Project

- The goal of this lab is to develop and optimize certain features of a database engine.
- Depending on the first results and individual preferences, the advanced tasks can be chosen individually.

⚠️ Basic knowledge about databases is required!
Sometimes, it makes sense to open a book!
What is a database engine?

- Software component used by a database management system to create, read, update and delete data.

- What are important design parts of a database engine?
  - Data structures (files, indexes, trees, hash tables, ...)
  - Algorithms (scheduler, selection, projection, joins, ...)
  - Storage hierarchy (disc, main memory, ...)
  - ...
What you will do in this lab

- Build your own analytical database engine
  - Basic requirements for each team
  - Possible extensions depending on individual preferences

What you will not do in this lab!

This course is not about writing applications that use database engines.
This course is not about modeling relational databases.
What you will do in this lab

- Build your own analytical database engine
  - Basic requirements for each team
    - Logical data structures
    - Physical data/storage structures
    - Buffer management
  - Query processing and operators (projection, selection, join, sort, aggregation...)
  - Scheduler
  - Cost functions and query optimizer
  - Performance measurement and optimization
- Possible extensions depending on individual preferences
  - Data compression
  - Inter- and intra-query parallization
  - Distribution of the query processing and the data on different machines
  - Compilation of queries to native code/byte code at runtime
  - Index structures

We want to encourage you to implement optimizations inspired by publications, e.g. Just-in-time Compilation for SQL Query Processing...
What you will not do in this lab

- Dynamic insert operations
- Transaction and concurrency control
- User and access control
Programming language

- Whatever you like!

- E.g. C, C++, Java, Scala, Phyton, ...

This might be a opportunity to learn and practise a new language!
Organization

- Teams of 3-4 students

- Every team develops its own database engine
  - analyses, designs, implements, tests, **tweaks***, and presents it

- Project divided into 6 or 7 phases
  - Each phase is 2 weeks in duration

- A meeting after every project phase

* This should be done systematically ;-)
What we expect

- A working and convincing database engine
- Report, source documentation, presentations
  - eMail us report and slides the night before each meeting
- Participation in programming, documentation, and presentations
  - Everyone in the group!
- Approximately 100-200h of work (1CP = 30h, 6CPs = 180h)
- Ask for help if there are problems
Meetings

- You deliver the lab report sketch
  - Deadline: midnight before meeting
- Each group presents
  - Current state of implementation (live demo!)
  - Approach taken
  - Progress
  - Problems
  - 4-5 slides (as PDF/PPT, delivered midnight before the meeting date)
- We present
  - The outline of the next phase
  - Relevant background theory where necessary
- We answer your questions
Mark Breakdown

- **40% Implementation + Source Code Documentation**
  - Intermediate results count!
  - Coverage
  - Design
  - Correctness
  - Documentation
  - (Performance)

- **30% Meetings + Presentations**
  - Participation
  - Presentation (clarity and completeness)
  - Slides

- **30% Lab Report**
  - Overview of implementation and user manual
  - Documentation of development process
  - 10-15 pages total
Contact & Infrastructure

- Contact
  - froemmge@dvs.tu-darmstadt.de

- Course Homepage
  - http://www.dvs.tu-darmstadt.de/teaching/dbed/

- Our lab is available 24/7
  - Transponders are available from RBG service center C119
Literature

Database Management Systems
Ramakrishnan/Gehrke
3rd Edition

Database Systems II
http://www.dvs.tu-darmstadt.de/teaching/db2/

Business Intelligence and Data Warehousing
http://www.dvs.tu-darmstadt.de/teaching/bidw/

...
Blueprint of the Solution

Query language for data retrieval

Performance benchmark

Parse expression
Generate execution plans
Choose most suitable execution plan
Execute plan (projection, selection, join, aggregation…)

Please allow the configuration of the maximum RAM consumption!

RAM
CPU

Store data on disk / in RAM

Inter- and intra-query parallelization

Disk

CSV

Import data

21.04.2015 | Department of Computer Science | Databases and Distributed Systems
<table>
<thead>
<tr>
<th>Date</th>
<th>Week</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>21.04.</td>
<td>(0)</td>
<td>Intro, query language, performance-benchmark</td>
</tr>
<tr>
<td>28.04.</td>
<td>(1)</td>
<td>Store data on disc / in RAM, execute selection</td>
</tr>
<tr>
<td>12.05.</td>
<td>(2)</td>
<td>Execute projection, join, sort</td>
</tr>
<tr>
<td>26.05.</td>
<td>(3)</td>
<td>Performance benchmark</td>
</tr>
<tr>
<td>09.06.</td>
<td>(4)</td>
<td>Optimizer / choose most suitable execution plan</td>
</tr>
<tr>
<td>23.06.</td>
<td>(5)</td>
<td>Advanced stuff</td>
</tr>
<tr>
<td>14.07.</td>
<td>(6)</td>
<td>Advanced stuff</td>
</tr>
<tr>
<td>04.08.</td>
<td>(7)</td>
<td>Final demonstration</td>
</tr>
</tbody>
</table>
Phase 0

ANALYSIS AND DESIGN
Phase 1: Analysis & Design

- System architecture?
- Programming language?
- Design query language?
- Develop concrete test cases!
  - Correctness and performance testing
- How will you measure the performance of the system?
- First basic performance measurements!

Be prepared to present your results next week!
We provide you data

- As a csv file at the webpage ([example1.zip](example1.zip))
- Will be extended in the next sessions!
- (StatisticId, Time, ExperimentId, NodeId) is unique

<table>
<thead>
<tr>
<th>Statistic Id</th>
<th>Statistic Name</th>
<th>Time</th>
<th>ExperimentId</th>
<th>Experiment Name</th>
<th>Node Id</th>
<th>Unit</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>109</td>
<td>managed/target density</td>
<td>2010-01-01 00:00:44</td>
<td>340</td>
<td>short-durable</td>
<td>37</td>
<td>bytes</td>
<td>704</td>
</tr>
<tr>
<td>109</td>
<td>managed/target density</td>
<td>2010-01-01 01:14:40</td>
<td>340</td>
<td>short-durable</td>
<td>19</td>
<td>seconds</td>
<td>32</td>
</tr>
</tbody>
</table>
Query language

- You do not have to use SQL!
- Some inspirations:
  - Some examples from the MSDN

```csharp
List<Customer> customers = GetCustomerList();
var orderCounts = from c in customers

List<Product> products = GetProductList();
var categories = from p in products group p by p.Category into g
                 select new { Category = g.Key, AveragePrice = g.Average(p => p.UnitPrice) };

var categoryCounts = from p in products group p by p.Category into g
                      select new { Category = g.Key, ProductCount = g.Count() };
```
Query language

- You do not have to use SQL!
- Some inspirations:
  - Use Scala for a domain specific language

```scala
// open DB
val db = new SimDB("myDB");

// read data cube
val ds = new ExperimentDBSourceDataCube(db)

// make query
val ds2 = ds.filter().equals("Experiment", 320).avg("Node", "Time").rotate("Statistic").orderBy("Time")

// materialize
val result = ds2.evaluate();
```
Think about Performance
Measure Performance

- Concentrate on Performance!
- This implies a basic understanding of the relevant knobs!

- How long does it take to read/write a certain amount on SSD?
  - Sequential access
  - Random access

- How long does it take to read/write a certain amount of RAM?
  - Sequential access
  - Random access

- Are there buffers?  
- Can you leverage multiple cores?
What’s next?

- Meeting in A213, 29.04.14, 13:30

- You will
  - present your results

- We will give you
  - feedback on your results & implementation idea
The Sorting Hat

- Choose your Team.