Databases and data mining TJTST19 Autumn 2013 assignments

Instructions

The goal of the demonstrations is to learn by doing and to apply and deepen the knowledge you get during the lectures.
Because this is a course at master level, we do not want to make lectures or demonstrations compulsory. You can attend the lectures or then not and the same holds for the demonstrations. Students at this level of studies should be able to decide themselves how to take the course. But I can say from experience that those students that did not make major part of the assignments have almost never got the highest grades. This is not so much because of the missing max 4 or 6 additional points on top of the points from the final exam, but rather because their answers to the problems in the final exam tend to be less complete than from those that worked on the assignments. Evidently, those that did not take part in demonstrations learned less during the course.

How to perform the demonstrations. The assignments are released on the previous Thursday or Friday, latest, in Optima. After that you should compile your solutions for the next week.
Primarily, you should come to the demonstration session as indicated in Korppi and be ready to present your solution and discuss them in the group. If it is impossible for some reason (living e.g. not in Jyväskylä, going to business trip, etc.) then you can provide the solutions to Mr. Semenov by email (Alexander.v.semenov@jyu.fi) or by giving him access to a URL where he can find the material. We also ask you to make your solution available to us before the demo session, or during the demo session. You can use a memory stick that we can attach to the USB port. The solutions should be named as explained below. We can use my computer or your own computer to show the solutions on the screen and the authors will explain the solutions to the group. If the solutions is an piece of program that should be run in our environment then you must be prepared accordingly.

Your solutions can be stored into .txt, Word, pdf, Excel, or Powerpoint file. It does not matter, which format you use, as long as the results can be shown on the screen and stored into Optima- and run if necessary against the database. Each file should contain the names of those who contributed to the solution and for xth assignment the headline Assx For instance, if two students, Smith and Chen, have worked together and compiled a solution into a Word 2007 file for assignment #3, they must name the word file containing the solution in the following way ass3_Smith_Xu.docx

We will put the solutions to a folder in Optima. For each week we will establish a separate folder. The solutions are accessible to all students who enrolled to the course, but not to a wider audience.
We will provide model solutions to the assignments, if appropriate. Hopefully we can pick them among the solutions you provide. A model solution is named assx_model_<authornames> . These will be included into the final exam material.

Finally, while gathering information from the web, always quote the source from where you got the information you present and when did you retrieve it.
Also, pay attention to the timeliness of the information. The web is full of pages that are 5-10 years old and thus the information, even if accurate 5-10 years ago, is now often outdated. This also holds for the Wikipedia articles. Always check when the article was updated the last time. And if you use Wikipedia, always give reference to the article, but also check the original sources quoted at the end of the article for accuracy of the information. If citations are missing, you should not trust the article - or those points where citations are missing according to the remarks added by the Wikipedia editors.

Assignments (demos) for week 45 (31.10.-7.11.)

Assignment 1. At the end of the lecture notes for week one there are ca. 10 articles (Reading list) that deal with the advances of the Data Management area. Some of them plus one in addition is retrieved and put into this demofolder.

Read Codd's article Codd_CACM1970 from the year 1970 in CACM. Then pick one newer article in the list and read it. Compare the two articles and find out what underlying assumptions have changed since Codd presented his approach ca. 40 years ago, in the light of the newer article. What concerns do the writers have in 1990 (3rdGDBSManifesto_1990), 1995 (date3rdRmanifesto1995), 1996 (Objectdbs-bancilhon1996), 2005 (atomicmanifesto_2005) and 2011 (meijerCACM2011), respectively? Do not be worried, if you do not understand everything that is written in the articles. The issues will be clarified during demonstrations.

4 dc.

Assignments (demos) for week 46 (8.-14.11)

You can choose two tracks from now on. The track A is for students who did not use databases before and will start from basic things. The other track is for students who already know how to use databases. This track B contains problems mostly with social media data. You can first try Stanford open data set. Later, we will move to real Twitter and perhaps Facebook datasets. You can change the track both ways, if you like.

Assignment 2A.

Assignment 2 is aimed at creation of COMPANY database (described on a lecture 3 (in Optima)), population of it with the information, connection to it and modifying and running the software which would make several queries. Personal access credentials for the database were sent to your e-mails at jyu.fi (if it didn't come please send message to Alexander.v.semenov@jyu.fi). Creation of the tables and population of the DB should be carried out using "psql" tool, queries should be made using software (described below). For completion of the demo you should perform query which would display employees working for Research department using one of the programs, described below, and present the source code of the program; database structure and data inside will also be considered.

3dc.

Tutorial for assignment 2:

Connection to UNIX

Connection to the DB can be made only from your users.jyu.fi UNIX account (https://www.jyu.fi/thk/ohjeet/how-to-nain-tehdaan/suorakaytto-unix).
For connection to UNIX machines you can use PuTTY software (http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html). You should download PuTTY, run it, and connect to halava.cc.jyu.fi (or jalava.cc.jyu.fi), then you will be in your home directory, from where you can connect to database. For authentification at UNIX server you should use credentials which you use for accessing University services.

Connection to Database

Connection to database can be performed using tool "psql". Example: psql –h vdb1.it.jyu.fi –U alsemeno alsemeno_db, using your own credentials instead of alsemeno and alsemeno_db (credentials were sent by e-mail to every participant of the course). After connection your password will be asked.
http://www.postgresql.org/docs/8.1/static/app-psql.html
after successful authentification you will be able to run PostgreSQL commands on database server and see the output.

Entering the data

Database description: Lecture3, pages 18-19
Values: Lecture3, pages 22-23.
Create statements can be found in Lecture4, page 19-20

Running C, JAVA, PHP programs

Despite of "psql" tool, interaction with the database can be carried out using your own software, which should be run from your users.jyu.fi account (not from database server or local computer). Currently we have 3 examples of such software: written in Embedded SQL (demo2.pgc), PHP (Demo2.php), and Java (JDBCExample.java); files are located in demo2 folder in Optima. At first, you should put source code to your account folder (e.g. enter it using "nano" text editor (nano file.c), or download it there), modify the code: you should add access credentials there, then compile it, if it is necessary, and then run.

Embedded SQL

Preprocessing: "ecpg demo2.pgc"

Compilation: "gcc demo2.c –lecpg"

Run : ./a.out

Java

Compilation: javac JDBCExample.java

Run: java -cp ~:/usr/share/java/postgresql-jdbc.jar JDBCExample

(be careful with names of the files)

PHP

PHP program can be run in two ways, compilation is not needed:

Command line: php file.php

Web-access (example is for web): you should put file to your web-folder (in users.jyu.fi) and run from the browser. Example: http://schoolsecurity.it.jyu.fi/db.php

Assignment 2B (Project).

This part of the task aims at analysis of social network data sets using PostgreSQL and other software

Tasks:

1) Download Small Facebook edges dataset from http://snap.stanford.edu/data/facebook_combined.txt.gz to your UNIX account folder using wget tool ("wget http://snap.stanford.edu/data/facebook_combined.txt.gz")

2) Extract the file (gzip -d facebook_combined.txt.gz)

3) Open the file (one possibility is 'nano' text editor in UNIX) and design the table which would fit for this dataset

4) Create the table and import the dataset into the table using COPY command/ Example: '\copy tst from '/autohome/nashome1/USERNAME/facebook_combined.txt' with delimiter ' ''

5) Design the following SQL queries:

a. Show total number of nodes

b. Show count of edges for each node

c. Show the id of the node with minimal degree (edges count), and with maximal

d. Extract the nodes, which are on 2nd level neighborhood from the node ('friends of the friends');

4dc

Assignments (demos) for week 47 (15.-21.11)

Assignment 3A.

Design the following queries:

1. extract average salary from EMPLOYEE table (from demo 2)
2. extract employees whose salary is below average
3. extract second maximum salary from the EMPLOYEE table

2 dc

Assignment 4A. Assignment 4 is aimed at creation of the table with 10000 instances and comparison of performance of the queries, with and without index. You would need to complete the following steps:

1. Connect to your database using psql tool, and create the table *demo3_ass1(id int, val varchar(16));*
2. Upload to your directory at users.jyu.fi file sql.sql from demo3 folder from optima

3. Load contents of this file to the database, using "-f filename" option, example: psql -h vdb1.it.jyu.fi -U username -f sql.sql username_db (using your name)
4. Connect to database and turn the timing on with command \timing (you will see "Timing is on.")
5. Run query which would select values from demo3_ass1 table 10 times and write down (or save somewhere) times of execution of the queries:
    a. WHERE id = SOME_ID
    b. WHERE id < SOME_ID (where SOME_ID is some digital value)
6. Create index on *id* column (CREATE INDEX ind_id ON demo3_ass1(id);)
7. Repeat queries from p.5
8. Compare performance of the queries and explain the results

1 dc

Assignment 5A. Assignment 5 is aimed at development of the software which would insert random values to the COMPANY database from assignment 2 (Lecture 3). In demo3 folder you can find 3 files: ffnames.txt, mfnames.txt, lnames.txt – lists of female first names, list of male first names, list of the last names, respectively. You should develop the software using preferred language (Java, PHP, Python, …) that would load **2000** rows to EMPLOYEE table. Names should be generated based on the names from files. You should pay attention to foreign keys, while populating the database. Address field could be set with '---' value. Salaries have to be in range from 10000 to 100000, people have to belong to different departments.

Perform the following (using psql tool with timing switched on):

1. Select the people from research department with salary more than 24000
2. Create partial index which would allow fast selection of employees from dept. no 5, perform query from p1 and compare the performance
3. Perform the query which would select certain employee using his first, last and middle names
4. Create multicolumn index on fname, minit and lname (create index ind_name on employee (fname, minit, lname);) perform query from p3 and compare the performance

4dc

Assignment 3B

1. For the previous table with data from Facebook design the query, which would extract second level friend networks, where links between second level friends are preserved, 2 dc
2. Have a look at Twitter API, https://dev.twitter.com/docs 1 dc
3. Create Twitter application (https://dev.twitter.com/apps), create OAuth keys for it, using Twitter OAuth tool. That would require Twitter account; register it if you do not have one .1 dc
4. Try to access Sample stream API endpoint via curl, https://dev.twitter.com/docs/api/1.1/get/statuses/sample (go to this link, login, go to "generate OAuth signature", then "See OAuth signature for this request", there you will see curl command, example is: `curl --get 'https://stream.twitter.com/1.1/statuses/sample.json' --`

```
header 'Authorization: OAuth oauth_consumer_key="------", oauth_nonce="------
", oauth_signature="------", oauth_signature_method="HMAC-SHA1",
oauth_timestamp="1384510234", oauth_token="113411992-------",
oauth_version="1.0"' --verbose
```

This command should be copied to your halava/jalava command line, 3dc

5. Have a look at the returned by the request data, design DB scheme for such data (hint: returned data is in JSON format), 2 dc

Assignments (demos) for week 48 (22.11-28.11)

Assignment 7A. Modify the program from assignment 5A, so that it could add (in addition to existing employees) 10 departments, 100 projects, and allocate projects to each employee (inserting into table works_on). Names of the projects could be e.g. "Project4", "Project5", etc

1dc

Assignment 8A.
a) Form for the query 'SELECT * FROM EMPLOYEE, DEPARTMENT WHERE EMPLOYEE.SALARY > 10000' the corresponding canonical relational algebra expression,

b) construct a query tree for the above expression, 1 dc

c) perform for the canonical expression heuristic optimization where projections and selections are pushed down in the tree and explain for each step the used transformation rule.

1 dc

Assignment 9A. Write a query which would select all the employees, working for one of the projects (choose one of the existing), and have salary less than some given value, in relational algebra form, construct query tree for it, perform a heuristic optimization for it and describe for each step the applied transformation rule. Argue why obtained tree is optimal.

2 dc

Assignment 10A. Execute the queries from assignment 8 and 9 using psql tool, using 'EXPLAIN ANALYZE' command, cf example below.

1. Present output of the command for both queries: find out from the output where are selectivity factors.
2. Drop indexes from the table employee (added in 5.2 and 5.4), run explain analyze for both queries
3. Index columns employee.salary and project.name; run explain analyze queries
4. Compare results of 10.1, 10.2 and 10.3

Example:

```
alsemeno_db=> EXPLAIN ANALYZE SELECT ssn FROM EMPLOYEE, DEPARTMENT;
                              QUERY PLAN
```

```
------------------------------------------------------------------------------------------------------------
 Nested Loop  (cost=1.09..198.09 rows=8800 width=13) (actual time=8.910..9.011 rows=24 loops=1)
   ->  Seq Scan on department  (cost=0.00..21.00 rows=1100 width=0) (actual time=8.885..8.888 rows=3 loops=1)
   ->  Materialize  (cost=1.09..1.17 rows=8 width=13) (actual time=0.007..0.022 rows=8 loops=3)
         ->  Seq Scan on employee  (cost=0.00..1.08 rows=8 width=13) (actual time=0.006..0.018 rows=8 loops=1)
 Total runtime: 9.096 ms
```

(5 rows)

At first, sequential scan of employee table was carried out (*Seq Scan on employee*) which returned 8 rows, below you can see the detailed analysis of the output; then results of the query are materialized (*Materialize*) (copied to the separate table which is saved; loops = 3 means that there were 3 loops, 8 rows were returned for each), also sequential scan of department table (*Seq Scan on department*) was conducted, where 3 rows were returned. Later, 24 rows were returned in nested loop.

"Seq Scan on employee (cost=0.00..1.08 rows=8 width=13) (actual time=0.006..0.018 rows=8 loops=1)":

(cost=0.00..1.08 rows=8 width=13) – estimated parameter set

cost=0.00..1.08:

0.00: "starting up" cost: start up time before first row can be returned

1.08: cost for returning all rows (measured in units of disk page fetches)

rows=8: estimated number of returned rows (selectivity)

width=13: estimated width of the row

(actual time=0.006..0.018 rows=8 loops=1) – actual parameters, loops parameter means number of repetitions of the node (can appear e.g. in nested joins)

http://www.postgresql.org/docs/8.1/static/sql-explain.html

4 dc

Assignment 4B

1. Create a table, which would store Twitter message, and information about the place (https://dev.twitter.com/docs/platform-objects/tweets), 2 dc
2. Develop a software which would access Twitter similarly to curl (see assignment 3B), and put the message, information about message and place into database, 2 dc
3. Design queries, which would display the number of the messages, depending on geographical location of the messages, where location is described by text (e.g. city name), or by bounding box coordinates. 3 dc

Assignments (demos) for week 49 (29.11-05.12)

Assignment 11A.

During the lectures conflict serializability was defined and graph characterization was given: History *H* over (formal) transactions is serializable iff SG(C(H)) is acyclic.

Let us define prefix of *H*, *H'*: it is *H*, but operations in the tail cut from some point on and the operations in the beginning of *H* are left in *H'*.

Prove the following theorem: Conflict serializability is a commit-prefix closed property; that is, if *H'* is an arbitrary prefix of serializable *H*, *H'* is also conflict serializable.

Hint: show that *SG(C(H'))* is a subgraph of *SG(C(H))*

4dc

Assignment 12A.

During the lectures View serializability was defined. Show that view serializability is not a commit-prefix closed property.

Ponder then, what kind of anomalies could occur should a dynamic scheduler allow view serializable schedules as defined during the lectures.

Hint: construct a history *H* that is view serializable, but that has a prefix *H'* that is not.

4dc

Assignment 13A.

Develop the procedure that would take as input employee's ssn, two names of the projects, and amount of working hours. Procedure should transfer given amount of employee's working hours from one project to another. Make sure, that it is database will remain consistent if procedure would run in parallel or program would suddenly quit (e.g. because of power failure)

1 dc

Assignment 5B

1. Modify the software from 4B, so that it would save timestamp of the messages, 1 dc
2. Collect 5000 tweets (message + location + timestamp), using the software developed in assignment 4B., 1 dc
3. Develop the queries which would find from your dataset numbers of messages per each hour, and number of messages per each minute (hint: http://www.postgresql.org/docs/8.1/static/functions-formatting.html). Present it as plot (e.g. in MS Excel) 3 dc.
4. Develop the queries, which would find the keyword in the text of the collected messages (not case sensitive), 2 dc