Query Processing – Practice Questions

1. Consider these relations with the following properties:

r(A, B, C)	s(C, D, E)
30,000 tuples	60,000 tuples
25 tuples fit on 1 block	30 tuples fit on 1 block

a) Estimate the number of disk block accesses required for a natural join of r and s using a nested-loop join if r is used as the outer relation.

b) Estimate the number of disk block accesses required for a natural join of r and s using a block nested-loop join if s is used as the outer relation. Assume that there are more than 2000 memory buffers available to facilitate this operation, where each memory buffer can buffer one disk block.

2. Consider the following SQL query on the schema *branch(branch_name, branch_city, assets)*:

```
select t.branch_name
from branch t, branch s
where t.assets > s.assets and s.branch_city = 'Burnaby';
```

Write an efficient relational algebra expression that is equivalent to this query and JUSTIFY your choice with an explanation.

3. Suppose we have the following relations:

employee(<u>emp_id</u>, salary, age, dept_id)
department(<u>dept_id</u>, budget, status)

Each *employee* record is 20 bytes long and each *department* record is 40 bytes long. There are 20,000 tuples in the *employee* table and 5000 tuples in the *department* table. The *dept_id* attribute in *employee* is a foreign key of the *department* relation. The file system supports a page size of 4000 bytes and there are 12 buffer pages available to the database. Assume we are using the number of page I/O's as the measure of a query's cost. The following indices exist:

- a clustering (*i.e.* primary) index on the *dept_id* attribute in *employee*,
- a non-clustering (*i.e.* secondary) index on the *age* attribute in *employee*,
- a clustering index on the *dept_id* attribute in *department*

a) Consider the SQL query

```
select * from employee where age > 30
```

Let N = the number of tuples retrieved with this query. For what values of N would a sequential table scan of the *employee* relation be cheaper than processing the query using the index? **Explain** your answer.

b) Consider the SQL query

```
select *
from employee, department
where employee.dept_id = department.dept_id
```

What evaluation plan would a query optimizer likely choose to get the least estimated cost?