1. Let \( r \) and \( s \) be relational schemas, both having the schema \( R \). Then, \( r \cup s = \{ t \mid t \in r \lor t \in s \} \), \( r \cap s = \{ t \mid t \in r \land t \in s \} \), and \( r - s = \{ t \mid t \in r \land t \notin s \} \). Consider the following query rewriting rules

\[
\pi_X((e_1)\theta(e_2)) = \pi_X(e_1)\theta\pi_X(e_2)
\]

\[
\sigma_f((e_1)\theta(e_2)) = \sigma_f(e_1)\theta\sigma_f(e_2)
\]

where \( \theta \) can be union (\( \cup \)), intersection (\( \cap \)), or set difference (\( - \)). There are six rules here. Two of these six rules do not hold. Find which two do not hold and give examples to show that they do not hold.

2. Let \( r_1(ABC) \), \( r_2(CDE) \), and \( r_3(ACF) \), and assume that the domain for all attributes is the set of integers. Rewrite the following queries so that they are optimal in the sense that all \( \pi \) and \( \sigma \) operations are done as early as possible. Resolve any conflicts between doing \( \pi \) or \( \sigma \) earlier, by doing \( \sigma \) earlier. In addition, for (b), show your work and justify each step with one of the equivalence laws in QueryRewriting.ppt or in the exercise above. In your justification, use the law numbers in QueryRewriting.ppt (the supplement in the schedule for query rewriting), and let all the valid laws in Exercise (1) above be law number 10.

   (a) \( \sigma_{B=4\land C=6}\land F\geq7}(r_1 \bowtie r_2 \bowtie r_3) \)
   (b) \( \pi_A\sigma_{C\leq9}(\pi_{AC}r_1 - \pi_{AC}r_3) \)
   (c) \( \sigma_{A\leq B}\pi_{ABD}(\pi_{AB}r_1 \bowtie \pi_{DE}r_2) \)
   (d) \( \pi_{AE}(r_1 \bowtie r_2) \)

3. Consider relations \( r(ABC) \) and \( s(CDE) \) and suppose that \( r \) has 10,000 tuples stored 20 per block and that \( s \) has 30,000 tuples stored 30 per block. (Here—and usually in this context—“access” means that the disk head moves. The ”estimates” are to be for the worst case, e.g., ignoring caches and lucky block placement.)

   (a) Estimate the number of block accesses required to execute \( \sigma_{A=1}r \) assuming that \( r \) is stored as a sequential file that is not contiguous and not sorted.
   (b) Estimate the number of block accesses required to execute \( \sigma_{A=1}r \) assuming that \( r \) is indexed on \( A \), which is the primary key, and the index is in memory. (Assume a sparse index.)
   (c) Estimate the number of block-read accesses required to execute \( r \bowtie s \) assuming that both \( r \) and \( s \) are stored as sequential files that are not contiguous and not sorted. Assume that the algorithm does the reads for processing the join as follows: it reads the first block of \( r \) and then all of \( s \), it then reads the second block of \( r \) and again all of \( s \), repeating this pattern until all blocks of \( r \) have been read and processed. Assume that blocks of \( s \) are not retained in memory between iterations.
   (d) Estimate the number of block-read accesses required to execute \( r \bowtie s \) assuming that both \( r \) and \( s \) are stored as sequential files that are not contiguous and not sorted. Assume that the algorithm does the reads for processing the join as follows: it reads the first block of \( s \) and then all of \( r \), it then reads the second block of \( s \) and again all of \( r \), repeating this pattern until all blocks of \( s \) have been read and processed. Assume that blocks of \( r \) are not retained in memory between iterations.
(e) Why are your answers for (c) and (d) different? Can you make a general rule for using this block-oriented join algorithm for doing a join with different size files?

The following database schema is for Exercises 4 and 5.

supplier(SName, City, Country) key: Sname
item(ItemNumber, ItemName, Color, Weight) key: ItemNumber
shipment(SName, ItemNumber, Quantity, ShipDate) key: {Sname, ItemNumber}

4. To see the process the database system uses to answer queries that use views, do the following.

(a) Write a SQL statement to create a view of Canadian suppliers whose shipments are in quantities greater than one hundred. The attributes of the view are to be SName, ItemNumber, Quantity, ShipDate, and City. Its name is to be LargeCanadianShipment.

(b) Using your view, write a SQL query to list SName, ItemNumber, and Quantity of the large shipments whose shipping date is 10 May.

(c) Express your SQL query in relational algebra as follows. (1) Write the view described in (a) as a relational algebra expression by itself; this expression has the name LargeCanadianShipment. (2) Write a relational algebra expression for (b) using the name LargeCanadianShipment as if it were a base table. (3) Write the full query without assuming that LargeCanadianShipment is a base table by substituting the relational algebra expression for (1) in place of LargeCanadianShipment in your relational algebra expression for (2), enclosed in parentheses. (4) Optimize this query using the rules for query optimization.

5. Consider the following view update of the view described above in Exercise 4: Remove large canadian shipments in the view that are from Toronto. Using the view, we could write

```
DELETE FROM LargeCanadianShipment
WHERE City = 'Toronto'
```

but the view is virtual. Show that this request is ambiguous by writing, in SQL, two different ways the view update could be carried out on the base relations so that when the view is materialized there would be no LargeCanadianShipment tuples with Toronto as the city. One of the two SQL statements must alter the supplier table; the other must alter the shipment table.