DB 2

Summer 2018

Torsten Grust
Universität Tübingen, Germany

1 Welcome all...

... to this course which is all about digging in the mud of database management system (DBMS) internals.

We will try to understand how DBMSs

1. **organize primary and secondary memory** (RAM and SSD/HDD) to store and query tables of data,
2. use **“data maps” (indexes)** to navigate huge tables,
3. internally **represent, optimize, and evaluate complex queries** over tables, and
4. **manage concurrent access to data** while avoiding inconsistencies and confusion.

But Why Would You Do That?

Insights into the DBMS kernel and its inner workings can explain, for example, ...

- why **query evaluation** takes (much) longer than expected (we are talking hours 😩 *vs.* msec 😀 here),
- whether **database growth** (by factor $n > 1$) will slow updates and/or queries by factor n (or n^2 or ...),
- whether the host's **resources** (CPU, cache, RAM, storage) are used effectively or if bottlenecks exist,
- how (known) **algorithms and data structures** hold up in the presence of huge inputs.

SQL is the Secret Key

I assume that you have working knowledge of **SQL,**[1] the ***"Intergalactic Dataspeak"*** used to manipulate and query tabular (or: relational) DBMSs.

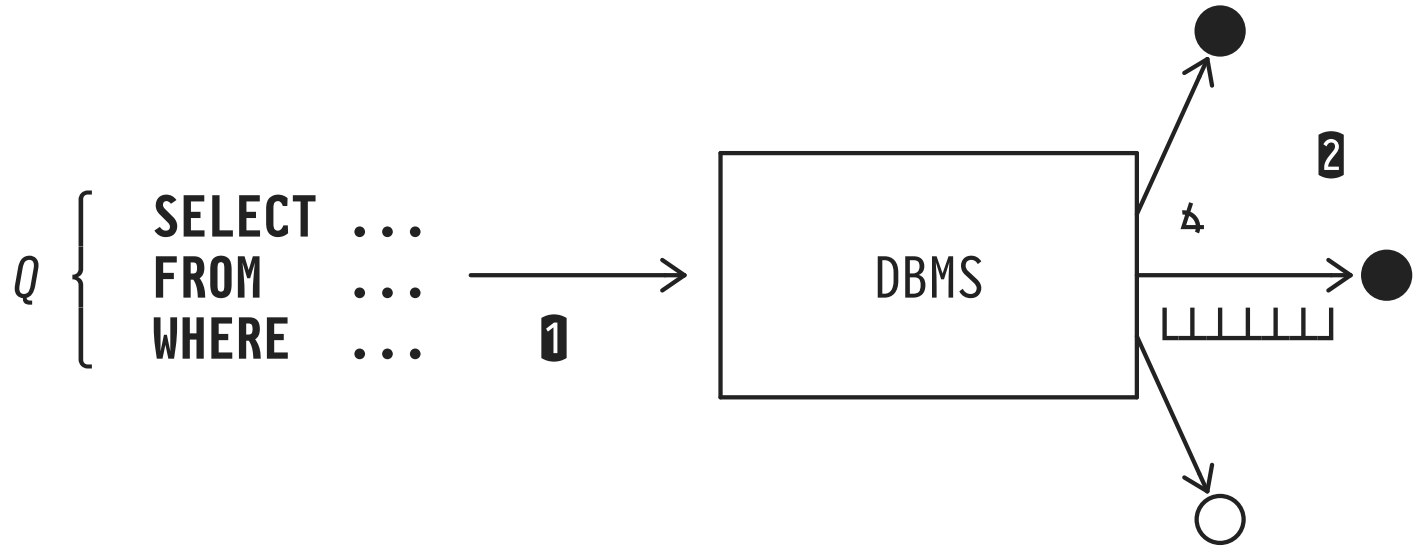
We will **submit a variety of SQL queries** Q to DBMSs and observe

- which components of the DBMS engage in processing Q,
- how DBMS-internal data structures and algorithms contribute to execute Q efficiently.

[1] As taught in courses "DB1" or "Advanced SQL", for example.

Sending “SQL Probes”

❶ Submit SQL probe query *Q* to DBMS.
❷ Observe/interpret DBMS response.



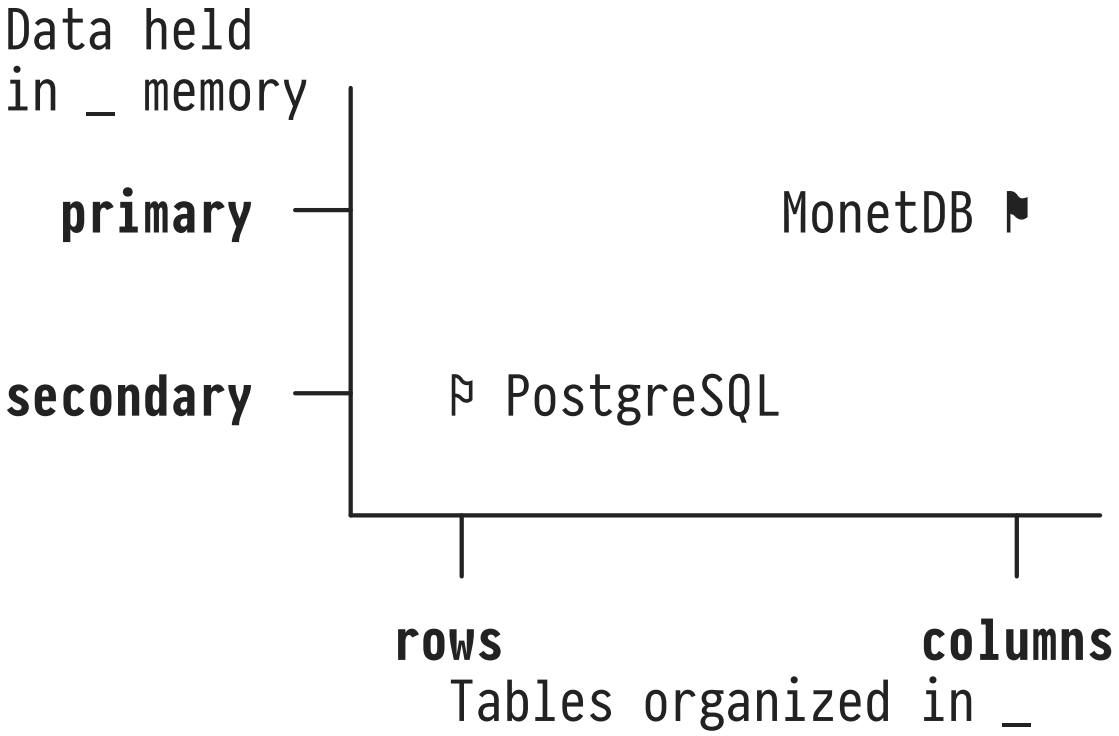
Thankfully, DBMSs provide several hooks and facilities that support the observation of their operation.

EXPLAIN Yourself, DBMS!

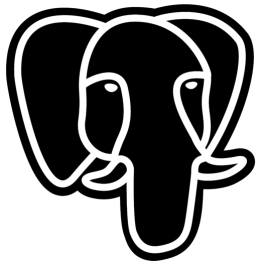
We will make extensive use of EXPLAIN facilities that act like a "DBMS X-ray." Instead of *Q*, submit EXPLAIN *Q* to reveal

- the **evaluation order** chosen to process the expressions/statements in *Q*,
- the **indexes** used to access tables,
- the **cost** and **resources** (# of I/O operations, memory buffers, CPU time) expended to process *Q*,
- the **algorithms** selected to access tables, manipulate rows, and compute *Q*'s results.

There is **one relational data model** but **a variety of DBMS** that implement it. We will focus on two DBMSs, **PostgreSQL** and **MonetDB**, whose internals deviate significantly:



PostgreSQL (postgresql.org)



- Row-wise table organization (▤)
- Data held in blocks on secondary storage (e.g., disks), database size virtually unlimited (32 TB in single table)
- Extensible database kernel (types, operations), support for a rich SQL:2011 dialect

- Actively developed since 1986 (derived from Ingres)
- In this course: **PostgreSQL ~~10.2~~ 10.3** (10.x will do)
- *Open source*, available on macOS , Linux , Windows

MonetDB (monetdb.org, developed at CWI Amsterdam)

- Column-wise table organization (🀫)
- Data completely mapped into primary memory for processing —once hot database size » RAM size, MonetDB struggles
- Optimized for CPU performance

- Actively used in database research since 1993 until today ("*the column store pioneers*")
- In this course: **MonetDB 5** (**v11.27.13**, “Jul2017-SP4")
- *Open source*, available on macOS , Linux , Windows

On the Bare Metal

Occasionally we will discuss/develop short **program fragments in C** to understand how DBMSs — MonetDB, in particular — process data internally.

No need to be a C whiz, but we will encounter:

- dynamic allocation of memory (`malloc(3)`)
- array types (`τ[]`), array access, array processing
- pointer types (`τ*`), pointer operations
- control flow (conditionals, loops)
- UNIX system calls (`open(2)`, `mmap(2)`, ...).

Torsten Grust?

Time Frame	Affiliation/Position
1989–1994	Diploma in Computer Science, TU Clausthal
1994–1999	Promotion (PhD), U Konstanz
2000	*Visiting Researcher*, IBM (USA)
2000–2004	Habilitation, U Konstanz
2004–2005	Professor Database Systems, TU Clausthal
2005–2008	Professor Database Systems, TU München
since 2008	Professor Database Systems, U Tübingen

- E-Mail: Torsten.Grust@uni-tuebingen.de
- Twitter: @Teggy (*Professor, likes database systems, programming languages, and LEGO ツ*)
- WSI, Sand 13, Room B318

Administrativa

Weekday/Time	Slot	Room
Monday, 10:15–11:45	Lecture	Sand 6/7, F119
Tuesday, 10:15–11:45	Lecture	Sand 6/7, F119
Thursday, 14:15–15:45	Tutorial	Sand 6/7, F119

- ⚠️ **No** lectures/tutorials on
  - Thursday, April 19 (tutorials start on April 26)
  - Tuesday, May 1
  - Thursday, May 10
  - Monday, May 21
  - Tuesday, May 22
  - Thursday, May 24
  - Thursday, May 31

Administrativa

End-Term Exam

- 90-min **written exam** on July 23, 10:00 (N10/Morgenstelle).
- You may bring a DIN A4 double-sided *cheat sheet*.
- Passing earns you 9 ECTS.

Weekly Assignments

- We will distribute, collect, and grade **weekly assignments** (Tuesday→Tuesday) via Github .
- Score $\geqslant \frac{2}{3}$ of the overall assignment points to be admitted to the exam and earn bonus points in the end-term exam.

Weekly Assignments & Tutorials

1. Expand on lecture material
2. Develop additional code, run additional examples, ...
3. Discuss solutions to weekly assignments

Organized and run by **Benjamin Dietrich** & **Christian Duta:**

- E-Mail: {b.dietrich,Christian.Duta}@uni-tuebingen.de
- WSI, Sand 13, Rooms B314 & B315

Assignments and tutorials will start in the second week of the semester once we have collected the first batch of interesting material.

Forum

In recent semesters we ran quite lively (and welcoming, friendly) forums for our courses. We aim to do so for **DB2**, too:

forum-db.informatik.uni-tuebingen.de/c/ss18-db2

- **Registration** (mandatory) and announcements
- Questions and answers (do *not* post complete solutions)
- Download additional code examples (SQL, MAL, and C)
- General discussion
- Quick turnaround (responses often within minutes)

Material

Course Homepage

db.inf.uni-tuebingen.de/teaching/DB2SS2018.html

- Download **slides** (PDF, also distributed via)
  New slide set developed while the semester runs — please be aware of bugs and report them. Thank you!

- **Contact information**
  Just drop by our offices (Sand 13), send e-mail first if you seek specific help/require longer attention.

Material

This course is *not* based on a single textbook but instead draws from

- a variety of scientific papers,
- textbook excerpts,
- blog and mailing list postings, Stack Exchange Q&As,[2]
- SQL references/standards,
- DBMS docs for PostgreSQL (★★★★★) and MonetDB (★★★☆☆),
- experience, and best practices.

[2] All of dba.stackexchange.com/questions/tagged/{sql,postgresql,monetdb} are worth a look.

Material (on Index Design and Usage)

- To the point, clear, and actionable advice on how to design “data maps”—or: **indexes**—and how to recognize whether a query can (not) benefit from an index.

- See use-the-index-luke.com [free HTML version] and sql-performance-explained.com [PDF 9.95€, Paperback 29.95€].