4. Experiment: Optimization, date values, recursion, access control

Exercise 4.1 (Optimierung Korrelation; 15 P.)
In exercise 2.5 you computed a correlation coefficient. The following shows the SQL query for the formula given in exercise 2.5, for convenience computing the correlation between population and area of a country.

```
WITH Average AS
(
  SELECT AVG(population) population, AVG(area) area
  FROM Country
  WHERE population IS NOT NULL AND area IS NOT NULL
)
SELECT
  SUM((Country.population - Average.population)*(Country.area - Average.area)) /
  (
    COUNT(*) *
    SQRT(SUM(POWER(Country.population - Average.population,2))/COUNT(*)) *
    SQRT(SUM(POWER(Country.area       - Average.area,      2))/COUNT(*))
  ) r
FROM Country CROSS JOIN Average
WHERE Country.population IS NOT NULL AND Country.area IS NOT NULL;
```

Examining the execution plan of this query you will see that the table *County* has to read twice. Reformulate the query (formula) such that a single pass over *Country* is enough. Indicate your conversion steps briefly.

Hint: So you have to remove the nested aggregations. The function `CORR` comoputes the same correlation coefficient. If you look at the execution plan of this function you can see which aggegations are sufficient. You can also reason how the used NVL2 expressions can substitute the not-null-conditions.

Exercise 4.2 (Korrelation Organisationen; 15 P.)
Compute the correlation between the per capita income and the membership in the European Union of a country. Consider only European countries. The formula is (point-biserial correlation):

$$r_{pb} = \frac{\overline{y}_0 - \overline{y}_1}{\sigma_y} \cdot \sqrt{\frac{n_0 n_1}{n^2}}$$

$\overline{y}_0$ and $\overline{y}_1$ are the arithmetic means of the variable (here: per capita income) of both groups, σ_y is the standard deviation of the variable (without distinction by group), n_0 is the number of members, n_1 the number of non-members and $n = n_0 + n_1$. There is no distinction between positive and negative correlations.

a) Write an SQL query that computes the correlation coefficient.

b) Now write the query in such a way that each required table is only read once. If you did this already in part a) you have nothing to do here. **Hint:** An outer join and `NVL2`, `DECODE` or `CASE` are useful.

c) Write a query that computes for each organization the correlation between the membership of a country in that organization and the per capita income of a country. Sort the result in ascending order by r_{pb}.

Exercise 4.3 (Direkter Mengenvergleich; 5 P.)
Create a view that contains all rivers as in *River*. Additionally, the view should have a set-valued column *countries* containing all countries (*code*) where the river is flowing through. Write a query over this view returning all (unordered) pairs of rivers flowing through the same set of countries (only *code*). **Hint:** Set comparisions on set-valued attributes over a simple datatype can be performed directly.

Exercise 4.4 (Datumsangaben; 20 P.)
The table *politics* contains a column with the independence date.

1. Select all countries with an independence date between 1300 and 1600.
2. Compute the arithmetic mean and the standard deviation of the independence date of the European countries.
3. Specify the independence date for all countries in a table with two columns. The first column should contain the country code, the second the date in the form of the following example (without quotes): `"18. Jan"`

4. Group all countries by the weekday of their independence date and compute the mode thereof (the weekday with the most independencies).

Exercise 4.5 (Fibonacci-Zahlen; 15 P.)
Compute the Fibonacci number f_n using a recursive WITH. An optimal solution would enable the specification of n in the WHERE-clause of the main query and return exactly n and f_n. **Hint:** Think about how to stop the computation at the given n. One possibility contains the usage of *rownum*.

Exercise 4.6 (Sicherheit; 15 P.)
The scripts `tax_employee.sql` and `tax_inspector.sql` provide a small tax aduit setup. There are two users contibuting tables to a global model: An employee manages the taxpayers in table *taxpyer* (being responsible for inserts, deletes, and updates). The tax inspector needs only read access on this table. He manages the tables *employment* with information about employers and *revenue* with the taxpayers' annual income. The employee does not have access here but only on a view with aggregated data for statistical reasons.

Install this scenario using two accounts of your group (replace "inspector" and "employee" with your user names) and get acquainted with it. You might notice a security flaw quite soon. Write a script showing how the employee can obtain undected access to confidential information.

Exercise 4.7 (Erdoberfläche; 5 P.)
How big is the amount of the earth's surface being covered by seas?

Deadline: 23.6.2010, 11h