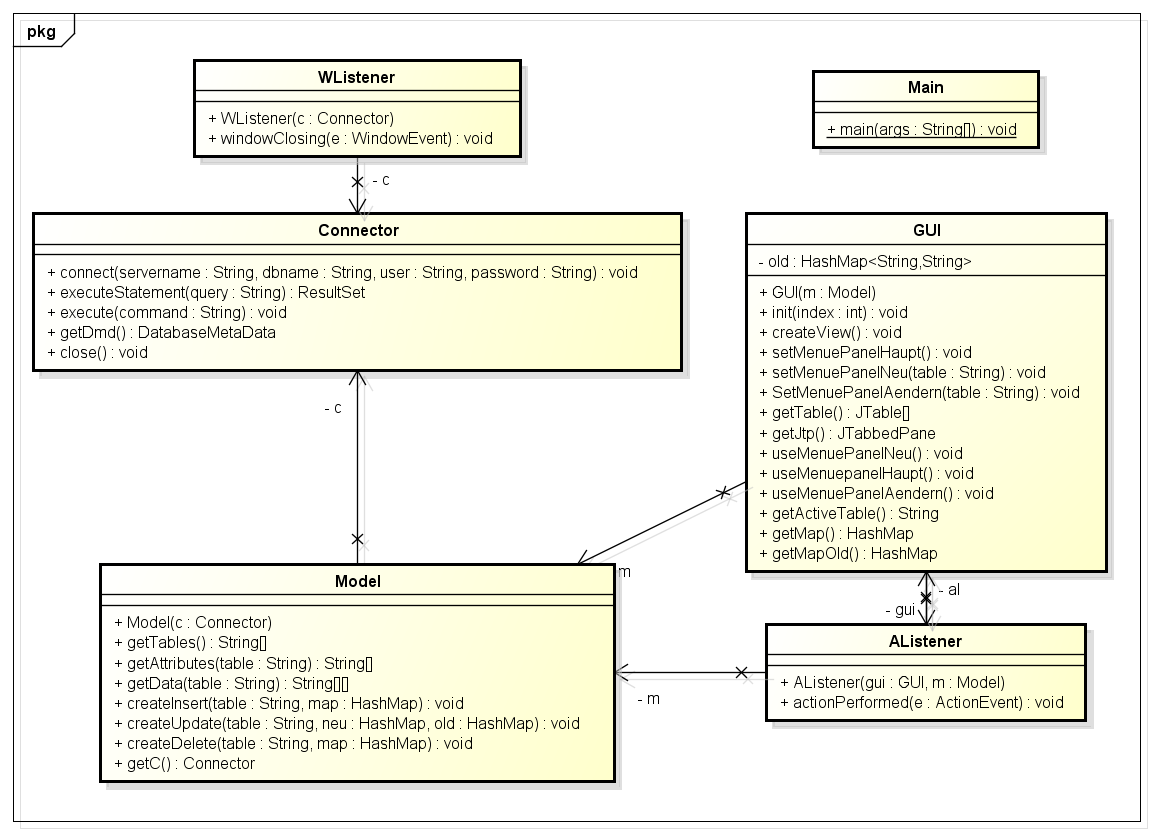
PGPASSWORD=segel psql -U segel -h localhost segelverein

## Java und JDBC

Erstellen eines Java Programms, welches sich mit der Datenbank verbindet. Dieses Programm ermöglicht einfache CRUD-Befehle auf alle Tabellen. In der *main* Methode kann man ganz einfach die zu verwendende Datenbank, den Benutzer und dessen Passwort ändern. [2][3]  
Mit Tabs am oberen Rand kann man ganz einfach zwischen den einzelnen Tabellen wechseln. In jeder Tabelle kann man anschließend mittels den Knöpfen im Menü am rechten Rand einfache CRUD-Befehle ausführen(Die Daten bearbeiten).  
Wenn man auf neu klickt wechseln die Knöpfe vom Hauptmenü zu Eingabefeldern und zwei Buttons zum Speichern oder Abbrechen. Wenn man nun korrekte Daten in die Felder einfügt und auf speichern klickt werden die Daten in die Datenbank gespeichert, die Tabellen aktualisiert und wieder das Hauptmenü angezeigt.  
Um einen Datensatz ändern zu können, muss man diesen zuerst per Mausklick auswählen und dann auf ändern klicken. Im Menüerscheinen Eingabefelder, befüllt mit den aktuellen Werten des ausgewählten Datensatzes und zwei Knöpfen, Speichern und Abbrechen. Jetzt kann man den/die gewünschten Änderungen vornähmen und anschließend auf speichern klicken. Wenn man nun korrekte Daten in die Felder einfügt und auf speichern klickt hat, werden die Daten in die Datenbank aktualisiert, die Tabellen aktualisiert und wieder das Hauptmenü angezeigt.  
Wenn man einen Datensatz löschen möchte muss man den gewünschten Datensatz zuerst per Mausklick auswählen und dann auf löschen Klicken. Der Eintrag wird aus der Datenbank gelöscht und die Tabellen werden aktualisiert.

## UML-Erzeugen

Mithilfe des Programms Astah ein UML-Diagramm zu meinem Java Programm erstellt.



## Git log

Eine log Datei meines Git repositorys erzeugt.  
*git log > JDBC/doc/log.txt*

# SQL-Anfragen

1. Geben Sie alle Mannschaften aus, die bei der Bodenseeregatta im Jahr 2014 teilgenommen haben. Wenn eine Mannschaft mit einem Boot mit der Segelfläche kleiner als 20 m2 teilgenommen hat, soll auch die ID des Bootes ausgegeben werden.

SELECT mname AS mannschaft, (SELECT id FROM sportboot WHERE segelflaeche<20 AND id=sportboot) AS boot\_id FROM nimmt\_teil WHERE rname = 'Bodenseeregatta' AND rjahr = 2014;

1. Geben Sie den Namen und das Geburtsdatum der jüngsten Trainer aus (können auch mehrere sein).

SELECT name,geburtsdatum FROM person ORDER BY geburtsdatum DESC LIMIT 25;

1. Geben Sie alle Personen geordnet nach Geburtsdatum aus, die sowohl Segler als auch Trainer sind, allerdings in keiner Mannschaft dabei sind.

SELECT name,key,geburtsdatum FROM person natural join trainer natural join segler WHERE key NOT IN (SELECT key FROM mannschaft) AND key NOT IN (SELECT key FROM bildet) ORDER BY geburtsdatum ASC;

1. Geben Sie alle Personen geordnet nach Geburtsdatum aus, die entweder Segler oder Trainer sind, jedoch nicht beides und vermerken Sie in einer Spalte, ob es sich um einen Trainer oder einen Segler handelt.

SELECT \*,'segler' AS funktion FROM person NATURAL JOIN segler WHERE key NOT IN (SELECT key FROM trainer) UNION SELECT \*,'trainer' AS funktion FROM person NATURAL JOIN trainer WHERE key NOT IN (SELECT key FROM segler) ORDER BY geburtsdatum ASC;

1. Geben Sie die Regatten (Name und Jahr) mit den wenigsten Wettfahrten an und geben Sie auch die Anzahl aus.

SELECT name,jahr,COUNT(datum) AS anzahl FROM wettfahrt GROUP BY name,jahr ORDER BY anzahl ASC LIMIT 25;

1. Geben Sie die Namen jener Trainer aus, die zwei oder mehr Mannschaften betreuen.

SELECT name,mannschaften FROM (SELECT name, (SELECT COUNT(name) FROM mannschaft WHERE key = person.key) AS mannschaften FROM person NATURAL JOIN trainer) AS trainer where mannschaften>1;

1. Welche Altersklasse ist am aktivsten (hat an den meisten Wettfahrten Punkte erzielt)?

SELECT altersklasse FROM (SELECT aklasse AS altersklasse,sum(punkte) AS punkte FROM erzielt LEFT JOIN mannschaft ON erzielt.mname=mannschaft.name GROUP BY aklasse ORDER BY punkte DESC) AS altersklassen LIMIT 1;

1. Um wieviel gehen Tourenboote durchschnittlich tiefer als Sportboote?

SELECT AVG(tiefgang)-(SELECT AVG(tiefgang) FROM sportboot NATURAL JOIN boot) as differenz FROM tourenboot NATURAL JOIN boot;

1. Geben Sie für alle Mannschaften aus, an wievielen Regatten sie bereits teilgenommen haben und wieviele Punkte sie dort erzielt haben.

SELECT mname AS mannschaft,COUNT(mname) AS anzahl, sum(punkte) AS punkte FROM (SELECT mname,wname,sum(punkte) as punkte FROM erzielt GROUP BY mname,wname) AS teilnehmen GROUP BY mname;

1. Welches Land bietet die längste Wettfahrtsstrecke und hat zusätzlich nicht die kürzeste?

SELECT land FROM (SELECT land,MAX(laenge) FROM regatta NATURAL JOIN wettfahrt GROUP BY land ORDER BY max DESC) AS laengen WHERE land NOT IN (SELECT land FROM regatta NATURAL JOIN wettfahrt GROUP BY land ORDER BY MIN(laenge) ASC LIMIT 1) LIMIT 1;

1. Wie heißt der Trainer, der die Manschaft mit den meisten Punkten trainiert hat?

SELECT name FROM person NATURAL JOIN (SELECT key,SUM(punkte) AS punkte FROM erzielt LEFT JOIN mannschaft ON mannschaft.name=erzielt.mname GROUP BY key ORDER BY punkte DESC) AS top\_trainer LIMIT 1;

1. Geben Sie für JEDE Mannschaft aus, wieviele Punkte Sie bei der 'Bodenseeregatta' in 'Oesterreich' erzielt haben.
   1. --Alle die teilnehmen

SELECT mname as name,sum(punkte) AS punkte FROM erzielt LEFT JOIN regatta ON erzielt.wname=regatta.name WHERE wname = 'Bodenseeregatta' AND land = 'Oesterreich' GROUP BY mname;

* 1. --Alle

SELECT mname as name,sum(punkte) AS punkte FROM erzielt LEFT JOIN regatta ON erzielt.wname=regatta.name WHERE wname = 'Bodenseeregatta' AND land = 'Oesterreich' GROUP BY mname UNION SELECT name,'0' AS punkte FROM mannschaft WHERE name NOT in (SELECT mname AS punkte FROM erzielt LEFT JOIN regatta ON erzielt.wname=regatta.name WHERE wname = 'Bodenseeregatta' AND land = 'Oesterreich' GROUP BY mname) GROUP BY name;

1. Geben Sie die ID und den Namen jender Sportboote aus, die mindestens an zwei Regatten teil enommen haben, aber keiner Mannschaft zugewiesen sind.

SELECT id, name FROM (SELECT sportboot AS id, COUNT(sportboot) FROM nimmt\_teil group by sportboot) AS teilnahmen NATURAL JOIN (SELECT id, name FROM (SELECT id, name FROM sportboot NATURAL JOIN boot WHERE id NOT IN (SELECT id FROM sportboot NATURAL JOIN zugewiesen)) AS boote) AS A;

1. Geben Sie die Regatten (Name, Jahr und Land) aus, die über die kürzeste Distanz gehen.

SELECT name,jahr,land,sum(laenge) AS laenge FROM wettfahrt NATURAL JOIN regatta GROUP BY name,jahr,land ORDER BY laenge ASC LIMIT 35;

# Quellen

[1] Doku über .sh Scripts  
<http://www.chemie.fu-berlin.de/chemnet/general/topics/scripts_sh.html>  
Zuletzt besucht: 18.03.2015

[2] Grundlegende Informationen über JDBC  
<https://elearning.tgm.ac.at/mod/resource/view.php?id=31157>  
Zuletzt besucht: 19.03.2015

[3] Java API  
https://docs.oracle.com/javase/8/docs/api/  
Zuletzt besucht: 20.03.2015

[4] PostgeSQL Manual  
<http://www.postgresql.org/files/documentation/pdf/9.3/postgresql-9.3-A4.pdf>  
Zuletzt besucht: 19.03.2015