Midterm II Review

A.1) (16 Points) A Memory buffer has three pages. Fill each of the three slots in the table below to show the contents of the memory buffer after each read for LRU and MRU policies. Check the corresponding checkbox at each step if a page read requires retrieving a page from disk. What is the total number of disk access for each of LRU and MRU policies, which policy is better for this scenario?

	LRU			Disk?	MRU			Disk?
1. Read P1				☐				☐
2. Read P2				☐				☐
3. Read P3				☐				☐
4. Read P4				☐				☐
5. Read P1				☐				☐
6. Read P2				☐				☐
7. Read P3				☐				☐
8. Read P4				☐				☐

A.2) (8 Points) Draw a page layout that includes variable length records. Your layout should show the page header, the empty space, and few variable length records.

A.3) (6 points) What is meant by an index-only plan? Show how we can employ an index only plan for the following query:

```
SELECT AVG (E.rating)
FROM  Employee E
WHERE  E.age > 50
```

A.4) (10 Points) Draw a B+-tree with the following properties:

- Fanout: 4 (Each node has four children)
- Root node has only one key
- Non-leaf nodes are not full
- The tree includes 12 records, with keys from 1 to 12

B.1) (30 Points) Consider the relations Sailors (Sid, name, rating, age), Reserves (Sid, Bid, Price) (keys are underlined). One disk page can accommodate 30 tuples from the Sailors relations or 60 tuples from the Reserves relation. Given that there are 150,000 sailors and 600,000 reservations. What is the estimated I/O cost for the following scenarios?

Use alternative 2. Assume one I/O is needed to retrieve a hash data entry and three I/Os are needed to retrieve a B+-tree data entry. Ignore the cost of writing out the results.

1) (5 points) Index Nested-loop join: Clustered B+-tree on Sid of Sailors (Sailors is inner)

2) (5 points) Index Nested-loop join: Unclustered B+-tree on Sid of Sailors (Sailors is inner)

B.1) (30 Points) Consider the relations Sailors (Sid, name, rating, age), Reserves (Sid, Bid, Price) (keys are underlined). One disk page can accommodate 30 tuples from the Sailors relations or 60 tuples from the Reserves relation. Given that there are 150,000 sailors and 600,000 reservations. What is the estimated I/O cost for the following scenarios?

Use alternative 2. Assume one I/O is needed to retrieve a hash data entry and three I/Os are needed to retrieve a B+-tree data entry. Ignore the cost of writing out the results.

3) (5 points) Index Nested-loop join: Clustered Hash-index on Sid of Reserves (Reserves is inner)

4) (5 points) Index Nested-loop join: Unclustered Hash-index on Sid of Reserves (Reserves is inner)

B.1) (30 Points) Consider the relations Sailors (Sid, name, rating, age), Reserves (Sid, Bid, Price) (keys are underlined). One disk page can accommodate 30 tuples from the Sailors relations or 60 tuples from the Reserves relation. Given that there are 150,000 sailors and 600,000 reservations. What is the estimated I/O cost for the following scenarios?

Use alternative 2. Assume one I/O is needed to retrieve a hash data entry and three I/Os are needed to retrieve a B+-tree data entry. Ignore the cost of writing out the results.

5) (5 points) Block Nested-loop join: you have 52 buffer pages available (Reserves is the outer)

6) (5 points) Hash join:

B.2) (15 points) You are the chief architect of the LateDB database system, and you are really running up against the product shipment deadline. You just figured out that you have not yet implemented any index structure or a join algorithm.

a) (5 points) You only have time to implement one join algorithm. Will you choose simple-nested-loop join, page-nested loop, block-nested-loop join, index-nested-loop join, sort-merge join, or hash join? Justify your answer.

b) (5 points) You only have time to build one type of index structure. Will you choose B+-trees or hashing? Justify your answer.

c) (5 points) You seem to have panicked too soon, and things are not as bad as they seem. So, you have time to implement one other join algorithm (in addition to the join algorithm you chose above). Which one will you choose and why? Justify your answer.

B.3) (15 points) Consider a relation R(A,B,C). Assume that the only index on relation R is an unclustered B+-tree on attribute A. The B+-tree has height 3 and are all in memory. B+-tree data entries are also in memory. R has 500,000 tuples in 10,000 pages. The values of attribute A are uniformly distributed from 1 to 1,000. Now consider the following query:

```
SELECT A, B FROM R WHERE A > val
```

Compute the minimum value of val such that the most efficient way of executing the above query is to use the B+-tree index. Assume that the cost metric is the number of page I/Os. In your analysis, you can assume the worst-case scenario in computing the cost of a range scan using an unclustered B+-tree index.