CSE 444 Practice Problems

Query Optimization

1. **Query Optimization**

   Given the following SQL query:

```
Student (sid, name, age, address)
Book(bid, title, author)
Checkout(sid, bid, date)

SELECT S.name
FROM Student S, Book B, Checkout C
WHERE S.sid = C.sid
AND B.bid = C.bid
AND B.author = 'Olden Fames'
AND S.age > 12
AND S.age < 20
```

   And assuming:

   - There are 10,000 `Student` records stored on 1,000 pages.
   - There are 50,000 `Book` records stored on 5,000 pages.
   - There are 300,000 `Checkout` records stored on 15,000 pages.
   - There are 500 different authors.
   - Student ages range from 7 to 24.

(a) Show a physical query plan for this query, assuming there are no indexes and data is not sorted on any attribute.

(b) Compute the cost of this query plan and the cardinality of the result.

(c) Suggest two indexes and an alternate query plan for this query.

(d) Compute the cost of your new plan.

(e) Explain the steps that the Selinger query optimizer would take to optimize this query.

2. **Query Optimization**

Consider the following SQL query that finds all applicants who want to major in CSE, live in Seattle, and go to a school ranked better than 10 (i.e., rank < 10).

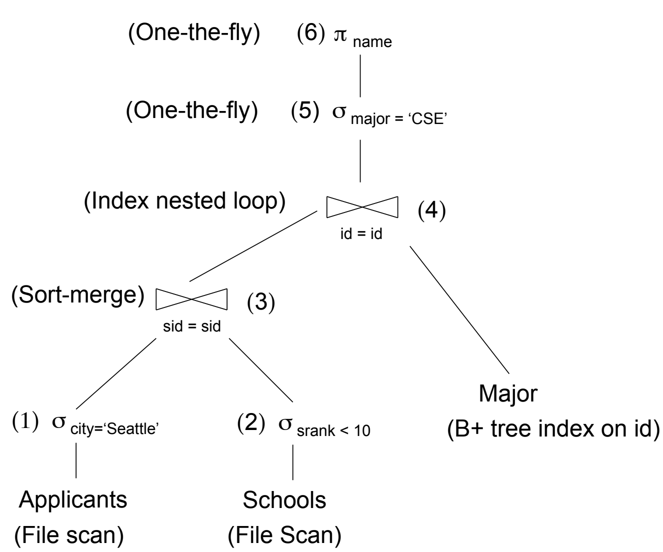
Relation	Cardinality	Number of pages	Primary key
`Applicants (id, name, city, sid)`	2,000	100	`id`
`Schools (sid, sname, srank)`	100	10	`sid`
`Major (id, major)`	3,000	200	`(id,major)`

```
SELECT A.name
FROM   Applicants A, Schools S, Major M
WHERE  A.sid = S.sid AND A.id = M.id
AND    A.city = 'Seattle' AND S.rank < 10 AND M.major = 'CSE'
```

And assuming:

- Each school has a *unique* rank number (`srank` value) between 1 and 100.
- There are 20 different cities.
- `Applicants.sid` is a foreign key that references `Schools.sid`.
- `Major.id` is a foreign key that references `Applicants.id`.
- There is an unclustered, secondary B+ tree index on `Major.id` and all index pages are in memory.

(a) What is the cost of the query plan below? Count only the number of page I/Os.



(b) The Selinger optimizer uses a dynamic programming algorithm coupled with a set of heuristics to enumerate query plans and limit its search space. Draw two query plans for the above query that the Selinger optimizer would NOT consider. For each query plan, indicate why it would not be considered.

3. **Query Optimization**

Consider the schema R(a,b), S(b,c), T(b,d), U(b,e).

(a) For the following SQL query, give two equivalent logical plans in relational algebra such that one is likely to be more efficient than the other. Indicate which one is likely to be more efficient. Explain.

```
SELECT   R.a
FROM     R, S
WHERE    R.b = S.b
         AND S.c = 3
```

(b) Recall that a *left-deep* plan is typically favored by optimizers. Write a left-deep plan for the following SQL query. You may either draw the plan as a tree or give the relational algebra expression. If you use relational algebra, be sure to use parentheses to indicate the order that the joins should be performed.

```
SELECT   *
FROM     R, S, T, U
WHERE    R.b = S.b
         AND S.b = T.b
         AND T.b = U.b
```

(c) Physical plans. Assume that all tables are clustered on the attribute b, and there are no secondary indexes. All tables are large. Do not assume that any of the relations fit in memory.

For the left-deep plan you gave in (b), suggest an efficient physical plan.

Specify the physical join operators used (hash, nested loop, sortmerge, etc.) and the access methods used to read the tables (sequential scan, index, etc.). Explain why your plan is efficient.

For operations where it matters, be sure to include the details — for instance, for a hash join, which relation would be stored in the hash tables; for a loop join, which relation would be the inner or outer loop. You should specify how the topmost join reads the result of the lower one.

(d) For the physical plan you wrote for (c), give the estimated cost in terms of $B(...)$, $V(...)$, and $T(...)$. Explain each term in your expression.