ONID: ______________________________

CS540 Database Management Systems

Winter 2021

School of Electrical Engineering & Computer Science
Oregon State University

Midterm Examination

Time Limit: 120 minutes

- Print your name and ONID below. In addition, print your ONID in the upper right corner of every page.

  Name and ONID: ______________________________

- The exam is open book and notes.
- Any form of cheating on the examination will result in a zero grade.
- The submissions should be in pdf format.
- Please make your answers clear and succinct; you will lose credit for verbose, convoluted, or confusing answers. *Simplicity does count!*

Question:	1	2	3	4	Total
Points:	4	5	6	5	20
Score:					

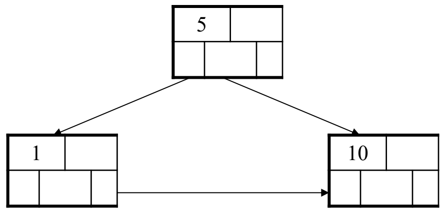

Figure 1: The B+tree for question 1.

1. **B+tree Index**

   Consider the B+tree with degree, i.e., order, $d = 1$ shown in Figure 1. The picture shows both keys and pointers in each node. We do *not* show the links to the data blocks on the underlying relation from the leaf nodes of the B+tree.

   (a) (4 points) Show the B+tree produced after inserting search keys with values 25 and 30. You should first insert 25 and then 30.

2. **Indexing**

   (a) (5 points) Consider the relation Employee(id, name, salary) and the following SQL query.

   ```
   SELECT name
   FROM Employee
   WHERE salary > 1000  and salary < 10000
   ```

   What index will improve the running time of this query more than any other index? You must explain the attribute(s) of the index, its type (B+tree or hash), and whether it is clustered. You should justify your answer to each part.

3. **Query Processing: Join Algorithms**
Consider relations *Student(ID, Name)* and *Enrollment(CourseID, SID)*. The size of relation *Student* is 40,000 blocks and the size of *Enrollment* is 400 blocks. Assume that the query processor has to choose between the internal memory (in-memory) join algorithms, block-based nested loop algorithm, and the sort-merge join with two-pass multi-way merge-sort algorithm for sorting to perform the join of $Student \bowtie_{ID=SID} Enrollment$.

   (a) (6 points) Given that we have only 4 blocks available in main memory, i.e., $M = 4$, which one of the aforementioned algorithms is the fastest one to perform the join? You should also provide the cost, i.e., number of I/O accesses, of the join using your proposed algorithm.

4. **Query Processing**
The *left anti-join* operator is a variant of the relational join. The basic idea is to output only those tuples in the left relation that do *not* join with any tuple in the right relation. For example, consider two relations R(A, B) and S(B, C). Assume that R has the following tuples: (1, 10), (1, 20), (2, 30), (2, 40) and S has the following tuples: (10, 75), (10, 85), (30, 95). The left anti-join of R and S on attribute B will produce the following tuples: (1, 20) and (2, 40).

   Given the above description of left anti-joins, answer the following question.

   (a) (5 points) How can you adapt the sort-merge join algorithm to process left anti-joins? Describe only essential modifications to the algorithm. You may use the two-pass multi-way merge-sort algorithm for sorting.