The TPC-H ad-hoc, decision support benchmark.

- (still) THE standard database OLAP benchmark
- By independent TPC organization
- All major DB vendors are members
- Official audited results (available online)

TPC-H Database Schema

- Synthetic data
- Database generator “dbgen”
- Variable database size:
  - Scale factor “SF”: 1, 3, 10, 30, 100, 300, …
  - SF-1 ~= 1 GB
- 22 query templates
- Query generator “qgen” to instantiate literals
TPC-H
ad-hoc, decision support benchmark

- “modes”
  - Single-client “power” (query time) test
  - Multi-client concurrent query throughput test

Official runs also include updates
Ignored / omitted here

Various metrics, also including price of system
Details online
Here: single-client query performance

Assignment 1

Optionals:
- Build the TPC-H data- and query-generator “dbgen” & “qgen” yourself:
  - Sources are in the “tpch_2_17_1.zip” archive on the course website
  - Go to ...
  - On Linux (and alike) build via `make -f Makefile.MonetDB`
    - Edit “Makefile.MonetDB” or “makefile.suite” accordingly for other systems
  - Generate the data:
    - In ...
    - This generates the data in ...
    - Change “1” to other number for other scale factors
  - Generate the queries:
    - In ...
    - This generates the queries in ...
    - Change “1” to other number for other scale factor

- Provided:
  TPC-H sources are in the “tpch_2_17_1.zip” archive on the course website
  In there,
  - the data- & query-generator “dbgen” & “qgen” are in ...
  - Data for SF-1 & SF-3 are in ...
  - Sample query results are in ...
  - In two formats: computer-readable .csv and human-readable .pretty
  - SQL schema creation and data loading scripts for MonetDB are in ...
  - Might also work for other DBMSs, possibly requiring minor syntax changes
  - Queries for MonetDB and SF-1 are provided in ...
  - If you want to run the queries on other scale factor than SF-1, you need to edit
    query 11 (“q11.sql”) as explained by the comment in “q11.sql”
    - Might also work for other DBMSs, possibly requiring minor syntax changes

Assignment 1

Tasks 1/2:
- Install MonetDB and one other DBMS of your choice
  - MonetDB: http://www.monetdb.org/
- With both systems, for at least scale factors SF-1 & SF-3 (using the provided scripts and data):
  - Create TPC-H schema
    - (.../dbgen/MonetDB/0-create_tables.sql)
  - Load TPC-H data
    - (.../dbgen/MonetDB/1-load_data.SF-*.sql)
  - (create constraints: primary- & foreign-keys)
    - (.../dbgen/MonetDB/2-add_constraints.sql)
  - Run TPC-H queries
    - (.../dbgen/MonetDB/q???.sql)
- Verify (for SF-1 & default query values) that results are correct
- Document in detail how and on what system you run:
  - hardware, OS, DBMS, configuration parameters, tuning parameters, etc.
  - Make sure that your documentation is sufficient for a third person to repeat your experiments
  - Compare query execution times between multiple runs of the same system and between systems
  - Graphically visualize times and differences
  - Explain – in your words and to the best of your knowledge – why the performances do (not)
    differ (per query) between runs and between systems

http://www.tpc.org/
Tasks 2/2:

- Implement queries Q1 & Q6 in a programming-, scripting-, statistical-, data analysis language (or system) of your choice (C, C++, Java, Python, R, ...)
  (hint: start with Q6, i.e., the simpler one of the two)
- Compare execution times of your implementation (for scale factors SF-1 & SF-3) to those of the DBMSs
  - If your implementation is single-threaded, you might want to compare to the DBMSs running both single- and multi-threaded (where applicable)
  - Explain – in your words and to the best of your knowledge – why the performances do (not) differ between your own implementation and the DBMS(s)

*Bonus points will be awarded for*
- each scale factor you use larger than SF-3 (with DBMSs and/or your own implementation)
- using more than one (significantly different) hardware platform (and discussing their effect on the observed performance)
- using more than two DBMSs (and discussing their effect on the observed performance)
- providing own implementations for Q1 & Q6 that “in fair comparison” are faster than MonetDB

TPC-H Q6:

```sql
SELECT
    SUM(l_extendedprice * l_discount) AS revenue
FROM
    lineitem
WHERE
    l_shipdate >= '1994-01-01'
    AND l_shipdate < '1994-01-01' + INTERVAL '1' YEAR
    AND l_discount BETWEEN 0.06 - 0.01 AND 0.06 + 0.01
    AND l_quantity < 24;
```

TPC-H Q1:

```sql
SELECT
    l_returnflag,
    l_linestatus,
    SUM(l_quantity) AS sum_qty,
    SUM(l_extendedprice) AS sum_base_price,
    SUM(l_extendedprice * (1 - l_discount)) AS sum_disc_price,
    SUM(l_extendedprice * (1 - l_discount) * (1 + l_tax)) AS sum_charge,
    AVG(l_quantity) AS avg_qty,
    AVG(l_extendedprice) AS avg_price,
    AVG(l_discount) AS avg_disc,
    COUNT(*) AS count_order
FROM
    lineitem
WHERE
    l_shipdate <= '1998-12-01' - INTERVAL '90' DAY (3)
GROUP BY
    l_returnflag, l_linestatus
ORDER BY
    l_returnflag, l_linestatus;
```

**Assignment 1**

- Work in groups of 5 students (13 groups?)
  - Preferably at least 3 CS students per group
  - Work together, cooperate, teach and learn from each other

**Produce:**

- A report (in PDF) that describes:
  - How you run the benchmark
  - such that the reader could repeat your experiments
  - How you implemented Q1 & Q6
  - How you verified that SF-1 results are correct
  - The results (execution times) you got
  - Your discussion of the results
- A compressed archive (e.g., zip) with
  - The scripts / programs you created and used
  - Your own implementation of Q1 & Q6
  - Query results achieved (with SF-1 & SF-3)
- Name your files as follows:
  - `<sorted-studentIDs>-report.pdf`, `<sorted-studentIDs>-archive.zip`

**Submit by email**

- To: S.Manegold@liacs.leidenuniv.nl
- Subject: [DBDM-2018] DB Assignment 1 (<sorted list of student IDs>)

**Deadline:** Sunday Oct 14, 2018, 23:59 CEST