Write all of your answers directly on this paper. Be sure to **clearly indicate** your final answer for each question. Also, be sure to state any assumptions that you are making in your answers.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Maximum</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. True/False Statements</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. Relational Algebra and SQL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. B+ Tree Indexes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. Query Evaluation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOTAL</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Question 1 [5 parts] True/False Statements**

State if the following statements are TRUE or FALSE. Please write your answers in the boxes below. No explanation is needed.

(a) Consider relational operators $\sigma$ and $\cup$. For any union compatible relations $R_1$ and $R_2$ and any predicate $p$, $\sigma_p(R_1 \cup R_2) \equiv \sigma_p(R_1) \cup \sigma_p(R_2)$ (‘$\equiv$’ means that the left expression and the right expression always return the same answer).

(b) Consider relational operators $\pi$ and $\setminus$. For any union compatible relations $R_1$ and $R_2$ and any attribute set $s$, $\pi_s(R_1 \setminus R_2) \equiv \pi_s(R_1) \setminus \pi_s(R_2)$.

(c) On average, random accesses to N pages are faster than a sequential scan of N pages because random I/Os tend to access different cylinders and therefore cause less contention.

(d) It is a good idea to create as many indexes as possible to expedite query processing because there is no advantage of having many indexes.

(e) Using an unclustered B+Tree index on age to retrieve records in sorted order of age is faster than performing a two-pass external merge-sort (assuming that we have enough memory to do so).
Question 2 [3 parts]: Relational Algebra and SQL

Consider the following relational schema.

- **Emp(eid, ename, age, salary)**
- **Works(eid, did)**
- **Dept(did, dname, budget, managerid)**

An employee can work in more than one department. And `managerid` in Dept is a foreign key referencing `eid` in Emp.

(a) Print the name and age of each employee who works in both the Hardware department and the Software department. Write an SQL statement for this query.

(b) Write an expression in Relational Algebra for the query in Part (a).

(c) For each employee who manages some department, print his name and the sum of the budgets of all the departments that he manages. Write an SQL statement for this query.
Question 3 [1 part]: B+ Tree Indexes

(a) Create a B+ tree where each node can hold at most 3 keys and 4 pointers when the following keys are inserted in the following order:
1, 10, 2, 11, 3, 4, 8, 5, 7, 6

Show the final tree below.
Question 4 [2 parts]: Query Evaluation

(a) Evaluation of Selection

Consider the following schema:

Employees(eid: integer, ename: string, sal: integer, title: string, age: integer)

Suppose that the following indexes, all using Alternative (2) for data entries, exist:

- An unclustered B+ tree index on sal,
- A clustered B+ tree index on <age, sal>.

The Employees relation contains 10,000 pages and 200,000 data records. Each Employees record is 100 bytes long and each index data entry is 20 bytes long.

Consider the following selection condition

\[ sal > 200 \land \text{title} = 'VP' \]

Assume that the reduction factor (RF) for an equality predicate is 1% and that for an inequality predicate is 10%. Compute the cost of the most selective access method (among the file scan and available index scans) for evaluating this selection condition.
Answer to Question 1:

(a) True.

(b) False. Suppose E1 and E2 have the same schema (name, gpa), and E1 has one tuple (‘Sam’, 4.0) and E2 has one tuple (‘Sam’, 3.0). \( \pi_{\text{name}}(E1 - E2) \) returns one tuple (‘Sam’), but \( \pi_{\text{name}}(E1) - \pi_{\text{name}}(E2) \) returns no result.

(c) False. N random I/Os have repeated seek time and rotational delay, hence slower than a sequential scan of N pages.

(d) False. Indexes consume a lot space and can slow down insertions.

(e) False. An unclustered index can lead to 1 random I/O for each record, in general worse than two-pass external sort. For details, see slides 22-25 in Lec 11.

Answer to Question 2:

(a) SQL statement

```sql
SELECT   E.ename, E.age
FROM     Emp E, Works W1, Works W2, Dept D1, Dept D2
WHERE    E.eid = W1.eid AND W1.did = D1.did AND D1.dname = 'Hardware' AND
         E.eid = W2.eid AND W2.did = D2.did AND D2.dname = 'Software'
```

(b) Expression in Relational Algebra

\[
\rho(R1, \pi_{\text{eid}}(\sigma_{\text{dname} = 'Hardware'}(\text{Dept} \bowtie \bowtie \text{Works})))
\]

\[
\rho(R2, \pi_{\text{eid}}(\sigma_{\text{dname} = 'Software'}(\text{Dept} \bowtie \bowtie \text{Works})))
\]

\[
\pi_{\text{ename, age}}((R1 \cap R2) \bowtie \bowtie \text{Emp})
\]

(c) SQL statement

```sql
SELECT   E.ename, sum(D.budget)
FROM     Dept D, Emp E
WHERE    D.managerid = E.eid
GROUP BY D.managerid
```

Answer to Question 3:

(a) The final B+ tree:

Root: 7

Level 1: (3, 5) (10)

Level 2: (1, 2) (3,4) (5,6) (7,8) (10,11)
Answer to Question 4:

(a) sal > 200 ∧ title = 'VP'

Option 1: filesan with a  cost of 10,000 pages.

Option 2: Unclustered B+tree.
- 2,000 (10,000/(100/20)) leaf pages * 10% matches = 200 pages.
- 20 * 10,000 * 10% matches = 20,000 matches. One I/O per match, then 20,000 I/Os. If use
  the refinement of sorting, still 10,000 pages.

Option 3: Clustered B+tree on <age, sal> does not help.

In this case, the file scan is the best available method to use, with a cost of 10,000.