Assignment 2

Query Tuning

Database Tuning

Presentation and Demo:	March 13, 2020
Report due:	March 27, 2020, 23:55
Grading:	5 points

Notes

- It is suggested that you also have a look at the report template before you start working on the assignment.

Access Parameters for PostgreSQL

- Host: `biber.cosy.sbg.ac.at`
- Port: `5432`
- Database: `dbtuning_ss2020`
- User/Password: you should have received them via email

The database server (`biber.cosy.sbg.ac.at`) is accessible only from within the university network. If you would like to work from home, please connect to `fanny.cosy.sbg.ac.at` via `ssh`. Java, the PostgreSQL client, and Python are installed on this machine.

Support

If there are any ambiguities or problems of understanding regarding the assignment, you have the following possibilities to clarify them:

- Upon request via email (martin.schaeler@kit.edu)

In this assignment you will gain hands-on experience in rewriting slow queries and in experimentally evaluating the rewritten queries.

Creating Tables and Indexes

Create a database with the following database schema once in the postgres server at university, and once inside you local DBMS from Assignement 1:

- `Employee(ssnum,name,manager,dept,salary,numfriends)`
  - unique index on `ssnum`
  - unique index on `name`
  - index on `dept`

- `Student(ssnum,name,course,grade)`
  - unique index on `ssnum`
  - unique index on `name`
- `Techdept(dept,manager,location)`
  - unique index on `dept`
  - a manager may manage multiple departments
  - a location may contain multiple departments

Populating the Tables

Fill the database with $100k$ employees, $100k$ students, and 10 technical departments. Only about 10% of the employees are in a technical department. The types of the attributes should make sense (e.g., `ssnum` should be an integer) but the values need not be meaningful (e.g., names can be random strings). U

Queries

Original Queries Choose two types of queries that might be hard for your database to optimize and tet them in both systems. Taking queries from the lecture notes is OK. However, *do not* use a query where the original query would use a temporary table.

Note: For at least one of your queries rewriting should make a difference.

Rewritten Queries and Execution Plans Rewrite the queries and consult the execution plans of the original and the rewritten query. The rewritten query must lead to the same output across all possible instances.

Experiments Run the original and the rewritten query and measure the runtimes.

Report

1. Describe your database instance (data types, how did you fill the tables?).
2. Give the original and the rewritten queries.
3. Show and explain the execution plans, i.e., the output of `EXPLAIN ANALYZE`. Graphical versions (e.g., `pgAdmin3` output) are *not enough*.
4. Report and briefly discuss the runtime results from your experiments.
5. What differences did you observe between the postgres dbms and you local dbms?

Please indicate the average time per group member that was spent solving this assignment. The time that you indicate will have *no* impact on your grade.

Grading scheme:

Category	Max. Points
Description of your setup	0.5
Original query 1: description of query plan	0.5
Rewritten query 1: description of query plan	0.5
Difference between original and rewritten query 1	0.5
Original query 2: description of query plan	0.5
Rewritten query 2: description of query plan	0.5
Difference between original and rewritten query 2	0.5
Interpretation of the runtime results	1
Explanation and interpreation of differences	0.5

Reminder: Additional questions about the involved topics/techniques will be asked during the meeting.

Important: If the grading scheme is unclear, ask the instructor!