Advanced SQL
Winter Term 2025/2026
Prof. Torsten Grust, Tim Fischer, Björn Bamberg
WSI — Database Systems Research Group

Assignment 7

Hand in this assignment until **Friday, December 12th 2025, 12:00 pm** at the latest.

🤔 Running out of ideas?
Are you hitting a roadblock? Are some of the exercises unclear? Do you just need that one hint to get the ball rolling? Refer to the `#forum` 🌐 channel on our Discord server—maybe you'll find just the help you need.

⚠️ The lecture evaluation is coming up!
We rely on your feedback to steer Advanced SQL in the right direction. We look forward to both your criticism and your praise! So please take part in the lecture evaluation by December 5th 2025. Thank you very much!

Ⓔxam-style Exercises
Exercises marked with Ⓔ are similar in style to those you will find in the exam. You can use these to hone your expectations and gauge your skills.

Task 1: Sorted Lists

Lists | Table functions

The file `sorted.sql` contains a definition for a table called `lists` that holds multiple lists of integers (type `int[]`). The table is set up such that each list is automatically assigned a unique ID through the sequence `serial`.

```sql
CREATE SEQUENCE serial START 1;

CREATE TABLE lists (
  id  int   DEFAULT nextval('serial') PRIMARY KEY,
  lst int[] CHECK (len(lst) > 0)
);
```

lists

id	lst
1	[1,2,3,5,6]
2	[4,3]
3	[7,6,9]
4	[2,2]

A **Write a SQL query** that scans table `lists` for lists **sorted in ascending order**.

Task 2: Lights

Grouping | Table functions

You are in a hallway with $N \geq 1$ other people. By pure coincidence, this hallway is lit by exactly N light bulbs dangling from the ceiling. Each bulb is connected to a pull cord which turns it on and off. Initially, it's pretty dark since all lights are off. Now, all N people go down the hallway, one by one, each pulling on a different set of cords.

- The first person pulls all cords $1, 2, 3, \ldots$
- The second person pulls every other cord: $2, 4, 6, \ldots$
- The third person pulls every third cord: $3, 6, 9, \ldots$
- This continues until, finally, the N-th person pulls cord N only.

A **Write a SQL query** that computes a single-column table with the positions of all lit bulbs after all N people have passed through the hallway.

☝️ Be honest with yourself!
Your query needs to work for all $N \geq 1$, so please make the N in your solution configurable using a macro 🌐. Something along the lines of the following should do the trick. 😉

```sql
CREATE OR REPLACE MACRO N() AS ...;
```

Task 3: Data Analysis

CSV | Grouping

The CSV file `analysis-data.csv` contains five data sets, identified by the first column. Each data set comprises a set of points (x, y) in the 2D plane.

A **Write a SQL query** which calculates the mean of `x` and `y` (in columns `x_mean` and `y_mean`), their respective standard deviations (in columns `x_stddev` and `y_stddev`) and the correlation coefficient (in column `correlation`). Please ensure that values in each result column have type `numeric(3,1)`.

> Hit the books to find what you need!
> DuckDB supports a wide range of statistical aggregate functions, so peruse its documentation to pick out the ones you need!

B Statistical measures are cool and all, but they do not tell the whole story contained within our data. To bring the unseen to light, **generate a 2D plot of the points in each data set**, `'a'` through `'e'`, and **give each data set a descriptive name** based on what you find.

> Quick and dirty plotting.
> There are many ways you can generate the plots of the data sets. If you are look for a method that doesn't require much setup, there are online tools that can the job.

> Show use what you see!
> We want to see the hidden pictures as well, so please hand in `.pdf` or `.svg` files of your five plots!

Task 4: `SEND + MORE = MONEY`

Lists Macros

In this task wou will solve the *digit assignment puzzle* to the right. A legal solution to the puzzle consists of a mapping between each letter in the puzzle to a unique digit between 0 and 9 such that each of `SEND`, `MORE`, and `MONEY` are actual numbers, *i.e.*, don't have any leading zeros, and the arithmetic actually works out.

```
   SEND
+  MORE
-------
= MONEY
```

A First, **write a macro** `val(digits)` which converts the list of integers `digits` into its actual integer representation, *i.e.*, calling `val([1,2,3,4])` should produce the integer 1234. If list `digits` is empty, the macro should yield `NULL`.

> Need help getting the ball rolling?
> You can implement this macro completly using the built-in function `list_reduce`.

B **Write a SQL query** that solves puzzle using the `val` macro you wrote in Subtask A. The query's results should comprise eight columns `S`, `E`, `N`, `D`, `M`, `O`, `R`, `Y`, each of type `int`, such that each column holds the respective letter's assigned digit. If the puzzle has multiple distinct solutions, your query should produce one row for each.

> Puzzled yet?
> You can start by trying each possible solution in a *brute force* manner, which is an entirely legitimate approach! It may simply take a few minutes to finish ($\approx 4\,\mathrm{mins}$). But you can drastically speed this search up (to $\approx 4\,\mathrm{secs}$) if you restrict the solution space you are iterating through.
>
> *As a freebie:* If you think about it, `M` must be `1`, which also strongly restricts the possible values for `S`.

Task 5: Mountain Altitudes Ⓔ

Lists Table functions

Write a SQL query to compute the height of each "vertical segment" of the mountain range in table `mountains`. The query should produce a table with the `(x, alt)` schema you know from the lecture, see Chapter 5, Slide 14.

mountains

y	mnt
1	`' #  '`
2	`'### '`
3	`'####'`

One # measures exactly 100 meters in height and you can assume that each vertical mountain segment is one continuous pile of # starting at the very bottom, **i.e.**, the minimum `alt` value is 100. The value of `x` ascends from left to right, starting at 1.

> Think back…
> Consider reusing the `list_positions` macro from Task 1 of Assignment 6.

> Unsure if you understood the task?
> If not sure that you understood the task, take a look at the result to the right, which your query should produce for the example instance of `mountains` above it.

x	alt
1	200
2	300
3	200
4	100