1. **[10 points] Relational Algebra**

   (a) [5 points] Name the two derived operations of relational algebra.

   (b) [5 points] Express the "intersection of two sets" operation in terms of the five basic operations of relational algebra.

2. **[40 points] Implementation of Operators**

   (a) [10 points] Consider the relational algebra operator "union". Suppose you need to union two tables R and S, which are two files on disk that are too large to fit into the buffer. This is a "set union", not a "bag union", operation. So each distinct tuple in R or S will appear only once in the output.
   Assume that R has M pages, S has N pages, the buffer has B pages, and $M<B(B-1)$ and $N<B(B-1)$. Describe how you can perform this "union" operator so that the total cost is at most 5M+5N. As usual, do not count the cost of writing the output to disk.

   (b) [10 points] The hash join algorithm uses TWO different hash functions. Explain where these two hash functions are used and why they have to be different.

   (c) [10 points] Consider doing a hash join for two relations R and S that are on disk. Suppose R has M pages and S has N pages. Assume further that $M<(B-2)(B-2)$, where B is the number of pages in the buffer. In the ideal case what is the cost of this hash join operation? Explain your answer. As usual, do not count the cost of writing the output of the join to disk.

   (d) [10 points] Consider a relation R = Enroll(sid, cid, grade), where "sid" is the student ID, "cid" is the course ID, and "grade" is the grade of the student in the course. R lists which student enrolls in which course and receives which grade.

   Consider a relation S = Students(sid, sname), where "sid" is the student ID and "sname" is the student name. Assume that "sid" is the key for relation S, and that each student enrolls in at most 3 courses. Assume also that R has 20 pages and S has 10 pages, that each page can hold 100 tuples, and that both R and S are on disk.

   Suppose we want to join R and S on "sid", using a memory buffer of 6 pages. What is the cost of the sort merge join? Explain your answer. As usual, do not count the cost of writing the join output to disk.

3. **[15 points] Query Optimization**

(a) [10 points] Consider the following SQL query Q:

```
SELECT A.name
FROM A, B, C
WHERE A.id = B.id and B.id = C.id and B.price > 5 and C.age >
30.
```

Briefly describe at least 12 physical query plans that the query optimizer can consider for the above query. You do not have to explicitly list all 12 plans (unless you want to). Clearly describing these 12 plans in words is sufficient.

(b) [5 points] Following up on Part a, suppose there is an index on B.id, so we consider joining table A with B using an indexed nested loop join. Can we push the selection operation (B.price > 5) to be below this join operation (in the physical query plan)? If yes, why? If not, why not?

4. **[15 points] Transaction Management**

(a) [5 points] What is a transaction in the context of relational databases?

(b) [5 points] Consider a transaction T that has two SQL queries. The first query subtracts $20 from the checking account, and the second query adds $20 to the saving account (so this transaction moves $20 from checking to saving). Describe a transaction V that can interfere with this transaction T, unless we enforce ACID properties. You should clearly describe the SQL queries in transaction V and discuss why V may interfere with T.

(c) [5 points] Briefly explain how each of the four ACID requirements (A, C, I, D) is ensured.

5. **[10 points] Recovery**

(a) [5 points] Briefly explain the idea of non-quiescent checkpointing. Does the database have to freeze (that is, stop accepting new transactions) during non-quiescent checkpointing?

(b) [5 points] In undo logging, when recovering from a crash, what kinds of transactions do we have to undo, and why?

6. **[10 points] Recovery**

Suppose after crash, the log is as below, and this is a redo log:

```
<start t1>
<t1, x, 5>
<start t4>
<t4, y, 3>
<commit t1>
<commit t4>
<start t5>
<t5, x, 2>
<start t6>
<start ckpt t5,t6>
<start t7>
<end ckpt>
<t7, z, 4>
<commit t7>
```

(a) [5 points] In the above log, after the system has written the record `<start ckpt t5,t6>` to the log, it will start flushing to disk all changes made by certain transactions. Which transactions are these?

(b) [5 points] Describe what actions we need to take to recover, using the above redo log. Once recovery is complete, what are the values for x, y, and z?