Q 1. [7pts] For the following questions, clearly circle True or False.

1. Sequential data access is typically orders of magnitude faster than random data access when reading data from non-volatile memory.
   False

2. When the pin count of a buffer frame is decremented, the page in that frame is said to be “unpinned.”
   False

3. In the unpacked layout for fixed-length records, deleting a record will alter the record IDs of other records on the same page.
   False

4. In the delimiter-based record format, we need to scan the record from the start even to retrieve a single field in the middle.
   True

5. It is possible to have multiple clustered B+ tree indexes on the same table.
   True

6. Redistribution of index keys among siblings following the overflow of a non-leaf node is the first preference for reorganizing a B+ tree index after an insert operation.
   False

7. A key benefit of a learned index over a B+ tree index reported in Kraska et al. (SIGMOD 2018) is that the former can be much smaller in size.
   True

Q 2. [11pts] Consider the following extendible hash index with 4 slots per bucket.

1. [5pts] What is the global depth after the following sequence of operations: delete 32*, insert 32*, insert 125*, insert 17*, and insert 36*?
(a) 1  (b) 2  (c) 3  (d) 4  (e) 5  (f) 6

**ANSWER:** (c) 3. Bucket B will split, incrementing global depth.

2. **[2pts]** After the above operations, what is the largest number of directory pointers pointing to the same bucket?

(a) 1  (b) 2  (c) 3  (d) 4  (e) 5  (f) 6

**ANSWER:** (d) 4. Bucket A and C would have merged, with local depth 1. With global depth at 3, this means 4 pointers point to this merged bucket.

3. **[4pts]** After the above operations, what is the minimum number of delete operations needed now for the global depth to decrease?

(a) 1  (b) 2  (c) 3  (d) 4  (e) 5  (f) 6

**ANSWER:** (b) 2. We just need the split pages of Bucket B to merge again.

**Q 3. [9pts]** Suppose we are sorting a relation with 1 million pages and we have 100 buffer pages for the external merge sort (EMS). A "pass" over the relation is defined as one read and write of the whole file. In all of the following, you have to include both the sort and merge phases. **Clearly circle** the correct answer for each of the following questions.
1. [3pts] How many passes will a multi-way EMS perform, assuming we use replace-ment sort for internal sorting?

(a) 2   (b) 3   (c) 4   (d) 5   (e) 6   (f) 7

ANSWER: (b) 3

2. [3pts] How many passes will a multi-way EMS perform, assuming we do not use re-placement sort for internal sorting but use blocked I/O with block sizes of 10 pages?

(a) 2   (b) 3   (c) 4   (d) 5   (e) 6   (f) 7

ANSWER: (e) 6

3. [3pts] How many passes will a multi-way EMS perform, assuming we use replace-ment sort for internal sorting along and double buffering but no blocked I/O?

(a) 2   (b) 3   (c) 4   (d) 5   (e) 6   (f) 7

ANSWER: (c) 4

Q 4. [12pts] We are given two tables $R(X, Y)$ and $S(X, Y)$ with the same schema, i.e., they are union-compatible. Assume attributes $X$ and $Y$ are attributes of the same size. The table sizes of $R$ and $S$ happen to be $1.8B$ and $6B$ pages respectively, where $B$ is the number of buffer pages given (in the millions). Assume there are no skews in any attribute distributions. Hash table fudge factor is 1.4.

Given all the above, what is the I/O cost for executing each of the following queries in terms of $B$ using the specified physical operator implementations? Exclude the cost of writing the output.

1. [3pts] $\pi_X(R)$ with sorting-based project
ANSWER: (a) 1.8B. Since X and Y are of same length, the intermediate non-deduplicated file is of size roughly 0.9B, which fits in memory for a full internal sort. So, we only need one read of R. Note that a hash-based project is more expensive, since it requires a repartitioning stage.

2. [3pts] \( \pi_Y(S) \) with hashing-based project

\[
(a) 3B \quad (b) 6B \quad (c) 9B \quad (d) 12B \quad (e) 15B
\]

ANSWER: (d) 12B. Read S and write out 5 partitions of the intermediate file with only Y, which is of size 3B. Hash table on each partition is of size \((1.4 \times 3/5)B = 0.805B\), which fits in memory. So, total I/O cost is \((6 + 3 + 3)B\). Sorting-based project will have the same cost.

3. [6pts] \( R \bowtie S \) with block-nested loop join (in-memory hash table is used on blocks)

\[
(a) 11.4B \quad (b) 19.8B \quad (c) 22.2B \quad (d) 23.4B \quad (e) 31.6B
\]

ANSWER: (b) 19.8B. The smaller R is the outer table. So, total I/O cost of BNLJ is \( N_R + N_U \cdot \lceil FN_R / (B - 2) \rceil \). Since B is in the millions, \( B / (B - 2) \approx 1 \). So, the I/O cost works out to \( 1.8B + 6B \cdot 3 = 19.8B \).

Q 5. [10pts] Are you on social media? Given the following relational database schema, translate each SQL query over this relational database into an equivalent relational algebra query. Circle the correct answer (only one is correct).

Person (ID, Name, Age)
Friends (ID1, ID2)

Person.ID is the primary key of Person. Both Friends.ID1 and Friends.ID2 are foreign keys referring to Person.ID.
1. [2pts] SELECT DISTINCT Name FROM Person WHERE Age > 20

(a) $\pi_{\text{Name}}(\sigma_{\text{Age}>20}(\text{Person}))$

(b) $\sigma_{\text{Age}>20}(\pi_{\text{Name}}(\text{Person}))$

(c) $\pi_{\text{Name,ID}}(\sigma_{\text{Age}>20}(\text{Person}))$

(d) $\sigma_{\text{Age}>20}(\pi_{\text{Name,ID}}(\text{Person}))$

(e) $\sigma_{\text{Name}}(\pi_{\text{Age}>20}(\text{Person}))$

(f) $\pi_{\text{Name}}(\pi_{\text{Age}}(\sigma_{\text{Age}>20}(\text{Person})))$

ANSWER: (a)

2. [2pts] SELECT ID1, COUNT(*) FROM Friends GROUP BY ID1

(a) $\sigma_{\text{ID1}}(\gamma_{\text{ID1,COUNT(*)}}(\text{Friends}))$

(b) $\pi_{\text{ID1}}(\gamma_{\text{COUNT(*)}}(\text{Friends}))$

(c) $\gamma_{\text{ID1,COUNT(*)}}(\sigma_{\text{ID1,ID2}}(\text{Friends}))$

(d) $\sigma_{\text{ID1,COUNT(*)}}(\text{Friends})$

(e) $\pi_{\text{ID1,COUNT(*)}}(\text{Friends})$

(f) $\gamma_{\text{ID1,COUNT(*)}}(\text{Friends})$

ANSWER: (f)

3. [6pts] SELECT DISTINCT P2.Name
FROM Person P1, Person P2, Friends F
WHERE P1.ID = F.ID1 AND P2.ID = F.ID2 AND P1.Name = "Thanos"
(a) $\pi_{\text{Name}}(\pi_{\text{ID1,ID2}}(\pi_{\text{ID}}(\sigma_{\text{Name}="Thanos"}(\text{Person}))) \bowtie_{\text{ID}=\text{ID2}} \text{Friends}) \bowtie_{\text{ID2}=\text{ID}} \text{Person}$

(b) $\pi_{\text{Name}}(\pi_{\text{ID1,ID2}}(\pi_{\text{ID}}(\sigma_{\text{Name}="Thanos"}(\text{Person}))) \bowtie_{\text{ID}=\text{ID1}} \text{Friends}) \bowtie \text{Person}$

(c) $\pi_{\text{Name}}(\pi_{\text{ID1,ID2}}(\pi_{\text{ID}}(\sigma_{\text{Name}="Thanos"}(\text{Person}))) \bowtie_{\text{ID}=\text{ID1}} \text{Friends}) \bowtie_{\text{ID2}=\text{ID}} \text{Person}$

(d) $\pi_{\text{Name}}(\pi_{\text{ID1}}(\pi_{\text{ID}}(\sigma_{\text{Name}="Thanos"}(\text{Person}))) \bowtie_{\text{ID}=\text{ID1}} \text{Friends}) \bowtie_{\text{ID2}=\text{ID}} \text{Person}$

ANSWER: (c)