Lab session DB(i), 26 April 2023

[Lab environment]

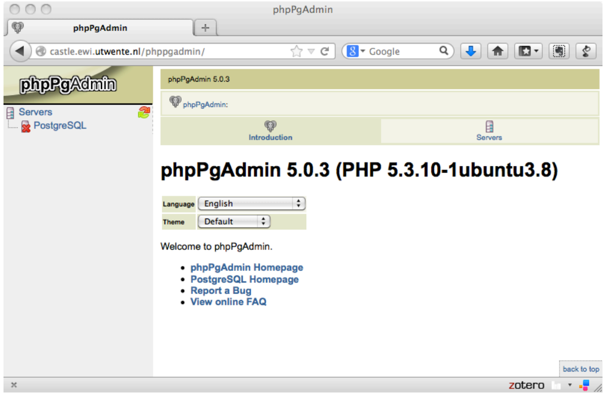
You will use a DBMS (database management system) for this and several other lab sessions as well as your project: PostgreSQL 10.12 running on server bronto.ewi.utwente.nl. Each group has their own database on this server. You can obtain a ***database for your group*** as follows (only one person of the group needs to do this to obtain a database for the whole group).

- Go to DAB: http://bronto.ewi.utwente.nl/dab/
- If you do not have an account in DAB, create one with "Register here". You need to use your student email address.
  NB: there is a known bug with the system that sometimes produces an error if you click on "Register here". The problem is that there is a "dab" missing in the URL: please change it from http://bronto.ewi.utwente.nl/register to http://bronto.ewi.utwente.nl/dab/register
  NB: some times you get site not found error on Chrome or Safari. The error is gone by clearing cache.
- Sign in and choose the course "di22232b" which stands for "Data and Information, year 2022/2023, quartile 2B,".
- Fill in your group number and click on "Get credentials".
- A database will be created for you and the system provides you with the credentials: a username (which is the same as the database name) and a password.
- For your convenience, your database already has two *schemas*: movies for the assignments, and one with the same name as your username for any experimenting of your own. In this way, you can keep the tables for these apart.
- You can always return to DAB to look at your credentials again and to reset your database.
  Warning: you loose everything in the database when you reset it, i.e., for all schemas in the database.

The same server also runs a web-based database administration tool, called PhpPgAdmin. PhpPgAdmin makes it easier to execute queries and various other tasks on your database. This is all web-based, so no installation is required. You can access it with this link: http://bronto.ewi.utwente.nl/phppgadmin. If you login with the credentials you obtained from DAB, you will see your database and the three schemas. You can easily create, inspect and drop the tables in your database with this tool.[1]

Below you see a series of PhpPgAdmin screens. PhpPgAdmin contains a lot more features than you'll need. Below are the instructions how to get to the two screens you need in order to do the exercises.

[1] You can also use any other PostgreSQL client such as PgAdmin (host = bronto.ewi.utwente.nl; port = 5432)



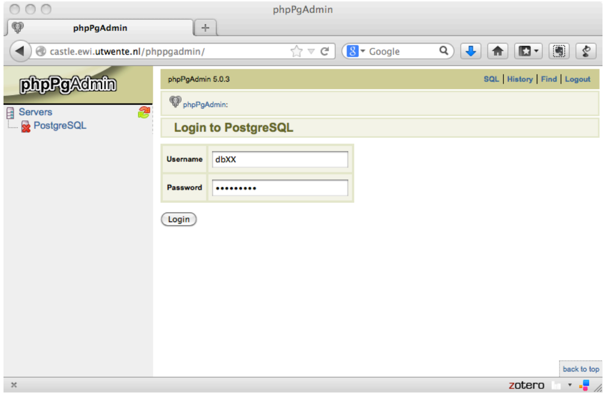
phpPgAdmin 5.0.3 (PHP 5.3.10-1ubuntu3.8)

Language: English

Theme: Default

Welcome to phpPgAdmin.

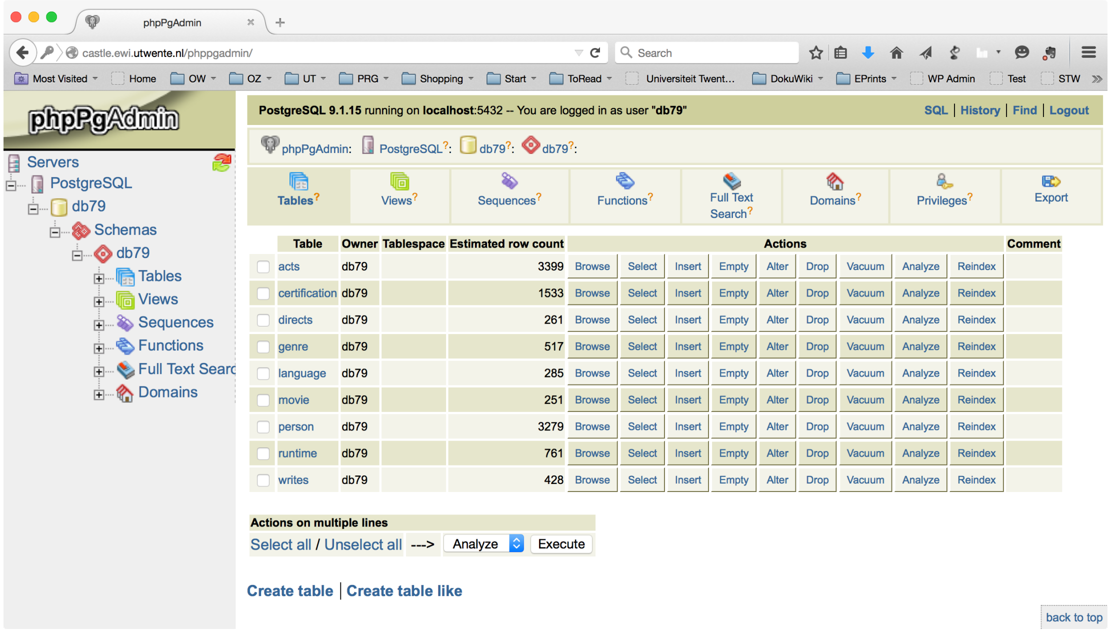
- phpPgAdmin Homepage
- PostgreSQL Homepage
- Report a Bug
- View online FAQ



Login to PostgreSQL

Username: dbXX

Password: •••••••••



PostgreSQL 9.1.15 running on localhost:5432 -- You are logged in as user "db79"

Table	Owner	Tablespace	Estimated row count	Actions									Comment
acts	db79		3399	Browse	Select	Insert	Empty	Alter	Drop	Vacuum	Analyze	Reindex	
certification	db79		1533	Browse	Select	Insert	Empty	Alter	Drop	Vacuum	Analyze	Reindex	
directs	db79		261	Browse	Select	Insert	Empty	Alter	Drop	Vacuum	Analyze	Reindex	
genre	db79		517	Browse	Select	Insert	Empty	Alter	Drop	Vacuum	Analyze	Reindex	
language	db79		285	Browse	Select	Insert	Empty	Alter	Drop	Vacuum	Analyze	Reindex	
movie	db79		251	Browse	Select	Insert	Empty	Alter	Drop	Vacuum	Analyze	Reindex	
person	db79		3279	Browse	Select	Insert	Empty	Alter	Drop	Vacuum	Analyze	Reindex	
runtime	db79		761	Browse	Select	Insert	Empty	Alter	Drop	Vacuum	Analyze	Reindex	
writes	db79		428	Browse	Select	Insert	Empty	Alter	Drop	Vacuum	Analyze	Reindex	

Actions on multiple lines

Select all / Unselect all ---> Analyze Execute

Create table | Create table like

On the start screen, click “PostgreSQL” in the left menu; this will allow you to log in. After logging in, you see all databases you have access to: one of these is “dab_ di22232b_NNN” (with NNN a number). For this database, you have all privileges required to perform queries and create and update tables, data, stored procedures, etc. Everyone has their own database with a copy of all data, so you cannot inadvertently mess up someone else’s database.

When you click on the database, on the tab “SQL” you get a screen in which you can enter any SQL query. You can cut and paste the query from a text editor or execute a script.
Another useful screen appears when you click on the “+” left of the database to unfold it and then “Tables”. It allows you to inspect the contents of your tables (Browse).

A convenient way of working is to write all SQL queries and scripts in a text editor and store them in one or more files.

[Database Schema]

The exercise below all use database with data about movies, persons, and various associations between them. These tables are contained in your database, i.e., everyone has their own copy.

The database has the following tables:

Movie	
mid	Integer KEY
name	text
year	numeric(4,0)
plot_outline	text
rating	numeric(2,1)

Person	
pid	integer KEY
name	text

Acts	
mid	integer
pid	integer
role	text

Directs	
mid	integer
pid	integer

Writes	
mid	integer
pid	integer

Genre	
mid	integer
genre	text

Language	
mid	integer
language	text

Certification	
mid	integer
country	text
certificate	text

Runtime	
mid	integer
country	text
runtime	numeric(3,0)

Actors, directors, and authors (script writers) of movies are indicated by means of tables which contain keys from the tables `Movie` and `Person`.
The tables in the right column contain further information about movies: genre (a movie can be associated with different genres); language(s) in which the movie is released; certification (age restrictions; there are different schemas for different countries); runtime (can vary per country).

Why the database has been designed this way is subject of a later lecture. Today we use the database only to practice writing SQL queries and stored procedures.

[Simple queries]

Exercise 1) Give name and year of recording for every movie with a rating between 8.7 and 9.0 [19 rows]

Exercise 2) Give the names of all movies which have a role name starting with "Dr." [39 rows]

[Hint: If you formulate the query like "`SELECT m.name FROM ...`", it could happen that the same movie title appears multiple times in the answer table (*why?*).
SQL offers the following construct to eliminate duplicate rows in a table:

```
    SELECT DISTINCT ... FROM ...
```

]

Exercise 3) Give an alphabetically ordered list of all actors who acted in movies with a rating of 8.9 or higher. [70 rows]

Exercise 4) Give the names of all authors of movies in which Harrison Ford acts. [11 rows]

[The basis of JDBC and Stored Procedures in PL/pgSQL]

Driver: PostgreSQL JDBC driver, version 4.2 (see https://jdbc.postgresql.org/)
Database host: bronto.ewi.utwente.nl; port 5432 (default PostgreSQL port)
See slides of lecture on DB-connectivity of *Wednesday 26 April* above for details and code examples for how to access the database from Java.

Importing jar file to InteliJ

1. Download the jar file provided on Canvas.
2. In InteliJ project go to File->Project Structure (or Ctrl+Alt+Shift+S)
3. Go to Modules
4. Select the + icon
5. Select JARs or Directories
6. Select the jar file from your file system
7. Click OK.
8. Make sure that the jar file is selected as option Compile on the right
9. Click Apply and OK

Importing jar file clipse

To import jar file in your Eclipse IDE, follow the steps given below.

10. Right click on your project
11. Select Build Path
12. Click on Configure Build Path
13. Click on Libraries and select Add External JARs
14. Select the jar file from the required folder
15. Click and Apply and Ok

Exercise 5) Take the solution of the previous exercise (4) and write a Java program which executes the query by means of JDBC and displays the result of the query on the screen (console).
[Shows 11 authors of movies in which Harrison Ford acts]

Exercise 6) Modify you program, making use of “PreparedStatement”, rather than “Statement”. Make “Harrison Ford” a parameter of the Java program, allowing you to execute the query for an arbitrary actor. The ResultSet should make use of a cursor, fetching 5 results in at a time.[2] [For “Bruce Willis” you get a result of 7 rows]

Exercise 7) Create a stored procedure which executes the query with the actor as parameter, like in Exercise 6. See “Chapter 39. PL/pgSQL - SQL Procedural Language[3]” of the PostgreSQL-documentation for information about the PL/pgSQL language with which one can write stored procedures. Execute the stored procedure with PhpPgAdmin in order to verify that it works correctly.

Exercise 8) Make a new version of your Java program which does the same as in Exercise 6, but now using the stored procedure.

[2] Zie https://jdbc.postgresql.org/documentation/head/query.html
[3] http://www.postgresql.org/docs/9.5/static/plpgsql.html

[Queries with subqueries]

Exercise 9) Give the names of all authors who wrote a movie that has no director.
[answer: Mogens Rukov, Thomas Vinterberg]

Exercise 10) Give the names of all authors who *only* wrote movies that have no director.
[Answer: Thomas Vinterberg]

[Determine where queries execute: in the database or application]

The exercises below ask you to measure executions time. In Java this can be done as follows:

```
int iters=100;
long startTime = System.currentTimeMillis();
      // preparatory statements outside the loop,
      // e.g., Statement st = conn.createStatement();
for (int i=0; i<iters; i++) {
      // statements that you want to do 100x
}
      // rounding up statements outside the loop.
      // e.g., st.close();
long stopTime = System.currentTimeMillis();
double elapsedTime = (stopTime - startTime) / (1.0*iters);
System.out.println("Measured time: "+elapsedTime+" ms");
```

As many students are doing the same, it can happen that the timings vary with the server load. Execute your program at least three times, with at least 10 seconds in between, so as to increase the chance of catching the server when it's not too busy.

Exercise 11) Take the query from the previous exercise (10). Write a Java method that executes the query and gets the results by means of a for-loop that executes the query 100x. Do this for the following three variants: using Statement (as in Exercise 5), using PreparedStatement (as in Exercise 6), using a store procedure (as in Exercise 8). In each case measure how long it takes per query and print the timings on the console, so that you can see what is fastest and what the differences are.
[On campus you should see times of several ms; off campus several tens of ms]

Exercise 12) (how not to do it ☺) Novice programmers often tend to avoid complex queries or stored procedures, but to write Java methods with simpler queries and combine these to more complex result in a Java program. Exercise 9, “Give the names of all authors who wrote a movie that has no director“, could be realized with three Java methods:

```
List<Integer> allMovies()          // mids of all movies
List<String> authorOfMovie(int mid)// authors of movie ‘mid’
boolean hasDirector(int mid)      // movie ‘mid’ has a director
```

The Java program contains a loop iterating over all results of allMovies(), in the body of the loop calling both other methods:

```
for (Integer mid : allMovies()) {
   if !(hasDirector(mid)) {
      List<String> aut=authorOfMovie(mid);
   }
}
```

Write a Java method/program that executes the query this way and measures the execution time as in the previous exercise. (Please use “iters=5” or less, rather than 100; with 5x the results will be accurate enough).

[Queries with GROUP BY]

Exercise 13) For every director who directed two or more movies in the genre `Action’, give the year in which they directed their first movie in this genre.
[Akira Kurosawa 1954, Steven Spielberg 1981, Ridley Scott 1982, James Cameron 1984]

Exercise 14) Give a list of directors who also acted, with the number of movies in which they acted. Order the list descending according to the number of movies. [23 rows]

[Control flow statements in Stored Procedures]

Exercise 15) The language PL/pgSQL offers a complete programming language, including loops, ifs, exceptions, etc. See “Chapter 39. PL/pgSQL - SQL Procedural Language”[4]. The program of exercise 12 could also be realized this way. (Still an example of “how not to do it”, but it can be done.) Define 4 stored procedures (allMovies, authorOfMovie, hasDirector, exercise15), which implement the Java program of Exercise 12 in PL/pgSQL. Apply the Java method to measure the performance of the execution of stored procedure “exercise15”.

Exercise Set 2

We would like to get some insights from the data we gathered during our online TScIT 33 conference. Consider the database schema below, which we derived from the full class diagram developed for the TScIT system.
TEACHER (ID PK, Name, Faculty, Research_Group);
STUDENT (ID PK, Name, Program);

[4] http://www.postgresql.org/docs/9.5/static/plpgsql.html

CONFERENCE_PAPER (ID PK, Supervisor, Supervisee, Topic, Title, FK Supervisee REF STUDENT(ID), FK Supervisor REF TEACHER (ID));
CONFERECE_SESSION (ID PK, Supervisee, Title, Duration, Nb_Attendees, FK Supervisee REF STUDENT(ID));

Exercise 1) Download the file dbdefinition.txt and use this file to create your own database schema on the database server.

Exercise 2) Write a SQL query to return the name of the student (supervisee) whose conference session (online thesis defense) had the highest number of attendees.

Exercise 3) Write a SQL query to show the names and programs of the students who were supervised by "M. van Keulen (Maurice)".

Exercise 4) Write a SQL query to show the names and programs of the students who were supervised by teachers from the same research group as "Faizan Ahmed".

Exercise 5) Write a SQL query to show the name of the research group(s)[there can be more than one] whose members supervised more students than the rest of the (research) groups.

Exercise 6) Write a SQL query to show the names of the teachers who did not supervise any students.

Exercise 7) Write a SQL query to show the research group(s) [there can be more than one] whose teacher supervised the maximum theses/research papers in the topic of "Artificial Intelligence".

Submission

You can submit your solution to Canvas. Combine everything into a single PDF file and submit it on Canvas. This is complete/incomplete assignment. We are looking for a serious attempt. We will publish solution on Wednesday April 26 at 1800. Therefore, there is no re-sit for this assignment.