Question 1: Trees (Part 2) . . . . . [280 points]

(i) **[10 points] More B+Trees:**
Distinguish between `std::multimap` and `std::map`.

(ii) **[10 points] More B+Trees:**
Distinguish between these two techniques for handling duplicate keys: (1) append record ID and (2) overflow leaf nodes.

(iii) **[10 points] More B+Trees:**
What is a partitioned B+tree? Justify its utility with an example.

(iv) **[10 points] More B+Trees:**
List a benefit and a limitation of a partitioned B+tree.

(v) **[10 points] More B+Trees:**
List a benefit and a limitation of a prefix B+tree.

(vi) **[10 points] More B+Trees:**
Explain why prefix compression is attractive in a B+tree.

(vii) **[10 points] Additional Index Magic:**
Distinguish between implicit and explicit indexes.

(viii) **[10 points] Additional Index Magic:**
Define data integrity. Is it related to the ACID properties?

(ix) **[10 points] Additional Index Magic:**
Are implicit indexes automatically created for enforcing referential integrity constraints? Justify your answer.

(x) **[10 points] Additional Index Magic:**
Define a partial index. Illustrate its utility with an example.

(xi) **[10 points] Additional Index Magic:**
Define a covering index. Illustrate its utility with an example.

(xii) **[10 points] Additional Index Magic:**
Define an index with include columns. Illustrate its utility with an example.

(xiii) **[10 points] Additional Index Magic:**
Distinguish between a covering index and an index with include columns.

(xiv) **[10 points] Additional Index Magic:**
Define a functional index. Illustrate its utility with an example.

(xv) **[10 points] Tries:**
Define a trie. Illustrate its utility with an example.

(xvi) **[10 points] Tries:**
Distinguish between a trie and a B+tree.

(xvii) **[10 points] Tries:**
What is the time complexity of operations in a trie? Is it dependent or independent of the length of the key? Is it dependent or independent of the number of keys?

Question 1 continues. . .

(xviii) **[10 points] Tries:**
Does a trie require rebalancing operations?

(xix) **[10 points] Tries:**
Define the span of a trie level. How does it affect the fan-out of each node? How does it affect the physical heightof the tree?

(xx) **[10 points] Radix Tree:**
Distinguish between a trie and a Radix tree.

(xxi) **[10 points] Radix Tree:**
Can a radix tree return false positives? Justify your answer.

(xxii) **[10 points] Radix Tree:**
Explain how `INSERT` operation works in a Radix Tree.

(xxiii) **[10 points] Radix Tree:**
Explain the necessity for binary comparable keys in a Radix Tree.

(xxiv) **[10 points] Radix Tree:**
What will happen if we do **not** flip the byte order for an unsigned integer key in a radix tree?

(xxv) **[10 points] Radix Tree:**
Distinguish between 1-bit Span and 8-bit Span Radix Trees with an example.

(xxvi) **[10 points] Inverted Index:**
Define an inverted index. When is it used?

(xxvii) **[10 points] Inverted Index:**
List three types of quries supported by an inverted index.

(xxviii) **[10 points] Inverted Index:**
How would you construct an inverted index that supports phrase searches with atmost three words?