Carnegie Mellon University
Department of Computer Science
15-826 Multimedia and Data Mining
C. Faloutsos, Fall 2024

Homework 1 - Solutions

Due: **hard copy, in class, at 2:00pm, on 09/06/2024**

VERY IMPORTANT - check-list:

1. Deposit **hard copy** of your answers, in class. For ease of grading, please **type** the full info on each page:
   - your *name* and *Andrew ID*,
   - *Course#* and *Homework#*.
2. **Typeset** all of your answers (eg., ascii, pdf, msword, etc). Handwritten responses may get **zero** points, at the discretion of the grader.
3. **Staple** them, if you use more than 1 page.

Reminders:

- *Plagiarism*: Homework is to be completed *individually*.
- *Late homeworks*: please follow standard policy, i.e., please email your homework
  - to the instructor
  - with the subject line exactly `15-826 Homework Submission (HW 1)`
  - and the count of slip-days you are using.

For your information:

- Graded out of **100** points; **2** questions total
- Rough time estimate: *2-6 hours*
- Weight: 1% of course grade.

Revision : 2024/09/10 13:58

Question	Points	Score
B-trees	10	
SQL	90	
Total:	100	

Question 1: B-trees . . . . . [10 points]

Consider B-trees of order d=2 (2*d+1 = 5 = maximum fanout). One such tree is in Figure 1.

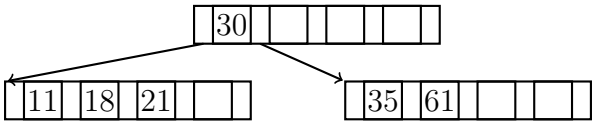

Figure 1: A B-tree of order d=2, with n=3 nodes, and height h=2.

NO NEED to justify your answers.

(a) **[5 points]** In an initially empty B-tree of order 2, insert the first 5 integers: 1,2,3,4,5 in this order. How many nodes will the tree have?

(a) **3**

(b) **[5 points]** In an initially empty B-tree of order 2, insert the first 15 integers 1,2,. . .,15, in this order. How many keys will the be in the root of the tree?

(b) **4**

Question 2: SQL . . . . . [90 points]

For this part, we will use `sqlite3` (version 3.7.17), which is available on the andrew unix machines (`ssh unix.andrew.cmu.edu`).

Set up

1. Download the (380MB) database file with the patent citation graph from `https://www.cs.cmu.edu/~christos/courses/826-resources/DATA-SETS-HOMEWORKS/patents/patents.db`
2. At the unix/linux prompt, open the database with the following command:

   ```
   sqlite3 patents.db
   ```

   which should bring you the `sqlite>` prompt.

Optional set-up steps

1. **Sanity checks**:
   (a) the command

   ```
   sqlite> .schema Patents
   ```

   should give:

   ```
   CREATE TABLE Patents( "CITING" TEXT, "CITED" TEXT );
   ```

(b) Check the count of rows - the command:

```
select count(*) from Patents;
```

should give

```
16522438
```

(= total number of rows)

2. **Fun fact**: At `https://ppubs.uspto.gov/pubwebapp/static/pages/ppubsbasic.html` you can look up for the full info about each patent (title, year, inventors, e.t.c.). This could help you double-check the correctness of your responses.

Data description: The `patents.db` database has one table `Patents`, listing which patent cites what patent. For example the following row in the table means that patent number `5856190` is citing patent number `4216617`.

```
CITING      CITED
----------  ----------
5856190     4216617
```

Queries, and what to hand in: For all the queries below, hand in hard copy of

- both the SQL **code** of your answer,
- as well as the **output** of your code.

Hint: Use `.headers on` and `.mode column` for easier debugging.

(a) **[30 points] Loops:** Check if there are any loops, that is 'A' cites 'B', and 'B' cites 'A'. Specifically, report all the pairs of patent-ids that form such loops. Self-loops should be included ('A' cites 'A'), if any.

Hint: It may take some time, since there are no indices

Solution: Code:

```
select p1.CITING, p1.CITED
    from Patents as p1, Patents as p2
    where p1.CITING = p2.CITED
        and p1.CITED = p2.CITING;
```

Grading info: full points for all correct alternatives (using 'views' is fine).

Grading info: no partial credit, if there are serious errors.

Solution: Output:

```
CITING   CITED
-------  -------
5489070  5489070
```

Grading info: *no partial credit, if there are serious errors.*

(b) **[30 points] Highly cited patents:** Find the patents (if any) that are cited more than n=700 times; list the patent-id (CITED) and the count of citations it has received; give the most cited ones first; break ties (if any) by smallest patent-id first.

(FYI - Relationship to data mining: Grouping, sorting, and spotting of 'heavy hitters' are vital, for several data mining tasks like information summarization and anomaly detection.)

Solution: Code:

```
select CITED, count( CITING)
    from Patents
    group by CITED
    having count(CITING) > 700
    order by count(CITING) desc, CITED;
```

Grading info: *full points for all correct alternatives (using 'views' is fine).*

Grading info: *no partial credit, if there are serious errors.*

Solution: Output:

```
CITED    count( CITING)
-------  --------------
4723129  779
4463359  716
```

Grading info: *no penalty if there are no column headers*

Grading info: *-1 if small errors*

Grading info: *if there are serious errors, partial credit of 1pt per correct tuple*

(c) **[30 points] Top-5 most-citing patents:** Almost the reverse of the previous question - find the 'encyclopedias': the patents that have the largest bibliographies, that is, the patents that cite a lot of other patents. For each of the top k=5 'encyclopedias', give the patent id (`CITING`) and number of patents it cites. Order the results by most citations first, and then sort by patent-id in ascending order (to break ties, if any).

Hint: use the keyword: `limit`.

Solution: Code:

```
select CITING, count( CITED)
    from Patents
    group by CITING
    order by  count(CITED) desc, CITING
```

```
        limit 5;
```

Grading info: *full points for all correct alternatives (using 'views' is fine).*
Grading info: *-1 for each small error (wrong ordering, etc)*
Grading info: *no partial credit, if there are serious errors.*

Solution: Output:

```
CITING   count( CITED)
-------  -------------
5795784  770
5887243  745
5856194  737
5855655  626
5891229  626
```

Grading info: *-1 if the ordering is wrong*
Grading info: *no penalty if there are no column headers*
Grading info: *-1 if other small errors*
Grading info: *if there are serious errors, partial credit of 1pt per correct tuple*