Exercise sessions are here to illustrate the material of the course with examples, special cases, etc.

- Homework every week: programming assignment and 2-3 problems
- Do 75% or better and get the bonus for the final grade
- Written exam at the end
- Slides on the website
- Email subject should start with [qo14]
Execution plan

- Students(Id, Name), Lecture(Id, Name), Attends (SId, LId)
- Find all students that attend lectures together with Schopenhauer, excluding Schopenhauer himself.
Execution plan

- Students(Id, Name), Lecture(Id, Name), Attends (SId, LId)
- Find all students that attend lectures together with Schopenhauer, excluding Schopenhauer himself.

```sql
select s2.name
from Students s1, Students s2,
     Attends a1, Attends a2
where s1.name='Schopenhauer'
and s1.id <> s2.id
and a1.sid=s1.id
and a2.sid=s2.id
and a1.lid=a2.lid
```
select distinct v.title
    from Lectures v, Professors p
    where v.prof_id = p.persnr
        and p.name = 'Kant'
        and v.sws = 2;

Reminder:

- cross product (from)
- add a selection (where)
- add a projection (select)
- the result is a tree
Logical optimization: preliminary

Cardinality and Selectivity
Logical optimization: preliminary

Cardinality and Selectivity
Selectivity of a predicate, selectivity of a join
Logical optimization: preliminary

Cardinality and Selectivity
Selectivity of a predicate, selectivity of a join

- example of a predicate with (very) high selectivity
Logical optimization: preliminary

Cardinality and Selectivity
Selectivity of a predicate, selectivity of a join

▶ example of a predicate with (very) high selectivity
▶ (now: with joins)
Logical optimization: preliminary

Cardinality and Selectivity
Selectivity of a predicate, selectivity of a join

- example of a predicate with (very) high selectivity
- (now: with joins)
- example of a predicate with (very) low selectivity
Logical optimization: preliminary

Cardinality and Selectivity
Selectivity of a predicate, selectivity of a join
  - example of a predicate with (very) high selectivity
  - (now: with joins)
  - example of a predicate with (very) low selectivity
  - (now: with joins)
Logical optimization: preliminary

Cardinality and Selectivity
Selectivity of a predicate, selectivity of a join

▶ example of a predicate with (very) high selectivity
▶ (now: with joins)
▶ example of a predicate with (very) low selectivity
▶ (now: with joins)
▶ independent and correlated conditions
Logical optimization

- $|\text{Students}| = 1000$
- $|\text{Lectures}| = 100$
- $|\text{Attends}| = 5000$
- $f_{s,l} = 0.001$
- $f_{a,l} = 0.01$

Find the students that attend the course 'Ethik'

- SQL query
- canonical transformation, compute cardinalities
- push down selections, compute cardinalities
Cost Estimation

The goal of optimization is to minimize the cost function

Reminder: $C_{out}$

$$C_{out}(T) = \begin{cases} 
0 & \text{if } T \text{ is a leaf } R_i \\
|T| + C_{out}(T_1) + C_{out}(T_2) & \text{if } T = T_1 \bowtie T_2
\end{cases}$$
Cost Estimation

The goal of optimization is to minimize the cost function

Reminder: $C_{out}$

$$C_{out}(T) = \begin{cases} 
0 & \text{if } T \text{ is a leaf } R_i \\
|T| + C_{out}(T_1) + C_{out}(T_2) & \text{if } T = T_1 \Join T_2 
\end{cases}$$

That's why we need join ordering!
Cost Estimation

The goal of optimization is to minimize the cost function

Reminder: $C_{\text{out}}$

$$C_{\text{out}}(T) = \begin{cases} 0 & \text{if } T \text{ is a leaf } R_i \\ |T| + C_{\text{out}}(T_1) + C_{\text{out}}(T_2) & \text{if } T = T_1 \bowtie T_2 \end{cases}$$

That's why we need join ordering!

$\begin{array}{c}
\triangledown \\
| R_1 | = 100 \\
| R_2 | = 200 \\
| R_3 | = 100 \\
f_{1,2} = 0.1 \\
f_{2,3} = 0.0001
\end{array}$
Cost Estimation

The goal of optimization is to minimize the cost function $C_{\text{out}}$

Reminder: $C_{\text{out}}$

$$C_{\text{out}}(T) = \begin{cases} 
0 & \text{if } T \text{ is a leaf } R_i \\
|T| + C_{\text{out}}(T_1) + C_{\text{out}}(T_2) & \text{if } T = T_1 \Join T_2
\end{cases}$$

That's why we need join ordering!
The goal of optimization is to minimize the cost function.

Reminder: $C_{\text{out}}$

$$C_{\text{out}}(T) = \begin{cases} 0 & \text{if } T \text{ is a leaf } R_i \\ |T| + C_{\text{out}}(T_1) + C_{\text{out}}(T_2) & \text{if } T = T_1 \Join T_2 \end{cases}$$

That’s why we need join ordering!
Info for Homework

- You can work in groups with up to two students
- Handwritten (and/or scanned) solutions will not be accepted. Use LaTeX (preferable) or Word.
- Programming assignment:
  - Implement your own query optimizer step by step
  - Initial code base (very simple database system) is available on the website
  - Language: C++11 (great opportunity to learn it btw)
  - Solutions should come with a Makefile and instructions on how to build/run it
  - Future assignments will build upon the current
Submit the whole project directory, not just separate source files (no binaries!)
You can work within the TinyDB directory, changing its structure if needed
(Briefly) comment the source code: every class, field, method, design choice
Give examples of the input queries for which you tested. How about unit tests?
Info

▶ Slides and exercises:
  http://www-db.in.tum.de/teaching/ss14/qo/
▶ Send any questions, comments, solutions to exercises etc. to
  andrey.gubichev@in.tum.de
▶ Exercises due: 9 AM, April 28