swp15-lib

Administratormanual

Projektleiter: Christian Blecha

Christian Blecha
26.05.2015

Inhaltsverzeichnis

1 Allgemeines zur DB

Beim Einfügen von Daten in die DB ohne die App muss darauf geachtet werden, dass Sonderzeichen, wie z.B. ß, ä, ü und ö, entweder nicht verwendet oder nach Html-Standards escaped werden. Trägt man diese mit Hilfe der App ein, können auch Sonderzeichen verwendet werden. Diese werden automatisch escaped und in die DB eingetragen. Alle Variablen, die zwischen „ “ stehen, sind Variablen, die von Außen (von der App) in die Statements eingefügt werden müssen! Strings, die übergeben werden sollen, müssen in Hochkommatas stehen. Das Zeichen ‘ ist ein Accent grave und über die Tastenkombination Shift+(Accent−Taste rechts neben Backspace) und dann Space als alleinstehendes Zeichen erzeugbar. In der Tabelle GPS existieren 2 Attribute: x und y. x ist dabei der Längengrad, weil dieser sich in x−Richtung ändert und y ist der Breitengrad, weil dieser sich in y−Richtung ändert. Um die Kartenbilder in die DB einzufügen, bietet sich die MySQL-Workbench an.

Weitere Hinweise zur DB sind in der „Entwurfdokumentation des Gesamtprojektes“ in Kapitel 4.1 und Kapitel 8 zu finden.

2 Richtlinien für die Benutzung der DB

Alle Kartenbilder, die eingefügt werden, müssen im JPEG-Format eingefügt werden. Im folgenden wird immer wieder von „beliebiger Stellenanzahl“ gesprochen. Dies ist nur im Bereich der Wertigkeit des Datentyps int möglich. Für die DB ist das z.B. -2147483648 bis 2147483647 also sind maximal 10-stellige Id’s möglich. Für die Erstellung der verschiedenen ID’s ist folgendes Schema zu befolgen:

- Jede ID beginnt mit der Ziffer 1. Dies dient dazu, dass am Anfang stehende 0en, auf Grund der Tatsache, dass alle ID’s vom Typ Int und nicht vom Typ String sind, nicht gelöscht werden.
- Jede ID besteht aus mehreren Teilen:
  - der Anfangsziffer,
  - der Routen-ID,
  - einer Ziffer zur Kennzeichnung des gerade verwendeten „Zahlenbereiches“ und
  - der entsprechenden ID für ein Haus, einen Routenpunkt, ...
- Dieses Schema gilt für alle IDs außer der Routen-ID. Diese besteht nur aus der Anfangsziffer und der 3-stelligen Zahl zur Identifizierung der Route an sich. Diese Zahl muss immer 3-stellig sein. D.h. die erste Route besitzt die ID: 1001.
- Als Zahlenbereiche sind die folgenden angedacht:
  - Die 0 beschreibt einen Bereich, der IDs zur Verwaltung verschiedener routenspezifischer IDs verwendet wird. Darunter zählen:

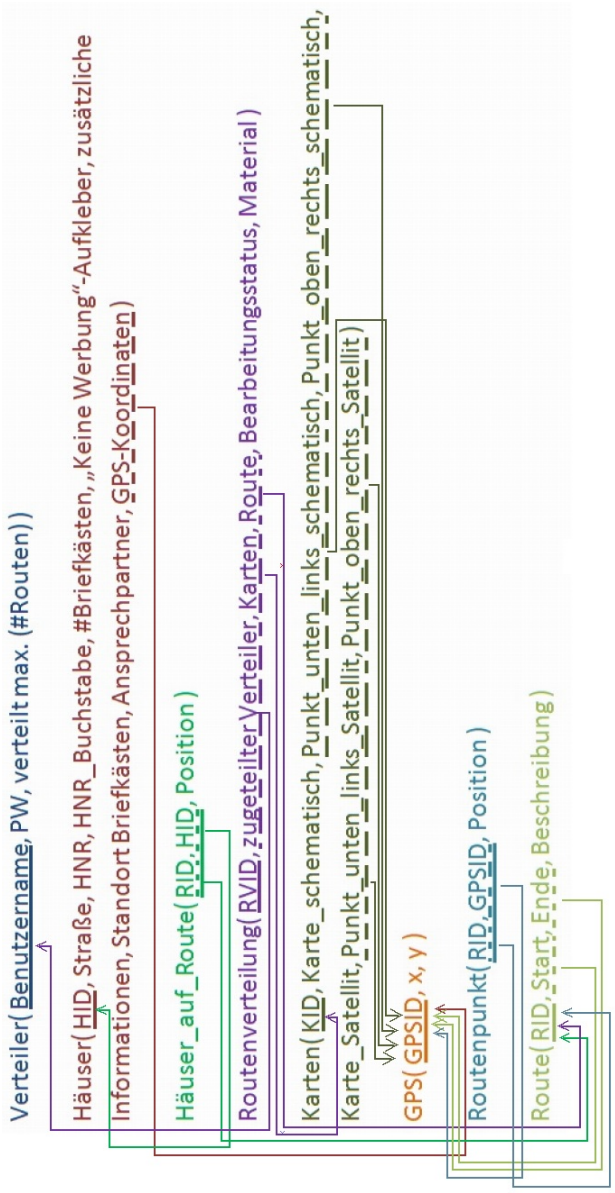
    - der Start einer Route (1− − −01),
    - das Ende einer Route (1− − −02),
    - die KID die zur Route gehört (1− − −03),
    - der Punkt_unten_links der schematischen Karten (1− − −04),
    - der Punkt_oben_rechts der schematischen Karten (1− − −05),
    - der Punkt_unten_links der Satellitenkarten (1− − −06),
    - der Punkt_oben_rechts der Satellitenkarten (1− − −07),
    - Routenverteilungs−IDs (aus dem Bereich 1− − −08 bis 1− − −0n (n ist bel. Zahl mit beliebiger Stellenanzahl)).
  - Die 1 beschreibt den Bereich, welcher die GPS-ID's der Routenpunkte einer Route enthält (1− − −11 bis 1− − −1m (m ist bel. Zahl mit beliebiger Stellenanzahl)).
  - Die 2 beschreibt den Bereich, welcher die GPS-ID's der Häuser einer Route enthält (1− − −21 bis 1− − −2p (p ist bel. Zahl mit beliebiger Stellenanzahl)).
  - Die 3 beschreibt den Bereich, welcher die HIDs der Häuser einer Route enthält (1− − −31 bis 1− − −3q (q ist bel. Zahl mit beliebiger Stellenanzahl)).

- Z.B. ein Haus behält die ID zum Zeitpunkt seiner Erstellung. D.h. ein Haus wird für eine bestimmte Route erstellt. Wird es später auf einer anderen Route verwendet, dann wird es einfach referenziert und nicht neu angelegt.

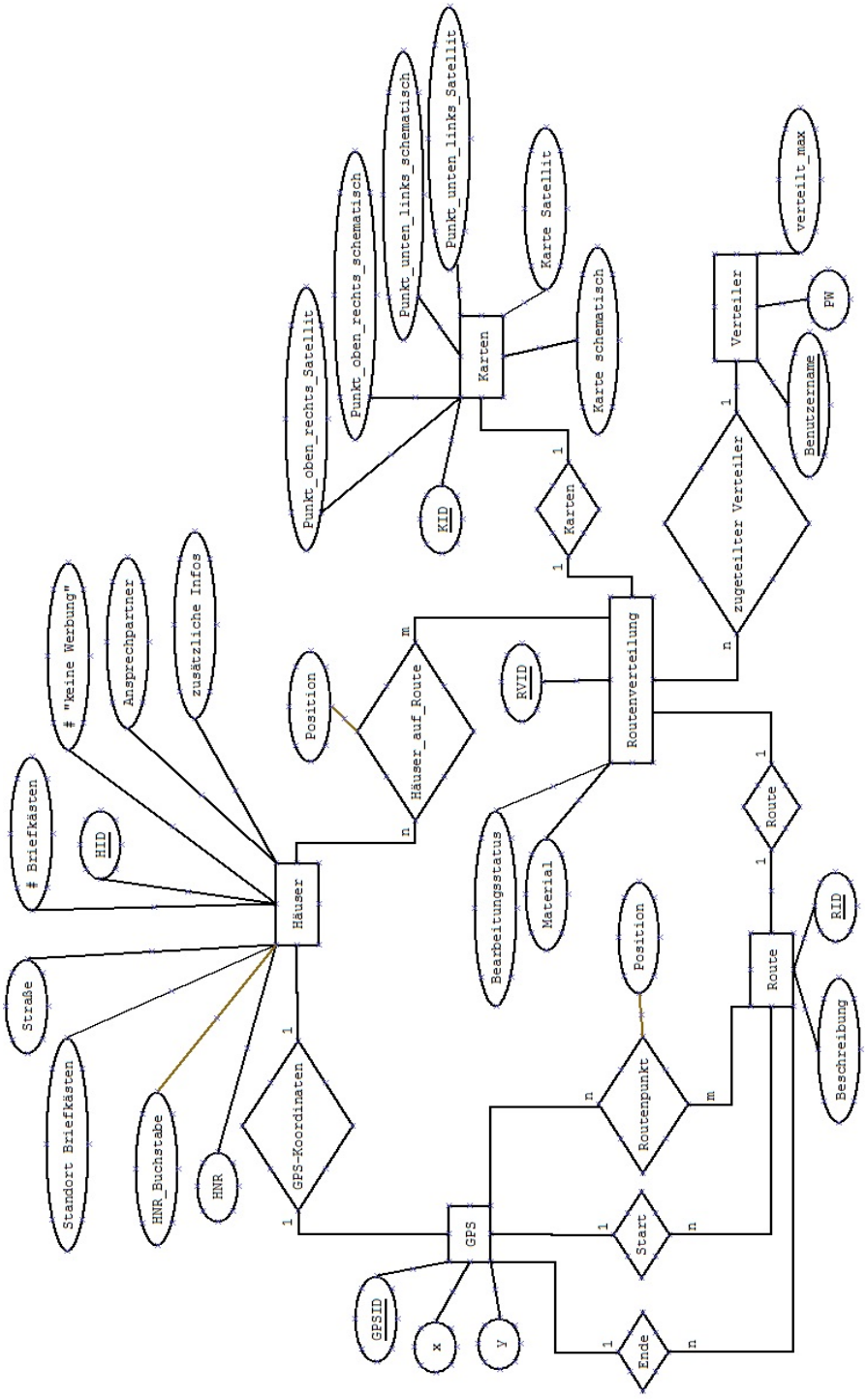
3 Allgemeines zum DB-Aufbau

Die hier aufgeführten SQL-Statements können so wie sie sind an eine MySQL-DB geschickt werden. Man sollte beim Kopieren nur darauf achten, dass aus design-technischen Gründen extra Zeichen bei den Statements eingefügt wurden, die man vor dem Abschicken der Statements löschen sollte.

4 Relationenmodell



5 ER-Modell


Punkt_oben_rechts_Satellit
Punkt_oben_rechts_schematisch
Punkt_unten_links_schematisch
Punkt_unten_links_Satellit
Karte Satellit
Karte schematisch
Karten
KID
Karten
1
1
vertellt_max
PW
Benutzername
Verteiler
1
zugeteilter Verteiler
n
Routenverteilung
RVID
m
Position
Häuser_auf_Route
n
Häuser
"keine Werbung"
Ansprechpartner
zusätzliche Infos
Briefkästen
HID
Straße
Standort Briefkästen
HNR_Buchstabe
HNR
GPS-Koordinaten
1
1
Beazbeitungsstatus
Material
1
Route
1
Route
RID
Beschreibung
Position
n
Routenpunkt
m
GPS
1
Start
n
GPSID
x
y
1
Ende
n

5.1 MySQL-Statements für das Erstellen der Tabellen

```
Create Table Verteiler (
        Benutzername Varchar(20) ,
        PW Varchar(30) Not Null ,
        verteilt_max Int Default 1 ,
        Primary Key (Benutzername) );

Create Table GPS (
        GPSID Int ,
        x Double Not null ,
        y Double Not Null ,
        Primary Key (ID));

Create Table Route (
        RID Int ,
        Start Int ,
        Ende Int ,
        Foreign Key (Start) References GPS (GPSID)
        On Delete Cascade
        On Update Cascade ,
        Foreign Key (Ende) References GPS (GPSID)
        On Delete Cascade
        On Update Cascade ,
        Primary Key (RID));

Create Table Haeuser (
        HID Int ,
        Straße Varchar(50),
        HNR INT Default 0,
        HNR_Buchstabe Varchar(1) Default Null,
        Briefkaesten Int Default 0,
        Keine_Werbung_Aufkleber Int Default 0,
        zusaetzliche_Informationen Varchar(100),
        Standort_Briefkaesten Varchar(100),
        Ansprechpartner Varchar(100),
        GPS_Koordinaten Int,
        Foreign Key (GPS_Koordinaten) References GPS (GPSID)
        On Delete Cascade
        On Update Cascade ,
        Primary Key (HID));
```

Create Table Haeuser_auf_Route (
RID Int ,
HID Int ,
Position Int,
Foreign Key (RID) References Route (RID)
On Delete Cascade
On Update Cascade ,
Foreign Key (HID) References Haeuser (HID)
On Delete Cascade
On Update Cascade ,
Primary Key (RID, HID, Position));

Create Table Routenpunkt (
RID Int ,
GPSID Int ,
Position Int Foreign Key (RID) References Route (RID)
On Delete Cascade
On Update Cascade ,
Foreign Key (GPSID) References GPS (GPSID)
On Delete Cascade
On Update Cascade ,
Primary Key (RID, ID, Position));

Create Table Karten (
KID Int ,
Karte_schematisch Longblob ,
Punkt_unten_links_schematisch Int ,
Punkt_oben_rechts_schematisch Int ,
Karte_Satellit Longblob ,
Punkt_unten_links_Satellit Int ,
Punkt_oben_rechts_Satellit Int ,
Foreign Key (Punkt_unten_links_schematisch) References GPS (GPSID)
On Delete Cascade
On Update Cascade ,
Foreign Key (Punkt_oben_rechts_schematisch) References GPS (GPSID)
On Delete Cascade
On Update Cascade ,
Foreign Key (Punkt_unten_links_Satellit) References GPS (GPSID)
On Delete Cascade
On Update Cascade ,
Foreign Key (Punkt_oben_rechts_Satellit) References GPS (GPSID)
On Delete Cascade
On Update Cascade ,
Primary Key (KID));

```
Create Table Routenverteilung (
        RVID Int ,
        zugeteilter_Verteiler Varchar(20) ,
        Karten Int ,
        Route Int ,
        Bearbeitungsstatus Varchar(1) Default 'n' ,
        Material Varchar(100) ,
        Foreign Key (zugeteilter_Verteiler) References Verteiler (Benutzername)
        On Delete Cascade
        On Update Cascade ,
        Foreign Key (Route) References Route (RID)
        On Delete Cascade
        On Update Cascade ,
        Foreign Key (Karten) References Karten (KID)
        On Delete Set Null
        On Update Cascade ,
        Primary Key (RVID));
```

5.2 MySQL-Statements für das Erstellen der Stored Functions

Funktion CheckMaxRoutenAnzahl:

```
Drop Function If Exists CheckMaxRoutenAnzahl;

Delimiter //

Create Definer=‘root‘@‘localhost‘ Function ‘CheckMaxRoutenAnzahl‘( ) RETURNS
int(11)
Begin
    Declare erg Integer;
    Declare anzahl Integer;
    Select count(*) into anzahl From (Select F.name From (Select Distinct (zuge-
teilter_Verteiler) as name, (‘test‘.‘GetRoutenAnzahl‘(zugeteilter_Verteiler)) as anzahl
From test.Routenverteilung) As F, test.Verteiler Where F.name=test.Verteiler.Benutzername
And F.anzahl > test.Verteiler.verteilt_max) as v;
    Set erg=1;
    If(anzahl>0) Then
        Set erg=0;
    End If;
RETURN erg;
End;
//
```

Funktion GetRoutenAnzahl:

```
Drop Function If Exists GetRoutenAnzahl;

Delimiter //

Create Definer=‘root‘@‘localhost‘ Function ‘GetRoutenAnzahl‘(name Varchar(20))
RETURNS int(11)
Begin
    /* Anzahl ermitteln */
    DECLARE anzahl int;
    Set anzahl = 0;
    Select count(*) into anzahl From test.Routenverteilung Where zugeteilter_Verteiler=name;
RETURN anzahl;
End;
//
```

5.3 MySQL-Statements für das Erstellen der Trigger

Hinweis: Die Trigger/ Funktionen nicht mit ExequteQuery oder Squirrel sondern mit Konsole oder über die MySQL-Workbench eingeben!

Trigger BooleanHackUpdate:

```
Drop Trigger If Exists BooleanHackUpdate;

Delimiter //

Create Trigger BooleanHackUpdate
Before Update On test.Routenverteilung
For Each Row
Begin
    If(New.Bearbeitungsstatus!=„n“ AND New.Bearbeitungsstatus!=„j“) Then
        Signal SQLState „45000“
        Set Message_Text=„An error occurred. The only working values for the attribute Routenverteilung.Bearbeitungsstatus are ’j’ or ’n’!“;
    End If;
End;
//
```

Trigger BooleanHackInsert:

```
Drop Trigger If Exists BooleanHackInsert;

Delimiter //

Create Trigger BooleanHackInsert
Before Insert On test.Routenverteilung
For Each Row
Begin
    If(New.Bearbeitungsstatus!=„n“ AND New.Bearbeitungsstatus!=„j“) Then
        Signal SQLState „45000“
        Set Message_Text=„An error occurred. The only working values for the attribute Routenverteilung.Bearbeitungsstatus are ’j’ or ’n’!“;
    End If;
End;
//
```

Trigger CheckRoutenIntegrity:

```
Drop Trigger If Exists CheckRoutenIntegrity;

Delimiter //

Create Trigger CheckRoutenIntegrity
After Insert On test.Routenverteilung
For Each Row
Begin
    If(test.CheckMaxRoutenAnzahl()=0) Then
        Signal SQLState „45001“
        Set Message_Text=„An error occurred. You can’t assign more routes to an
user as they had set as theire max. routes!“;
        End If;
End;
//
```

5.4 MySQL-Statements für das Erstellen der Stored Procedures

Prozedur DeleteHelp:

```
Drop Procedure If Exists DeleteHelp;

Delimiter //

Create Definer=‘root‘@‘localhost‘ Procedure ‘DeleteHelp‘(IN name Varchar(20))
Begin
    Declare Continue Handler for sqlexception,not found, sqlwarning
    Begin

    End;

    /* Delete User */
    Set @s = Concat('Drop User ', name);
    Prepare stmt From @s;
    Execute stmt;
    Deallocate Prepare stmt;

    /* Delete Views */
    Set @v1 = Concat('Drop View ', name,'_r');
    Prepare stmtv1 From @v1;
    Execute stmtv1;
    Deallocate Prepare stmt;
    Set @v2 = Concat('Drop View ', name,'_rv');
    Prepare stmtv2 From @v2;
    Execute stmtv2;
    Deallocate Prepare stmtv2;
    Set @v3 = Concat('Drop View ', name,'_rp');
    Prepare stmtv3 From @v3;
    Execute stmtv3;
    Deallocate Prepare stmtv3;
    Set @v4 = Concat('Drop View ', name,'_gps');
    Prepare stmtv4 From @v4;
    Execute stmtv4;
    Deallocate Prepare stmtv4;
    Set @v5 = Concat('Drop View ', name,'_h');
    Prepare stmtv5 From @v5;
    Execute stmtv5;
    Deallocate Prepare stmtv5;
    Set @v6 = Concat('Drop View ', name,'_har');
    Prepare stmtv6 From @v6;
```

```
        Execute stmtv6;
        Deallocate Prepare stmtv6;
        Set @v7 = Concat('Drop View ', name,' _k');
        Prepare stmtv7 From @v7;
        Execute stmtv7;
        Deallocate Prepare stmtv7;

        /* Delete Verteiler-Datensatz */
        Set @d = Concat('Delete from test.Verteiler where Benutzername=',name,';');
        Prepare stmtd From @d;
        Execute stmtd;
        Deallocate Prepare stmtd;
    End;
    //
```

Prozedur NewCreateAdmin:

```
    Drop Procedure If Exists NewCreateAdmin;

    Delimiter //

    Create Definer=`root`@`localhost` Procedure `NewCreateAdmin`(IN name Varchar(20), IN pw Varchar(30))
    Begin

        /* Admin erstellen */
        Set @s = Concat('Create User ', name);
        Prepare stmt From @s;
        Execute stmt;
        Deallocate Prepare stmt;

        /* alle Rechte geben */
        Set @r = Concat('Grant ALL PRIVILEGES on *.* to ', name,' with grant option');
        Prepare stmtr From @r;
        Execute stmtr;
        Deallocate Prepare stmtr;
        /* Passwort setzen */
        Set @p = Concat('Set Password For ', name, ' = Password(',pw,')');
        Prepare stmtp From @p;
        Execute stmtp;
        Deallocate Prepare stmtp;
    End;
    //
```

Prozedur NewCreateUser:

```
Drop Procedure If Exists NewCreateUser;

Delimiter //

Create Definer=‘root‘@‘localhost‘ Procedure ‘NewCreateUser‘(IN name Varchar(20),
IN pw Varchar(30), IN anzahl Int)
Begin

        /* Exception-Handler erstellen */
        Declare Exit Handler For sqlexception
        Begin
                Signal SQLState „45002“ Set Message_Text=„An error occured. If the
user already exists correctly, use another username else call the stored procedure De-
leteHelp.“;
                Rollback;
        End;

        Start transaction;

        /* User erstellen */
        Set @s = Concat('Create User ', name);
        Prepare stmt From @s;
        Execute stmt;
        Deallocate Prepare stmt;

        /* alle Rechte entfernen */
        Set @r = Concat('REVOKE ALL PRIVILEGES on *.* FROM ', name);
        Prepare stmtr From @r;
        Execute stmtr;
        Deallocate Prepare stmtr;

        /* Passwort setzen */
        Set @p = Concat('Set Password For ', name, ' = Password(',pw,')');
        Prepare stmtp From @p;
        Execute stmtp;
        Deallocate Prepare stmtp;

        /* Datensatz in Verteilertabelle ablegen */
        Set @i = Concat('Insert Into test.Verteiler(Benutzername,PW,verteilt_max) Va-
lues (', name,',',pw,',',anzahl,')');
        Prepare stmti From @i;
```

```
Execute stmti;
Deallocate Prepare stmti;

/* Routenverteilungsview erstellen */
Set @v1 = Concat('Create View ',name,'_rv as Select * From test.Routenverteilung
Where zugeteilter_Verteiler='',name,'');
Prepare stmtv1 From @v1;
Execute stmtv1;
Deallocate Prepare stmtv1;

/* Routenview erstellen */
Set @v2 = Concat('Create View ',name,'_r as Select Distinct * From test.Route
where Rid in (Select Distinct Route From test.',name,'_rv)');
Prepare stmtv2 From @v2;
Execute stmtv2;
Deallocate Prepare stmtv2;

/* Haeuser_auf_routeview erstellen */
Set @v3 = Concat('Create View ',name,'_har as Select * From test.Haeuser_auf_route
where Rid in (Select RID From test.',name,'_r)');
Prepare stmtv3 From @v3;
Execute stmtv3;
Deallocate Prepare stmtv3;

/* Haeuserview erstellen */
Set @v4 = Concat('Create View ',name,'_h as Select * From test.Haeuser where
hid in (Select HID From test.',name,'_har)');
Prepare stmtv4 From @v4;
Execute stmtv4;
Deallocate Prepare stmtv4;

/* Routenpunktview erstellen */
Set @v5 = Concat('Create View ',name,'_rp as Select * From test.Routenpunkt
where rid in (Select RID From test.',name,'_r)');
Prepare stmtv5 From @v5;
Execute stmtv5;
Deallocate Prepare stmtv5;

/* Kartenview erstellen */
Set @v7 = Concat('Create View ',name,'_k as Select Distinct * From test.Karten
where KID in (Select Distinct Karten From test.',name,'_rv)');
Prepare stmtv7 From @v7;
Execute stmtv7;
Deallocate Prepare stmtv7;
```

```
/* GPSview erstellen */
Set @v6 = Concat('Create View ',name,'_gps as Select * From test.GPS where
GPSID in (Select GPSID From test.',name,'_rp UNION Select Start From test.',name,'_r
UNION Select Ende From test.',name,'_r Union Select GPS_Koordinaten From test.',name,'_h
UNION Select Punkt_unten_links_schematisch From test.',name,'_k UNION Select
Punkt_oben_rechts_schematisch From test.',name,'_k UNION Select Punkt_unten_links_Satellit
From test.',name,'_k UNION Select Punkt_oben_rechts_Satellit From test.',name,'_k)');
Prepare stmtv6 From @v6;
Execute stmtv6;
Deallocate Prepare stmtv6;

/* Select-Rechte für Views setzen */
Set @r1 = Concat('Grant Select on ',name,'_rv to ',name);
Prepare stmtr1 From @r1;
Execute stmtr1;
Deallocate Prepare stmtr1;
Set @r2 = Concat('Grant Select on ',name,'_r to ',name);
Prepare stmtr2 From @r2;
Execute stmtr2;
Deallocate Prepare stmtr2;
Set @r3 = Concat('Grant Select on ',name,'_har to ',name);
Prepare stmtr3 From @r3;
Execute stmtr3;
Deallocate Prepare stmtr3;
Set @r4 = Concat('Grant Select on ',name,'_h to ',name);
Prepare stmtr4 From @r4;
Execute stmtr4;
Deallocate Prepare stmtr4;
Set @r5 = Concat('Grant Select on ',name,'_rp to ',name);
Prepare stmtr5 From @r5;
Execute stmtr5;
Deallocate Prepare stmtr5;
Set @r6 = Concat('Grant Select on ',name,'_gps to ',name);
Prepare stmtr6 From @r6;
Execute stmtr6;
Deallocate Prepare stmtr6;
Set @r14 = Concat('Grant Select on ',name,'_k to ',name);
Prepare stmtr14 From @r14;
Execute stmtr14;
Deallocate Prepare stmtr14;

/* Update-Rechte für rv.Bearbeitungsstatus setzen */
Set @r7 = Concat('Grant Update (Bearbeitungsstatus) on ',name,'_rv to ',na-
```

```
me);
        Prepare stmtr7 From @r7;
        Execute stmtr7;
        Deallocate Prepare stmtr7;

        /* Insert-Rechte setzen */
        Set @r8 = Concat('Grant Insert on ',name,'_gps to ',name);
        Prepare stmtr8 From @r8;
        Execute stmtr8;
        Deallocate Prepare stmtr8;
        Set @r9 = Concat('Grant Insert on ',name,'_h to ',name);
        Prepare stmtr9 From @r9;
        Execute stmtr9;
        Deallocate Prepare stmtr9;
        Set @r10 = Concat('Grant Insert on ',name,'_har to ',name);
        Prepare stmtr10 From @r10;
        Execute stmtr10;
        Deallocate Prepare stmtr10;
        Set @r11 = Concat('Grant Insert on ',name,'_rp to ',name);
        Prepare stmtr11 From @r11;
        Execute stmtr11;
        Deallocate Prepare stmtr11;

        /* Update-Rechte setzen */
        Set @r12 = Concat('Grant update on ',name,'_h to ',name);
        Prepare stmtr12 From @r12;
        Execute stmtr12;
        Deallocate Prepare stmtr12;
        Set @r13 = Concat('Grant update (position) on ',name,'_har to ',name);
        Prepare stmtr13 From @r13;
        Execute stmtr13;
        Deallocate Prepare stmtr13;
        Commit;
    End;
    //
```

6 MySQL-Statements die in der App verwendet werden

6.1 Routenauswahl Datengrundlage herunterladen (Methode: getRouteShort)

Parameter: Benutzername als name

Select-Statement MapChooseMenu Liste füllen:

SELECT Route, Material, Beschreibung
FROM „name“_rv as rv „name“_r as r
WHERE where Bearbeitungsstatus='n' And rv.Route=r.RID

Ergebnis: Liste mit Routenobjekten, bei denen die RID, das Material und die Beschreibung sowie der Bearbeitungtsstatus (durch Suche nach Bearbeitungsstatus='n') gefüllt sind und in der Liste angezeigt werden können.

6.2 Alle Haueser einer Route auslesen (Methode: getHouses)

Parameter: Benutzername als name, Routenid as rid

SELECT h.HID, h.Strasze,h.HNR,h.HNR_Buchstabe,h.Briefkaesten,h.Keine_Werbung_Aufkleber, h.zusaetzliche_Informationen, h.Standort_Briefkaesten, h.Ansprechpartner, gps.x as GPS_x,gps.y as GPS_y, har.position
FROM „name“_har as har, „name“_h as h, „name“_gps as gps
WHERE har.HID=h.HID And har.RID=„rid“ and h.GPS_Koordinaten=gps.gpsid
ORDER BY h.HID ASC

Ergebnis: Liste mit Häusern und deren Infos, ohne Referenzen.

6.3 Routenpunkte herunterladen (Methode: getWaypoints)

Parameter: Benutzername als name, Routenid as rid

Select-Statement Start- und Endpunkt auslesen:

SELECT gps1.x as Start_x,gps1.y as Start_y, gps2.x as Ende_x, gps2.y as Ende_y
FROM „name“_r as r, „name“_gps as gps1, „name“_gps as gps2
WHERE r.rid =„rid“ and r.Start=gps1.gpsid and r.Ende=gps2.gpsid

Ergebnis: Start- und Endpunkt einer Route (ohne Referenz).

Select-Statement restliche Routenpunkte herunterladen:

SELECT x,y,position
FROM „name“_rp as rp„name“_gps as gps
WHERE rp.RID=„rid“ And rp.GPSID=gps.GPSID

Ergebnis: Liste mit allen Routenpunkten einer Route (außer den Start- und Endpunkt)(ohne Referenzen) geordnet nach deren Position.
Alles zusammen muss nur noch in eine Liste der Reihe nach eingefügt werden: Startpunkt, Liste mit allen Routenpunkten, Endpunkt.

6.4 Karten herunterladen

Parameter: Benutzername als name, Routenid as rid und das Material als material

Select-Statement KID auslesen (Methode: getMapID):

SELECT Karten
FROM „name“_rv
WHERE Routen=“rid“ And Material=“material“

Ergebnis: Die Karten-ID der gerade verwendeten Route.

Parameter: Benutzername als name, Karten-ID as kid

Select-Statement GPS-Koordinaten Punkt_unten_links_schematisch (Methode: getCornerPoint):

SELECT x,y
FROM „name“_k as k, „name“_gps as gps
WHERE KID=“kid“ And Punkt_unten_links_schematisch = gps.gpsid

Ergebnis: Linker unterer Eckpunkt der schematischen Karte.

Parameter: Benutzername als name, Karten-ID as kid

Select-Statement GPS-Koordinaten Punkt_oben_rechts_schematisch (Methode: getCornerPoint):

SELECT x,y
FROM „name“_k as k, „name“_gps as gps
WHERE KID=“kid“ And Punkt_oben_rechts_schematisch = gps.gpsid

Ergebnis: Rechter oberer Eckpunkt der schematischen Karte.

Parameter: Benutzername als name, Karten-ID as kid

Select-Statement GPS-Koordinaten Punkt_unten_links_Satellit (Methode: getCornerPoint):

SELECT x,y
FROM „name“_k as k, „name“_gps as gps
WHERE KID=“kid“ And Punkt_unten_links_Satellit = gps.gpsid

Ergebnis: Linker unterer Eckpunkt der Satellitenkarte.

Parameter: Benutzername als name, Karten-ID as kid

Select-Statement GPS-Koordinaten Punkt_oben_rechts_Satellit (Methode: getCornerPoint):

SELECT x,y
FROM „name“_k as k, „name“_gps as gps
WHERE KID=“kid“ And Punkt_oben_rechts_Satellit = gps.gpsid

Ergebnis: Rechter oberer Eckpunkt der Satellitenkarte.

6.5 Update des Bearbeitungsstatus & Resynchronisieren mit dem Server

Parameter: Benutzername als name, Routenid als rid und Material als material

Update-Statement Bearbeitungsstatus setzen (Methode: updateStatus):

UPDATE „name“_rv
SET Bearbeitungsstatus = ’j’
WHERE Route = „rid“ And Material=“material“

Ergebnis: Der Status der Route in der DB wird von „nicht bearbeitet“ als ’n’ zu „bearbeitet“ als ’j’.

Parameter: Benutzername als name, die neue HID als hid, die neue Straße als straße, die neue HNR als hnr, der neue HNR_Buchstabe als hnr_Buchstabe, die neue Anz. der Briefkaesten als briefkaesten, die neue Anz. der „Keine-Werbung-Aufkleber“ als keineWerbungAufkleber, die neuen zusätzlichen Informationen als zusätzliche Informationen, der neue Standort der Briefkaesten als Standort Briefkaesten, der neue Ansprechpartner als Ansprechpartner, die neuen GPS-Koordinaten als Referenz(GPSID) als

GPS_Koordinaten

Update-Statement Haus updaten (Methode: updateHouse):

UPDATE „name“_h
SET Strasze = „straße“, HNR = „hnr“, HNR_Buchstabe = „hnr_Buchstabe“, Briefkaesten = „briefkaesten“, Keine_Werbung_Aufkleber = „keineWerbungAufkleber“, zusaetzliche_Informationen = „zusaetzliche Informationen“, Standort_Briefkaesten = „Standort Briefkaesten“, Ansprechpartner = „Ansprechpartner“, GPS_Koordinaten = „GPS_Koordinaten“
WHERE HID=„hid“

Ergebnis: Der Status der Route in der DB wird von „nicht bearbeitet“ als 'n' zu „bearbeitet“ als 'j'.

Parameter: Benutzername als name, Routenid als rid

Select-Statement Max. Haus−GPSID auslesen (Methode: getMaxHouseGPS):

SELECT Max(g.GPSID) as max_ID
FROM (**SELECT** gps.GPSID
FROM „name“_gps as gps, „name“_h as h, „name“_har as har
WHERE h.GPS_Koordinaten=gps.GPSID And har.RID=„rid“ And har.HID=h.HID
) as g

Ergebnis: Es wird die größte Haus-GPSID im System für die Route zurückgegeben, damit nach den in 2 auf Seite 3 neue Häuser eingefügt werden können.

Parameter: Benutzername als name, neue GPSID als gpsid, die neue x-Koordinate als x und die neue y-Koordinate als y

Insert-Statement Insert eines GPS-Punktes (Methode: insertGPS):

INSERT INTO „name“_gps (‘GPSID‘,‘x‘,‘y‘)
VALUES(„GPSID“„x“„y“)

Ergebnis: Es wird ein neuer GPS-Punkt in der DB eingefügt, solange eine GPSID verwendet wird, die noch nicht existiert.

Parameter: Benutzername als name, Routenid als rid

Select-Statement Max. Haus-ID auslesen (Methode: getMaxHouseId):

SELECT Max(ha.HID) as max_HID
FROM (**SELECT** h.HID

FROM „name“_h ash, „name“_har as har
WHERE har.HID=h.HID And har.RID=„rid“) as ha

Ergebnis: Es wird die größte Haus-ID im System für die Route zurückgegeben, damit nach den in 2 auf Seite 3 neue Häuser eingefügt werden können.

Parameter: Benutzername als name, neue HID als hid, die neue Straße als straße, die neue HNR als hnr, der neue HNR_Buchstabe als hnr_Buchstabe, die neue Anz. der Briefkaesten als briefkaesten, die neue Anz. der „Keine-Werbung-Aufkleber“ als keineWerbungAufkleber, die neuen zusätzlichen Informationen als zusätzliche Informationen, der neue Standort der Briefkaesten als Standort Briefkaesten, der neue Ansprechpartner als Ansprechpartner, die neuen GPS-Koordinaten als Referenz(GPSID) als GPS_Koordinaten

Insert-Statement Insert eines Hauses (Methode: insertHouse):

INSERT INTO „name“_h (‘HID‘,‘Strasze‘,‘HNR‘,‘HNR_Buchstabe‘,‘Briefkaesten‘,‘Keine_Werbung_Au
VALUES(„hid“„straße“„hnr“„hnr_Buchstabe“„briefkästen“„keineWerbungAufkleber“„zusätzliche Informationen“„Standort Briefkaesten“„Ansprechpartner“„GPS_Koordinaten“)

Ergebnis: Es wird ein neues Haus in der DB eingefügt, solange eine GPSID verwendet wird, die schon existiert/ vorher eingefügt wurde und eine HID verwendet wurde, die noch nicht existiert.

Parameter: Benutzername als name, neue HID als hid, Routenid als rid und Position des Houses in der Route as position

Insert-Statement Insert des Hauses auf der Route (Methode: putHouseOnRoute):

INSERT INTO „name“_har (‘RID‘,‘HID‘,‘position‘)
VALUES(„rid“„hid“„position“)

Ergebnis: Es wird ein Haus auf der aktuellen Route in der DB eingefügt, solange die HID schon existiert.

Parameter:
Commit-Statement nach Inserts (Methode: sendCommit)
commit;
Ergebnis: Die eingefügeten Daten werden bestätigt und fest in die DB geschrieben.
Parameter: Benutzername als name, HID als hid, Routenid als rid und neue Position des Houses in der Route as position

Update-Statement Update der Position eines Houses in einer Route(Methode: updateHouseOnRoutePosition):

UPDATE „name"_har
SET Position=„position“
WHERE RID=„rid“ And HID=„hid“

Ergebnis: Die Position des Houses in der Route wird geändert.