Question 1: Larger-than-Memory Databases . . . . . [270 points]

(i) **[10 points] Debugging:**
List two techniques you found to be helpful in debugging your buffer manager.

(ii) **[10 points] Device Price:**
What is approximate cost of: (1) 1 GB DRAM, (2) 1 GB NVM, (3) 1 GB SSD, and (4) 1 GB HDD?

(iii) **[10 points] System Price:**
Which one is the most expensive component in a regular Macbook: (1) compute, (2) memory, or (3) disk?

(iv) **[10 points] Expenses:**
Distinguish between capital and operational expenses. Which expense dominates the overall cost in a datacenter?

(v) **[10 points] Larger-than-Memory DBMS:**
Define a larger-than-memory DBMS. Distinguish it from a disk-oriented DBMS.

(vi) **[10 points] Larger-than-Memory DBMS:**
Why is in-memory access tuple-oriented? Why is disk access block-oriented?

(vii) **[10 points] Anti-Caching:**
Distinguish between caching and anti-caching.

(viii) **[10 points] Cold Data:**
What is cold data? How is it identified?

(ix) **[10 points] Eviction Timing:**
When is it useful to evict based on a threshold instead of only on-demand?

(x) **[10 points] Evicted Tuple Metadata:**
Define a tuple tombstone. What does it contain?

(xi) **[10 points] Bloom Filter:**
Define a Bloom Filter. Why is it approximate?

(xii) **[10 points] Bloom Filter:**
Why can Bloom Filter return false positives?

(xiii) **[10 points] Evicted Tuple Metadata:**
List the steps involved in locating the evicted tuple using an in-memory Bloom Filter and an on-disk index.

(xiv) **[10 points] Data Retrieval Granularity:**
Make the case for merging only the tuples accessed by a query in the retrieved disk block as opposed to all the tuples.

(xv) **[10 points] Retrieval Policy:**
When is synchronous retrieval a better choice than abort-and-restart retrieval?

(xvi) **[10 points] Retrieval Policy:**
When is abort-and-restart retrieval a better choice than synchronous retrieval?

Question 1 continues…

(xvii) **[10 points] Memory Mapping:**
How is `mlock` used for larger-than-memory data management?

(xviii) **[10 points] Pointer Swizzling:**
Define pointer swizzling.

(xix) **[10 points] LeanStore:**
How can the pointer be used to determine the location of a block? Why is this technique more scalable compared to a page table?

(xx) **[10 points] LeanStore:**
Justify the randomized block eviction policy.

(xxi) **[10 points] LeanStore:**
How does a block hierarchy preclude the need for a centralized page table?

(xxii) **[10 points] LeanStore:**
Why can the DBMS evict a page only when its children are also evicted?

(xxiii) **[10 points] Umbra:**
Justify the need for a variable-sized buffer pool.

(xxiv) **[10 points] Umbra:**
Distinguish between a fixed-size and variable-sized buffer pool.

(xxv) **[30 points] MemSQL:**
Define the mutable delta store. What storage model is used in this component?
Define the immutable main store. What storage model is used in this component?
How does this architecture support HTAP workloads?