Database Practicum

Course info

- Michal Rjaško – rjasko@dcs.fmph.uniba.sk
- Webpage: http://www.dcs.fmph.uniba.sk/~rjasko/dbpraktika2026.html
- **Evaluation:**
  - **2–3 homework assignments worth 45/30 points each, totaling 90 points (assigned roughly at one-third intervals throughout the semester)**
  - **12 lessons of 1 point each, at least 9 must be obtained**
    - within 6 days after the lesson, the solved tasks (at least half) must be se over by e-mail
  - A: 92 and more
  - B: 84 - 91 points
  - C: 76 - 83 points
  - D: 68 - 75 points
  - E: 60 - 67 points

Course plan

- SQL, DDL, DML
  - (datalog)
- working with database in Java
- Explain - query analysis / optimization
- SQLite, PostgreSQL
- NoSQL (MongoDB a pod)
- (Vector databases, AI / RAG)

Core motivation

- Most software needs to store data persistently.
- In doing so, it is necessary to address rather non-trivial problems:
  - How to store structured data in binary form
    - "Tabular" data - relational databases
    - NoSQL databases
    - Objects, serialization
  - Integrity and consistency
    - Resilience to failures
    - Prevent invalid, inconsistent data
  - Concurrency, Transactions (ACID)
  - Security
  - How to query the data, find the data we need
  - How to do it efficiently

Core motivation

- Database systems (DBMS) solve most of the problems mentioned above
- Course „Introduction to databases“ learns you how to solve many of these problems
- We are focused more practically, i.e. how to use the DBMS
  - How to query the data (SQL)
  - How to store the data (DDL, DML)
  - How to communicate with the DBMS

Relational databases

- Most common type of DBMS
- Store structured data in „connected“ tables
- Tables
  - **Rows** → individual records / entities
  - **Columns** → attributes
- Tables represent math relations
  - Relation <=> Table
  - Row in a table

    -> tuple in a relation

STUDENT

id	first_name	last_name
1	John	Smith
2	Anna	Brown
3	Peter	Novak

(1, "John", "Smith"),
(2, "Anna", "Brown"),
(3, "Peter", "Novak")

COURSE

id	name
10	Databases
11	Java

ENROLLMENT

student_id	course_id
1	10
1	11
2	10

Querying the data

- Declarative programming
  - The program defines **what** needs to be calculated, but does not describe **how**
  - All common database languages (e.g. SQL) do not even contain the means to describe the calculation
  - The database itself chooses the calculation procedure according to the query and existing data
    (non-trivial optimization)
- SQL is the most widely used query language for relational databases

SQL:

```
SELECT name FROM students
WHERE grade > 3;
```

Procedural programming:

```
for every student:
    open student record
    read grade
    if grade > 3:
        add student to res
```

SQL - Structured Query Language

SQL is the most used database query language today.

- Standard since 80ties (1986); no changes last 10-20 years
- If there is the standard, „SQL“ is varies among different database systems
- Queries written for one database system (PosgreSQL) need not to work in another (MySQL), mostly because of usefull extensions of the SQL standard the database systems implement
- SQL contains also standards Data Manipulation Language (DML) and Data Definition Language (DDL)

Basic syntax:

- **SELECT** <attributes>
  FROM <relations>
  WHERE <condition>
  [ORDER BY attribute1, attribute2...]
  [LIMIT 100] [OFFSET 0]

Example of a SQL query

```
SELECT
     concat(e.firstname,’ ‘,e.lastname) AS ename,
     (CASE
          WHEN e.comm IS NULL THEN e.sal
          ELSE e.comm + e.sal
     ) AS ‘total_salary’
FROM emp
WHERE deptno>=20 AND lower(e.firstname)=‘john’
```

Operators, expressions, functions

- You can use many operators in WHERE clause:
  - =, <>, >, <, >=, <=, BETWEEN, LIKE, IN, IS NULL, IS NOT NULL
  - AND, OR, ! (NOT)
- Also it is possible to use arithmetic expressions and many more functions
  - E.g. *concat(e.firstname,' ', e.lastname)*
  - Functions to work with date and time
  - List of supported functions depends on a database system
    - https://www.postgresql.org/docs/current/static/functions.html

Multisets

- SQL threats relations as multisets, i.e. multisets can contain duplicate rows (opposite to relations, which are sets).
- If you want to strip duplicates you must enforce it using some commands (**UNIQUE** constraint when creating table – more in DDL – or using **DISTINCT** keyword in queries).
  - SELECT DISTINCT job FROM emp;

PostgreSQL / SQLite

- For simplicity, we start with SQLite database system
- SQLite:
  - “Embedded” SQL database system
  - “Serverless” – does not need to run as a standalone server
  - “Single-file” – the database is contained in a single file
  - “Cross-platform” – runs everywhere
  - “Small footprint” – the SQLite library is under 500 KB
  - Quick setup, but not suitable for „large scale“ data / many concurrent users
- We will also work with the PostgreSQL database system
  - „full“ client/server DBMS

PostgreSQL

- Most database systems have a client-server architecture
  - Server
    - contains data
    - understands SQL queries
    - clients connect to it mostly via socket (TCP/IP), or named pipes or other channels supported by the OS
  - client
    - An application that needs to work with data
    - sends queries to server in SQL language
    - displays/processes response from the server
- Some features can be implemented on any side (e.g., pagination).

SQLite

- To work with SQLite you can use:
  - Download and use command line tools
    - https://www.sqlite.org/download.html
    - https://www.sqlite.org/cli.html
  - Or use some UI
    - https://sqlitebrowser.org/