CSC540

Written Assignment 4

Due date Dec. 4, midnight. (e-submission)

Qu. 1.

Consider the attribute set *R* = *ABCDEGH* and the FD set *F* = *{AB* → *C, AC* → *B*, *AD* → *E*, *B* → *D*, *BC* → *A*, *E* → *G}*.

a.) For attribute set *ABCEG*, do the following:
(i) Compute the set of dependencies that hold over the set and write down a minimal cover.
(ii) Name the strongest normal form that is not violated by the relation containing these attributes.
(iii) Decompose it into a collection of BCNF relations if it is not in BCNF.

b.) Which of the following decompositions of *R* = *ABCDEG*, with the same set of dependencies *F*, is (a) dependency-preserving? (b) lossless-join?. Explain your answer. (you may use examples to explain)

(i) *{AB*, *BC*, *ABDE, EG }*
(ii) *{ABC, ACDE, ADG }*

Qu. 2.

Suppose you are given a relation *R(A,B,C,D)*. For each of the following sets of FDs, assuming they are the only dependencies that hold for *R*, do the following: (a) Identify the candidate key(s) for *R*. (b) State whether or not the proposed decomposition of *R* into smaller relations is a good decomposition and briefly explain why or why not.

i.) *B* → *C, D* → *A*; decompose into *BC* and *AD*.
ii.) *AB* → *C, C* → *A, C* → *D*; decompose into *ACD* and *BC*.
iii.) *A* → *BC*, *C* → *AD*; decompose into *ABC* and *AD*.
iv.) *A* → *B*, *B* → *C*, *C* → *D*; decompose into *AB* and *ACD*.
v.) *A* → *B*, *B* → *C*, *C* → *D*; decompose into *AB*, *AD* and *CD*.

Qu. 3.

Consider the following relational schema and SQL query. The schema captures information about employees, departments, and company finances (organized on a per department basis).

```
Emp(eid: integer, did: integer, sal: integer, hobby: char(20))
Dept(did: integer, dname: char(20), floor: integer, phone: char(10))
Finance(did: integer, budget: real, sales: real, expenses: real)
```

Consider the relational expression

$\Pi_{D.dname,F.budget}(((\Pi_{E.did}(\sigma_{E.sal>=59000,E.hobby="yodelling"}(E)))\text{join}$

$\Pi_{D.did,D.dname}(\sigma_{D.floor=1}(D)))\text{join }\Pi_{F.budget,F.did}(F))$

for the SQL query:

```
SELECT D.dname, F.budget FROM Emp E, Dept D, Finance F
WHERE E.did=D.did AND D.did=F.did AND D.floor=1
AND E.sal ≥ 59000 AND E.hobby = 'yodeling'
```

Suppose that the following additional information is available: Unclustered B+ tree indexes exist on *Emp.did*, *Emp.sal*, *Dept.floor*, *Dept.did*, and *Finance.did*. The system's statistics indicate that employee salaries range from 10,000 to 60,000, employees enjoy 200 different hobbies, and the company owns two floors in the building. There are a total of 50,000 employees and 5,000 departments (each with corresponding financial information) in the database. The DBMS used by the company has just one join method available, *index nested loops*.

(a) Draw a relational expression plan tree (nodes are annotated with physical operators) that is equivalent to the expression.

(b) For each of the query's base relations (Emp, Dept, and Finance) estimate the number of tuples that would be initially selected from that relation if all of the non-join predicates on that relation were applied to it before any join processing begins.

(c) Given your answer to the preceding question, estimated cost of evaluating the plan if the order of order of join processing is as given i.e. ((E join D) join F)