Universität Hamburg
Department Informatik
Wissenschaftliches Rechnen
Dr. J. Kunkel, H. Hatzel, J. Gresens

Übungsblatt 3 zur Vorlesung
Big Data Analytics
im WiSe 2015/2016
Abgabe: 05.11.2015, 465 P

Bitte dokumentieren Sie die benötigte Bearbeitungszeit für die einzelnen Aufgaben in `bearbeitungszeit.txt`. Bitte denken Sie auch daran uns Feedback zur Veranstaltung zu geben:

`http://goo.gl/forms/B01IIDHvW2`

Bei der Abgabe von Source Code kommentieren Sie diesen bitte.

1 Beobachtungsdaten im NetCDF (Python & R) (150 P)

NetCDF ist ein weit verbreitetes wissenschaftliches Datenformat. Es erlaubt die Ablage von N-dimensionalen Daten gemeinsam mit Beschreibungen (Metadaten) dieser Daten.
Berechnen Sie den Mittelwert von NetCDF Daten mit den Programmiersprachen Python und R: Errechnen Sie die mittlere Temperatur für jedes Jahr im Messzeitraum in Hamburg. Nehmen Sie hierfür die Temperatur in zwei Meter Höhe. Geben sie den Mittelwert in $°C$ aus.

1.1 Daten-Exploration

Verwenden Sie Methoden der deskriptiven Statistik mit R um die Gesamtheit der Daten zu analysieren als auch das Sample von Hamburg. Vergleichen Sie dann die Ergebnisse von Hamburg mit denen von Tokio.
Verwenden Sie hierfür:

- Zusammenfassungen (`summary()`)
- Histogramme
- Plots der Dichte
- Zeitreihen der Temperaturverläufe

Dokumentieren Sie einige Beobachtungen. Erstellen Sie mit LaTeX ein PDF, das alle Plots, Ausgaben von R und Ihre dazu passenden Beobachtungen enthält.

1.2 Temperaturunterschiede

Nun wollen wir die Datenpunkte ermitteln, deren Temperatur von der ihrer Nachbarn deutlich abweicht. Diese Datenpunkte sind für Analysen besonders interessant, da sie nur mit einem Vorhersagemodell mit hoher Auflösung berechnet werden können.
Hierfür gehen Sie wie folgt vor:

- Ermitteln Sie für einen festen Zeitpunkt und jeden Punkt der 2D-Matrix die erwartete Temperatur basierend auf der Temperator seiner Nachbarn $et(x, y) = \frac{t(x-1,y)+t(x+1,y)+t(x,y-1)+t(x,y+1)}{4}$. Hierbei gehen wir davon aus, dass t die Matrix für die Temperatur an einem gewählten Zeitpunkt enthält.
- Bilden Sie die Differenz zum aktuellen Wert: $td(x, y) = t(x, y) - et(x, y)$
- Erstellen Sie hierfür ein Histogramm für einen beliebigen (von Ihnen gewählten) Zeitpunkt

- Schreiben Sie eine Funktion für das Ermitteln des mittleren und maximalen Temperaturunterschied über die gesamte Matrix für einen Zeitpunkt über td.
- Plotten Sie den mittleren und maximalen Temperaturunterschied in einer Zeitreihe. Beschreiben Sie kurz Ihre Beobachtungen.

1.3 Hinweise

Mithilfe des Kommandozeilen-Werkzeugs `ncdump` können Sie sich die Struktur der NetCDF Datei `/home/hatzel-/bigdata/atls14-CyGl1B.nc` ansehen. Auf dem Rechenknoten `abu1` sind die notwendigen Pakete installiert. Nehmen Sie die folgenden Koordinaten an:

```
Hamburg                    Tokio
longitude:  10.5           longitude:  140.25
latitude:   53.25          latitude:   35.25
```

1.3.1 Codegerüst für Python

In Python sollten Sie das Paket `netCDF4` verwenden[1]. Vorsicht: Die Zeitstempel in der Datei sind in Stunden nach 1900 angegeben, d.h. eine Konvertierung ist nötig.

```python
import netCDF4
from datetime import datetime, timedelta

file = netCDF4.Dataset("atls14-CyGl1B.nc")
print(file.variables)
print(file.dimensions)

# konvertiere 333 Stunden beginnend mit dem Jahr 2000 zu einem neuen Datum
date = datetime(2000,1,1) + timedelta(hours=int(333))
# Schauen Sie sich die Datentypen von datetime an
# help(datetime)

file.close()
```

1.3.2 Codegerüst für R

In R lassen sich NetCDF Dateien mit dem Paket `ncdf4` einlesen[2]. Eine exakte Konvertierung der Zeitstempel zu entsprechenden Jahren ist hier nicht notwendig.
Die Funktionen $which()$ und $ddply()$ könnten sich als nützlich erweisen.

```r
library(ncdf4)
library(plyr)

d = nc_open("atls14-CyGl1B.nc", write=FALSE,readunlim=TRUE, verbose=FALSE,)
# print(d) # shows information
# names(d) # zeigt Informationen über die verfügbaren Attribute, hier bspw. dim
# names(d$dim) # zeigt verfügbare Dimensionsvariablen an
d$dim$NAME_DER_VARIABLE$vals
data = ncvar_get(d, "VARNAME")

# Erstellen Sie ein Dataframe (Tabelle) mit N Zeilen mit den Werten für Hamburgddply(df, c("year"),
    ↪ summarize, count = sum(count), mean = sum(value))
df = data.frame(year=numeric(N), data=numeric(N))

# Berechnen Sie daraus den Mittelwert für jedes Jahr...
```

[1] https://github.com/Unidata/netcdf4-python/
[2] https://cran.r-project.org/web/packages/ncdf4/ncdf4.pdf

Abgabe:

`1-hamburg.(py|R)`	Der Source-code in Python bzw. R.
`1-auswertung.pdf`	Ihre Datenexploration in R.

2 Einfache Datenverarbeitung in PostgreSQL (120 P)

Den kleinen Ausschnitt der Wikipedia wollen wir nun in eine PostgreSQL Datenbank importieren und verarbeiten. Dies geschieht in folgenden Schritten:

1. Erstellen Sie für die Daten ein triviales Datenbankschema und führen Sie dieses auf der interaktiven Shell `psql` aus.
2. Schreiben Sie ein Programm in Python, welches einen Ausschnitt der um Annotationen bereinigte CSV-Datei (`wiki-clean.csv`) verarbeitet und mittels SQL direkt in die Datenbank importiert.
3. Schreiben Sie ein weiteres Programm in Python, das einen SQL-Select Aufruf durchführt, welcher die Worthäufigkeiten eines jeder einzelnen Artikel zählt (Groß/Kleinschreibung ignorieren). Dokumentieren Sie zusätzlich den SQL-Aufruf für die Ermittlung der Worthäufigkeiten eines Artikels.
4. Speichern Sie das Ergebnis in einer CSV-Datei in der für jede ArtikelID ein Dictionary mit den Wörtern und dessen Häufigkeiten enthält.
5. Messen Sie die Laufzeit Ihres Programms für den Import und die Verarbeitung der Daten mit dem `time` Modul.

2.1 Hinweise

Für jede Gruppe haben wir eine kleine PostgreSQL-Datenbank angelegt (das Passwort und Ihre Gruppen Nr. sollten Sie per Email erhalten haben). Um diese interaktiv nutzen zu können, verwenden Sie:
`psql -U group<NR> -h abu1`
Für den Import verwenden wir das Modul `psycopg2`[3]. Ein Codegerüst für Python ist gegeben.
Die PostgreSQL Funktionen zur Verarbeitung von regulären Ausdrücken[4] und `unnest()` könnten sich als hilfreich erweisen.

2.1.1 Python Codegerüst

```
#!/usr/bin/python3
import psycopg2
import sys
import json
import itertools

# for the importer use
def parseCSV():
    fd = open("wikipedia-text-tiny.csv", "r")
    #  TODO process lines
    return lines

def main():
  # connect to the database
  conn = psycopg2.connect("host='abu1' dbname='group1' user='group1' password='secret'" )

  # conn.cursor() returns a cursor object which allows to execute SQL queries
  cursor = conn.cursor()
```

[3] https://wiki.postgresql.org/wiki/Using_psycopg2_with_PostgreSQL
[4] http://www.postgresql.org/docs/9.4/static/functions-matching.html

```
19
20    # run a SQL query
21    cursor.execute("SELECT * FROM tableX")
22    # retrieve response tuples from the query
23    records = cursor.fetchall()
24
25    # escaping of strings is performed by psycopg, prevents errors and SQL injections
26    cursor.execute("INSERT into X values(%s, %s)", (5, "te'st'") )
27
28    # Commit your transaction to persist changes
29    conn.commit()
30
31    # to output word frequencies into a CSV file:
32    out = open('output.csv', 'w')
33    cout = csv.writer(out)
34
35    # write article ID, total number of words, and all words with their frequencies
36    cout.writerow([4, 4, json.dumps({"word1":3, "word2":1})])
37
38    # to convert an array of tuples to a dictiory e.g. from x = [["word1",3], ["word2",1]]
39          # d = dict(x)
40
41
42  if __name__ == "__main__":
43    main()
```

Abgabe:

`2-wortcount.txt`	Ihre SQL-Befehle für die Erstellung des Schemas und der SQL Abfrage und die Ergebnisse der Zeitmessungen.
`2-wortcount-import.py`	Das Python-Programm, welches die CSV-Datei importiert.
`2-wortcount.py`	Das Python-Programm für das Zählen der Wikipedia-Daten.

3 Relationales Schema für die Wikipedia (90 P)

Entwickeln Sie Ihr relationales Schema von *Blatt 2, Aufgabe 3 („Erstellung eines Datenmodells für Wikipedia“)* weiter. Überprüfen Sie ob dieses normalisiert ist, falls nicht, normalisieren Sie es. Dokumentieren Sie das Schema als ER-Diagramm (bspw. unter Verwendung von OpenOffice oder Inkscape). Zeichnen Sie ebenfalls gewählte Schlüssel ein.
Erstellen Sie mit SQL eine Sicht, die den Zugriff auf die Worthäufigkeiten einzelner Artikel direkt ermöglicht, d.h. beispielsweise Anfragen der Form:
`SELECT artikel, count FROM view WHERE wort == 'bigdata'`
korrekt verarbeitet.
Entwickeln Sie für Ihr Datenbankschema SQL Anfragen, welche die in *Blatt 2, Aufgabe 3 („Erstellung eines Datenmodells für Wikipedia“)* dargestellten Operationen theoretisch umsetzen könnten.
Schreiben Sie eine SQL-Anfrage um die Worthäufigkeiten eines bestimmten Wortes bei allen mit einem Artikel verwandten Artikel zu ermitteln[5] und geben Sie diese Artikelnamen aus, d.h.
`SELECT verwandterArtikelName, wordCount FROM ....`
Nutzen Sie hierfür einen JOIN mit ihrer SQL Sicht. Diskutieren Sie die die Verwendung von Indizes für die einzelnen Relationen.

[5] Bislang haben Sie die verwandten Artikel noch nicht explizit berechnet, das spielt aber keine Rolle.

Abgabe:

`3-schema.pdf`	Ihr normalisiertes Datenbank-Schema als ER-Diagramm inklusive gewählte Schlüssel und Diskussion der Indexe.
`3-view.txt`	Ihr SQL-Befehl zur Erstellung der Sicht.
`3-operations.txt`	SQL-Befehle für die einzelnen Operationen mit Kommentaren.
`3-join.txt`	SQL-Befehle für den Join zwischen Artikeln.

4 Relationales Schema für die Beobachtungsdaten (60 P)

Entwickeln Sie ein einfaches normalisiertes relationales Modell für die Beobachtungsdaten, die in Aufgabe 1 verwendet werden. Integrieren Sie neben der mittleren Temperatur auch die Niederschlagsmenge in das Modell.
Schreiben Sie eine SQL-Anfrage um die mittlere Temperatur für jedes dokumentierte Jahr in Hamburg zu ermitteln. Erweitern Sie die Anfrage um bei der Aggregation nur die Messungen zu berücksichtigen bei denen die Niederschlagsmenge einen bestimmten Schwellenwert überschreitet.
Nun nehmen wir an, dass wir zusätzliche Informationen in der Datenbank erfassen. Berücksichtigen Sie folgende Entitäten:

- Städte und Länder mit Ihren Koordinaten.
- Einwohner (Bürger) mit ihren Wohnort (Koordinaten) und den Beziehungen **befreundet mit** und **arbeitet für**.

Erstellen Sie ein ER-Diagramm und denken Sie daran Ihr Datenbankschema zu normalisieren.
Schreiben Sie eine SQL-Anfrage um an einem festen Tag die Wetterbedingungen für alle Freunde eines Einwohners zu ermitteln[6].

Abgabe:

`4-beobachtung.txt`	SQL-Anfragen mit Kommentaren.
`4-er.pdf`	ER-Diagramme für das erweiterte Modell.

5 Data-Warehouse Schema für die Beobachtungsdaten (45 P)

Erzeugen Sie ein faktenbasiertes Schema (OLAP-Cube) für die Beobachtungsdaten und Einwohnerdaten aus Aufgabe 4. Dokumentieren Sie die Entitäten der Faktentabelle und Attribute der Dimensionen[7]. Die Beziehungen der Bürger können Sie bei der Modellierung ignorieren oder alternativ durch ein sinnvolles Modell ersetzen.
Erstellen Sie ein Star-Schema, das den OLAP-Cube in ein relationales Modell abbildet. Dokumentieren Sie den SQL-Befehl zur Erstellung des relationalen Modells.
Schreiben Sie eine SQL-Anfrage für die Ermittlung des gesamten Niederschlags am Wohnort eines Einwohners in den einzelnen Jahren. Gehen Sie davon aus, dass keine Aggregation (entlang der Dimensionen) vorgenommen wurden und Sie alle Fakten verarbeiten müssen.

Abgabe:

`5-olap-schema.txt`	Beschreibung des OLAP Schemas (Fakten, Dimensionen).
`5-rolap.txt`	SQL-Befehl für die Erstellung des Star-Schemas und der Anfrage.

[6] Hinweis: Wir haben für die Einwohner hierbei keine Testdaten vorgegeben, falls Sie Ihre Anfrage testen wollen, so müssen Sie selbst Daten einfügen.
[7] *Hinweis:* Überlegen Sie sehr genau, was die abzulegenden Fakten sind.