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

Exercise Sheet 3
Big Data Analytics
WiSe 2016/2017
Due Date: 2016-11-11, 420 P

Please document your code, without sufficent documentation you won't recive any points.

1 Exploration Diamonds Data Set (R) (150 P)

We explore a data set of diamonds, each observation contains the price of a diamond and multiple criteria of it's quality. Document your results. Create a PDF using LaTeXor R Studio, containing your results and their analysis.

1.1 Guiding questions

1. With which measure does the price correlate most?
2. How does the price change with the quality of the cut?

1.2 Hints

Import the dataset from R's data set collection:

```
data(diamonds)
summary(diamonds)
```

Submission:

`1-analysis.pdf` A PDF document, i.e., a lab notebook with your R code and its results.

2 Relational Database Schema for Wikipedia Articles (90 P)

Create a relational schema for Wikipedia data in (at least) third normal form. Document it as an Entity-relationship diagram (use your favorite tool for visualization, we suggest either LibreOffice, Inkscape). Make sure the chosen keys are marked in the diagram.

Each article should store the following properties:

- Title
- Text
- Category (and subcategories)
- Links to related articles

The following operations should be implemented efficiently:

- Access article details based on the articles title
- Finding related articles (those that link to one another)
- Retrieving all articles of one category

Discuss the usefulness of indexes for each of the relations.

2.1 Word Distribution

Create a view that allow directly accessing a given articles word distribution. A query like:
`SELECT article, count FROM view WHERE word == ’bigdata’`
should be handled as expected.

Submission:

`2-schema.pdf`	Your normalized database schema in the form of an ER-diagram, including discussions of chosen indexes and keys.
`2-view.txt`	The SQL-Command for creating your view.
`2-operations.txt`	Each operations SQL statements, including comments.

3 Data Ingestion in PostgreSQL (Python) (120 P)

We will import a small fraction of the Wikipedia into our PostgreSQL database and work with the data:

1. Create a trivial database scheme, use the Postgres shell: `psql`, you may use a subset of the schema developed in the exercise before.
2. Write a small Python script to insert a section of the cleaned Wikipedia data into our PostgreSQL database. Read the data from `/home/bigdata/3/wiki-clean.csv` and insert it using SQL-statements.
3. Create an additional Python script which, using an SQL-select query, retrieves the counts of words for each article (ignoring the case of the words).
4. Save the result to a CSV file; a row contains the article ID and the word occurrences (as JSON or Python dictionary).
5. Use Python‘s `timeit` module to measure your program‘s runtime for importing/ingesting the data.

3.1 Hints

3.1.1 Setup

We created a PostgreSQL database for each participant. Each group can use the command below to start the interactive shell: `psql`
Use the Python module `psycopg2`[1] for importing the data into the Postgres database. We have supplied a code-skeleton for your Python code.
You will need to work with regular expression functions [2] and `unnest()` to implement as much as possible in Postgres itself.

3.1.2 Python Code

```python
#!/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
```

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

```python
    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()

  # run a SQL query
  cursor.execute("SELECT * FROM tableX")
  # retrieve response tuples from the query
  records = cursor.fetchall()

  # escaping of strings is performed by psycopg, prevents errors and SQL injections
  cursor.execute("INSERT into X values(%s, %s)", (5, "te'st'") )

  # Commit your transaction to persist changes
  conn.commit()

  # to output word frequencies into a CSV file:
  out = open('output.csv', 'w')
  cout = csv.writer(out)

  # write article ID, total number of words, and all words with their frequencies
  cout.writerow([4, 4, json.dumps({"word1":3, "word2":1})])

  # to convert an array of tuples to a dictiory e.g. from x = [["word1",3], ["word2",1]]
        # d = dict(x)


if __name__ == "__main__":
  main()
```

Submission:

3-wortcount.txt	Your SQL-Commands and the results of your runtime measurements.
3-wortcount-import.py	The Python program for importing the data.
3-wortcount.py	Your python script for counting the Wikipedia data.

4 Data-Warehouse Schema for Analytics Data (60 P)

Create a fact based schema (OLAP-Cube). Based on logs of the Wikipedia website the following data is created:

- IP-address
- country of visitor
- city of visitor
- access date
- time spent on the website
- browser's user agent

Build a useful OLAP Cube schema for the data.
Perform the following steps:

1. Discuss which data should be part of the fact table and which should be an attribute of the dimensions. Consider carefully which attribute should be a fact.

2. Create a star schema to map your OLAP-cube to a relational model. Document SQL Commands for creating your relational model.

3. Write an SQL query for retrieving the time spent on the website by users from a certain country within a specific month. Assume no aggregation within the dimensions has been done, thus, the query must process all individual facts.

Submission:

`4-olap-schema.txt`	Detailed description of your OLAP schema (facts, dimensions).
`4-rolap.txt`	SQL queries for creating and querying the star schema based on the OLAP model.