Name Peter Perfecto

Class Account: _____

UNIVERSITY OF CALIFORNIA
Department of EECS, Computer Science Division

CS186 Hellerstein
Spring 2009 Midterm Exam #1

Midterm Exam: Introduction to Database Systems

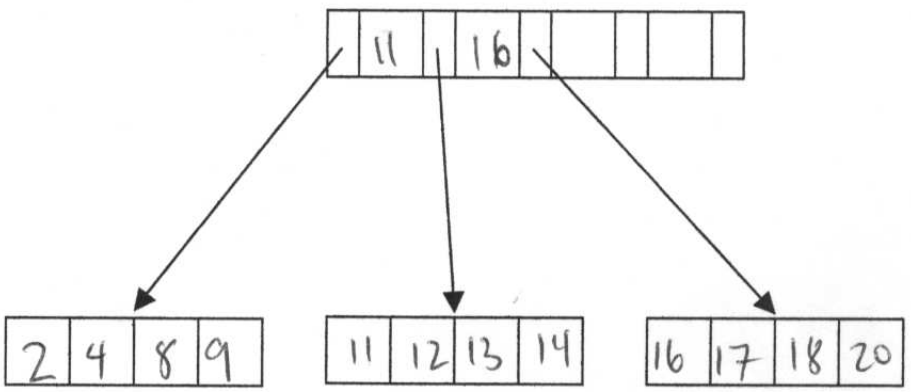
This exam has four problems worth different amounts of points each. Each problem is made up of multiple questions. You should read through the exam quickly and plan your time-management accordingly. Before beginning to answer a problem, be sure to read it carefully and to *answer all parts of every problem!*

You **must** write your answers on the exam. Extra answer space has been provided at the back in case you run out of space while answering. If you run out of space, be sure to make a "forward reference" to the page number where your answer continues. ***Do not tear pages off of your exam!***

1. Indexes [14 points]

Consider a B+ tree containing the elements 2, 4, 8, 9, 11, 12, 13,14, 16, 17, 18, 20. The tree has order d=2. This means that internal nodes have at least 2 keys and 3 pointers, and at most 4 keys and 5 pointers; leaf nodes have at least 2 data entries and at most 4 data entries.

a) **[2 point]** In the picture below, draw what the tree looks like if the leaves are as densely packed as possible. Draw directly on the picture.

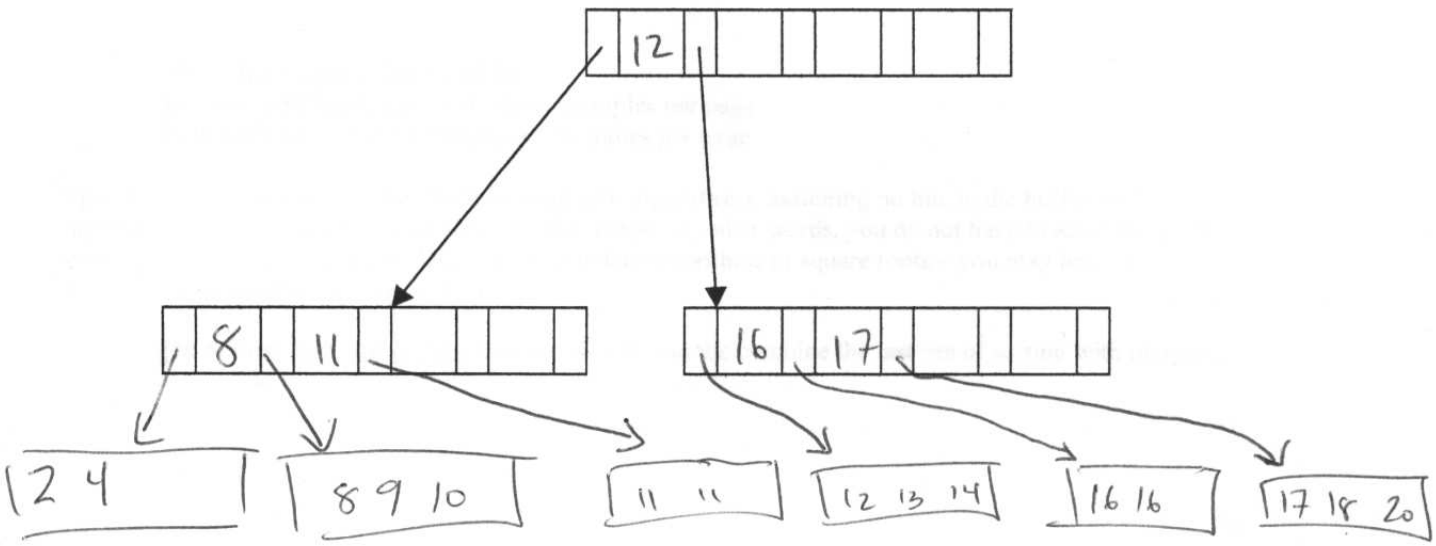


b) **[6 points]** Based on your original tree from (a), what is the **minimum** number of inserts that could cause a root split?

3

Name_____

c) **[6 points]** Draw the tree just after the root has split with a minimum of insertions as you described in (b). Use any values you like that would cause the split. We have started you off by drawing the root and internal nodes; you need to fill them in and draw the leaves.



(2 pts given for a legal tree)

1. **Joins**

Consider the following schema of a medical records system. Primary keys are underlined.

```
patient(patient_id, name, age)
visit(visit_id, patient_id, doctor_id, cost)
```

Consider the following SQL query:

```
SELECT *
FROM patient, visit
WHERE patient.patient_id = visit.patient_id;
```

Given: the buffer manager has 52 buffers.

Specify the number of IOs for the following join algorithms, assuming no hits in the buffer pool. You are not required to calculate logarithms or square roots – you may leave them written as expressions (e.g. $\log_{12}(749)$). There should be no variables in your answers.

a) **[10 points]** Sort-Merge Join. Use quick-sort and the combine the last run of sorting with merging, if possible

Answer
Number of passes to sort VISITS: ceiling($\log_{51}$ (pages(VISITS) / 52)) = 3
Read then write VISITS 2 times to ~190, then 4 runs
Number of passes to sort PATIENTS: ceiling($\log_{51}$ (pages(PATIENTS) / 52)) = 2
Read then write PATIENTS to 20 sorted runs
Read then write PATIENTS and VISITS to sort and merge
= penultimate-pass sort VISITS + penultimate-pass sort PATIENTS + Read, merge, write both relations
*= 4*10,000+2 * 1000 + 2 * (10,000 + 1000) = 64,000*

b) **[10 points]** Index Merge Join. There is a **clustered** B+ tree index on `visit`.patient_id. Tree height = H. A patients visits fit on a single page.

2*1000 + 1000 + (10,000*H)
sort
read patient in order
descend to left most Scan →

Name_____

3. Buffer Management [12 points]

Suppose you have a Buffer Pool that can hold 3 pages. There are 26 blocks on your disk; for simplicity we will refer to each block with a letter of the English alphabet between A and Z. An "access pattern" is a string of letters that log the requests to the Buffer Pool for pages. For each access pattern below, write down next to it the number of I/Os that would occur *starting with an empty buffer pool each time,* for both the LRU and MRU replacement policies. (Spaces in the access patterns are there just for legibility.)

access pattern	LRU	MRU
ZYXW ZYXW	8	5
ABCD DCBA ABCD	6	6
ACEG BDFH ACEG BDFH	16	13

[2 points per box]

Name_____

4. ER [21 points]

We are asked to design a database to model doctor's offices across California. In addition to enumerating the various entities and relationships as we have done below, we need to define a number of constraints to ensure that we fit the semantics of an office as closely as possible.

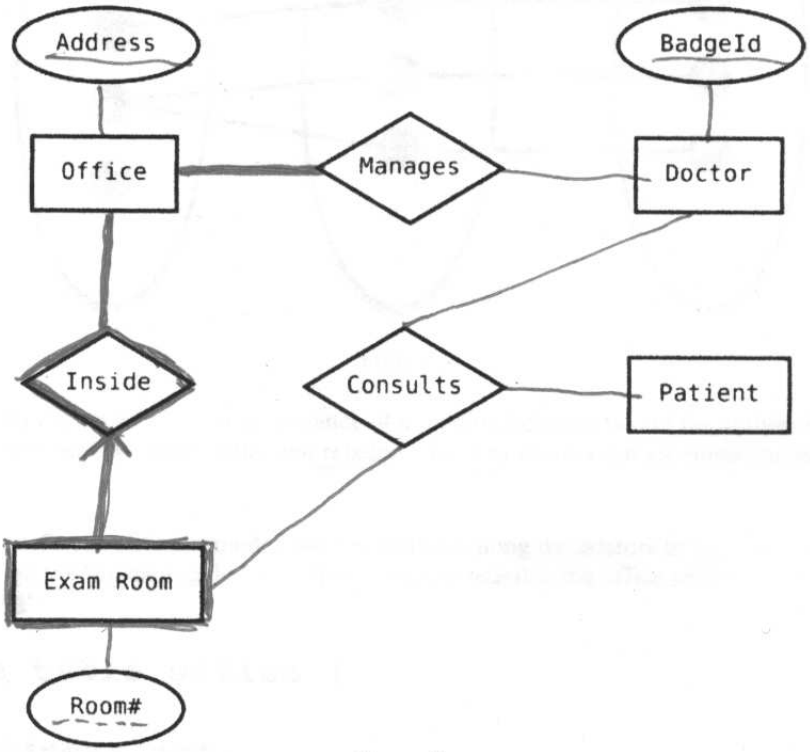


Figure 1.

Draw the answers to the questions below in Figure 1.

a) **[2 points]** An office may be managed by more than one doctor, but we must ensure that every practice has at least one. A doctor is uniquely identified by their BadgeId, and may manage more than one office. Draw the necessary constraints to capture this requirement.

b) **[3 points]** An office is identified by its Address, and contains one or more exam rooms. A room can be identified by its room number, and the office that it is in. Create the appropriate constraints, and modify or add the necessary attributes.

c) **[5 points]** When a patient visits an office, she has a consultation with a doctor in an examination room. For there to be a consultation, there must be at least one doctor, patient, and room.

Name_____

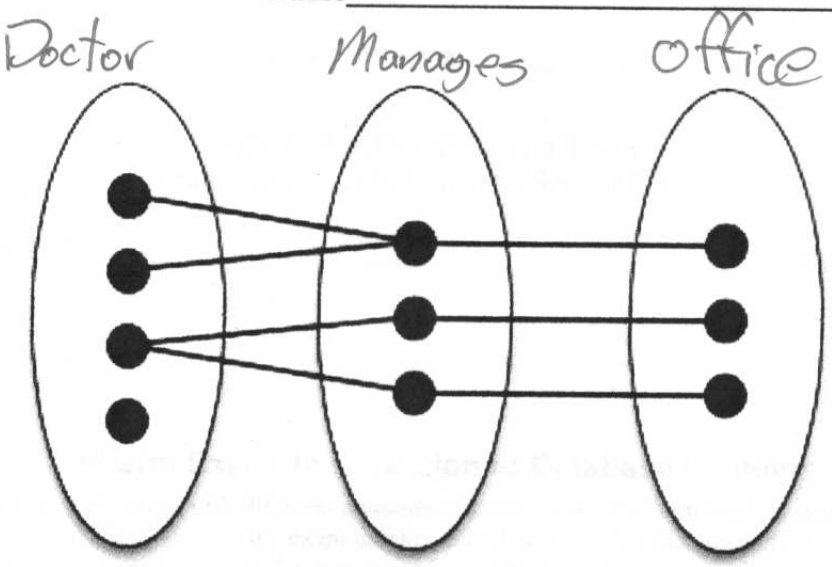


Figure 2.

d) **[5 points]** Figure 2 describes an instance of a relation between two of the entities in our model. Label the 3 ovals with the entity and relation names to which each set could correspond.

e) **[6 points]** To simplify our model, we decide that among the doctors in a given practice, one is designated the "managing doctor". How could be redefine the office relation express this constraint?

```
create table office (
    address int,
    badge
    doctor_id  int NOT NULL
    Primary key (address)
    Foreign key (badge_id) references doctor
)
```