CS150A Homework 1 – Writing

School of Information Science and Technology
October 3, 2022

1 SQL (40 PTS)

Consider the following relations with underlined primary keys:
Student (sname, sid, gpa, level, deptid)
Dept (dname, deptid)
Course (cid, cname, deptid, units)
Takes (sid, cid)

1. Write a SQL query that returns the names (i.e., snames) of students who have taken more courses outside their department than inside their department. For this question, you can assume that all students in the database have taken at least one course inside their department.

SELECT S.sname from Student S
WHERE (SELECT COUNT (*)
FROM Takes T, Course C
WHERE S.sid = T.sid AND T.cid = C.cid AND C.deptid = S.deptid)
< (SELECT COUNT(*)
FROM Takes T2, Course C2
WHERE S.sid = T2.sid AND T2.cid = C2.cid AND C2.deptid != S.deptid)
(Note: The above solution does not even require that each student takes at least one course in her department)

Another variation seen in solutions:
SELECT S.sname
FROM Student S, Takes T, Course C
WHERE S.sid = T.sid AND T.cid = C.cid AND S.deptid = C.deptid

GROUP BY S.sid, S.sname
HAVING count(*) <
(SELECT COUNT (*)
FROM Takes T1, Course C1
WHERE S.sid = T1.sid AND T1.cid = C1.cid AND C1.deptid != S.deptid)

2. Which of the following queries returns the department numbers of those departments for which there are no courses being offered? (The correct answer may be more than one).

   A. SELECT D.deptid
      FROM Dept D, Course C
      WHERE D.deptid NOT EQUAL C.deptid;

   B. SELECT C.deptid, COUNT(C.deptid)
      FROM Course C
      GROUP BY C.deptid
      HAVING COUNT (C.deptid) = NULL;

   C. SELECT C.deptid
      FROM Course C
      WHERE C.deptid NOT IN (SELECT * FROM Dept);

   D. **SELECT D.deptid**
      FROM Dept D
      WHERE NOT EXISTS (SELECT * FROM Course C
      WHERE C.deptid = D.deptid);

   E. None of the above

3. Which of the following queries returns the id of the student with the highest GPA? (The correct answer may be more than one).

   A. SELECT S.sid
      FROM Students S
      WHERE S.gpa = MAX(S.gpa);

   B. SELECT S.sid, MAX(S.gpa);
      FROM Students S

GROUP by S.gpa

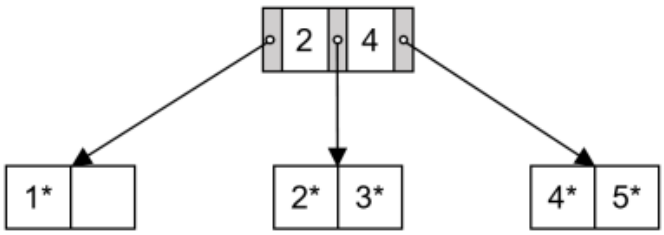
C. SELECT S.sid
FROM Student S
WHERE S.gpa > ALL (SELECT S.gpa FROM Student S);

D. **SELECT S.sid**
FROM Student S
Where S.gpa = (SELECT MAX(S.gpa)
FROM Student S);

E. None of the above

2 INDEX AND B+ TREES (30 PTS)

1. The following B+ Tree (height=1) has order 1 (max fan-out of 3) and each leaf node can hold up to 2 entries. Answer each of the follow questions independently of each other.

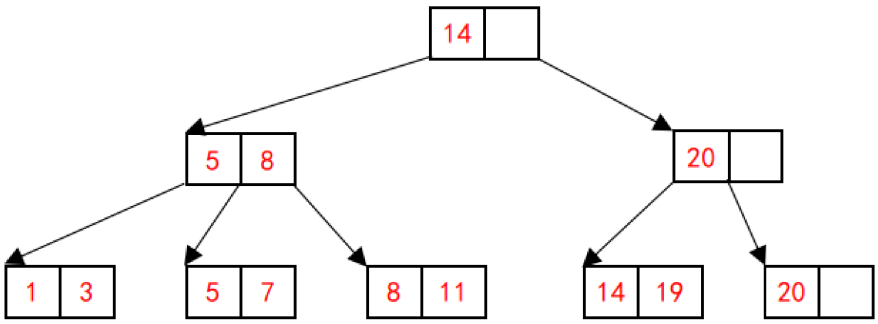


   A. What value(s) would be in the root node if we were to insert 0?
   2,4

   B. What value(s) would be in the root node if we were to insert 6?
   4

   C. Starting with the height 1 tree in the picture above, suppose we start inserting keys 6, 7,8, ... and so on. After inserting what key will the height of the tree become 3?
   10

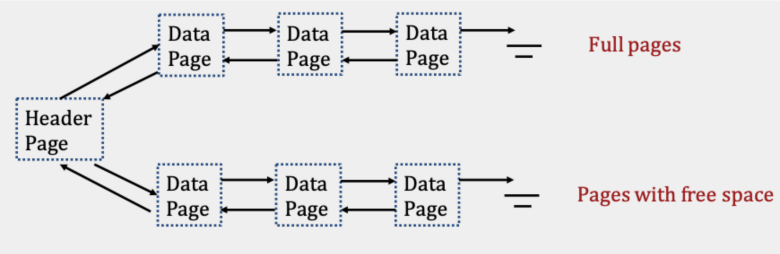
2. What is the resulting B+ tree after bulk loading the keys: 3, 8, 5, 19, 1, 20, 11, 7, 14? Assume we are building an order d = 1 index with a fill factor of 2/3.



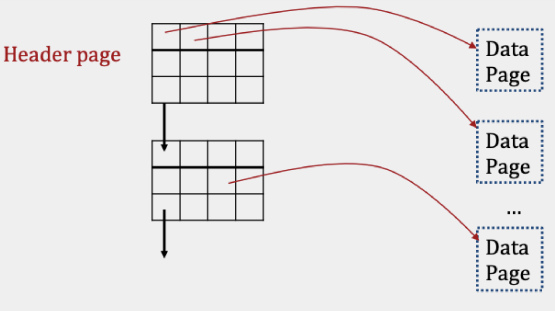
3 File Organization (30 pts)

For heap files, we have two implementation methods:

- Linked List Implementation



- Page Directory Implementation



1. Suppose we have a heap file A implemented with a linked list. It has 6 pages with free space, at least one of which has enough space to insert a record, and 12 full pages. In the worst case, writing a record to the final page with free space will require how many I/Os? Give your result and brief explanations.
   (Hint: Consider what happens when after writing, the page becomes full and assume that the header page can insert at the beginning of the full pages linked list.)

   Solution:

   12 I/Os.

   First, you need 7 I/Os to find the final page: read in the header and then the 6 pages in the free pages linked list before finding a page with enough free space in the final page.

   Then you need 1 I/O to write the record to the final page, 1 I/O to change the pointer from the previous free page, 1 I/O to change the header page (since the final page is now full and in the front of the full pages), and 2 I/Os to change the previous front of the full pages (read from disk, modify, and write to disk).

2. Suppose we have a heap file B implemented with a page directory. One page in the directory can hold 16 page entries. There are 86 pages in file A in total. In the worst case,

writing a record to a page with free space will require how many I/Os (assuming at least one free page with enough space to insert a record exists)? Give your result and brief explanations.

Solution:

9 I/Os.

First, you need 6 I/Os to find the free page: There will be $\lceil 86/16 \rceil = 6$ pages in the directory. In the worst case, we will need to look through all 6 pages in the directory. (The directory contains the amount of free space for each page.)

Then we also have to read the page with the free space (1 I/O), write a record to that free page (1 I/O), and write to the page directory (1 I/O), for a total of 9 I/Os.