Bachelorthesis

Herve Ghislain Ngamadjeu Youssa

Entwicklung einer graphischen Oberfläche für die Datenbank des Center for Demand Side Integration zum Abruf, zur Manipulation und zur Visualisierung von Mess-und Simulationsdaten

Fakultät Technik und Informatik
Department Informations- und Elektrotechnik

Faculty of Engineering and Computer Science
Department of Information and Electrical Engineering

Herve Ghislain Ngamadjeu Youssa

Entwicklung einer graphischen Oberfläche für die Datenbank des Center for Demand Side Integration zum Abruf, zur Manipulation und zur Visaulisierung von Mess-und Simulationsdaten

Bachelorthesis eingereicht im Rahmen der Bachelorprüfung

im Studiengang Informations- und Elektrotechnik

am Department Informations- und Elektrotechnik

der Fakultät Technik und Informatik

der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer : Prof. Dr. Ing Franz Schübert

Zweitgutachter : Prof. Dr. Ing Karin Landenfeld

Abgegeben am 06. Oktorber 2015

Herve Ghislain Ngamadjeu Youssa

Thema der Bachelorthesis

Entwicklung einer graphischen Oberfläche für die Datenbank des Center for Demand Side Integration zum Abruf, zur Manipulation und zur Visualisierung von Mess-und Simulationsdaten.

Stichworte

Python2.7, Spyder, Entwicklungswerkzeug Qt-design, Datenbanken, PgAdmin3, PostgreSQl, SQL, GUI, Plot

Kurzzusammenfassung

Die vorliegende Arbeit beschreibt die ständige Manipulation einer Datenbank durch die graphische Benutzeroberfläche in Verbindung mit Python-Code. Bevor die Daten aus der Datenbank auf der graphische Benutzeroberfläche gezeigt werden, wird die Datenbank durch mit Python Skriptmodus durch Psycopg2 verbunden. Die graphische Benutzeroberfläche(GUI) wird direkt von Qt-Designer erstellt, wird als Quellcode in Spyder übersetzt und ausgeführt. Die gespeicherten Daten aus der Datenbank, werden von der GUI abgeruft und angezeigt. Die Daten können per Zeitintervalloptionen in Tages, in Stunden, in Minuten, in Viertelstunden oder in Sekunden(mittel)werte konvertiert werden. Zusätzlich könne die Daten über die GUI in eine CSV-Datei gespeichert werden.

Herve Ghislain Ngamadjeu Youssa

Title of the paper

Developing a graphical user interface for the database of the Center for Demand Side Integration for retrieval, manipulation and visualization of measurement and simulation data.

Keywords

Python, Spyder (Python 2.7), development tools Qtdesign, C4DSI database pgAdmin3, PostgreSQL, SQL database language

Abstract

This report describes the construction of of an important Controller for microprocessors. Before the data from the database on the graphical user interface will be shown, the Tool using Python Script is linked by Psycopg2. The graphical user interface (GUI) is created directly from Qt designer, is translated as a source in Spyder and executed. The data stored in the database are called by the GUI and displayed directly. The data could custom user in several options (average in days, in hours, in fifteen minutes, in minutes and seconds) Convert and be saved in CSV formats.

Danksagung

An dieser Stelle möchte ich mich bei Herrn Prof. Franz Schubert und Prof. Karin Landenfeld für die Überlassung dieses interessanten Themas und die intensive Betreuung der Arbeit bedanken.

Des Weiteren gilt mein Dank an allen Kollegen des Center for Demand Site Integration(C4DSI), die mich stets bei meiner Arbeit unterstützt haben.

Mein desondererDank geht an Hern Sebastian Farrenkopf für die Hilfestellung beim Programmieren, Hern Bastian Hey bei der Installation der verschiedenen Software und Hern Petrit Vuthi für die Rückfrage und das Verständnis des Themas.

Ein großes Dankeschön an meine Familie und Freunde für die moralische und die sorgliche Unterstützung während meines Studiums und der stressigen Prüfungwochen.

Contents

1 Einleitung 10
1.1 Motivation 10
2 Aufgabestellung 11
3 Aufbau der Arbeit 12
4 Grundlagen / Stand der Technik 13
4.1 Datenbank 13
4.1.1 Definition und Struktur einer Datenbank 13
4.1.2 Definition einer Datenbanksprache 15
4.2 PgAdmin III (Benutzerschnittstelle zur Kommunikation mit Datenbankserver) 17
4.2.1 Definition und Anwendung der PgAdmin III 17
4.3 Programmiersprache Python 18
4.3.1 Definition Python 18
4.3.2 Entwicklungsgeschichte 18
4.3.3 Installation und Anwendung der Programmiersprache 19
5 Anforderung 25
5.1 Aufgabestellung 25
5.2 Anforderungsanalyse 25
6 Technische Analyse 27
6.1 Datenbank C4DSI 27
6.1.1 Externe Verbindung für die Datenbank C4DSI (AnyConnect Secure Mobility Client) 27
6.1.2 Struktur und Anwerndung der Datenbank C4DSI 28
6.1.3 Pflichtspalte „date“ 29
6.1.4 Postgresql war eine Anforderung von C4DSI-Gruppe 29
6.2 Programmiersprache Python 30
6.2.1 Python war eine Anforderung vom C4DSI(Python als Anforderung vom C4DSI) 31
6.3 Entwicklungswerkzeug für graphische Benutzeroberfläche 31
6.3.1 GUI Entwicklungswerkzeug Thinker 31
6.3.2 GUI Entwicklungswerkzeug Qt Design 31
6.3.3 Die Wahl der Qt-Design 37
7 Systementwurf 38
8 Implementation und Test 39
8.1 Anmeldefenster 39
8.2 Aufbau des Navigations Tools 41
8.3 Menu Zeitintervall 42

8.4 Menu DB_Übersicht(Auswahl) im Navigation Tool 44
8.4.1 Aufbau der Menu DB_Übersicht(Auslesen der Datenbank Schemata, Tabellen und Spalten) 44
8.4.2 Datenauswahl einer Spalte oder einer Tabelle 47
8.5 Menu Löschen der Daten aus dem Navigation Tool 52
8.6 Menu Tabellenfenster 54
8.6.1 Export 54
8.6.2 Import 55
8.6.3 Grafik darstellen 56
8.7 Manipulation der Datenwerte in der Datenbank 57
8.7.1 Query Mittelwertbildung, wenn zu viele Datenwerte für Intervall vorhanden sind 58
8.7.2 Query um fehlende Datenwerte mit NANs zu füllen 59
8.7.3 Query um fehlende Datenwerte mit dem Letzten bekannten Datenwert zu ersetzen 60
9 Zusammenfassung und Ausblick 61
10 Anhang 63
11 Literaturverzeichnis 66

1. Abkürzungsverzeichnis

HAW Hamburg.....Hochschule für Angewandte Wissenschaften Hamburg

C4DSI…….....Center for Demand Side Integration

GUI…….....Graphic User Interface

TW..... Tablewiget

VNC.....Virtuele Network Computing

VPN.....Virtuelle Private Netzwerke

CSV.....Comma-Separeted Value

CMD.....Commander

GBO.....Graphische Benutzeroberfläche

DML.....Data Manipulation Language

DCL.....Data Control Language

DDL.....Data Definition Language

SQL..... Structured Query Language

DBS.....Datenbankserver

ORDBMS.....objektrelationales Datenbankmanagementsystem

IDLE .....Integrated Developement Environement

TCL .....Tool Command Language

2. Abbildungsverzeichnis

Abbildung 1: Struktur einer Datenbank mit Schemata, Tabelle, Spalte 15
Abbildung 2: Die Funktionalität der PgAdmin3 17
Abbildung 3: : IDLE-Editor Fenster 20
Abbildung 4: IDLE-Shell Fenster 20
Abbildung 5: Python(x,y) interaktive Konsole(Ipython mit Console 2.7) 21
Abbildung 6: Python Interpreter für Python Version(2.7) 22
Abbildung 7: Der Interpreter führt das Programm Stück für Stück aus. 22
Abbildung 8: Ein Compiler übersetzt Quellecode in Objektcode 22
Abbildung 9: Ausführung der Funktion „Div“ in Spyder 24
Abbildung 10: VPN-Tunnel für Verbindung der C4DSI-Datenbankserver 27
Abbildung 11: SQL-Editor um die SQL-Querys zu testen 28
Abbildung 12 : Tabelle mit Date Abbildung 13: Tabelle ohne Date 29
Abbildung 14: Kommunikation zur Datenbank C4DSI 30
Abbildung 15: Qt Designer das Fenster für die Bearbeitung der GUI 33
Abbildung 16: Aufbau einer Dialog in Qt Designer zur Multiplikation einer Zahl 34
Abbildung 17: Cmd-Fenster für die Transformation der Gui-Fenster in Python-Code 35
Abbildung 18: GUI Fenster zur Multiplizieren eines eingegeben Wert 35
Abbildung 19: Die Übersetze GUI in Python QuelleCode 36
Abbildung 20: Ordner für die umgewandte python-Dateien und die UI-Datei 36
Abbildung 21: Systementwurf zwischen Datenbank und C4DSI Navigation-Tool 38
Abbildung 22: Anmeldung-Fenster des Navigations-Tools 39
Abbildung 23: Navigation Tool 41
Abbildung 24: Auswahl des Zeitintervalls und Darstellung der Referenzspalte im Tablewidget 43
Abbildung 25: Flussdiagramme zur Abruf der Daten mit Menu DB_Übersicht 44
Abbildung 26: Auswahl einer Spalte mit der Menu DB_Übersicht 45
Abbildung 27: Ergebnis der gewählte Spalte mit der Menu DB_Übersicht 46
Abbildung 28: Flussdiagramm eines Abrufs einer gewählte Tabelle oder einer gewählte Spalte 47
Abbildung 29: Darstellung der Daten einer Spalte nach Zeitintervall mit jedem Zeilenwert 51
Abbildung 30: Das Tablewidget vor das Löschen mit einer zu löschen ausgewählte Spalte 52
Abbildung 31: Darstellung der Daten nach der löschen Betätigung mit den restlichen Spalten 53
Abbildung 32: Speicherung der Daten in der CSV-Datei mit der CSV-name Stundenwerte.csv 54
Abbildung 33: Das Importieren der Daten von gewählten CSV-Date im Tablewidget 55
Abbildung 34: Das Plotten der geklickte Spalten aus dem Tablewidget 56
Abbildung 35: Auswahl des Zeitintervalls und Darstellung der Referenzspalte im Tablewidget 57
Abbildung 36: Bildung der Mittelwert von Minutenwerte in Viertelstundenwert 58
Abbildung 37: Bildung des Einsetzen der Werte mit dem Wort NAN statt Wert in leeren Zeilen 59
Abbildung 38: Bildung des Einsetzen der Werte mit jeder vorigen Wert in leeren Zeilen 60
Abbildung 39: Flussdiagramme zur Abruf der Daten mit Menu DB_Übersicht 63
Abbildung 40: Flussdiagramme zur Abruf einer Gesamte Tabelle und für einer Spalte 64

1 Einleitung

1.1 Motivation

Zunehmend müssen Daten aus verschiedenen Gründen langfristig, kostengünstig und wiederwendbar aufbewahrt werden. Auch in Datenbanksysteme belasten die Daten, mit denen längere Zeit nicht gearbeitet wird und die von „aktikven “ Daten unterschieden werden sollen. Das Hauptproblem bei dieser anfallenden Datenmenge ist, den Überblick über die Daten gleichzeitg auf mehrere Benutzer abzurufen. Bei zunehmender Datenmenge wird es zum wiederfinden unumgänglich, die Mess-daten abzurufen und zu manipulieren. Mit dieser Konsequenz stellt sich die Arbeitsgruppe C4DSI(Center for Demand Side Integration) der Haw-Hamburg über einen auf Datenbankserver bezogenes externes Tool vor, bei dem die Daten auf der Ebene der Datenbankanwendung (Transaktionen zwischen Schnittstelle und Datenbank durch die Aufruf der Daten mit einer hergestellten graphischen Benutzeroberfläche) bestimmt werden. Im Kontext des Aufrufs der Daten von einem Datenbankserver bedeutet anwendungsorientiertes Navigieren einer Erweiterung der Zugriffschnittstelle. Die Zugriffschnittstelle geben die Möglichkeiten der Programmiersprache Python die gespeicherte Mess-daten von der C4DSI-Datenbank zu holen.

2 Aufgabestellung

Im Rahmen meiner Bachelorarbeit werden verschiedene bestehende Einzelkonzepte zu einem ganzen zusammengeführt, das sich zu einem „Navigation Tool“ für Datenbanken vereinigt. Zu diesem Tool wird die Kommunikation bzw. die Übertragung der verschiedene Mess-Daten (wie z.B. Anlagenmessdaten, Wetterdate.etc) , die in der Datenbank gespeichert sind, abgefragt sind. In der Arbeit werden die Daten von dem strukturierten Datenbankserver mit der Erstellung einer graphischen Benutzeroberfläche abgerufen, visualisiert und in einer CSV-Datei gespeichert oder in einer graphischen Kurven nach mehreren vorgestellten Optionen darsgestellt.

Die Hauptanforderungen sind dabei, dass das Datenbanksystem eine postgresql Datenbank ist und das „Navigation Tool“ mit der Programmiersprache Python entwickelt werden soll. Bei der Realisierung des Navigation-Tools soll der Benutzer die Möglichkeit haben, die Daten der Datenbank ohne Kenntnisse der Datenbanksprache abzurufen.

Die dazu notwendigen Softwarepakete sollen analysiert und beschrieben werden.

3 Aufbau der Arbeit

In Diesem Teil werden die Kapitel meiner Arbeit vorgestellt.

Im Kapitel 1 (Grundlagen/Stand der Technik) werden

- Grundbegriffe beschrieben
- Grundlagen der Kommunikation der Datenbank mit einer Datenbanksprache erläutert.
- die Umsetzung der Qt-Designer-Herstellung in Python-Skript zusammen in Kommunikation mit der Datenbankserver beschrieben.

Danach folgen im Kapitel 2 (Anforderungen) und Kapeitel 3 (Technische Analyse) mit der Bechreibung der Anforderungen und Beschreibung einer technischen Analyse des gesamten Systems.

Im Kapitel 4 (Systementwurf) wird den Systementwurf vorgestellt.

Im Kapitel 5 (Implementation und Test) wird eine Implementation des Systems erläutert und getestet.

Eine Zusammenfassung wird meine Arbeit im Kapitel 6 beschrieben und mögliche Erweiterungen gegeben.

4 Grundlagen / Stand der Technik

In diesem Kapitel sollen die Grundbegriffe, die zur meiner Ausarbeitung gehören, eingegangen werden. Dies Beinhalten eine detaillierte Erklärung der Anwendung der Software Python als Programmiersprache und die Entwicklungswerkzeug Qt-Designer. Zudem werden die Grundlagen der Datenbank, der Datenbanksystem, die Kommunikation der Datenbank mit einer Datenbanksprache erläutert und werden die Umsetzung der Qt-Designer-Herstellung in Python-Skript zusammen in Kommunikation mit der Datenbankserver beschrieben.

4.1 Datenbank

4.1.1 Definition und Struktur einer Datenbank

Eine Datenbank ist eine Sammlung von Daten, Tabellen, Spalten und Objekten. Sie ermöglicht die Daten zu hinterlegen. Es gibt mehrere Datenbanke um die Daten zu speichern. Diese Datenabanke sind folgende: PostgreSQl, SQL, MYSQL und Oracle. Meine Arbeit bezieht sich nur auf die Datenbank PostgreSQL. Die Datenbank soll in einer Datenbankserver aufgebaut werden. Ein Datenbankserver ist eine Plattform der Datenbank, die nach einem speziellens Anmelde-ID(Identifikationsnummer oder Benutzername)/ Kennwort-Paar für die Berechtigung der Benutzer anfordern wird.
Die Struktur einer Datenbank ist in drei Ebenen unterteilt, dieser sind:

- Schemata:

  Ein Schemata(auch Instanzen genannt) ist ein Speicherstruktur, der in der Datenbank die Objekte erstellen, exportieren oder importieren kann. In einen Schema lassen sich Tabellen erstellen und verwalten. Ausserdem lassen sich für Schema Datenbankfunktionen sowie Triggerfunktionen speichern. Die Anzahle für mehrere Schemas sind in der Datenbank nicht begrenzt. Die Konfiguration für das erstellende Schema gibt das Recht nur zu lesen und nicht zu schreiben. Mit den folgenden Befehlen lässt sich ein Schema mit dem Namen Leiwartenarchiv anlegen.

```
-- Schema: leitwartenarchiv
-- DROP SCHEMA leitwartenarchiv;
CREATE SCHEMA leitwartenarchiv
```

- Tabelle:

Eine Tabelle ist eine zweidimensionale Struktur der Datenbank, in der Daten gespeichert werden. Diese Struktur besteht aus Spalten und Zeilen. Jedes Element(Spalte) in der Tabelle ist mit einem Datentyp definiert. Jede Tabelle kann bis zu 32 TeraByte(TB) umfassen und die Anzahl der Datensätze ist unbegrenzt. Damit eine Tabelle angelegt werden kann, muss das zugehörige Schema mitangeben werden.Ein Beispiel mit dem folgenden Befehl zeigt wie eine Tabelle gegründet wird und welche Spalten mit den Datentypen diese Tabelle beinhaltet.

```
-- Table: leitwartenarchiv.zenonempfangen
-- DROP TABLE leitwartenarchiv.zenonempfangen;
CREATE TABLE leitwartenarchiv.zenonempfangen
(
  "ID" serial NOT NULL,
  "NAME" character varying(45) NOT NULL,
  "DATUMZEIT" timestamp without time zone NOT NULL,
  "ZEIT_MS" smallint NOT NULL,
  "WERT" character varying(50) NOT NULL,
  "STATUS" integer NOT NULL,
  "ACK_SRV" integer,
  "ACK_SB" integer,
  "INSERTZEIT" timestamp without time zone,
  CONSTRAINT zenonempfangen_pkey PRIMARY KEY ("ID")
)
```

- Spalte:

Eine Spalte ist ein Speichort für den Datensatz. Jeder Spalte besitzt seinen eigenen Datentyp, damit ein Datensatz eindeutig definiert werden kann zu einem bestimmten Zeitpunkt muss ein Primary Key (Primär schlüssel) definiert werden. (Sie hierzu das vorherige Beispiel). Der Datentyp definiert für die Spalte den zulässigen Datenwert und beschreibt die Art der Daten oder des Inhalts. Das bedeutet in einer Spalte sind nur die Werte zulässig, die diesem Typ entspricht. Die Spalte stellt die dritte und letzte Ebene in der Datenabank dar. Die Anzahl der Spalten und deren Reifenfolgen sind fest. In einer

Tabelle können maximale Anzahl 1600 Spalten angelegt werden . Spalten können bis zu 1GB Daten.

Mit dieser folgenden Abbildung können die Daten in der Datenbank schritt für schritt hintergelegt werden. Um dieser Daten zu haben sollte der Benutzer mit Hilfe der SQL-Datenbanksprache nach Spalte, Tabelle und Schema abfragen.

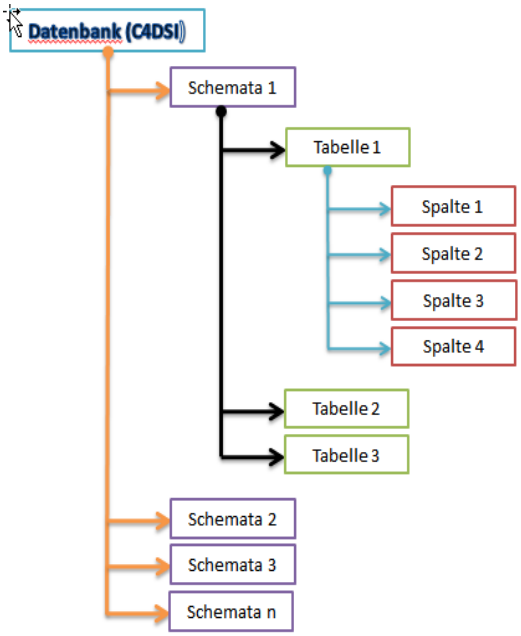


Abbildung 1: Struktur einer Datenbank mit Schemata, Tabelle, Spalte

4.1.2 Definition einer Datenbanksprache

Um die Datenbank ständig zu kommunuzieren, braucht der Benutzer eine Datenbanksprache. SQL ist die Abkürzung für strutured Quers language und heißt übersetzt „strukturierte Abfragensprache“. SQL dient in Datenbanksystemen zur Definition von Daten und zur Informationsgewinnung. Das Konzept relationaler Datenbank basiert auf mathematischen Ansätzen aus den frühen siebziger Jahren. Grundprinzip ist, dass die Daten in Form von der Tabelle gespeichert werden, die logisch miteinander verknüpft sein können. In diesem Zusammenhang wurde auch die SQL-sprache entwickelt, um auf diese Daten zugreifen zu können. Die Hauptaufgabe von SQL-Ausdrücken ist das lesen oder Verändern von vorhandenen Daten oder das Hinzufügen von Daten in die Datenbank. Die Beherrschung der SQL-Befehlssyntax ist deshalb für einen effektiven Umgang mit SQL-Datenbanken

unverzichtbar. SQL verfügt allgemein über Kommandos zur Datenbearbeitung, die so genannte Data Manipulation Language(DML), und Kommando, mit denen das Datenbankdesign definiert bzw. Eine Datenhilfesprache besteht aus zwei Teilen: eine Datendefinitionssprache(Data Definition Language) und einer Datenbearbeitungssprache(Data Manipulation language):

- Datendefinitionssprache(Data Definition Language -DDL):
  Die DDL ist eine Datenbanksprache, die es der Datenbankarchitektur ermöglicht, die Entitäten zu beschreiben, die für die Anwendungen und die Beziehungen benötigt werden. Das Datenbankschema wird mit Hilfe einer der DDL, die zur Beschreibung eines neuen oder zur Modifikation eines bestehenden Schemas benutzt, geschrieben werden. Die Datenbankanweisungen werden mit der logische Struktur der Datenbank des Schema, der Tabelle mit folgenden Befehle beschrieben oder verändert.
  CREATE SCHEMA : neue Basisschema erstellen
  CREATE TABLE: neue Basistabelle erstellen

- Datenbearbeitungssprache (Data Manipulation Language- DML)
  Die DML (auch kurz genannt) ist eine Sprache, die eine Reihe von Funktionen zur Unterstützung der grundlegenden Datenbearbeitungsoperationen für die in der Datenbank enthaltenen Daten bereitstellt. Die Datenbearbeitungsverfahren beinhalten normalerweise folgenden Funktionen:
  * Das Speichern von Daten in der Datenbank
  * Die Modifikation der gespeicherten Daten in der Datenbank
  * Das Löschen von Daten in der Datenbank
  * Die Abfrage von Daten aus der Datenbank
  Die folgende Befehle gehören zur Verarbeitung der Daten an das Datenbanksystem:
  SELECT: Daten aus der Datenbank lesen
  DELETE: Zeilen einer Tabelle Löschen

4.2 PgAdmin III (Benutzerschnittstelle zur Kommunikation mit Datenbankserver)

4.2.1 Definition und Anwendung der PgAdmin III

PgAdmin III ist eine Open-Source-Software zur Entwicklung und Administration von PostgreSQL-Datenbanken und abgeleiteten Datenbanken. Eine graphische Benutzeroberfläche ermöglicht die Erleichterung der Administration von der Datenbanken und ermöglicht durch eine native Anbindung an PostgreSQL den Zugriff auf die gesamte PostgreSQL-Funktionalität. Der PgAdmin III ist als das populärste Open-Source Administrationswerkzeug für PostgreSQL. Die Software wird in Versionen für mehrere Betriebssysteme(Linux, Mac OS X, FreeBSD, Solaris, Windows) angeboten. Die folgenden Abbildung zeigt die Struktur der Benutzeroberfläche von PgAdmin III :

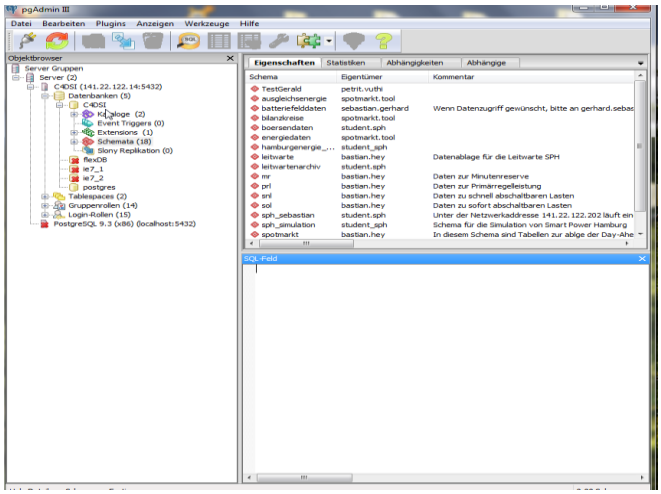


Abbildung 2: Die Funktionalität der PgAdmin3

Der PgAdmin III enthält eine graphische Verwaltungsoberfläche, ein SQL-Query-Werkzeug, einen SQl-Editor für prozeduralen Code und weietere Werkzeuge, mit denen der Datebankbenutzer seine Entwicklungs und Vervaltungsaufgaben auf einfache Weise erledigen kann. Die linke Seite des Hauptfenster, mit dem Titel Objektbrowser, zeigt die Hierachiestruktur des Server mit allen seinen Datenbankobjekten. In diesem Objektbaum ist es zu sehen, welche Datenbanken, Nutzergruppen der Server verwaltet. Es ist möglich neue Datenbanken, Nutzer oder Gruppen anzulegen und vorhanden Tabellen, Tabellen anzusehen oder anzulegen. Die Obere rechte Seite der Abbildung zeigt, wie z.B Eigentümer,

Objektidentifikator(OID), Kodierung einer ausgewählten Objektkategorie in der Objekthierarchie. Der Untere rechte Teilbereich, das SQL-Feld, enthält die SQL-Skripte.

4.3 Programmiersprache Python

4.3.1 Definition Python

Python ist eine objektorientierte Programmiersprache und wird die interpretierte Skriptsprache zugeordnet. Python ist für eine höhere Programmiersprache, die von denen anderen Programmiersprache(C , C++, Perl und Java) bereits gehört. Der Name Python verdankt nicht der gleichnamigen schlangenart, auch wenn diese inzwischen zu ihrem Symbol geworden ist, sondern die englische Komikertruppe Monty Python. Sie unterstützt mehrere Programmiersprache, zum Beispiel: die objektorientierte, die Aspektorientierte und die funktionale Programmierung. Es bietet ferner eine dynamische Typierung und wird von Python mehrere dynamische Sprachen oft als Skriptsprache genutzt. In dieser Programmiersprache gibt es eine offene, gemeinschaftsbasiertes Entwicklungsmodell, das durch die gemeinnützige Python Software Foundation, die von der Definition der Sprache in der Referenzumsetzung Cpython pflegt, gestützt wird. Als einfach zu erlernende Sprache, verfügt eine Klare und übersichtliche Syntax. Sie besitzt auch eine umfangreiche Standardbibliothek und zahlreiche im Python Package Index.

4.3.2 Entwicklungsgeschichte

Angefangen wurde ihre Entwicklung 1990 von Guido van Rossun am CWI(Centrum Wiskunde & Informatica) in Amsterdam. Python wurde ursprünglich in der Betriebsystemsforschung eingesetzt. Nach mehreren Jahren in der Forschung wurde die ersten Vollversion unter der Bezeichnung Python 1.0 im Januar 1994 veröffentlicht. Python 2.0 erschien am 20.Oktorber 2000 mit neuen Funktionen, die eine vollfunktionsfähige Automatische Speicherbereinigung und die Unterstützung für den Unicode-Zeichensatz umfassten. In Version 2.6 wurde eine Hilfe eingebaut, mit der angezeigt werden kann.

4.3.3 Installation und Anwendung der Programmiersprache

Python ist einfache und erlernende Programmiersprache. Die Installation diese Software konnte sehr hilfreich für mehrere Benutzer. Da die Lizenz
dieses Software frei ist, konnte die Installation sehr leicht ausführbar sein. Die C4DSI-Hompage gibt die Möglichkeiten direkt als ein Komplettes Paket zu downloaden. Aber es bietet nicht nur eine Option sondern mehrere Varianten Python zu haben. Hier wird eine Version von Python(x,y) haben, die einfach Zugriff auf die PostgreSQl-Datenbank und Qt-Designer mittels der Moduls Pyscopg2 ermöglicht. Python(x,y) besteht aus vier Teilen, die automatisch mitinstallieren wurden. Diese vier Teile konnten direkt nacheinander ausgeführt werden. Spyder wird als großer Teil zu beschäftigen von der vier Teile. Meine Python-Code wird in Spyder geschrieben, bearbeitet und ausgeführt. Über die Anmeldung auf der C4DSI-Hompage ist sicherer die Installation als auf Google. Da das volle Version Python soll komplett mit seinen Schnittstelle herunterladen. Mit einem Klick auf diesem folgenden Link wird Python für das komplette Paket heruntergeladen:
Python(x,y)-2.7.10.0.exe

In der nächste Abschnitt werden die vier Teile der Python(x,y) detaillieren:

4.3.3.1 IDLE(Integrated Development Environment)

IDLE ist eine integrierte Entwicklungsumgebung für Python, die mit der Standardimplementierung der Sprache gebündelt wurde. als vollständige in Python, ist es der schriftlichen Thinker GUI-Toolkit und als optionales Teil der verspackte Python-Verspackung. IDLE soll eine für Anfänger geeignet, vor allen in einem pädagogischen Umfeld. IDLE ist leistungsfähig genug, um bearbeiten, ausführen kurze Skripte und bestehst aus mehrere Teilen(Grundzüge):

- Mehrfenster-Text-Editor mit Syntax-Hervorhebung, Auto-Vervollständigung , smart Gedankentisch.
- Python-Shell mit Syntax-Highligting
- Integrierter Debugger mit Schritt, anhaltende Haltepunkte und der Call-Stack Stickbarkeit.

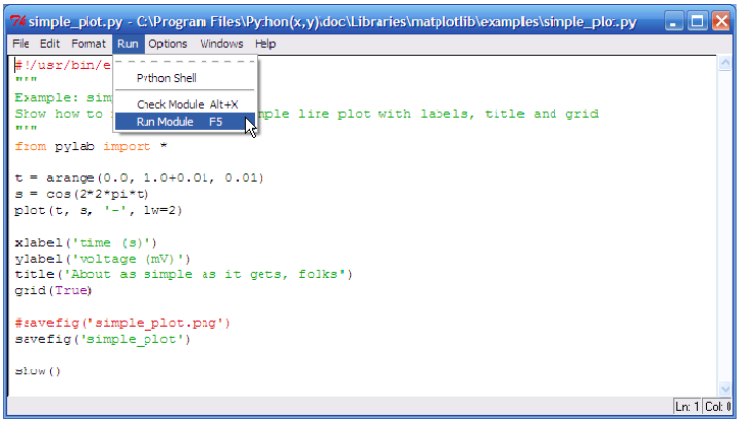


Abbildung 3: : IDLE-Editor Fenster

4.3.3.2 IDLE Shell-Fenster

IDLE Shell wird auch Python Shell genannt. Es bietet einen Python-Interpreter um interaktiv Code auszuführen. Im Vergleich mit der Matlab IDE, scheint IDLE Python sehr beschränkt zu sein. Der Zweck ist eine einfache effiziente IDE mit Programmiersprache Python zu beginnen.

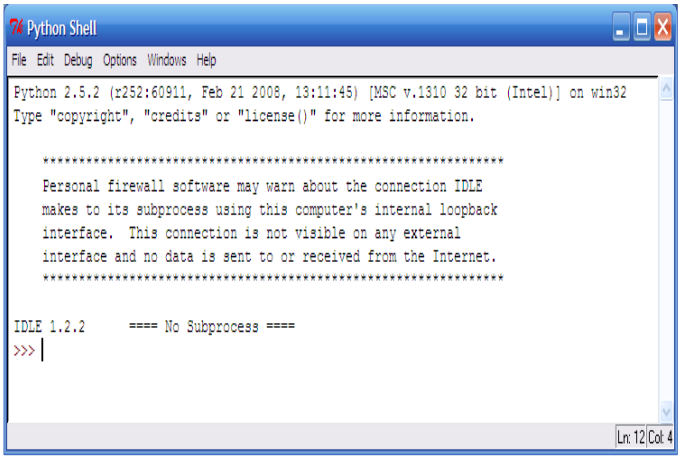


Abbildung 4: IDLE-Shell Fenster

4.3.3.3 Ipython

Ipython ist eine verbesserte Python-Shell, in der matplotlib Funktionen sind durch die verfügbaren Pylab-Schnittstelle, die eine Syntax sehr nah an MATLAB bietet. Ipython mit matplotlib, numpy und Scipy Unterstützung von seinem Startmenu-Eintrag(„Python-

Matplotlib console“) oder von einer Datei-Ordner mit der rechten Maustaste darauf durchgeführt werden.

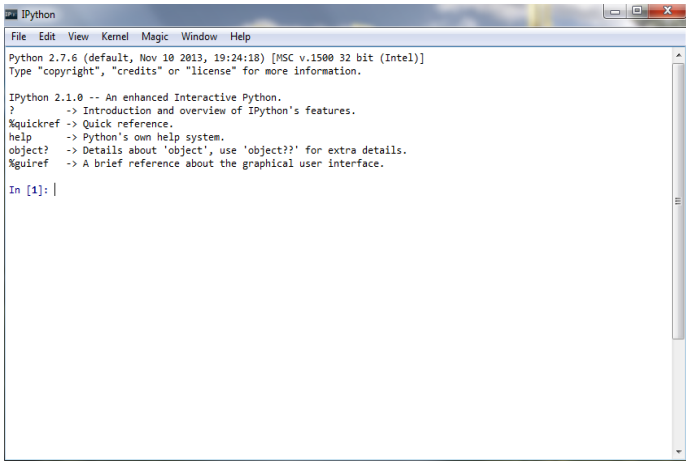


Abbildung 5: Python(x,y) interaktive Konsole(Ipython mit Console 2.7)

Python als höhere Programmiersprache , hat für die Bearbeitung zwei Arten von Programmen erforderlich: Interpreter oder Compiler. Ein Interpreter liest ein in einer höheren Programmiersprache geschriebenes Programm und führt es aus was das Programm tun soll. Der Interpreter führt das Programm dabei Stück für Stück aus, liest immer wieder Zeilen und führt die entsprechenden Berechnungen durch. Zwei Möglichkeiten, um den Interpreter zu verwenden, sind:

- **Interaktive Modus**

  Im Interaktiven Modus werden viele Python-Progemmen eingetippt , und das Ergebnis wird direkt durch den Interpreter gezeigt. Die Eingabeaufforderung erfüllt immer die Anforderungen der Python-Interpreter. Die Eingabaufforderung signalisiert immer, dass es bereit ist.

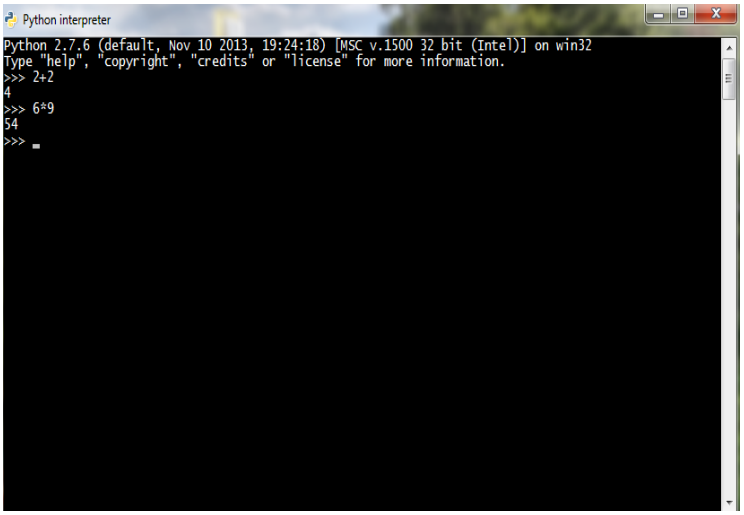


Abbildung 6: Python Interpreter für Python Version(2.7)

- **Skript Modus**

Der Interpreter führt immer den Inhalt einer Datei, die als Skript oder Skriptmodus bezeichnet. Die Namen von Python-Skripten enden nach den Konvention immer mit .py um diese Skripten auszuführen.

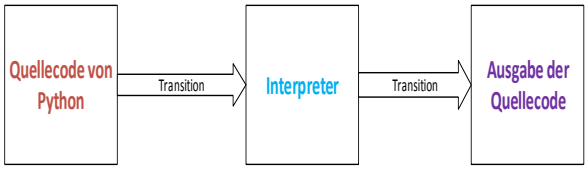


Abbildung 7: Der Interpreter führt das Programm Stück für Stück aus

- Compiler

Ein Compiler liest das gesamte Programm ein und übersetzt es vollständig, bevor es ausgeführt werden kann. In diesem Zusammenhang bezeichnet man das Programm in der höheren Programmiersprachen als Quellcode und das übersetzte Programm als Objekt-Code beziehungsweise ausführbare Datei.

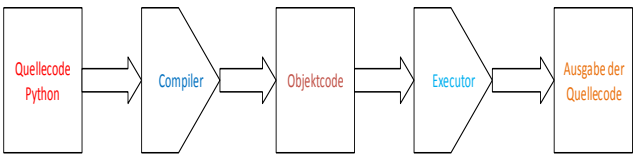


Abbildung 8: Ein Compiler übersetzt Quellecode in Objektcode

4.3.3.4 Spyder

Spyder ist ein Open-Source- Cross-Plattform-IDE für wissenschaftliche Programmierung Python(oder genannt eine wissenschaftliche Entwicklungsumgebung für Python). Als Inspiration von Maltab IDE soll Spyder eine Migration von Matlab zu Python zu erleichtern. Im Vergleich zu anderen IDE s für die Wissenschaftliche Entwicklung Spyder hat einen einzigartigen Open-Source, der in Python nicht Copyleft –Lizenz geschrieben hat und zur Verfügung steht. Spyder arbeitet mit der integrierten Software(Numpy, Scipy, Matplotlib und Ipython) und besteht aus Vier Elementen:

- Project Explorer:
  Project Explorer spielt eine große Rolle um viele erstellete Projekte und Module eine sichere Speicherort zu haben und direkt anzusehen oder zuzugreifen.

- Editor:
  Nach dem das esrtelle Projekte und Module gespeichert sind, kann der Benutzer in der Editor seine QuelleCode schreiben.

- Variable Explorer:
  Variable Explorer ermöglicht den Programmierer die Inforamtionen über die Variable, die in der Editor geschrieben wurden, zu haben.

- Console:
  Mit diesem Fenster lassen sich die Quelle Code testen und die Endergebnisse sehen. Die Console zeigt auch den Fehler wenn Spyder nicht die Quelle Code ausführen kann.

Die folgende Abbildung zeigt die Fünktionalität der Spyder Mit einer Projektname „Division“ und seiner Module „division_zwei_zahlen“. Hier wurde eine Funktion (div) in der Editor geschrieben. Bei Ausfürung der Funktion „div“ zeigt deutlich das Endergebnis in der Console.

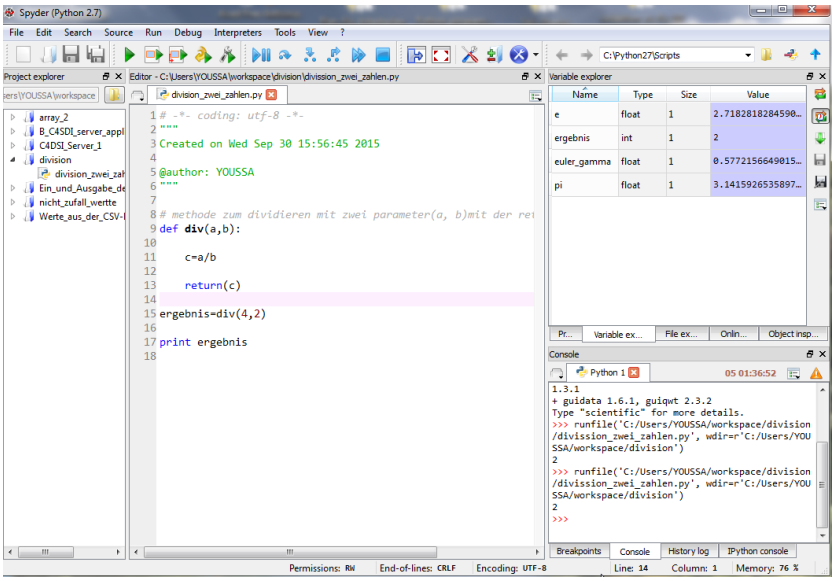


Abbildung 9: Ausführung der Funktion „Div“ in Spyder

5 Anforderung

Im vorherigen Kapitel wurden die Grundlagen näher erläutert, die für die Arbeit notwendig sind. In diesem Kapitel sollen die Anforderungen näher beschrieben werden

5.1 Aufgabestellung

Ziel der Arbeit ist es eine GUI zu erstellen, mit der es möglich ist ohne SQL Kenntnisse die Mess-und die Simulationsdaten von der C4DSI Datenbank abzurufen, zu visualisieren und als CSV-Datei zu exportieren. Zusätzlich sollen die Mess-und Simulationsdaten manipuliert werden können. Das Bedeutet zum einen, dass das Zeitintervall in Minute, Viertelstunden, Stunden und Tag ausgewählt werden kann. Zum anderen müssen wegen dem Zeitintervall die Daten manipuliert werden. Wenn in einem Zeitintervall mehr Daten vorhanden sind, muss ein Mittelwert für das Zeitintervall gebildet werden. wenn in einem Zeitintervall weniger oder gar keine Werte vorhanden sind, muss dieser aufgefüllt werden. Zum auffüllen von fehlenden Daten sind zwei Optionen vorgesehen. Die Erste ist, den fehlenden Wert mit NAN(Not a Number) zu kennzeichnen und die zweite ist den letzten bekannten Wert wieder zu verwenden. Die GUI soll mit Modularenfunktionen und in der Programmiersprache Python entwickelt werden.

5.2 Anforderungsanalyse

Das System soll die gespeicherte Mess- und Simulationsdaten durch die erstellenden graphischen Benutzeroberflächen abrufen und visualisieren. Zudem konnten die Python-Skript als Programmieren-Skript erstellt werden. Die Programmiersprache-Skripte sind als Schnittstelle für die selbst in Qt-Designer aufgebaute graphische Benutzeroberflächen zur Verbindung der C4DSI-Datenbank, wo die Daten gespeichert sind.
Mit Hilfe der Bibliothek in Python, werden die in Qt-Designer übersetzen Klassen zur den GUI gehören, in Python importiert und getestet. Die programmiersprache-Skripte besorgen die Abruf der Mess- und Simulationsdaten in der graphischen Benutzeroberfläche. Die GUI

sollen nicht nur die Mess-und Simulationsdaten auf gewähltem Zeitintervall anzeigen lassen, sondern in CSV-Format speichern und nach Zeitintervall in mehrere Kurven verlaufen. Der gewählten Zeitintervall führen nicht gleich an die gewünschten Daten in der GUI anzuzeigen. Deshalb bei entsprechenden Daten sollen die Mittelwert durch die geschriebene Querys in der SQL-Datenbank bilden. Die beiden Optionen sollen nicht die leere Zeile nur ergänzen. Bei fehlenden Daten, muss der Benutzer das Navigation-Tool nicht nur die beiden Optionen haben. Der Navigation-Tool soll zu dem Anwender andere Optionen geben. Der fehlende Daten in der Leeren Zeilen sollen mit NAN ersetzen um nichts bezeichnen oder Mit Vorigen Wert um die vorigen Wert zu nehmen.

6 Technische Analyse

Nachdem die Anforderungen und die Aufgabenstellung erläutert wurden, soll in diesem Kapitel eine technische Analyse des gesamten Systems beschrieben werden.

6.1 Datenbank C4DSI

Die Datenbank C4DSI befindet sich in einer erstellte Datenebankserver in PgAdmin III. Um die Datenbank zu kommunizieren muss eine VPN-Verbindung erstmal gebraucht werden.

6.1.1 Externe Verbindung für die Datenbank C4DSI (AnyConnect Secure Mobility Client)

Der Datenbankserver liegt im Rechenzentrum der HAW-Hamburg und smoit in einem privaten Netzwerk. Damit der Datenbankserver von extern, sowie auch vom internen HAW-WLAN angesprochen werden kann, muss eine VPN-Verbindung geöffnet werden. Herzu wird das Programm AnyConnect Secure Mobility Client verwendet. Das Programm ist kostenlos von der HAW Webseite zu beziehen. Die Anmeldung erfolgt über die Benutzerdaten der HAW, die für auch für das Web-Mail Programm genutzt werden. die folgende Abbildung zeigt das Anmeldefester vom AnyConnect Secure Mobility Client:

Abbildung 10: VPN-Tunnel für Verbindung der C4DSI-Datenbankserver

6.1.2 Struktur und Anwerndung der Datenbank C4DSI

In der PgAdmin III befindet zwei Datenbankserver(Datenbankserver C4DSI und Datenbankserver PostgreSQL9.3), die gleiche Localhost haben. Der Datenbankserver C4DSI beinhaltet In der Objektbrowser fünf Datenbanken. Die Datenbank C4DSI ist aus dem relationalen Datenmodell mit Hilfe von SQL aufgebaut worden und besteht aus Schemata, Kataloge und Extensions:

- Kataloge:
  Die Kataloge enthalten das Informationsschema, alle Systemkatalogue von PostgreSQL und zeigen die genutzten SQL-Standards der Datenbank.

- Extensions:
  Gründen eine Extension um eine Verlängerung in eine Datenbank laden.

- Schemata:
  Die Schemata enthalten die verschiedenen Datenbankobjekte wie zum Beispiel: Tabellen, Sequenzen, Funktionen, Triggerfunktionen und Sichten. Die Datenbank C4DSI besteht aus 18 Schemata. Die Schemata beinhalten nicht nur die Mess- und Simulationsdaten sondern auch Energiepreis- und Wetterdaten.

6.1.2.1 SQL Editor

Die SQL Editor ermöglicht die SQL_querys mit viele Befehle der Datenbanksprache zu testen. Die folgende Abbildung zeigt ein Test einer SQL-Query.

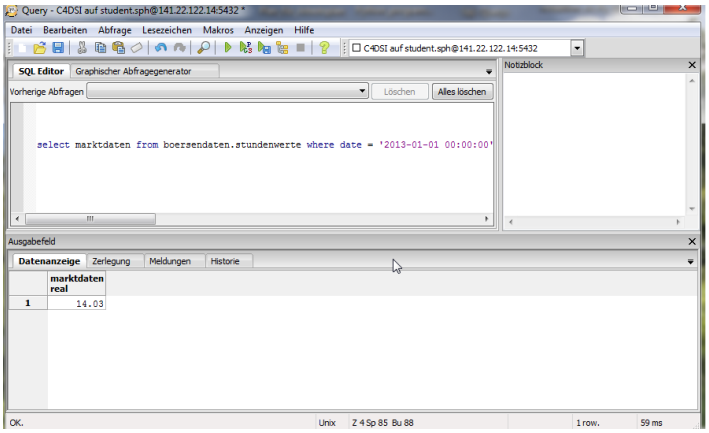


Abbildung 11: SQL-Editor um die SQL-Querys zu testen

6.1.3 Pflichtspalte „date“

Größenteils sind alle Daten in der Datenbank mit einem Zeitstempel versehen. Hierbei fällt auf, dass die Datumsspalten in den jeweiligen Tabellen unterschiedliche Spaltenbezeichnung haben. Damit besser die Daten in den Tabellen gesucht werden kann, muss eine Einheitliche Struktur in den Tabellen vorgegeben werden. Hierfür wurde mit den Mitarbeitern des C4DSIs gesprochen und entschieden, dass für den Zeitstempel die Spaltenbezeichnung „date“ mit dem Datentyp Timestamptz(TIMESTAMP with Time Zone) eingeführt und festgelegt wird. Der Datentyp Timestamptz wurde gewählt, weil dieser eindeutig die Zeitumstellungen kennzeichnet. Die beiden folgenden Abbildungen zeigen zwei Tabellen die mit und ohne der Spalte „ date“ sind aus der Datenbank vom C4DSI.

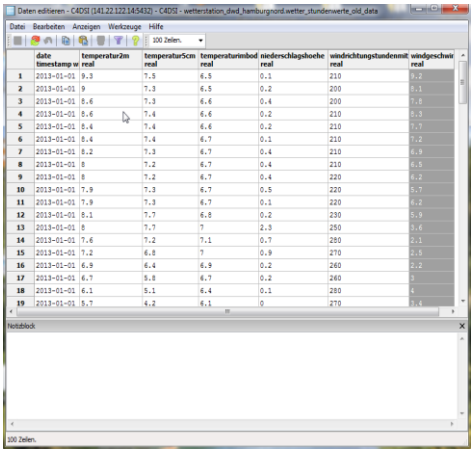


Abbildung 12 : Tabelle mit Date

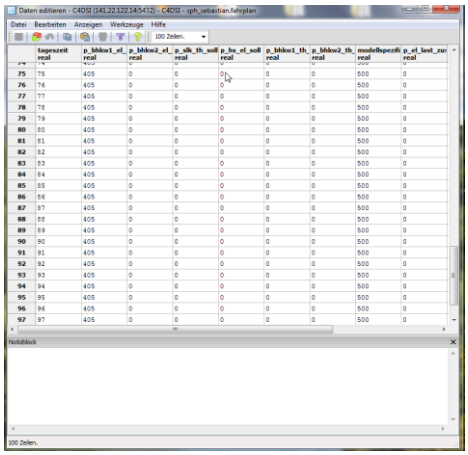


Abbildung 13: Tabelle ohne Date

6.1.4 Postgresql war eine Anforderung von C4DSI-Gruppe

Die objektrelationalen Datenbanken sind jedoch nicht zur direkten Abbildung von Objekten der Programmiersprache vorgesehen. Zur Nutzung des Konzepts der Vererbung bei Definition und Abfrage von Tabellen werden ähnliche Feldstrukturen genutzt. Damit die Datenbank für den Menschen lesbaren und in der verständlichen Form vorliegen, wurde mit den Mitarbeitern des C4DSIs gesprochen und entschieden mit Posgresql zu arbeiten.Postgresql vereinfacht die Handhabung von objektorientierten Datenbanken und erlaubt es den Benutzer das System um selbstdefinierte Datentypen. Operatoren und Funktionen zu erweitern.

6.2 Programmiersprache Python

Python ist eine objektorientierte Programmiersprache und wird die interpretierte Skriptsprache zugeordnet. Es gibt mehrere Versionen(1.1, 1.2. 2.0, 2.6, 2.7, 3.5) zu installieren. Um die Datenbank ständig abzufragen, wurde die Version 2.7 und die Schnittstelle Psycopg2 für meine Arbeit empfohlen. Psycopg2 ist die beliebteste PosgreSQL-Datenbankadapter für die Programmiersprache Python. Psycopg2 lässt sich mit folgendem Befehl über die Kommandozeile(Cmd) installiert:

$pip install psycopg2

Die folgende Abbildung zeigt ein Programm in Spyder mit der Kommunikation zur Datenbank C4DSI erzeugt wird:

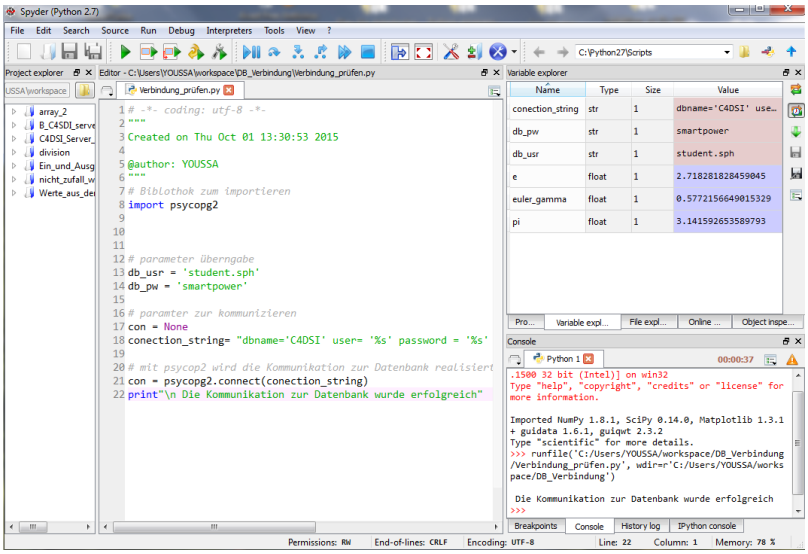


Abbildung 14: Kommunikation zur Datenbank C4DSI

Befor die Kommunikation stattfinden kann, muss Psycopg2 importiert werden. Im nächsten Schritt werden die Benutzerdaten für die Datenbank in Variablen angelegt. Mit „con= psycopg2.connect(connection_string)“ wird die Kommunikation realisiert. Über die „Console“ (rechts unten in der Abbildung) wird angezeigt ob die Verbindung erfolgreich war. In dem „Variable Explorer“ (lonks oben in der Abbildung) werden die verschiedenen Variablen mit ihren Datentypen angezeigt.

6.2.1 Python war eine Anforderung vom C4DSI(Python als Anforderung vom C4DSI)

Das C4DSI hat früher viel mit dem Programm Matlab/Simulink gearbeitet. Durch die hohen Lizenzkosten, die Matlab fordert, wurden in der Arbeitsgruppe nach alternativen gesucht. Hierbei wurden Python entdeckt, was eine ähnliche Programmiersprache wie Matlab ist. Der größte Vorteil von Python ist, dass es kostenlos und viele Module frei zugänglich sind. Python bietet mit dem Modul Psycopg2 einen einfachen Zugriff auf die PostgreSQL Datenbanken. Aus diesem Grund war gefördert, dass die Arbeit mit Python realisiert wird.

6.3 Entwicklungswerkzeug für graphische Benutzeroberfläche

Die Implementierung einer graphischen Benutzeroberfläche wird mit Hilfe einer Bibliothek auch Toolkit genannt, durchgeführt. Es gibt verschiedene Toolkits, die mit Python verwendet werden können. Unter einem Toolkit versteht man eine Bibliothek, mit deren Hilfe sich Programme mit graphischer Benutzeroberflächen erstellen lassen. Diese Toolkits sind zumeist für C(GtK) oder C++(Qt, wxWidgets) geschrieben, lassen sich jedoch durch sogenannte Bildings auch mit Python ansprechen. In folgende Teil werden die wichtigsten python-Blindings für GUI-Toolkits aufgelistet:

6.3.1 GUI Entwicklungswerkzeug Thinker

Das Toolkit TK wurde ursprünglich für die Sprache Tcl (Tool Command Language) entwickelt und ist das einzige Toolkit, in das in der Standardbibliothek Python enthalten ist. Das Modul Thinker(Tk interface) erlaubt es, TK-Anwendungen zu schreiben, und bietet damit eine interessante Möglichkeit, kleinere Anwendungen mit einer graphischen Benutzeroberflächen zu verstehen.

6.3.2 GUI Entwicklungswerkzeug Qt Design

Bei Qt handelt es sich um ein umfassendes Framework, das von der norwegischen Firma Qt Software (ehemals Trolltech) entwickelt wird. Es bietet ein GUI-Toolkits, dass viele GUI-Funktionalität enthält. Das durch und durch objektorientierte C++ Framework ist die Basis der freien Desktop-Umgebung KDE(K Desktop Environment)

Qt ist eine Klassenbibliothek für die Objektorientierte Entwicklung graphischer Benutzeroberflache in mehreren Programmiersprachen. Es ist sehr lauffähig, verfügbar für verschiedene Betriebssysteme (Windows, Linux und MAC) und natürlich sehr geeignet, um Applikationen zu entwickeln. Qt ist ein Open-Source Entwicklung, die Lizenz ist frei und ohne Kosten für die Verwendung. Komplexe Dialoge werden problemlos mit Hilfe der Stellung Qt-Designer editiert. Qt-Designer ist ein komfortabler Editor um graphische Benutzeroberflächen zu designen. Die Beschreibung der graphischen Oberfläche enthält keine Logik und es ist nicht erforderlich, auch wenn es möglich ist, diese Manuel zu programmieren. Mit der erstellten Oberfläche von Qt-Designer wird durch die Qt-Bibliothek an mehrere Programmiersprachen (Python, C++)gebunden. Die verwendeten Qt Elementen Variablen und Werte werden zugeordnet, wenn die Oberfläche erstellen wird. Das folgendes Bild zeigt die Bearbeitung der graphische Oberfläche mit alle Elemente, die man benutzen kann.

6.3.2.1 Installation und Anwendung des Entwicklungswerkzeug

Die Installation-Pakete werden in Python Ordner unter der PyQt(Python-Qt) abgelegt. Es gibt mehrere Varianten um Qt-Designer zu installieren:

- Die Installation des Qt-Designer durch Ubuntu mit folgender Befehl realisiert:
  Sudo apt-get install qt4-designer

Mit diesem Befehl erfolgt unter Unbuntu ein automatischer download mit Installation

- Es wird auf der Homepage von PyQt4 die Quellecode-Pakete nach dem Betriebssystem und nach Anzahl der Bit für Setup installiert:
  PyQt-win-gpl-4.11.4.zip für das Betriebssystem Windows
  PyQt4-4.11.1-GPL-PL-Py3.4-Qt4.8.7-x64.exe für Windows-64 Bit-Installationsprogramm

Nach dem Klicken der Benutzer zum Starten sieht der Qt Designer in folgenden Abbildung zur Bearbeitung der GUI aus:

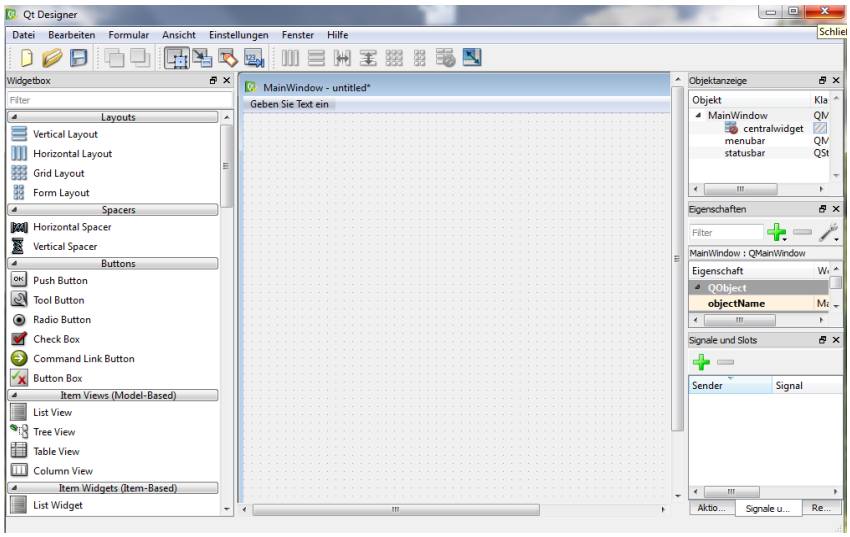


Abbildung 15: Qt Designer das Fenster für die Bearbeitung der GUI

Nach der Installation wird Qt-Designer als eigenständige Anwendung gestartet, erscheint ein Assistent, über den verschiedene Fenster und Dialoge anhand von Vorlagen erstellt werden können:

- Widgetbox:
  Um die Objekte in der Werkzeugleiste in Widgetbox ausgewählt werden zu können.
  Schnelle Dialoge bzw. Oberflächen können als Vorlagen von dem Designer ausgewählt werden.

- Objektanzeige:
  Hier werden die gewählten Objekte und ihre Klasse angezeigt. Jeder Objekt gehört zur jeder Klasse.

- Aktionseditor:
  Um die Signals and Slots zu aktivieren.
  Elemente(oder Aktionen) müssen miteinander verknüpft werden. Ein Element sendet ein Signal, woraufhin der verbundene Slot(Funktion) eines anderen Elements

aufgerufen wird. Diese Verknüpfungen können teilweise mit Hilfe des Qt-Designers vorgenommen werden.

- Eigenschaften-Fenster:
  Über einen Editor können Eigenschaften der Elementen bearbeitet werden.

- Bearbeitung-Fenster:
  Für die Erstellung einer Benutzeroberfläche mit Qt-Designer empfiehlt es sich Layouts zu verwenden. Die Objekte werden korrekt dargestellt, sobald ein Fenster vergrößert oder verkleinert wird und die Anwendung auf verschiedene Plattformen ausgeführt wird.

Für Qlabels kann ein Tastenkürzel vergeben werden. Es wurden mehrere Labels per Drap& Drop definieren. So werden mehrere Elemente danach bei Bestätigung des Tastenkürzels des Labels markiert. Tabulatorreihenfolge werden bearbeitet.

6.3.2.2 Transformation der erstellende GUI von Qt-Designer in Python-Quellcode

In Qt Designer wird erstmal das Fenster mit den gewählten Objekten bearbeitet und gespeichert. Die GUI lässt sich sich in Qt Designer mit der Name Endung *.ui speichern. In der folgenden Abbildung ist ein einfaches Beispiel GUI erstellt. Wenn der Button aktiviert wird, wird der eingegeben Wert im Eingabefeld multipliziert mit 2 und im Feld Ergebnis angezeigt.

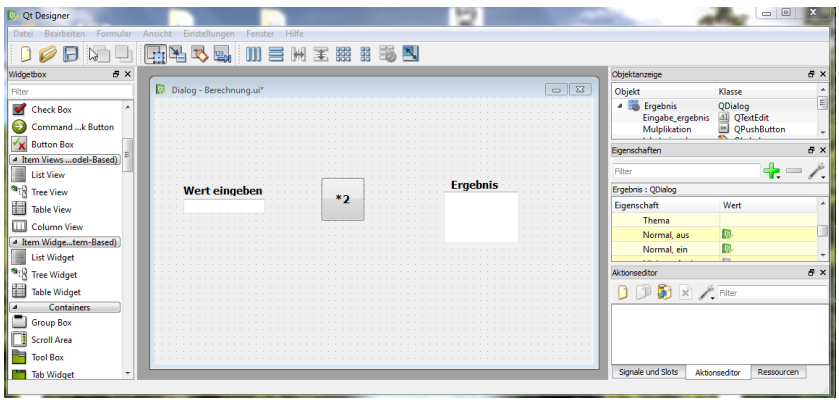


Abbildung 16: Aufbau einer Dialog in Qt Designer zur Multiplikation einer Zahl

Das Programm PYUIC4 bekommt den Pfad zu UI-Datei als Kommandozeilenparameter übergeben und gibt die daraus erstellte standartmäßig auf dem Bildschirm aus. Um die Ausgabe in einer Programmdatei umzulenken, kann in die Kommandozeilenoption –o verwendet werden. Der Befehl um eine python Datei sieht sieht wie folgt aus:

```
C:\Users\YOUSSA\Miniconda\Lib\site-packages\PyQt4\qt_C4DSI\pyuic4 -x
Berechnung.ui –o  Berechnung.py
```

Die folgende Abbildung zeigt die Übersetzung von Qt-Designer nach Python über das im Kommandozeile(cmd) ausgeführt wird.

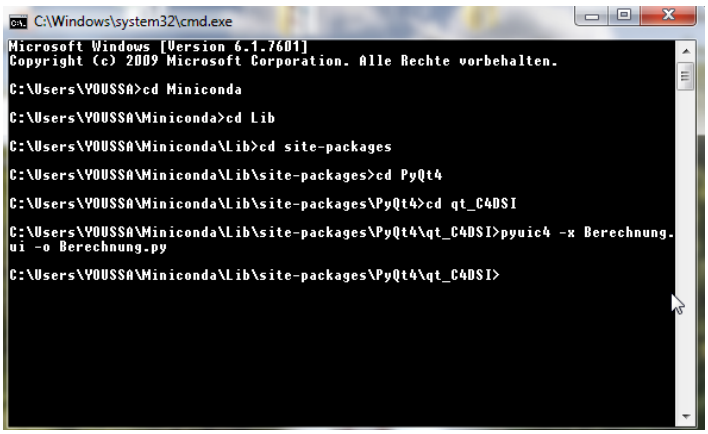


Abbildung 17: Cmd-Fenster für die Transformation der Gui-Fenster in Python-Code

Nach der Übersetzung im Cmd-Fenster konnte das Programm als Klasse Importiert werden. Die folgende Abbildung zeigt die Bearbeitung der Übersetzung in Spyder befor das Starten, zeigt die andere Quelle Code um in der GUI das entsprechende Feld den Wert einzugeben und um das Endergebnis im Feld anzuzeigen wenn der Benutzer auf das Bullton Multiplizieren geklickt hat.

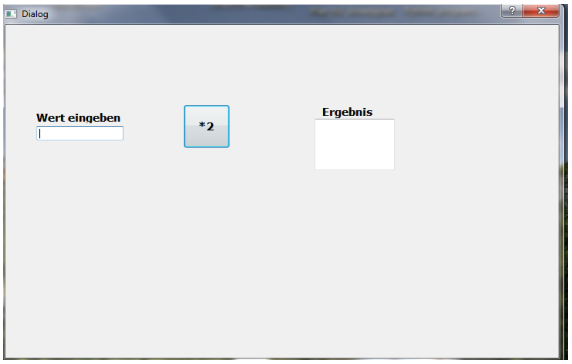


Abbildung 18: GUI Fenster zur Multiplizieren eines eingegeben Wert

Nach der Ausführung dieses Programm in Spyder, wird direkt ein Fenster anzeigt. Dort kann der Benutzer direkt die Werte in Feld und kann danach auf das Button Multiplzieren um das Ergbnis ins andere Feld zu sehen. Das Programm wird in der Projekt „rechner_zwei_zahlen“ mit zwei Modulen(„Berechnung1 und Multiplikation_einer_Zahl“) gelegt.

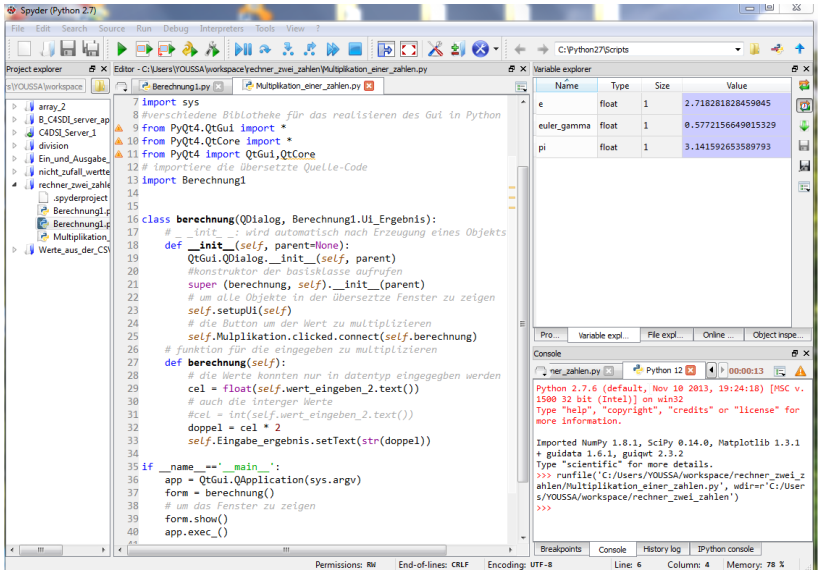


Abbildung 19: Die Übersetze GUI in Python QuelleCode

Die folgende Abbildung zeigt die Speicherort des erstellende Qt-Designer Fenster und die Python-Code nach der Transformation. Die Python-Code ermöglicht automatisch genau das Qt-designer in Python Programmiersprache erfüllen

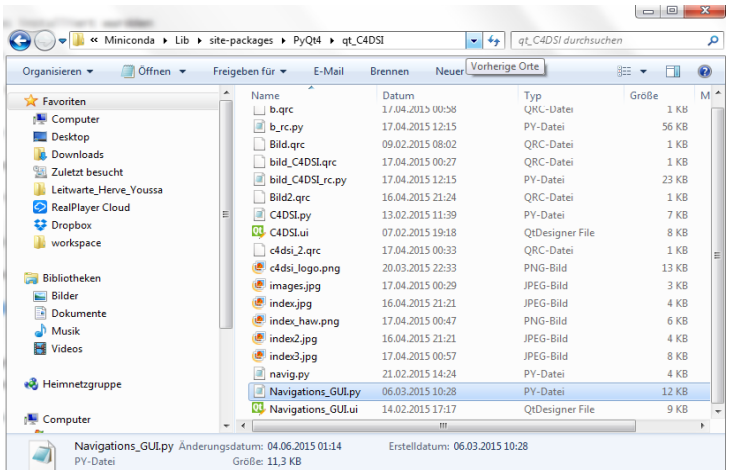


Abbildung 20: Ordner für die umgewandte python-Dateien und die UI-Datei

6.3.2.3 Transformation der gespeicherte Bilde von Qt-Designer in Python Quellcode

In die GUI sollen Bild-Dateien dargestellt werden. Hierzu sind Bilder in dem Ordner Qt C4DSI gespeichert. Damit Bilddateien im Qt-Designer verwendet werden können müssen diese umgewandelt werden. Hierzu ist es notwendig, dass das Bild in eine Ressourcendatei mit dem Format qrc konvertiert wird. Dazu muss in der Kommandozeilen(Cmd) der folgende Befehl genutzt werden:

```
C:\Python32\Lib\site-packages\PyQt4\Pyrrc4 –o Bild_C4DSI_rc.py   Bild_C4DSI.qrc
```

Die übersetze Datei(Bild_C4DSI-rc.py) wird sich automatisch im Ordner Qt_C4DSI befinden und wird von dem Module in dem Projekt C4DSI_server_1 verwendet.

6.3.3 Die Wahl der Qt-Design

Der große Vorteil von Thinker ist, dass es in der Standardbibliothek enthalten ist und somit nicht separat installiert werden muss. Aber Thinker ist nicht mehr zeitgemäße und eignet sich eher zum Schreiben von Protypen oder kleineren GUI-Anwendungen.
Qt ist nicht nur vom einen Toolkit sondern von einem Framework im Zusammenhang. Qt-Framework hat viel mit der Programmierung graphischer Benutzeroberflächen zu tun und enthält anderweitige nützliche Funktionalität breitzustellen. Qt Framework enthält Beispiele klasse zum Arbeiten mit XMI-Daten oder zu Netzwerkkommnunikation.

7 Systementwurf

Nach dem in der technischer Analyse das Zusammenspiel zwischen dem QT Designer und Python geklärt wurde und die Datenbank des C4DSIs näher analysiert wurde, soll in diesem Kapitel der Systementwurf vorgestellt werden.

Das GUI Tool, bekommt den Namen „Navigatons Tool“. Das Navigation Tool soll eine Schnittstelle zwischen dem Nutzer und der Datenbank darstellen. Dabei soll der Nutzer ohne Kenntnisse von Datenbanksprachen mit der Datenbank interagieren können. Dabei soll das Navigation Tool die ausgewählten Daten anzeigen, visualisieren und einen CSV Export zur Verfügung stellen. Außerdem soll der Nutzer die Datenmanipulieren können in dem er das Zeitintervall auf Sekunden, Minuten, Viertelstunden, Stunden oder Tage setzen kann und die dazugehörigen Daten bekommt.

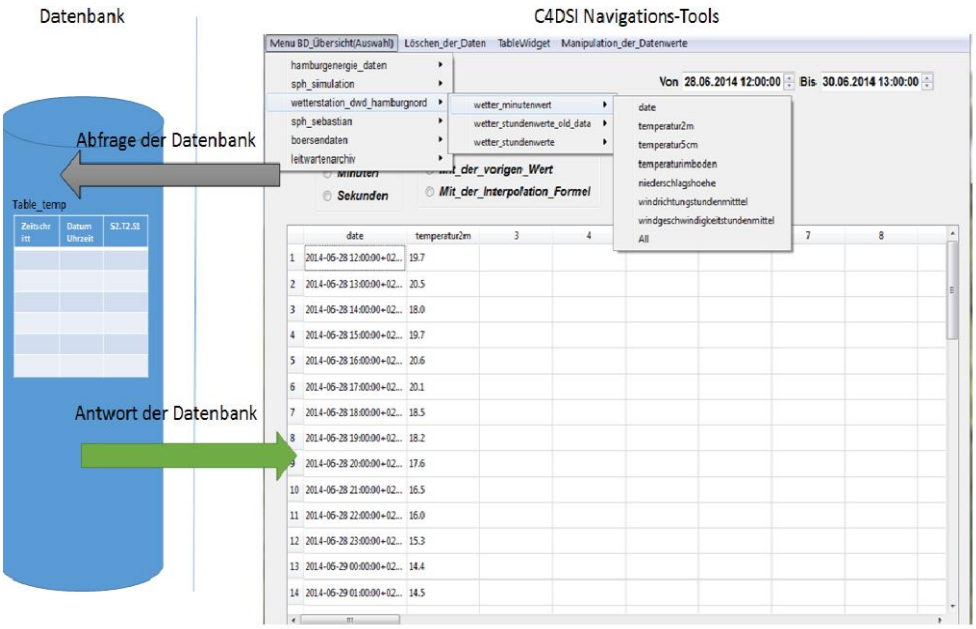


Abbildung 21: Systementwurf zwischen Datenbank und C4DSI Navigation-Tool

Diese Bild zeigt die Manipulation der Daten aus der Datenbank über das Navigation Tool. Der Nutzer fragt die Datenbank auf das Navigation Tool mit der Menu DB_Übersicht ab. Die Datenbank antwort SQL _Abfrage mit dem Anzeigen der Daten im Tablewidget. Mit den restlichen Menus konnte die gewählte Spalte in einer CSV-Datei importiert, exportiert und einer Kurve geplottet werden.

8 Implementation und Test

8.1 Anmeldefenster

Bevor das Navigation Tool genutzt werden kann, muss sich der Benutzer über ein Anmeldefenster anmelden. Anmeldedaten müssen im Datenbankserver hintergelget sein

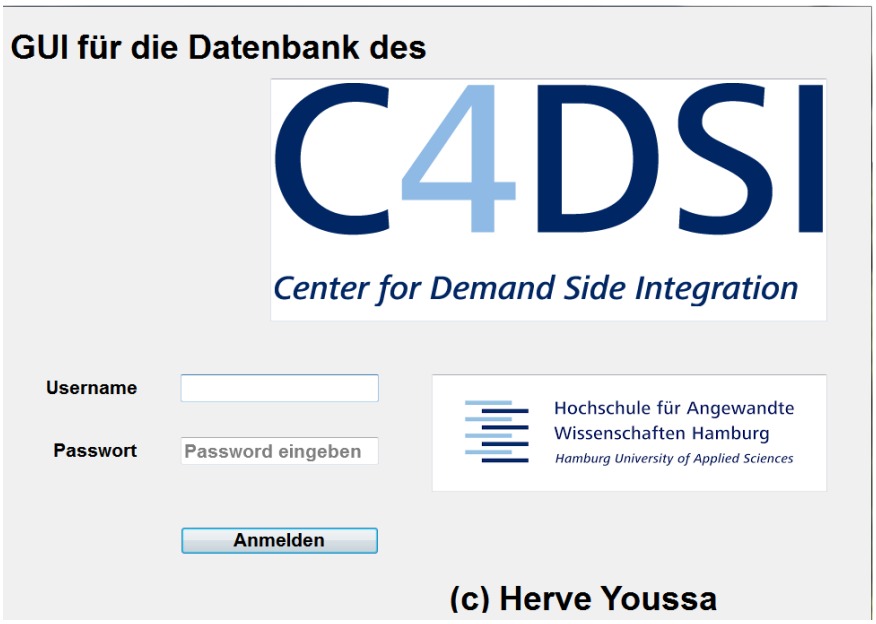


Abbildung 22: Anmeldung-Fenster des Navigations-Tools

Die Anmeldung der C4DSI ermöglicht nur eine Erstellung eines graphische Benutzeroberfläche mit einem Python-Code, der im Verbindung mit der GUI setzt. Die Erstellung des Anmeldungs-Fensters in Qtdesigner wird in Python-Code übersetzt. Der übersetzte Code lässt sich mit dem Haupt-Programm Python-Code in Verbindung mit einer Class-Erstellung. Unter diese Class definieren mehrere Objekte, die nützlich bei der Aufruf des Anmeldung-Fensters sind. Jede Objekte in Qtdesigner bekommt eine Objektename und lässt sich dimensionieren. Das heißt, in der Class der Quellcode in Python werden alle Komponente eines Objekt detaillieren. Die Übersetzung der Anmeldung-Fenster wird in der Projektname(C4DSI_Server1) von Spyder unter die Module Name C4DSI.py gespeichert. Bevor das Haupt-Programm in Pyhon geschrieben wird, werden in Spyder die Projektename

erstellt. Die Quellcode in Python sind unter die selbst genannte Module zu schreiben. Das Haupt-Programm dieser Arbeit wird in Projektname(C4DSI_server1) mit einer selbst genannte Module(ImC4DSI)erstellt. Die Anmeldung-Fenster Code unter die Module C4DSI.py setzt im Verbindung mit dem Haupt-Programm(C4DSI_server1). Die Verbindung zwischen die beiden Module wird unter dem Import einer der beiden Module in ein Modul gemacht. Es Bedeutet: die Module C4DSI.py lässt sich in der Module ImC4DSI importieren. Die Class C4DSI wird in Module(ImC4DSI) mit zwei Parametern als keine Class geschrieben um die Anmelden-Fenster nach der Ausführung des Haupt-Programms zu öffnen. Unter diese Class wird die Basis-Konstruktor des Class initialisiert, die mehrere Objekte von Qtdesingner mit der Befehl(self) aufrufen soll. Mit self konnten die Signale und Slot in Verbindung bleiben. Es wird ein Connect(Objekt von Qtdesigner) als Signal für das Anmelden-Fenster benutzt. Nach der Öffnung des Anmelden-Fenster werden die Benutzername und die Passwort nachgefragt. dem Connect-Button ermöglichen die Anmeldung nach der richtigen Eingabe der Benutzername und des Passwort. In der Class C4DSI ruft die Connect eine Funktion(Db_verbindung) mit Hilfe seine Objektname und der Bibliothek PyQt4.QtCore für Signal auf. In der Funktion Db_verbindung werden alle Parameter zu der Verbindung der Datenbank geprüft. die Parameter besteht aus Datenbankname, Passwort, Benutzername, Hostnummer, Portnummer. Mit der Psycopg2.connect() wird die Verbindung der Datenbank stattgefunden. Die Psycopg2.connect prüft alle Komponente zu dieser Verbindung. Nach falschen Eingabe der Benutzername oder Passwort wird eine Warning-Fenster aufgetaucht um der Fehler der Benutzer zu korrigieren. Im Erfolg zu der Eingabe oder wenn der Eingabe erfolgreich ist, kommt automatisch ein Andere Fenster(Navigation-Fenster) um die Daten aus der Datenbankserver aufzuzeigen.

8.2 Aufbau des Navigations Tools

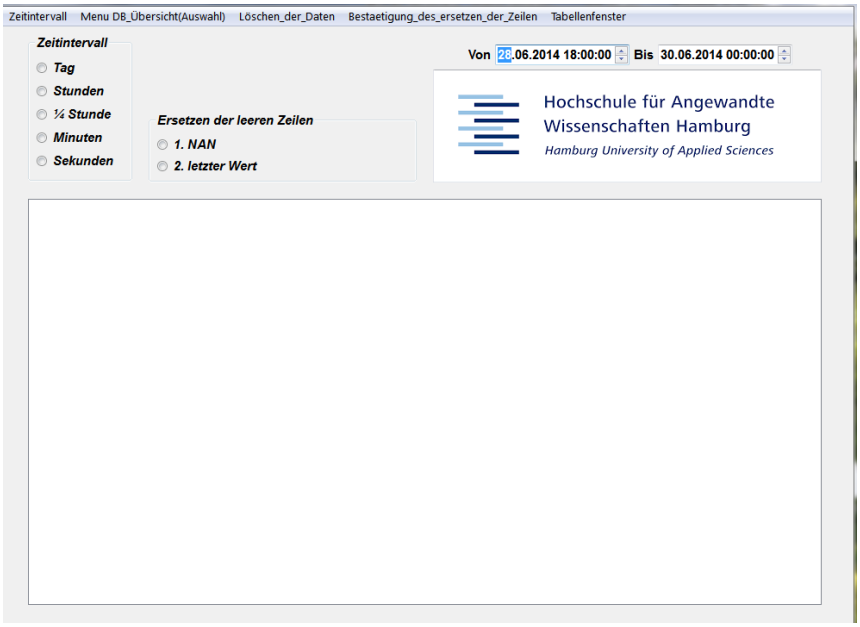


Abbildung 23: Navigation Tool

Das Navigation Tool ermöglicht den Abruf, die Manipulation und due Darstellung der Daten auf der Datenbank. Dieses Fenster wird mit dem QT Designer erstellt und besteht aus mehrere Bedienelemente. Das Menu im Fenster besteht aus fünf Hauptmenus:

- Zeitintervall:
  Dieser Menu wird für die Bestätigung des Zeitintervalls benötigt. Nachdem Anfansdatum, Enddatum und ein Zeitintervall definiert wurden und das Menu Zeitintervall aktiviert wird, erscheint im Tablewidget die ausgewählte Zeitreihe in der ersten Spalte.

- Menu DB Übersicht:
  Menu DB Übersicht zeigt die Spalten, Tabellen und Schema aus der Datenbank an. Hierdrüber können die Daten von der Datenbank abgerufen werden.

- Löschen der Daten:

Mit diesem Menu lassen sich die Daten aus der Tabletwidget löschen.

- Bestätigung des Ersetzen der Zeilen:
  Sollten beim Abruf Daten fehlen, muss beim erstetzten der Daten dieses Menu für die Bestätigung ausgewählt werden.

- Tabellenfenster:
  Das Menu Tabellenfenster verwendet die ausgewählten Daten, die im Tabletwidget angezeigt werden. Hierdrüber können die Daten visualisiert, importiert und exportiert werden.

Der Datenzeitraum wird über zwei Datenseingabefelder mit Anfangsdatum und Enddatum definiert. Um die zeitliche Auflösung(Intervall) zu definieren wurden im Fenstern mehrere Radiobuttons mit der Überschrift Zeitintervall angelegt. Für das ersetzen von fehlenden Datenpunkten sind ebenfalls Radiobuttons angelegt. Damit die ausgewählten Daten im Fenster angezeigt werden können, wurde in der Mitte des Fenster das Element Tabletwidget angelegt. Die Elementen und ihre Funktionen werden in den nächsten Kapiteln beschrieben.

8.3 Menu Zeitintervall

Die Daten in der Datenbank sind mit einem Zeitstempel in unterschiedlichen Zeitintervallen (Tages-, Stunden-, Viertelstunden-, Minuten- oder Sekundenwerte) gespeichert. Um die Daten vergleichbarer zu machen müssen die Daten mit einem gleichen Zeitstempel/Zeitintervall versehen werden. Somit muss der originale Zeitstempel in das gewünschte Zeitintervallformat konvertiert werden. Dazu kann der Benutzer nach seinem gewünschten Zeitintervall die Daten aus der Datenbank in das Tablewidget importieren. Dazu muss der Benutzer Anfangsdatum und Enddatum in den Datumsfelden definieren. Über Radiobuttons kann der Benutzer das Zeitintervallformat bestimmen. Damit das Format n das Tabletwidget zu übernehmen, muss der Benutzer über das Hauptmenu „Zeitintervall“ seine Eingabe bestätigen. Erst danach wird die Zeitreihe in der ersten Spalte des Tabletwidgets angezeigt und dient ab sofort als Referenzzeitreihe.

Die Referenzzeitreihe wird von der Datenbank erstellt, dafür nach dem bestätigen des Menus Zeitintervall die Funktion „Auswahl Option “ Über ein Trigger Signal aufgerufen. In dieser Funktion wird zunächst die Datumseingabe geprüft, wenn das Enddatum

kleiner ist als Anfangsdatum erscheint eine Warnung, dass die Eingabe Falsch ist und die Funktion wird beendet.

Wenn die Datumeingabe in Ordnung ist, werden die Radiobuttons für das gewünschte Zeitintervall ausgewertet. Das aktivierte Zeitintervall, aktiviert den PostgeSQL Query mit dem die Datenbank angefragt wird. Die Query Anfrage wird mit der Funktion CAST Generate series() realisiert. Diese benötigt die Übergabe parameter Anfangsdatum, Enddatum und Zeitintervall. Wenn der Query erfolgreich abgearbeitet ist, sendet die Datenbank die zeitreihe zurück. Damit die Zeitreihe gelesen werden kann werden die Daten in eine Liste übergeben, dafür wird die Funktion Cur.Fettchall() genutzt von Python. Anschließend wird die erste Spalte mit dem Spaltename Date versehen und das ausgelesene Zeitintervall wird der Spalte zugewiesen. Die folgende Abbildung zeigt, die ausgewählte Zeitreihe im Tablewidget.

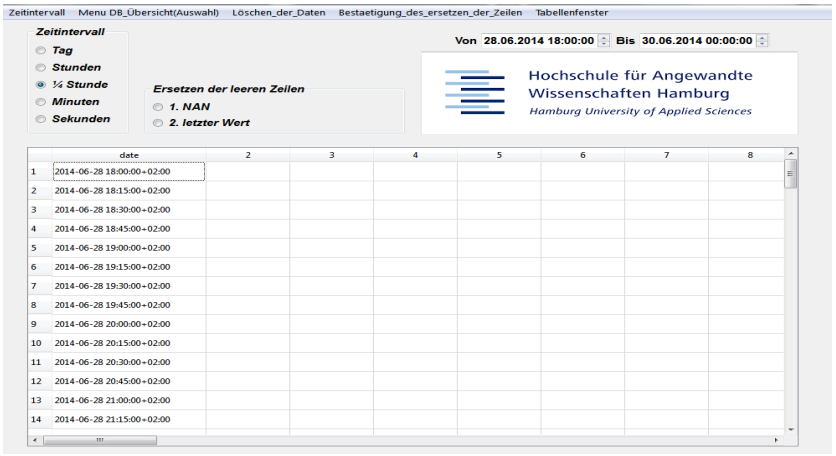


Abbildung 24: Auswahl des Zeitintervalls und Darstellung der Referenzspalte im Tablewidget

8.4 Menu DB_Übersicht(Auswahl) im Navigation Tool

8.4.1 Aufbau der Menu DB_Übersicht(Auslesen der Datenbank Schemata, Tabellen und Spalten)

Das Flussdiagramm zeigt, wie das Menu mit dem Namen der Schematas, Tabellen und Spalten gefüllt wird.

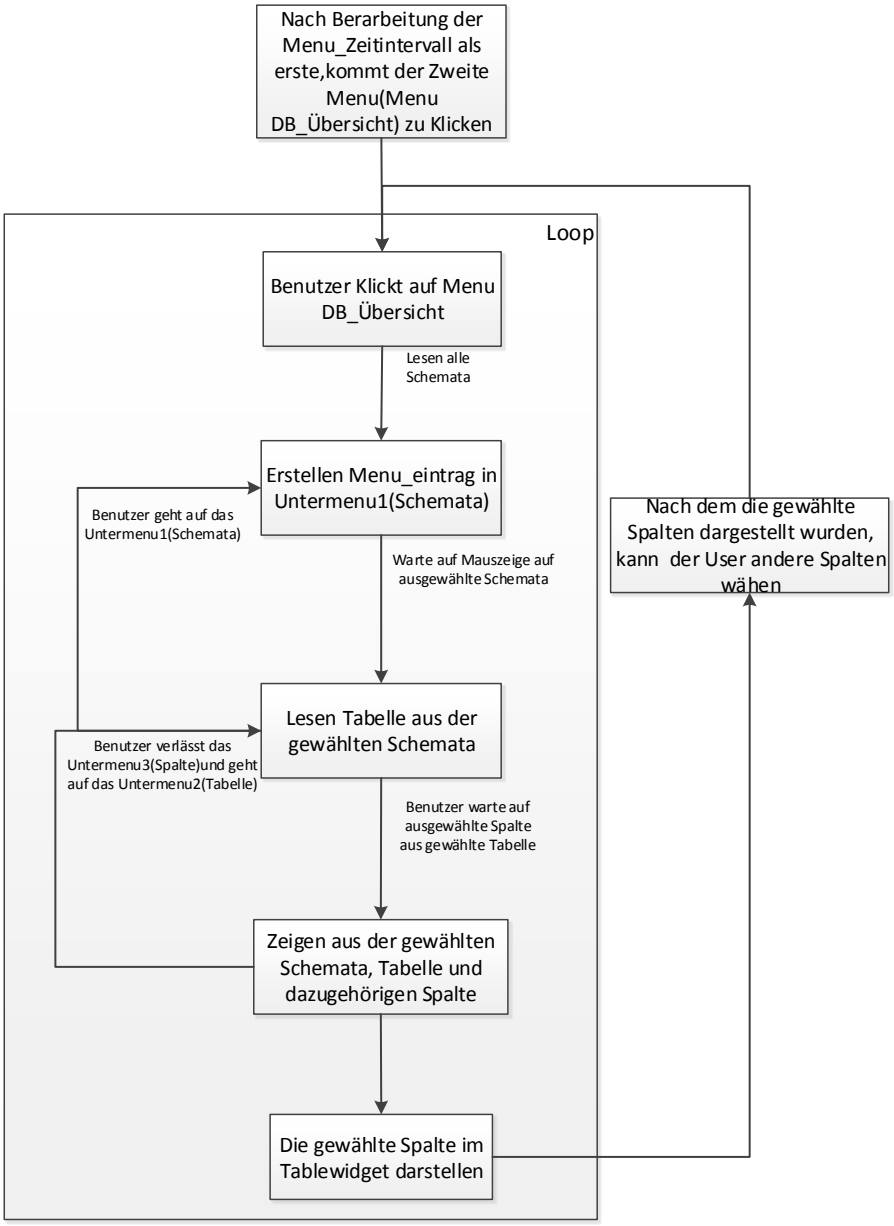


Abbildung 25: Flussdiagramme zur Abruf der Daten mit Menu DB_Übersicht

Die folgende Abbildung zeigt die Wahl einer Spalte auf dem Navigations Tool

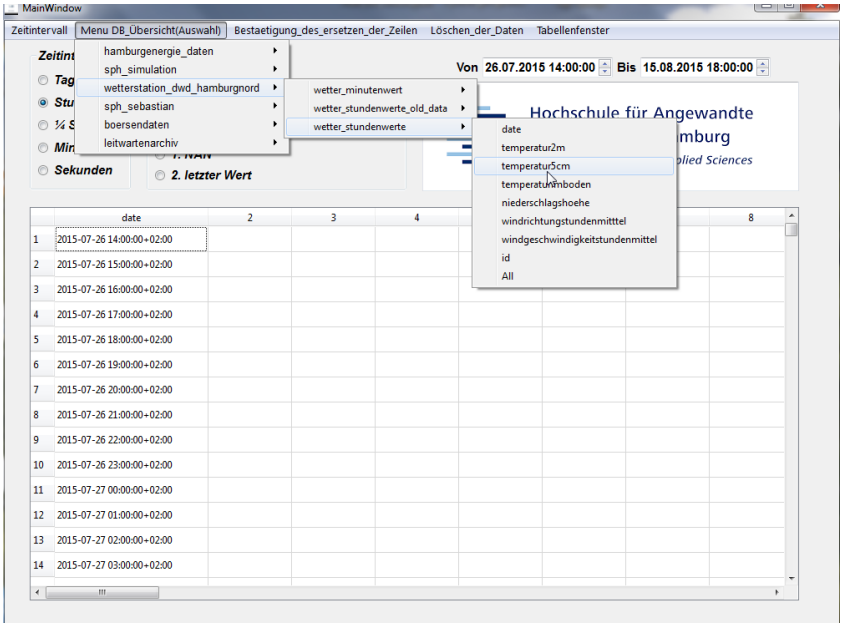
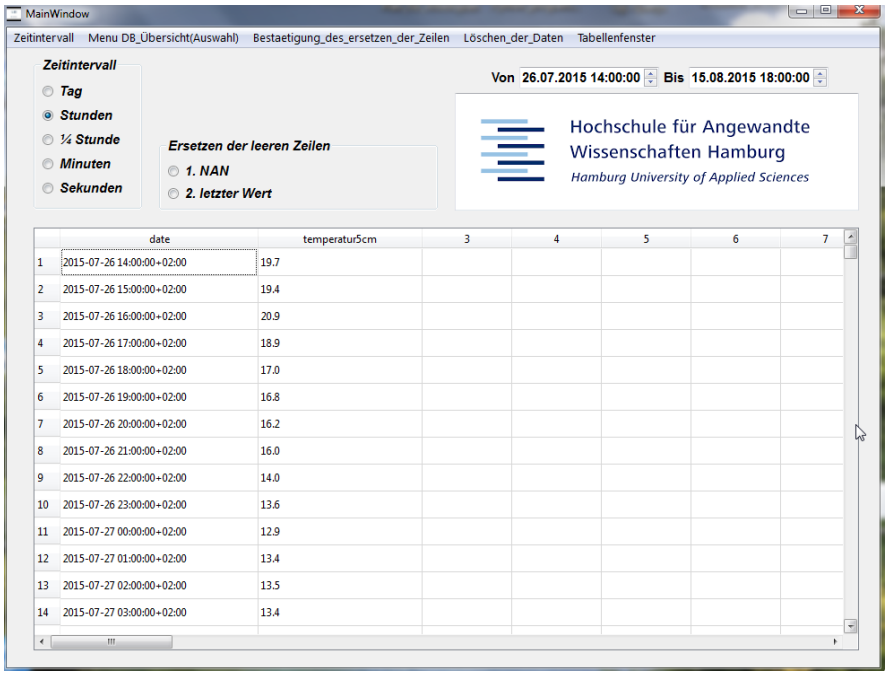


Abbildung 26: Auswahl einer Spalte mit der Menu DB_Übersicht

Das Menü DB_Übersicht im Navigations Toos spielt die größte Rolle um die Daten der Datenbank abzurufen. Es ist das einzige Menü, das direkt mit dem Datenbankserver(C4DSI) durch interaktives Klicken kommuniziert. Die Klasse für das Navigations-Tool wird in dem Haupt-Programm(C4DSI_server1) mit der QtGui.QMainWindow für die GUI-Window und der Übersetzung der Klasse Navigation_GUI_2.Ui_Navigation_GUI geschrieben. Während der Öffnung des Navigations-Tool ruft die Klasse „Navigation“ die Funktion „MenusErstellung“ auf. Der Quellcode dieser Funktion wird schrittweise erklärt. Der Befehl „ Curseur cur = con.cursor“ selektiert alle Schemata des Datenbankservers mit dem SQL-Anfrage. Die Daten werden mit der Funktion „cur.fetchall“ angezeigt. Die Untermenüs entsprechen den Tabellen und den dazugehörigen Spalten. Die Untermenus für Tabelle werden in einer Variablen der Menu-Schemas erstellt und werden mit jedem der entsprechenden Schemas durch die SQL-Befehl selektiert. Zum Anzeigen der Tabelle als Untermenu ruft die Funktion Tabellen = cur.fetchall() mit einer Variablen (Tabellen) auf. Im Gegenteil zu den anderen Untermenus für Tabelle betrachtet die MenusErstellung die Untermenus für Spalten als Aktions. Sie werden mit der Hilfe des SQL-Anfrage angezeigt, der die Spalten selektieren kann. Wie bei der Tabelle ruft die Funktion Spalten = cur.fetchall() mit der Variablen „Spalten“ auf und wird am Ende als Aktion aufgebaut. Das Auslesen der

Datenbank „Schemata“, „Tabelle“ und „Spalten“ führt sich unter die drei For-Schleifen aus. Jeder For-Schleife entspricht für das Durchzählen der Anzahl der Schemata, Tabelle und Spalte.

Die folgende Abbildung zeigt das Ergebnis einer gewählten Spalte aus der Datenbank



	date	temperatur5cm	3	4	5	6	7
1	2015-07-26 14:00:00+02:00	19.7					
2	2015-07-26 15:00:00+02:00	19.4					
3	2015-07-26 16:00:00+02:00	20.9					
4	2015-07-26 17:00:00+02:00	18.9					
5	2015-07-26 18:00:00+02:00	17.0					
6	2015-07-26 19:00:00+02:00	16.8					
7	2015-07-26 20:00:00+02:00	16.2					
8	2015-07-26 21:00:00+02:00	16.0					
9	2015-07-26 22:00:00+02:00	14.0					
10	2015-07-26 23:00:00+02:00	13.6					
11	2015-07-27 00:00:00+02:00	12.9					
12	2015-07-27 01:00:00+02:00	13.4					
13	2015-07-27 02:00:00+02:00	13.5					
14	2015-07-27 03:00:00+02:00	13.4					

Abbildung 27: Ergebnis der gewählte Spalte mit der Menu DB_Übersicht

Nachdem die Zeitintervall gewählt und angezeigt wurde, wurde mit der Menu DB_Übersicht nachher die Daten an die entsprechende Zeitintervall an die Datenbank C4DSI an die Spalte „ Temperatur5cm“ abfragt und direkt im Tablewigget angezeigt.

8.4.2 Datenauswahl einer Spalte oder einer Tabelle

Flussdiagramme eines Abrufs einer gewählte Tabelle oder gewählte Spalte

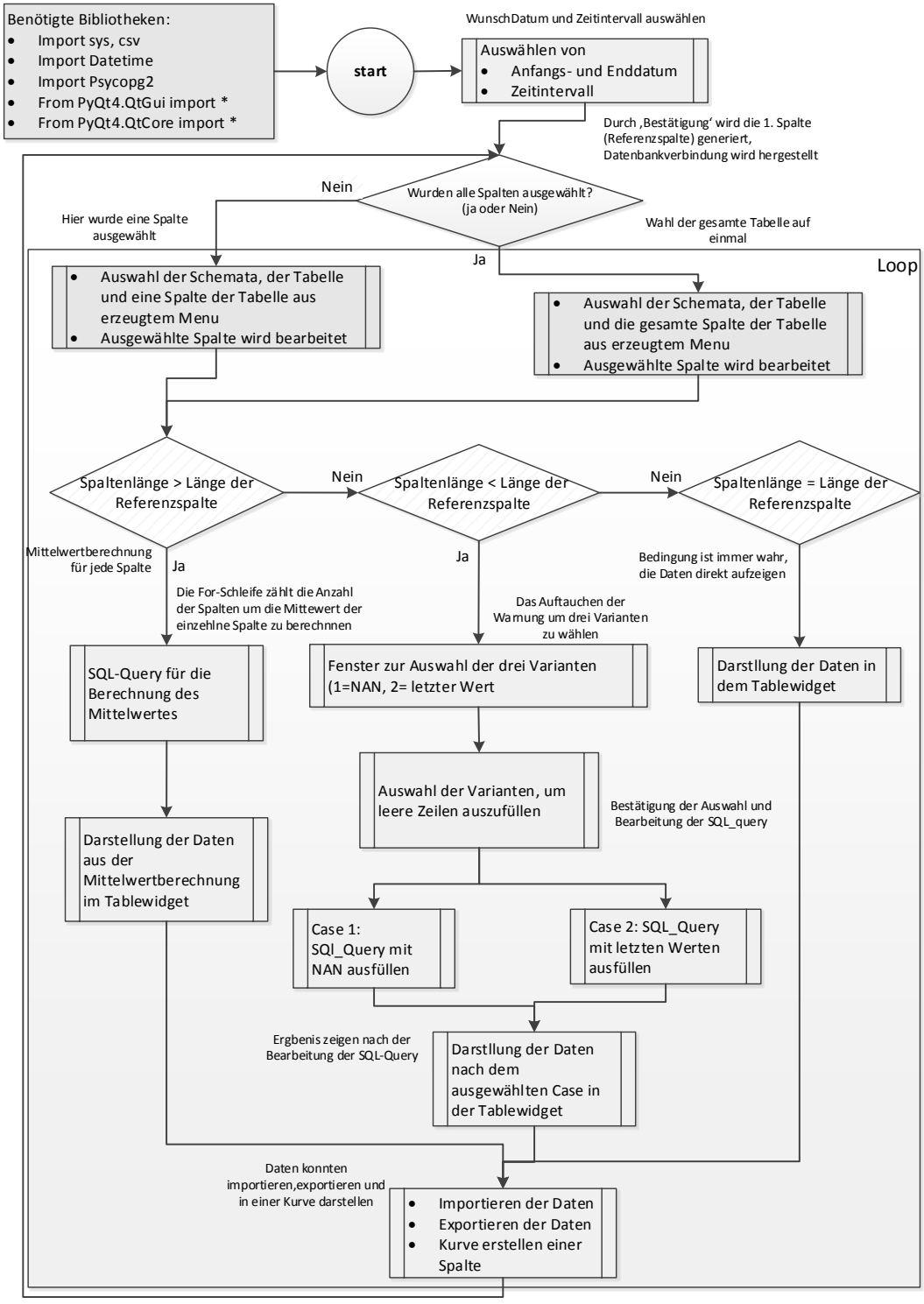


Abbildung 28: Flussdiagramm eines Abrufs einer gewählte Tabelle oder einer gewählte Spalte

In der obere Feld wird das Programm mit der benötigen Bilblotheken gestartet. Nach dem die Daten und die Zeitintervall gewählt wurden, wurde nach der Auswahl einer Spalte oder alle Spalten einer Tabelle abgefragt. Nach der Bestätigung einer gewählte Spalte oder gewählte Tabelle wurden die Vergleichung zwischen die Spaltelänge der gewählte Spalte oder Tabelle und die Länge der Referenzspalte für drei Fälle geprüft:

- 1) Die Spaltelänge gleich die Länge der Referenzspalte:
  Wenn die Spaltelänge einer gewählte Spalte oder Tabelle gleich die Länge der Spalte ist, wurden direkt die gewählte Daten in einer Tabelle oder einer Spalte im Tablewidget direkt dargestellt.

- 2) Die Spaltelänge kleiner als die Länge der Referenzspalte:
  In der zweite Fall wird erstmal geprüft ob die die Spaltelänge kleiner als die Länge der Referenzspalte wird. Nach der Bestätigung wurde direkt einer Warnung angezeigt um die fehlende Daten in der gewählten Spalte oder Tabelle zu ersetzen. In der nächste Schritt wurden die entsprechende SQL_Query für die gewählte Varianten bearbeitet und nachher die Daten direkt im Tablewidget angezeigt.

- 3) Die Spaltelänge größer als die Länge der Referenzspalte:
  Wenn die Spaltelänge größer als die Länge der Refernzspalte geprüft wurde, wurde die Mittelwertberechnung in einer SQL-Query für die gewählte Spalte oder gewählte Tabelle ausgeführt und direkt im Tablewidget angezeigt.

Nach der Bearbeitung und dem Anzeigen einer gewählte Spalte oder Tabelle im Tablewidget konnten die Mess-und Stimulationdaten importiert, exportiert und geplottet werden. In einer Loop der Flussdiagramm wird diese Aktion mehrfach realisiert. Das heißt der Benutzer kann mehrmals die Datenbank C4DSI abfragen um mehrere Spalten oder Tabellen zu haben.

8.4.2.1 SQL_Befehl zur Abruf einer gesamten Tabellen

query_all = "select * from %s.%s where date between '%s' and '%s' " % (Schemata, tabelle, anfang_datum_geandert, ende_datum_geandert)
cur.execute (query_all).

Der Befehl der SQL_Query selektiert alle Spalte einer Tabelle mit viele Integrierten Funktionen und Verknüpfungen. Die Funktion cast(generate_series) generiert die Daten oder die Werte in wachsend mit der umgewandelte Datentyp. Die Verknüpfung JOIN ordnet die Spalte DATE ganz Links auf dem Tablewidget. Die Verknüpfung TIMESTAMPTZ generiert das Integral der Uhrzeit und des Datums für jede Zeile mit Time Zone in der Spalte DATE. Die verschiedene Variable(anfang_datum_geandert, ende_datum_geandert, Intervall, Schemata, tabelle) in der SQL_query ermöglicht aus dem gewählten Datum und gewählten Tabelle in der Datenbank zu haben. Die Variable sollen mit den Variablen der Datenbank übereinstimmen um die Werte nach dem Klicken im Tablewidget zu zeigen. Wenn einer der Variable falsch ist, wird die SQL-Query nicht sich erfüllt.

8.4.2.2 Beschreibung der Darstellung der Daten im Tablewidget

Nach der Verbindung der Datenbank mit der cur = self.db_parameter_zur_vb(), wird die Aufruf der gesamte spalten einer Tabelle durch das Klicken in der Menu Datenbank_Auswahl ausgeführt. Das Klicken ALL , das auf SpalteAktion sich befindet, entwickelt ein Signal triggered, das eine Funktion Daten_alle_spalten Aufruf. Die aufgerufene Funktion präsentiert sich mit viele Parametern, die in der Datenbank erkennt und benutzt werden sollen. In der Funktion Daten_alle_spalten wird direkt die Verbindung zur der Datenbank erstellt. Bevor die SQl_query_all in der Funktion bearbeiten soll, muss erstmal die Tablewidget über die Anzahl der Spalte abgefragt werden. die Abfrage des Tablewidget soll die Anzahl der Spalten versichern ob die Anzahl der Spalten soll immer grösser als null sein worin die Spalte an der erste Stellung muss die Referenzspalte sich befindet. Die kommende Spalten nach der Erfüllung der SQL_query_all sollen nicht die erste Spalten sein. Wenn die Abfrage der Anzahl der Spalten erfolgreich ist, wird die SQl_query_ALL ausgeführt und wird als Liste von mehrere Spalten mit vielen Werte an die Funktion Cur.fetchall gelegt. Die Cur.fetchall weist die Liste anschließend an die rows_alle Variable zu. Bevor die Liste im Tablewidget auftaucht, werden erstmal die Länge der gesamten Spalten und die Länge der referenzspalte

ermittelt. In den kommenden Teil der Funktion Daten_alle_spalten wird die Länge der Spalten aus der Daten mit der Länge der referenzspalte verglichen.

8.4.2.3 SQL_Befehl zur Abruf einer Spalte aus einer Tabelle

Query_spalte = "select %s date from %s.%s where date between '%s' and '%s' " % (spalte, schemata,tabelle,anfang_datum_geandert, ende_datum_geandert).

Die Befehl der Query_spalte selektiert einer Spalte mit viele gegebene Variablen und Einschränkungen(Where,and,Between). Die Variablen sind in den Parameter der Funktion daten_einer_spalte() vordefiniert worden. Die Where, and, Between limitieren die fordernden Werte aus der Datenbank. Die Variable werden in der Query mit %s ersetzen. Die Variable(anfang_datum_geandert, ende_datum_geandert, Intervall, Schemata, tabelle) in der Query_spalte ermöglicht aus dem gewählten Datum und gewählte Spalte in der Datenbank zu haben. Die Variable sollen mit den Variablen(Spalte, Schemata, Tabelle) der Datenbank übereinstimmen um die Werte nach dem Klicken in der Tablewidget zu zeigen. Wenn einer der Variable falsch ist, wird die Query_spalte nicht ausgeführt.

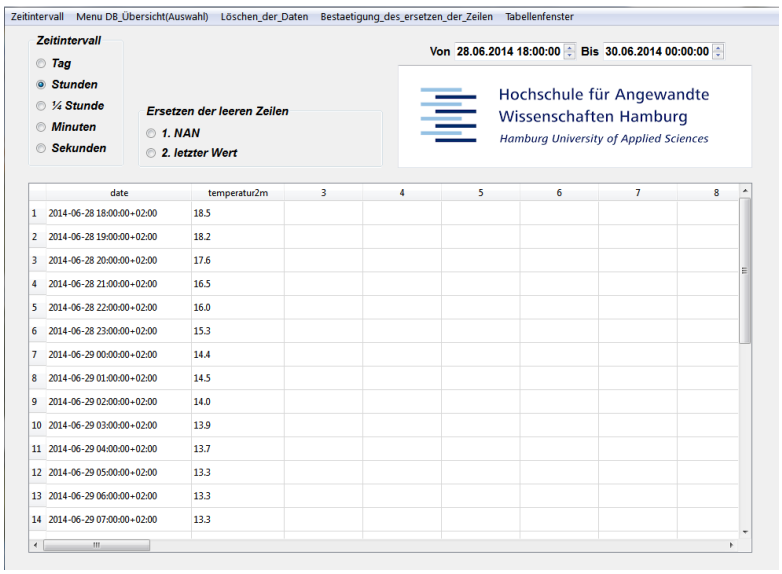
8.4.2.4 Beschreibung der Darstellung der Daten im Tablewidget

Nach der Verbindung der Datenbank mit der cur = self.db_parameter_zur_vb(), wird die Aufruf einer Spalte durch das Klicken in der Menu DB_Auswahl ausgeführt. Das Klicken auf einer gewählte Spalte entwickelt ein Signal triggered, das eine Funktion Daten_einer_spalte Aufruf. In der Funktion Daten_einer_spalte wird direkt die Verbindung zur der Datenbank erstellt. Bevor die Query_spalte ausführen soll, muss erstmal die Spaltenname mit folgenden Einsatz field_names = [i[0] for i in cur.description] definiert werden. Die Funktion cur.Fetchall() bearbeitet die ausgeführte Query_spalten als liste. Vor das Auftauchen der Daten einer Spalte, muss erstmal die Tablewidget über die Anzahl der Spalte abgefragt werden. Die Abfrage soll über die Länge einer gewählte Spalte und die Länge einer Referenzspalte vergleichen.

Die Abfrage teilen sich in drei Fälle wie bei der Funktion Daten_alle_Spalten, wobei die Manipulation der Daten einer Spalte durch die Aufruf SQL_Querys der Mittelbildung und die Interpolation eintreten sollen.

Wenn einer der Drei Bedingungen erfüllt ist, wird automatisch die entsprechende Query für eine Spalte in der Datenbank bearbeitet und direkt als Liste an die Funktion cur.fetchall() zugewiesen. Das gleiche Prozeduren wie für alle Spalten wird betrachtet Und an die Tablewidget aufzeigen. In der Verarbeitung für mehrere Spalten werden die Bedingungen deutlich erklärt dass die Wiederholung für eine Spalte nicht nötig ist. Das folgende Bild zeigt einer der Drei Bedingungen.

Das Tablewidget weist die Daten aus Daten die aus der gleiche Länge der Spalte mit der Referenzspalte auf. Für jede Zeile entspricht genau das Datum und der Uhrzeit mit dem rausgeholte aus Pgadmin-Tool Wert. Nach der Bestätigung der Zeitintervall, berücksichtigt die SQl_Query der ausgewählte Spalte auch die Zeitintervall. Die Zeilenwerte der Spalte(Temperaturimboden) führen genau die Werte, die in der Datenbank gelegt wurden.

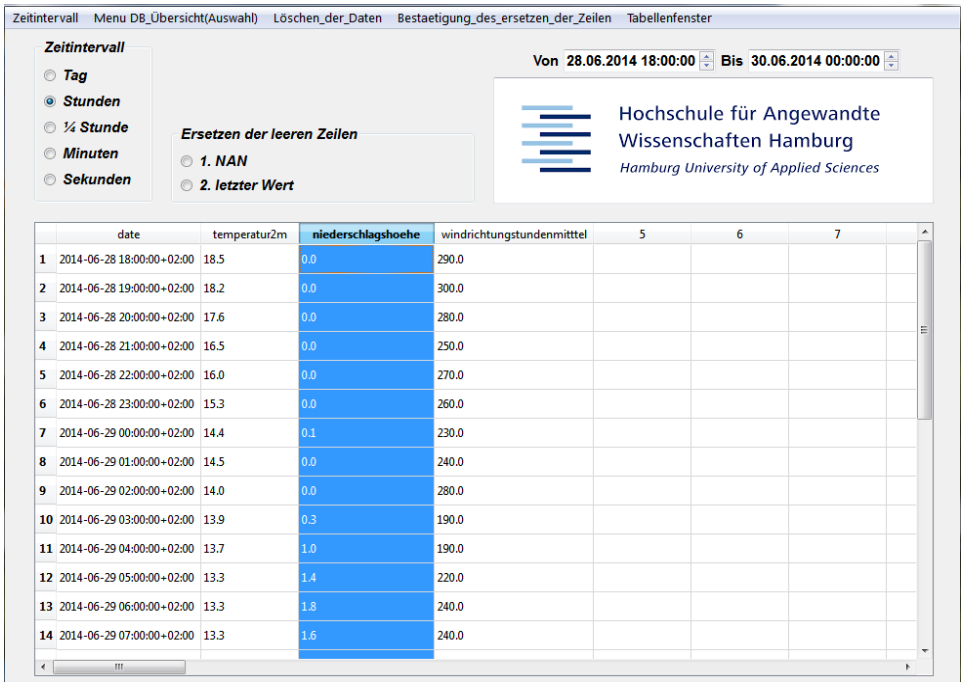


	date	temperatur2m	3	4	5	6	7	8
1	2014-06-28 18:00:00+02:00	18.5						
2	2014-06-28 19:00:00+02:00	18.2						
3	2014-06-28 20:00:00+02:00	17.6						
4	2014-06-28 21:00:00+02:00	16.5						
5	2014-06-28 22:00:00+02:00	16.0						
6	2014-06-28 23:00:00+02:00	15.3						
7	2014-06-29 00:00:00+02:00	14.4						
8	2014-06-29 01:00:00+02:00	14.5						
9	2014-06-29 02:00:00+02:00	14.0						
10	2014-06-29 03:00:00+02:00	13.9						
11	2014-06-29 04:00:00+02:00	13.7						
12	2014-06-29 05:00:00+02:00	13.3						
13	2014-06-29 06:00:00+02:00	13.3						
14	2014-06-29 07:00:00+02:00	13.3						

Abbildung 29: Darstellung der Daten einer Spalte nach Zeitintervall mit jedem Zeilenwert

8.5 Menu Löschen der Daten aus dem Navigation Tool

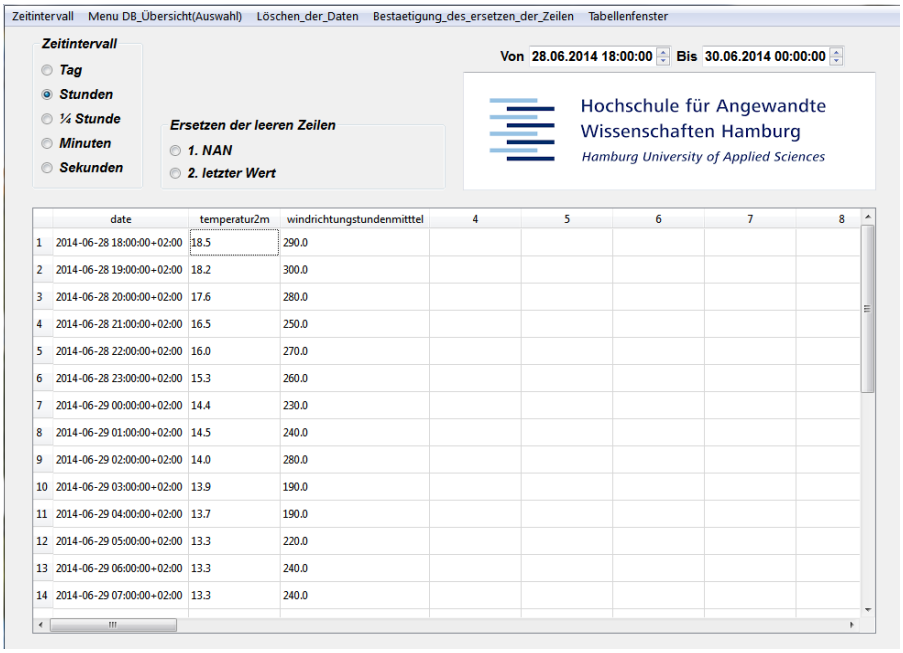
Nachdem das Tablewidget mit vielen ausgewählten Spalten gefüllt wurde, können die nicht interessierten Spalten gelöscht werden. Damit Spalten aus dem Tablewidget gelöscht werden können, muss der Benutzer zunächst die gewünschten Spalten markieren. Anschließend muss der Benutzer mit dem Mauszeiger auf das Menu „Löschen der Daten“ gehen und das Untermenu aktivierten. Dieses Untermenu triggert die Funktion Löschen. In der Funktion Löschen wird erstmal eine leere Liste erstellt. Diese wird mit den selektierten Spalten in die leeren Liste eingefügt und zu sortieren. Mit einem Index in der Liste wird mit einer For-Schleife die ausgewählten Spalte aus der Liste gelöscht. Die nächsten zwei Bilder zeigen wie eine Spalte Aus dem Tablewidget gelöscht wird. Die Abbildung zeigt drei ausgewählte Spalten (temperatur2m, niederschlagshoehe und windrichtungsstundenmittel). Alle drei Spalten stammen aus der tabelle DWD Wetterdaten und sind im Schema DWD Wetterdaten hinterlegt. Zum Löschen wird hier die Spalte niederschlagshoehe im Tabletwidget markiert.



	date	temperatur2m	niederschlagshoehe	windrichtungstundenmittel	5	6	7
1	2014-06-28 18:00:00+02:00	18.5	0.0	290.0			
2	2014-06-28 19:00:00+02:00	18.2	0.0	300.0			
3	2014-06-28 20:00:00+02:00	17.6	0.0	280.0			
4	2014-06-28 21:00:00+02:00	16.5	0.0	250.0			
5	2014-06-28 22:00:00+02:00	16.0	0.0	270.0			
6	2014-06-28 23:00:00+02:00	15.3	0.0	260.0			
7	2014-06-29 00:00:00+02:00	14.4	0.1	230.0			
8	2014-06-29 01:00:00+02:00	14.5	0.0	240.0			
9	2014-06-29 02:00:00+02:00	14.0	0.0	280.0			
10	2014-06-29 03:00:00+02:00	13.9	0.3	190.0			
11	2014-06-29 04:00:00+02:00	13.7	1.0	190.0			
12	2014-06-29 05:00:00+02:00	13.3	1.4	220.0			
13	2014-06-29 06:00:00+02:00	13.3	1.8	240.0			
14	2014-06-29 07:00:00+02:00	13.3	1.6	240.0			

Abbildung 30: Das Tablewidget vor das Löschen mit einer zu löschen ausgewählte Spalte

Nach dem das Menu „Löschen der Daten“ aktiviert wurde ist im nächsten Bild zu sehen, dass die markierte Spalte(niederschlagshoehe) gelöscht wurde. Außerdem wurde die vierte Spalte windrichtungsstundenmittel in die Spalte drei verschoben.



	date	temperatur2m	windrichtungstundenmitttel	4	5	6	7	8
1	2014-06-28 18:00:00+02:00	18.5	290.0					
2	2014-06-28 19:00:00+02:00	18.2	300.0					
3	2014-06-28 20:00:00+02:00	17.6	280.0					
4	2014-06-28 21:00:00+02:00	16.5	250.0					
5	2014-06-28 22:00:00+02:00	16.0	270.0					
6	2014-06-28 23:00:00+02:00	15.3	260.0					
7	2014-06-29 00:00:00+02:00	14.4	230.0					
8	2014-06-29 01:00:00+02:00	14.5	240.0					
9	2014-06-29 02:00:00+02:00	14.0	280.0					
10	2014-06-29 03:00:00+02:00	13.9	190.0					
11	2014-06-29 04:00:00+02:00	13.7	190.0					
12	2014-06-29 05:00:00+02:00	13.3	220.0					
13	2014-06-29 06:00:00+02:00	13.3	240.0					
14	2014-06-29 07:00:00+02:00	13.3	240.0					

Abbildung 31: Darstellung der Daten nach der löschen Betätigung mit den restlichen Spalten

8.6 Menu Tabellenfenster

Das Hauptmenu Tabellenfenster beinhaltet drei weitere Untermenus. Das sind Export, Import und Grafik darstellen.

8.6.1 Export

Nachdem Daten ausgewählt wurdden sind, können diese in ein CSV Datei exportiert werden. Mit Hilfe der Funktion Daten_speichern wird in der Quellcode sorgt für die Speicherung der Daten in einer CSV-Datei. Das Dateiformat CSV (Comma separated Values) beschreibt den Aufbau einer Textdatei zur Speicherung der strukturierten Daten.

Sobald das Untermenu Export aktiviert wurde, wird die Funktiom Daten_speichern über ein Triggersignal aktiviert. Es öffnet sich ein Fenster zum speichern von Dateien. Dort kann ein belibiger Name definiert werden und die CSV Datei über den Bütton „speichern“ abgespeichert werden. Die Daten werden in der Liste (Rowdata) für jeder Spalte und jeder Zeile aus dem Tablewidget kopiert. Der geöffnete Pfad kopiert auch die Liste (rowdata) Zeile für Zeile. Der Pfad soll mit den Namen der CSV Datei in das Projekt(C4DSI) gespeichert werden.

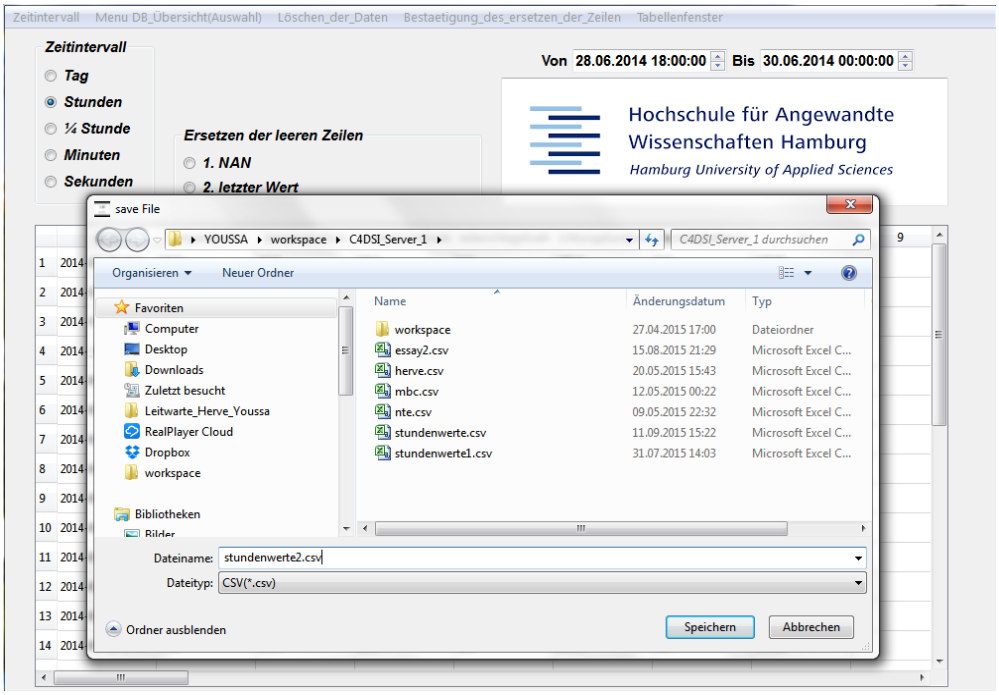


Abbildung 32: Speicherung der Daten in der CSV-Datei mit der CSV-name Stundenwerte.csv

8.6.2 Import

Nachdem die Daten in einer CSV-Datei gespeichert sind, können diese Daten wieder in das Tablewidget importiert werden. Dabei spielt es keine Rolle ob das Tablewidget leer oder mit anderen Daten gefüllt ist.Das Untermenu sendet das Signal triggered zu der Funktion „DatenImportieren“.

Es öffnet sich ein Fenster zum Öffnen von Daten. Aktuel ist ein fester Speicherort definiert, wo alle gespeicherten CSV Dateien liegen. Nachdem ein CSV geöffnet ist, wird die Liste „Rowdata“ nach Spalte und Zeile gelesen und direkt im Tablewidget entsprechend nach Spalte und Zeile eingefügt.

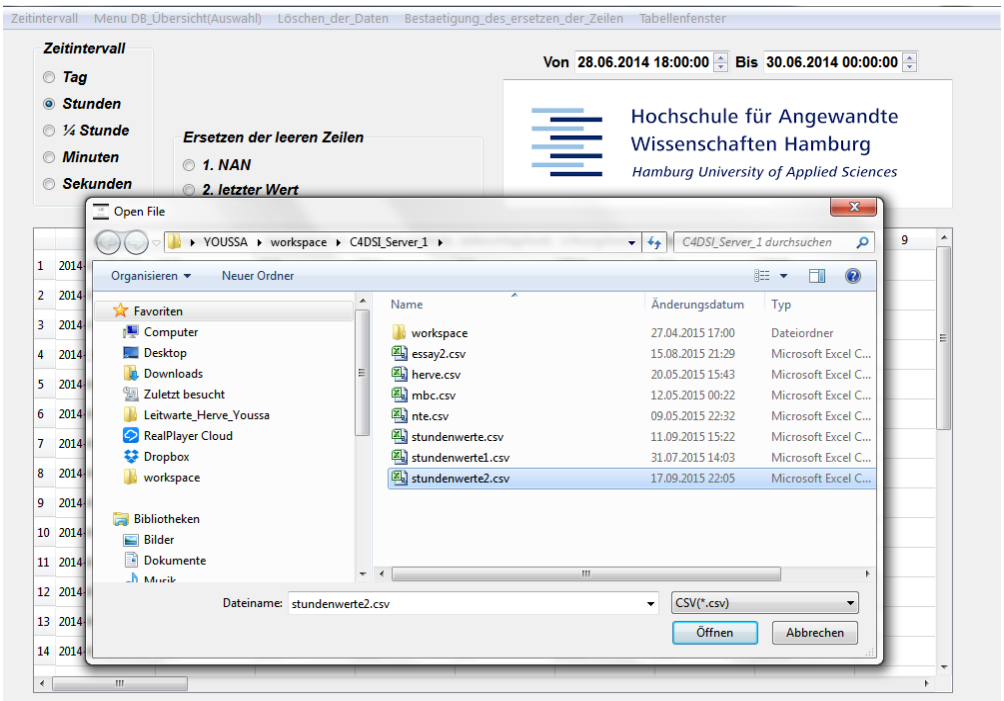


Abbildung 33: Das Importieren der Daten von gewählten CSV-Date im Tablewidget

8.6.3 Grafik darstellen

Bevor dieses Untermenu aktiviert wird, müssen Spalten im Tablewidget markiert werden. Nur dann lassen sich die Daten anzeigen (plotten). Die X-Achse ist immer mit der Spalte date i Tabletwidget definiert.

Wenn das Untermenu aktiviert wird, startet ein trigger die Funktion. In dieser Funktion werden erstmal die markierten Spalten eingelesen. Dabei wird die Spalte „date“ in einer X-Liste und die markierten Spalten in einer Y-Liste zugewiesen. Zum Plotten der Daten wird die Bibliothek Matplotlib benötigt, diese bietet die Funktionen „Plt.show()“. Mit dieser Funktion lassen sich die Daten plotten.

Zusätzlich bietet die Funktion, zoomen, Zeitbereiche ausschneiden und das Bild abzuspeichern.

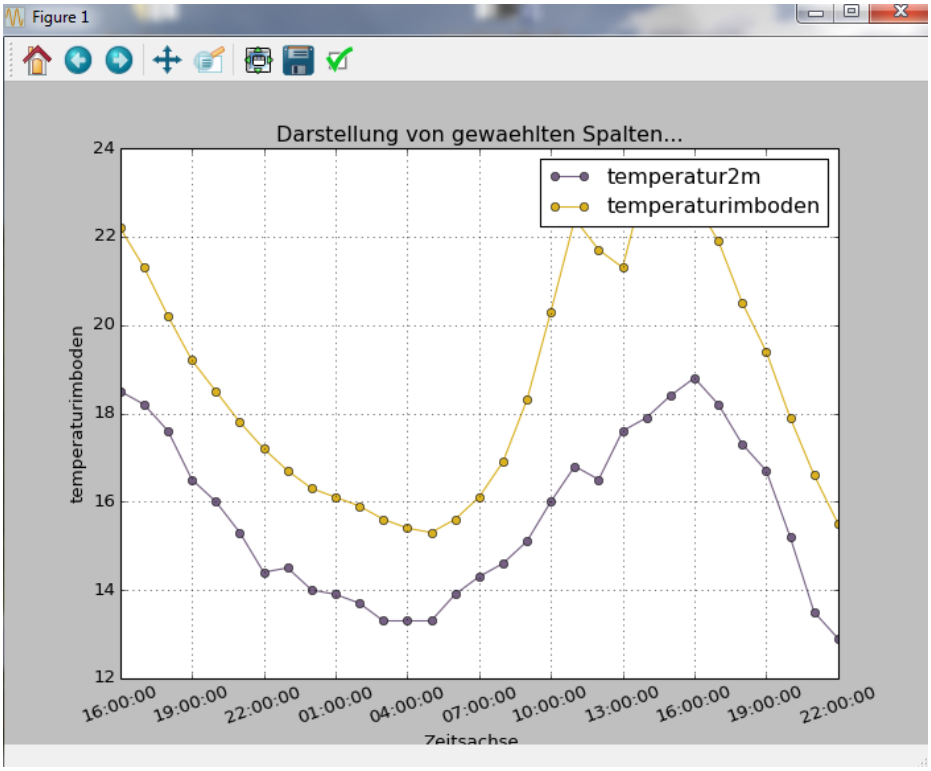


Abbildung 34: Das Plotten der geklickte Spalten aus dem Tablewidget

8.7 Manipulation der Datenwerte in der Datenbank

Die Daten in der Datenbank sind mit Zeit und Ohne Zeit in verschieden Arten(Tag, Stunden, Viertelstunden, Minuten und Sekunden) gespeichert. Die Datenbank speichert die Daten nach Datum und ermöglicht diese Daten nach mehreren Jahre wieder zu haben. Um der Benutzer nach seinem wünschten Zeit die Daten aus der Datenbank in der Tablewidget rauszuholen, wird eine Referenzzeit erstellt. Mit dem gewählten und bestätigten Zeitintervall konnte die Referenzspalte sich Zeile nach Zeile im Tablewidget deutlich präsentieren. Nach der Wahl und Bestätigung der Zeitintervall, wird ein Signal Triggered geschickt um die Funktion Auswahl_Option aufzurufen. In dieser Funktion muss erstmal die Eingabe des Datums geprüft werden. Eine Warnung stoppt der Verlauf der Funktion bei falschen Eingabe und in Gegenteil prüft die Funktion welche Optionen durch die Radiobutton gewählt wurden. Die Bestätigung der geklickte Radiobutton führt erstmal zur Verbindung der Datenbank und zur Bearbeitung der SQl_Query. Die SQL_Query generiert die Referenzzeit mit der Funktion SQL CAST Generate_series() in Betracht der Eingabe der Benutzer mit der umgewandte Datentyp. Die CAST Generate_series() behält drei Komponente(Starten, Stoppen, Schritt-Intervall) und erzeugt Reihe nach Reihe vom Datum von Anfang an zu stoppen mit einer Schrittweite von Schritt. Wenn die SQL-Query ohne Fehler nach mit Erfolg erreicht, wird als Liste an die Funktion Cur.Fettchall() zugewiesen. Um die Erste Spalte im Tablewidget zu bekommen, sollte erstmal abfragt werden ob eine Spalte als erste eingekommen ist. Die erste Spalte nach der Prüfung bekommt als Spaltenname Date damit dies nicht nur das Zeitintervall sondern auch die Spaltenname behält.

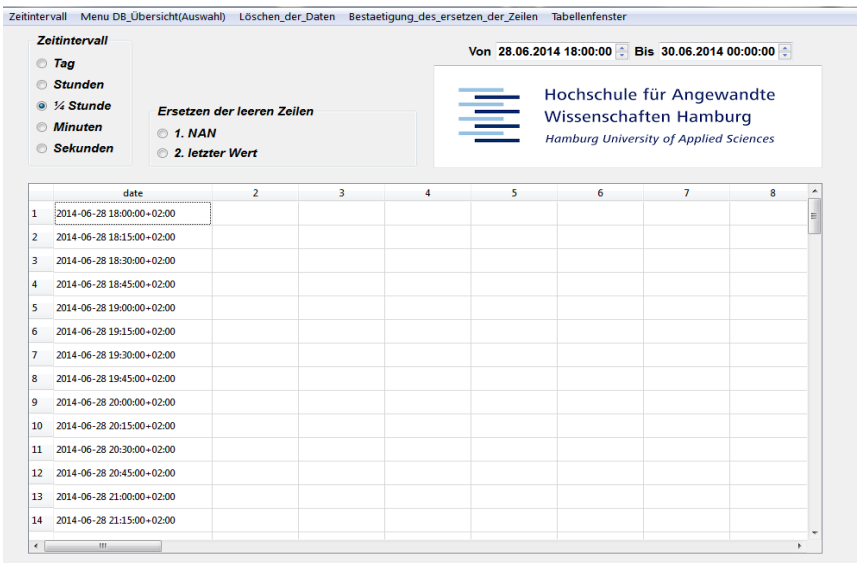
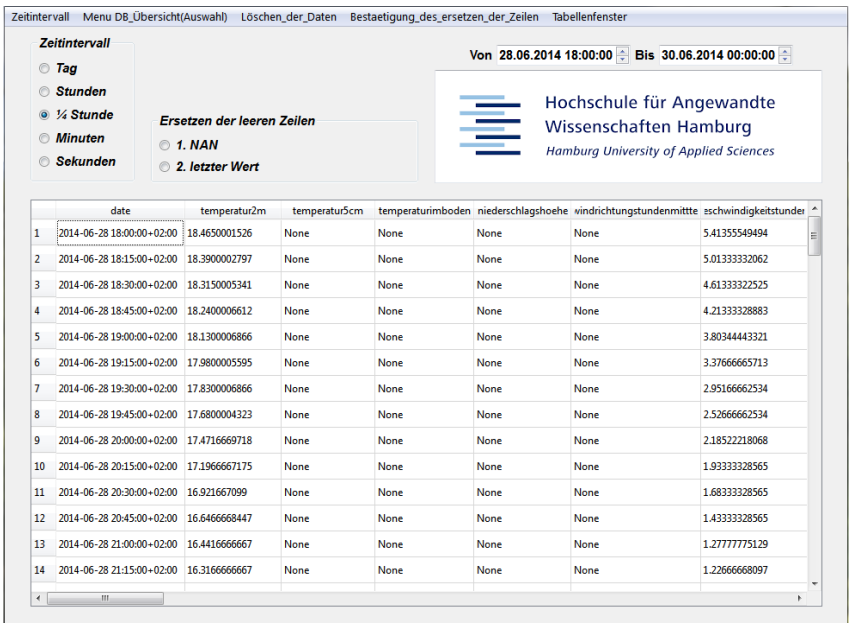


Abbildung 35: Auswahl des Zeitintervalls und Darstellung der Referenzspalte im Tablewidget

8.7.1 Query Mittelwertbildung, wenn zu viele Datenwerte für Intervall vorhanden sind

query_mittelwert_alle = "SELECT t1.date, (Select avg(%s) From %s.%s where %s.%s.date >= t1.date AND %s.%s.date < t1.date + interval '%s') as %s From (Select cast (generate_series ((SELECT(TIMESTAMPTZ '%s')), (SELECT (TIMESTAMPTZ '%s')) ,'%s') as TIMESTAMPTZ) as date) as t1 order BY t1.date " % (spalte, schemata, tabelle, schemata, tabelle, schemata, tabelle, interval, spalte, anfang_datum_geandert, ende_datum_geandert, interval).

Die SQL_Query selektiert alle einer Tabelle um die Mittelwert für jeder Spalte zu mit dem Intervall, eine SQl_funktion und die Verknüpfung TIMESTAMPTZ zu bilden. Der Mittelwert ist hier geschrieben wenn die Länge der gewählten Spalte größer als die Länge der Referenzspalte ist. In der Query wird erstmal nach Datum der Spalte mit dem Intervallsdatum verglichen. Wenn dieser Bedingung erfüllt ist, generiert die Funktion cast(generate_series) der Mittelwert der Werte nach der Anfangsdatum, Endesdatum und das Intervall mit der umgewandte Datentyp. Die Mittelwert für eine Spalte präsentiert sich in Betracht mit dem Intervall von jeder Zeile von Anfangsdatum bis Endesdatum.



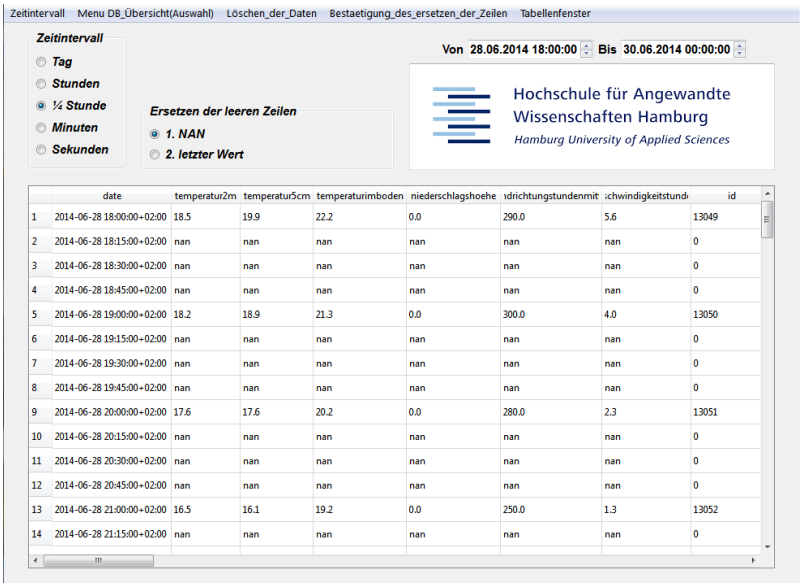
	date	temperatur2m	temperatur5cm	temperaturimboden	niederschlagshoehe	windrichtungstundenmittte	eschwindigkeitstunder
1	2014-06-28 18:00:00+02:00	18.4650001526	None	None	None	None	5.41355549494
2	2014-06-28 18:15:00+02:00	18.3900002797	None	None	None	None	5.01333332062
3	2014-06-28 18:30:00+02:00	18.3150005341	None	None	None	None	4.61333322525
4	2014-06-28 18:45:00+02:00	18.2400006612	None	None	None	None	4.21333328883
5	2014-06-28 19:00:00+02:00	18.1300006866	None	None	None	None	3.80344443321
6	2014-06-28 19:15:00+02:00	17.9800005595	None	None	None	None	3.37666665713
7	2014-06-28 19:30:00+02:00	17.8300006866	None	None	None	None	2.95166662534
8	2014-06-28 19:45:00+02:00	17.6800004323	None	None	None	None	2.52666662534
9	2014-06-28 20:00:00+02:00	17.4716669718	None	None	None	None	2.18522218068
10	2014-06-28 20:15:00+02:00	17.1966667175	None	None	None	None	1.93333328565
11	2014-06-28 20:30:00+02:00	16.921667099	None	None	None	None	1.68333328565
12	2014-06-28 20:45:00+02:00	16.6466668447	None	None	None	None	1.43333328565
13	2014-06-28 21:00:00+02:00	16.4416666667	None	None	None	None	1.27777775129
14	2014-06-28 21:15:00+02:00	16.3166666667	None	None	None	None	1.22666668097

Abbildung 36: Bildung der Mittelwert von Minutenwerte in Viertelstundenwert

8.7.2 Query um fehlende Datenwerte mit NANs zu füllen

query_2 = "select t1.date, case when t2.%s is NULL then 'NAN' else t2.%s end from (select cast (generate_series ('%s'::timestamptz, '%s', '%s') As timestamptz) as date) as t1 left join %s.%s t2 USING (date) order by t1.date"% (spalte, spalte, anfang_datum_geandert, ende_datum_geandert, interval, schemata, tabelle).

Die Query ersetzt die leere Zeilen die, die Datenwerte fehlen. Die Leere Zeilen werden mit Hilfe der SQL_Funktion(cast_generates_series) , Verknüpfung(Left Join und Timestamptz) und das Schlüsselwort USING Mit NAN statt NULL gefüllt. Es wird in der Query nach dem ersten Datum unter der Bedingung, wenn die leere Zeile mit NULL aufgetaucht ist, selektiert. Die Funktion cast_generate_case erzeugt an die Leere Zeile das Wort NAN nach Anfangsdatum, Endesdatum und Intervall mit der umgewandte Datentyp. Die Verknüpfung ordnet das Datum wo das erzeugte Wort sich referenziert. Mit der Verknüpfung Timestamptz stellt die Zeit nach dem Datum, Uhrzeit und time Zone vor.



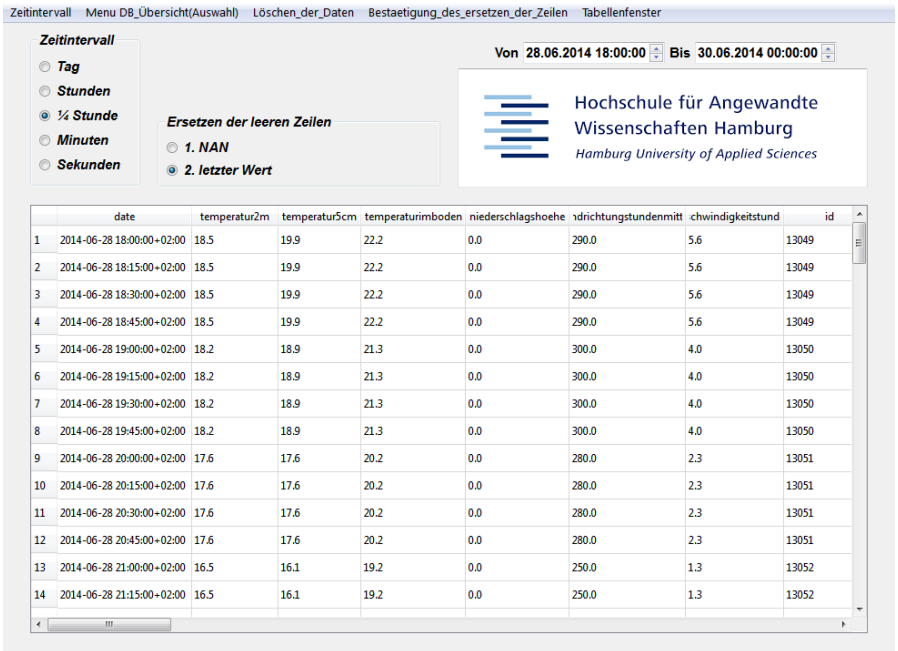
	date	temperatur2m	temperatur5cm	temperaturimboden	niederschlagshoehe	idrichtungstundenmit	schwindigkeitstund	id
1	2014-06-28 18:00:00+02:00	18.5	19.9	22.2	0.0	290.0	5.6	13049
2	2014-06-28 18:15:00+02:00	nan	nan	nan	nan	nan	nan	0
3	2014-06-28 18:30:00+02:00	nan	nan	nan	nan	nan	nan	0
4	2014-06-28 18:45:00+02:00	nan	nan	nan	nan	nan	nan	0
5	2014-06-28 19:00:00+02:00	18.2	18.9	21.3	0.0	300.0	4.0	13050
6	2014-06-28 19:15:00+02:00	nan	nan	nan	nan	nan	nan	0
7	2014-06-28 19:30:00+02:00	nan	nan	nan	nan	nan	nan	0
8	2014-06-28 19:45:00+02:00	nan	nan	nan	nan	nan	nan	0
9	2014-06-28 20:00:00+02:00	17.6	17.6	20.2	0.0	280.0	2.3	13051
10	2014-06-28 20:15:00+02:00	nan	nan	nan	nan	nan	nan	0
11	2014-06-28 20:30:00+02:00	nan	nan	nan	nan	nan	nan	0
12	2014-06-28 20:45:00+02:00	nan	nan	nan	nan	nan	nan	0
13	2014-06-28 21:00:00+02:00	16.5	16.1	19.2	0.0	250.0	1.3	13052
14	2014-06-28 21:15:00+02:00	nan	nan	nan	nan	nan	nan	0

Abbildung 37: Bildung des Einsetzen der Werte mit dem Wort NAN statt Wert in leeren Zeilen

8.7.3 Query um fehlende Datenwerte mit dem Letzten bekannten Datenwert zu ersetzen

query_voriger_wert_2 = "select t1.date, (select %s from %s.%s where date between '%s' and t1.date order by date desc limit 1) from (select cast (generate_series ('%s'::timestamptz, '%s', '%s') As timestamptz) as date) as t1 left join %s.%s t2 USING (date) order by t1.date" %(spalte, schemata, tabelle, anfang_datum_geandert, anfang_datum_geandert, ende_datum_geandert, interval, schemata, tabelle) cur.execute(query_voriger_wert_2)

Die Query zum Ersetzen die leere Zeile mit den vorigen Werten, funktioniert mit viele Schlüsselwörter, Verknüpfungen und SQL_Funktion. Die Query zeigt mit dem Schlüsselwort SELECT die Spalte mit einer Festlegung der Suchbedingung, die die zurückgegebenen Zeilen einschränkt, an. Die Klausel desc Lmit 1 sortiert die Werte in der angegebene Spalte in absteigende Reihenfolge vom höchstens bis eine Zeile mit LIMIT 1. Die SQL_funktion cast_generate_series erzeugt die Werte unter die Verwendung der Date Spalte(t1.date) auf ganze links der betreffenden Spalten mit der umgewandelte Datentyp und ORDER BY t1:DATE legt die Sortierung für das Resultset nach Date-Spalte fest.



	date	temperatur2m	temperatur5cm	temperaturimboden	niederschlagshoehe	ndrichtungstundenmitt	chwindigkeitstund	id
1	2014-06-28 18:00:00+02:00	18.5	19.9	22.2	0.0	290.0	5.6	13049
2	2014-06-28 18:15:00+02:00	18.5	19.9	22.2	0.0	290.0	5.6	13049
3	2014-06-28 18:30:00+02:00	18.5	19.9	22.2	0.0	290.0	5.6	13049
4	2014-06-28 18:45:00+02:00	18.5	19.9	22.2	0.0	290.0	5.6	13049
5	2014-06-28 19:00:00+02:00	18.2	18.9	21.3	0.0	300.0	4.0	13050
6	2014-06-28 19:15:00+02:00	18.2	18.9	21.3	0.0	300.0	4.0	13050
7	2014-06-28 19:30:00+02:00	18.2	18.9	21.3	0.0	300.0	4.0	13050
8	2014-06-28 19:45:00+02:00	18.2	18.9	21.3	0.0	300.0	4.0	13050
9	2014-06-28 20:00:00+02:00	17.6	17.6	20.2	0.0	280.0	2.3	13051
10	2014-06-28 20:15:00+02:00	17.6	17.6	20.2	0.0	280.0	2.3	13051
11	2014-06-28 20:30:00+02:00	17.6	17.6	20.2	0.0	280.0	2.3	13051
12	2014-06-28 20:45:00+02:00	17.6	17.6	20.2	0.0	280.0	2.3	13051
13	2014-06-28 21:00:00+02:00	16.5	16.1	19.2	0.0	250.0	1.3	13052
14	2014-06-28 21:15:00+02:00	16.5	16.1	19.2	0.0	250.0	1.3	13052

Abbildung 38: Bildung des Einsetzen der Werte mit jeder vorigen Wert in leeren Zeilen

9 Zusammenfassung und Ausblick

Das Ziel der Arbeit war es eine graphischen Oberfläche zu erstellen, mit der es möglich ist die Mess- und Simulationsdaten von der Datenbank C4DSI abzurufen, zu visualisieren und als CSV-Datei zu exportieren. Mit Hilfe dieses Tools ist es möglich ohne SQL Kenntnisse auf die Daten zuzugreifen. Der Nutzer kann interaktiv mit der Datenbank kommunizieren. Die Daten lassen sich in unterschiedlichen Zeitintervallen(wie z.B Minuten, Sekunden, Viertelstunden, Stunden und Tag) aus der Datenbank abrufen. Um die Daten in demselben Zeitintervallformat zu haben, mussten die Daten in der Datenbank entwerder zu Mittelwerten zusammengefasst werden, wenn zu viele Werte in dem Zeitintervall liegen, oder wenn keine Daten in dem Zeitintervall liegen, lassen sich diese entwerder als NaNs (Not a Number) kennzeichnen oder die letzten bekannten Daten werden wieder verwendet.
Die GUI ist mit dem Qt-Designer aufgebaut und wurde nach Python als Klasse übersetzt. Die Übersetzung erfolgt manuell über den Aufruf der Kommandozeile. Danach kann die erzeugte Klasse in ein Python Skript importiert werden und mit weiteren Funktionen realisiert werden. Eine wesentliche Schnittstelle zur Kommunikation mit der Datenbank stellt das Psycopg2 dar. Diese muss ebenfalls geladen sein. Psycopg2 ist ein PostgreSQL Datenbankadapter, dieser stellt Funktionen für die Kommunikation mit der Datenbank zur Verfügung.

Die Software PgAdmin 3 ermöglicht es über die Datenbanksprache PostgreSQl die Daten aus der Datenbank C4DSI abzurufen. Die Implementierten Postgresql Abfragen sind zunächst mit der Software PgAdmin 3 getestet worden. Diese Abfragen sind dann so flexibilisiert worden, dass über das Navigation-Tool die ausgewählten Schemata, Tabellen und Spalten der Abfrage zugewiesen werden und dann an die Datenbank gesendet wird. Die Daten, die von der Datenbank zurückgesendet werden, werden in dem Navigation-Tool in Tabellenform angezeigt. Damit die Daten Einheitlich abgerufen können, musste in den Datenbanktabellen eine Pflichtspalte definiert werden, diese trägt den namen date und besitzt den Datentyp TimestampTZ(Timestamp with timezone). Diese Spalte stellt einen eindeutigen Zeitstempel für die Daten dar, wodurch die Daten selektiert und in unterschiedlichen Zeitintervallen abrufen lassen.
Das Navigation-Tool ist mit Standardfunktionen ausgestattet. Erweiterungen für das Tool wären zum einen, wenn bei fehlende Daten, in ausgewählten Zeitintervallen, die Option einer Interpolation möglich wäre. Und zum anderen wäre die Erweiterung einer automatischen Erkennung von Abhängigkeiten(Fremdschlüssel). Da es vorkommt, dass in einer Tabelle ein

Fremdschlüssel vorhanden ist und die eigentliche Bezeichnung (oder auch andere Informationen) in einer anderen Tabellen liegt und diese vielleicht angezeigt werden soll. Außerdem könnte das Navigations-Tool mit weiteren Such/Auswahloptionen, wie kleiner als größer gleich für die angezeigten Daten implementiert werden.

10 Anhang

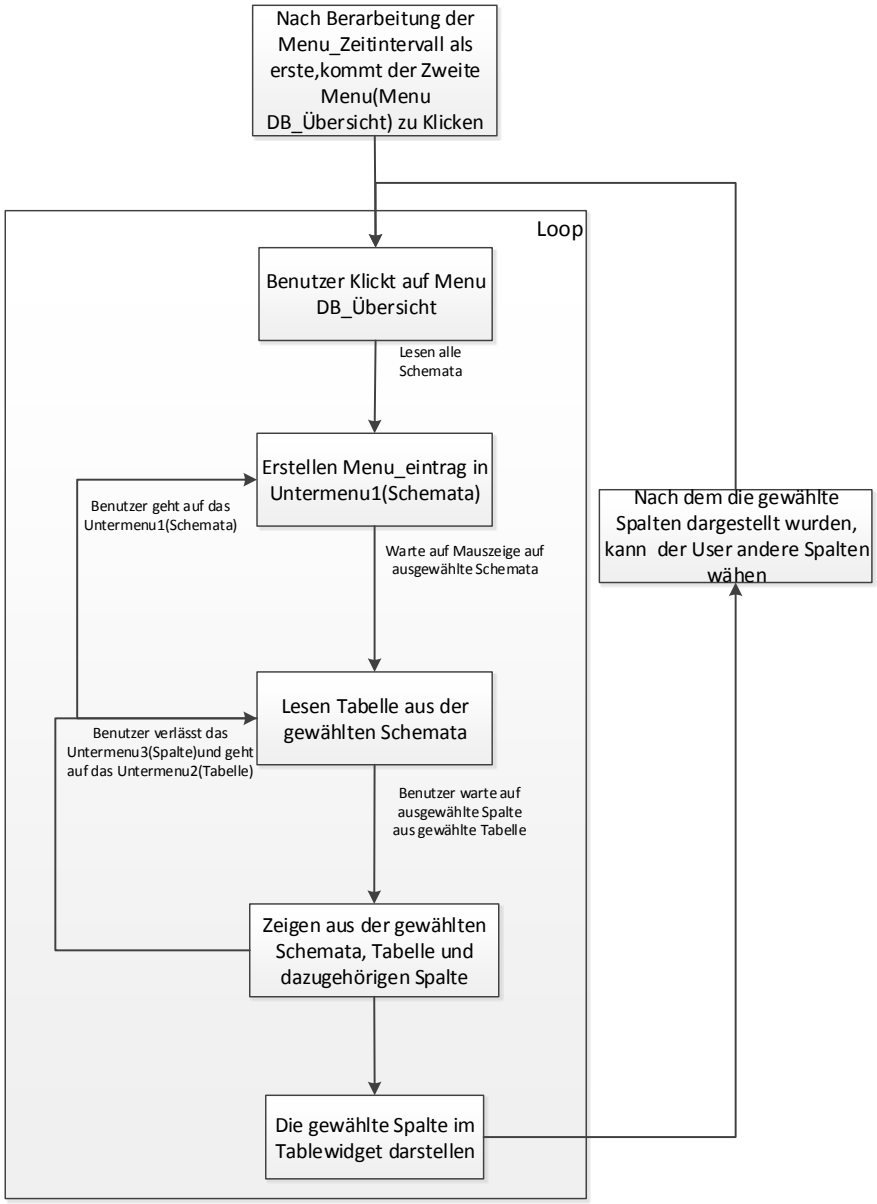


Abbildung 39: Flussdiagramme zur Abruf der Daten mit Menu DB_Übersicht

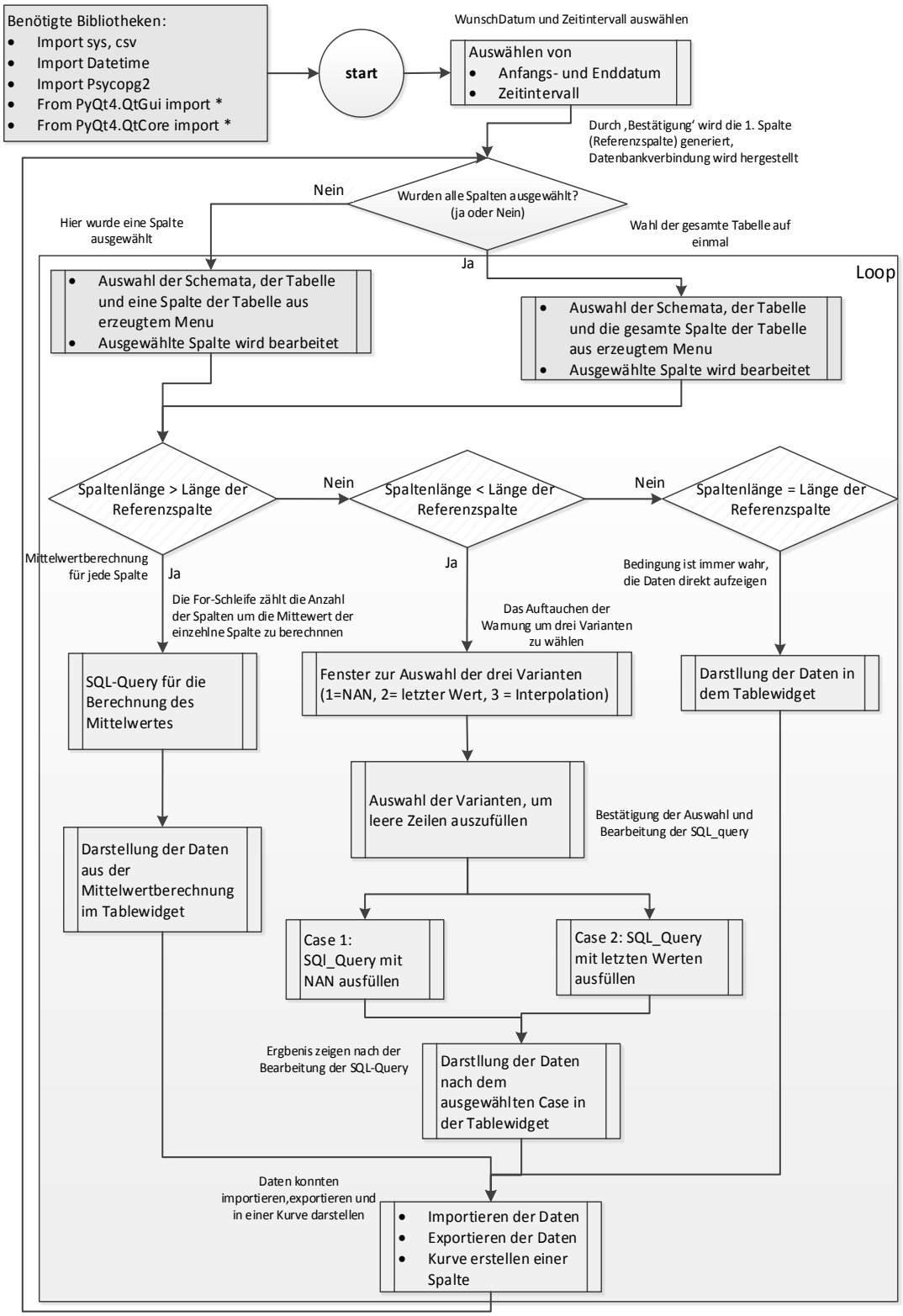


Abbildung 40: Flussdiagramme zur Abruf einer Gesamte Tabelle und für einer Spalte

Inhalt CV

Der Inhalt der Beiliegenden CD konnte die Dateien beschrieben werden

Datei	Beschreibung
C4DSI.py	Python-Datei als GUI für das Anmelder-Fenster
Navigations_GUI.py	Python-Datei als GUI für Navigation Tool
Im_C4DSI.py	Python-Datei für die Abruf , die Manipulation und Visualisierung der Mess- und Simulationdaten vom C4DSI-Datenbank
Berechnung1.py	Python-Datei als GUI für zur Multiplizieren
Multiplikation_einer_zahlen.py	Python-Datei zur Eingabe des Werts und das Ergebnis zeigen
Die geschriebene Arbeit	Barchelorarbeit in Pdf-Datei

11 Literaturverzeichnis

[HA02] Herbolsheimer, Andreas: Datenbank Programmierung. 1.Auflage, München. Addison-Wesley Verlag, 2002

[HR03] Steiner, Reine´: Grundkurs Relationale Datenbanken. 5.erweiterte und Verbesserte Auflage, Braunschweig.Vieweg Verlag, 2003

[SH04] Sauer, Hermann: Relationale Datenbanken.4.Auflage, München. Addison Wesley Longman Verlag, 1998

[DA05] Downey, Allen B: Programmieren Lernen mit Python. 2. Auflage, Köln. O´Reilly Verlag, 2012

[HA06] Heuer, Andreas; Saake Grenter: Datenabank:Konzepte und Sprache. 2. Auflage, Bonn.Verlag, 2000

[CT07] Connolly, Thomas; Begg, Carolyn; Strachan; Anne: Datenbanksysteme Eine Praktische Anleitung zu Design Implementierung und Management .1. Auflage, München.Addison-Wesley Verlag,2002

[TM08] Throll, Marcus; Bartosch, oliver: Einstieg in SQL verstehen, einsetzen, nachschlagen. 2.Auflage, Bonn.Verlag, 2010

[SAP05] Kunick, Stefan(2005): die erste Schritte in der Administration der Postgre SQL-Datenbank
Internet:http://www.postgres.de/download/1Admin_V3.pdf,
abgerufen am 17.06.2015

[PAP13] Regionales Rechenzentrum für Niedersachsen(2013): PostGre-SQL-Anweisungen in PgAdmin
Internet:htpp://www.luis.uni-hannover.de/fileadmin/kurse/material/PostGre_SQL.pdf
abgerufen am 22.06.2015

[WE07] Chemnitzer Linux-Tage(2007):Workshop, Entwicklen mit Qt4
Internet:http://www.qt4-buch.de/download/clt07-workshop.pdf,
Abgerufen am 24.062015

[GS15] Getting Started With PYQT and Qt Designer(2015):
Internet:http://www.nikolak.com/pyqt-qt-designer-getting-started,
Abgerufen am 25.06.2015

[CA07] Creating GUI Applications with PyQt and Qt Designer (2007):
Internet:http://www.boddie.org.uk/david/Projects/Python/Qt/

PyCon_UK_2007_PyQt_and_Qt_Designer.pdf,
abgerufen am 26.062015

[AD10] Aufbau einer Datenbank am Beispiel einer Adressdatenbank(2010):
Internet:http://www.robert.familiegrosskopf.de/opneoffice_adressen/Datenbankaufbau.pdf,
Abgerufen am 20.06.2015

[CP12] Computergrundlagen Programmieren in Python(2012):
Internet:http://www.icp.uni-stuttgart.de/~icp/mediawiki/images/d/D9/Slides-python_CG_WS12.pdf,
abgerufen am 22.09.2014

[PT12] Python Tutorial(2012):
Internet:http://www.marvin.cs.uidaho.edu/Teaching/CS15/pythonTutorial.pdf,
abgerufen am 15.11.2014

[PS08] Python & Statistische Methoden der Datenanalyse(2008):
Internet:http://www.zeuthen.desy.de/~kolanosk/smd_ss08/uebungen/pythonForStat.pdf,
abgerufen am 24.09.2014

[DS14] Digitale Signalverarbeitung mit Python(2014):
Internet:http://www.christian-muenker.de/sites/default/files/201203-Muenker-Intro_Python_DSP.pdf,
abgerufen am 20.12.2014

Erklärung zur Bachelorarbeit

Ich versichere, dass ich meine Bachelorarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Hamburg, 06.10.2015

_____

Herve Ghislain Ngamadjeu Youssa