Name:

CS 33007 Introduction to Database System Design, Fall 2017

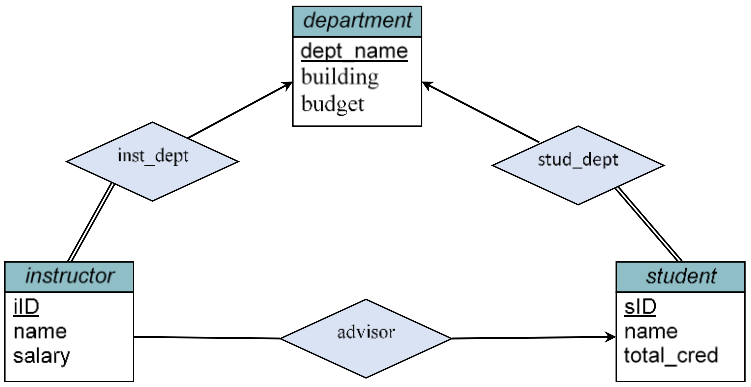
Final Exam Solution

Instructions:

- *This examination is closed book (no access to book, lecture notes, phone, laptop, tablet etc.). But you can have notes on one side of a paper.*
- *Please write your answer in the given blank space for each question. If your answer doesn't fit in the given space, you can use back side of the papers but write question number.*

Total Points:100 (25 points for each question) **Time:** 12:45 PM – 3PM

1.
   a. Convert the following ER diagram to non-redundant relation schemas. **[15 points]**
   b. The *advisor* relationship shown in the ER diagram is not practical- Explain why? **[10 points]**



Answer:

a.

Relation schemas:
instructor (<u>iID</u>, name, dept_name, salary)
student (<u>sID</u>, name, dept_name, total_cred)
department (<u>dept_name</u>, building, budget)
advisor(<u>iID</u>, sID)

b. In practice, number of students are much higher than number of instructors. The advisor relationship of the ER diagram restricts an instructor to advise at most one student. So many students will be left without any advisor and won't be possible to assign any advisor even when required.

2.

a. Between BCNF and 3NF, which one allows data redundancy and why? **[10 points]**

b. Given F = {A $\rightarrow$ B, A $\rightarrow$ C , CG $\rightarrow$ H, CG $\rightarrow$ I, B $\rightarrow$ H}, set of functional dependencies on R(A, B, C, G, H, I), check if AG is a candidate key of R. **[15 points]**

Answer:

(a) 3NF allows data redundancy to ensure functional dependency preservation.

(b) (i) check if AG is super key (AG->R) (ii) Then AG is candidate key only if none of A->R and G->R holds.

[Apply algorithm of finding closure of attribute sets to know if AG->R, A->R or G->R hold]

Calculate $(AG)^+$:
Initialization:
result = AG
Steps in while loop:
1. *result = ABCG(A* $\rightarrow$ *C* and *A* $\rightarrow$ *B)*
2. *result = ABCGH* *(CG* $\rightarrow$ *H* and *CG* $\subseteq$ *AGBC)*
3. *result = ABCGHI* *(CG* $\rightarrow$ *I* and *CG* $\subseteq$ *AGBCH)*

As, $(AG)^+ \supseteq$ R, so AG->R holds, in other words AG is super key of R.

Calculate $(A)^+$:
Initialization:
Result = A
Steps in while loop:
1. Result = ABCH (for A->B, A->C and B->H)
2. Result = ABCH, No change – exit loop

$(A)^+$ = ABCH, so A->R doesn't hold. A is not a super key.

Calculate $(G)^+$:
Initialization
Result = G
Steps in while loop:
1. Result =G (no change), exit from loop

$(G)^+$= G, So G->R doesn't hold, G can't be a super key

So, we can say, AG is the minimal attribute set to be super key for R. So, AG is candidate key for R.

3.

a. How does RAID provide high capacity, speed and reliability? **[15 points]**

b. What are the benefits of accommodating more sectors in outer track than inner tracks of a disk. **[10 points]**

Answer:

a.

i. High capacity – creating large storage system with many disks.

ii. High speed – allowing parallel data access from multiple disk.

iii. High reliability – By redundant data (storing duplicate data, recovery data etc) which can be used to recover the system from disk failure.

b. he main performance effect of storing more sectors on the outer tracks and fewer sectors on the inner tracks is that the disk's **data transfer rate will be greater on the outer tracks than the inner tracks**. This is because the disk spins at a constant rate, so more sectors pass underneath the drive head in a given amount of time when the arm is positioned on an outer track than when on an inner track. In fact, some high-performance systems are optimized by storing data only in outer tracks, so that disk arm movement is minimized while maximizing data-transfer rates.

4.

a. What are the advantages and disadvantages of sparse and dense indices. **[12 points]**

b. Suppose available buffer size to perform a join operation is 1GB (1024MB). Block size is 10MB. Two tables to be joined are ***students*** and ***instructors***. Table *students* has 500 records which takes 100 blocks to store. Table *instructors* has 100 records and take 400 blocks to store. If you use nested join loop algorithm for joining.

i. Which table should be used in inner loop and why.

ii. What will be the number of block transfers and seeks. **[13 points]**

Answer:

a.

sparse indices:

Advantage – Less space

Disadvantage – slower than dense indices to locate records

Dense Indices:

Advantage: Fast to locate records.

Disadvantage- large space to store index file.

b.

i. In nested loop, all blocks of inner loop table need to be accessed for every record of outer loop table. So inner table is accessed frequently. Since the students table entirely can fit into the buffer, loading whole student table in buffer and using it in inner loop will reduce overall number of block transfers and seeks.

ii. Each table need to be read only once from the disk. So total block transfer = 100+400 =500. Each table need to seek only once and then access whole table sequentially. So, total number of seeks is 2.