COMP624121 Querying Data on the Web

Lab Work QS

The goal of this lab work is to move one step further from the preceding lab work and explore questions of database scale. You will still work with **sqlite3**.

For this purpose, we have made available five synthetically generated versions of the popular **BBD** (for **B**ar, **B**eers, and **D**rinkers) example database in which the sizes grow by a factor of approximately 10 relative to the preceding largest.

There is a **bbd.db** database that is a classical version, with imaginary (but realistic) data about bars, beers and drinkers. Play with it so that you develop an intuitive feel for the world it captures.

In respect of the databases we have generated, there are **five** versions with sizes roughly as follows:

bbd_1.db	100KB
bbd_10.db	2MB
bbd_100.db	40MB
bbd_1000.db	120MB
bbd_10000.db	1.3GB

Note that the largest of all is, for the capabilities of the **sqlite3** DBMS, very very large indeed.

So, make sure that you work on the smaller sizes (i.e., from **bbd_1.db** to **bbd_100.db**) before you attempt the two largest. Even them, while **bbd_1000.db** may be usable, the very largest one may lead to very large response times or even not terminate at all.

Within bounds of common sense, if the script timeouts on the smaller three databases, then increase the time out sensibly (half as much again, or double, or triple).

It may not be possible to run the script in sensible time for the largest databases. In all cases, note down for every query that run the time it took and for those that timed out, note that they did so by assigning them a notionally infinite time.

Task 1: Write **eight** SQL queries over the **BBD** schema which in terms of relational algebra are of the following types:

(1) a simple selection (σ) query over a single relation
(2) a simple projection (π) query over a single relation
(3) a σ query over the Cartesian product[1] (×) of two relations
(4) if possible, reuse the previous query and ensure duplicate (δ) removal
(5) if possible, reuse the previous query and instead of Cartesian product, used the natural join (⋈)
(6) a union (∪) query
(7) an intersection (∩) query
(8) an aggregation (γ) query

Task 2: Using the knowledge you've acquired in the lectures about these query types and in last week's lab work about the **sqlite3** optimizer, for each of the eight queries write a small justification as to whether you'd expect that query to throw a challenge to **sqlite3**. Just as you did in the preceding lab work, take timings as recommended for experimental analysis.

Task 3: Summarize your investigation with one or more plots, accompanied by interpretation and comment, that evaluates the response time of **sqlite3** on the eight queries over the five databases. In particular, comment (in the light of your expectations in Task 2) whether any actual result seems surprising. Again, you should work with .sql scripts and generate .log files to make your work more systematic and agile.

---

[1] If you don't know already, read the **sqlite3** documentation to find out how to specify a Cartesian product (also called cross product, or, less commonly, cross join).

Marking

- Each of query in Task 1 is worth up to 5 marks, for a total of up to 40 marks.
- Each justification in Task 2 is worth up to 5 marks, for a total of up to 40 marks.
- Task 3 is worth up to 60 marks.
- The whole lab is worth up to 140 marks and contributes up to 14 marks to the final mark for the course unit.

Software/Data

BBD databases

The **BBD** databases are self-explanatory. Play with the **bbd.db** version in order to develop a feel for what the data means.

You need the database to be local to you, so that you have write permissions on it. Data is always under:

    /opt/info/courses/COMP62421/data

For this lab work, you will use the **BBD** databases in

    /opt/info/courses/COMP62421/data/relational

You should copy the database and the properties files to you working directory

    /opt/info/courses/COMP62421/data/relational/bbd.db
    /opt/info/courses/COMP62421/data/relational/bbd_1.db
    /opt/info/courses/COMP62421/data/relational/bbd_10.db
    /opt/info/courses/COMP62421/data/relational/bbd_100.db
    /opt/info/courses/COMP62421/data/relational/bbd_1000.db
    /opt/info/courses/COMP62421/data/relational/bbd_10000.db

sqlite3

As before, you may benefit from passing the initialization file we prepared (setting parameters for the **sqlite3** command-line interface). This is available in

    /opt/info/courses/COMP62421/data/relational/sqliterc

If you have copied it into your working directory, you can invoke **sqlite3** as follows:

    sqlite3 -init sqliterc bbd.db

for the original BBD database. Of course, for the five other versions, which are the ones that you will really use for Task 2, you should use the other databases you have copied, which are suffixed with an indication of their order or magnitude.

For **sqlite3**, the documentation in the website is very good, but if you prefer learning from books, you have free access (from an UoM IP address) to this one:

    The Definitive Guide to SQLite
    Grant Allen, Mike Owens
    ISBN: 978-1-4302-3225-4
    Apress, 2010
    http://link.springer.com/content/pdf/10.1007%2F978-1-4302-3226-1.pdf