1. [13 points] Joe764 is back. He now wants to propose “precise R-trees”, in which the entries are not (bounding box, ptr) but rather (precise polygon, ptr). That is, instead of just using bounding boxes, he wants to store detailed polygons that precisely describe the boundary of the objects in the subtree (if the entry is in a non-leaf node) or the object itself (if the entry is in a leaf.) Is this a good idea? Why or why not?
2. [13 points] *(This is a difficult question because it is very open-ended. You may want to leave it for the end of the exam.*) Consider the love-hate hint buffer management algorithm, in which a page is marked as “loved” (if the accessing code thinks it is likely to re-reference it) or “hated” (if the accessing code thinks it will not re-reference it.) The replacement policy never replaces a loved page if there is still a hated page in the buffer pool; it uses LRU among loved pages and hated pages. What is a query locality set model that is easy to approximate with love-hate hints? What is one that is hard?
3. [12 points] In the Stonebraker and Hellerstein data model paper, the authors state that schema-later is a niche market. What do they mean by “schema-later”?
4. [13 points] Based on your reading of the Map-Reduce vs. Parallel DBMS paper, why might someone choose to use Map-Reduce for a large data processing task? Why might someone choose to use a parallel DBMS for a large data processing task?
5. [12 points] Why is adding schema inheritance to the relational data model likely to be easier than adding ADTs and their methods to a relational DBMS?
6. [12 points] Why might one expect cache utilization to be better with a column store than a traditional row store? For what kind of query do you expect this to be important?
7. [12 points] Describe what the following XQuery computes:

```xml
<big_publishers>
  FOR $p IN distinct(document("bib.xml")//publisher)
  LET $b := document("bib.xml")/book[publisher = $p]
  WHERE count($b) > 100
  RETURN $p
</big_publishers>
```
8. [13 points] Consider a bitmap index built on column B of a table \( R(A, B, C) \). Suppose \( R \) has 1,000,000 rows in it, that the minimal size of an \( R \) tuple is 100 bytes, and that data pages can hold 10000 bytes of user data. Furthermore, suppose that \( R \) currently is stored on 13,000 pages (that is, tuples of \( R \) appear on 13,000 pages) and \( R.B \) has 200 distinct values. Finally, also assume that the method described in class for mapping bit position to record in page is being used.

   a. How many bitmaps will be in the index?

   b. How many bits will be in each bitmap?

   c. What is the total number of “1” bits in all of the bitmaps?