Carnegie Mellon University
Computer Science Department
15-445/645 – Database Systems (Spring 2026)
Prof. Andy Pavlo and Jignesh Patel

Homework #2 (by Saransh)
Due: **Sunday Feb 8, 2026 @ 11:59pm**

IMPORTANT:
- Enter all of your answers into **Gradescope by 11:59pm on Sunday Feb 8, 2026.**
- **Plagiarism**: Homework may be discussed with other students, but all homework is to be completed **individually**.

For your information:
- Graded out of **100** points; **3** questions total
- Rough time estimate: ≈4-6 hours (1-1.5 hours for each question)

Revision : 2026/01/28 10:44

Question	Points	Score
Slotted Pages and Log-Structured	30	
Storage Models	35	
Database Compression	35	
Total:	100	

Question 1: Slotted Pages and Log-Structured . . . . [30 points]

(a) **[10 points]** Which of the following statements are true for database systems using *log-structured storage*? Select all that apply.

☐ Log structured storage can cause write amplification

☐ Log structured storage requires the DBMS to check previous records on update operations

☐ Log structured storage can cause fragmentation

☐ In leveled compaction, SSTables within a level (except Level 0) are non-overlapping on key ranges

☐ None of the above

(b) **[10 points]** Which of the following statements are true for database systems using *slotted-page storage*? Select all that apply.

☐ Slotted-page storage can cause write amplification

☐ Applications should treat RIDs in slotted-page storage systems as stable identifiers and may rely on them

☐ Slotted-page storage can increase random reads and writes

☐ Slotted-page storage does not allow variable length tuples

☐ None of the above

(c) **[10 points]** You are asked to compare *log-structured storage* to *slotted-page storage* for a new system. Ignore any indexes and overhead from metadata. Select all true statements.

☐ Both storage systems update tuples in-place to avoid extra I/O

☐ For an append-only workload, both achieve comparable performance

☐ Slotted-page requires storing schema metadata with the tuple, while log-storage does not

☐ Both designs may require background maintenance to reclaim space after numerous inserts/updates/deletes

☐ Log-structured storage requires less disk space than slotted-page storage

☐ None of the above are true

Question 2: Storage Models .... [35 points]

Consider a database with a single table G(game_id,dev_id,total_sales,player_count), where game_id is the *primary key*, and all attributes are the same fixed width. Suppose G has 10,000 tuples that fit into 400 pages. You should ignore any additional storage overhead for the table (e.g., page headers, tuple headers). Additionally, you should make the following assumptions:

- The DBMS does *not* have any additional meta-data.
- G does *not* have any indexes (including for primary key game_id).
- None of G's pages are already in memory. The DBMS can store an infinite number of pages in memory.
- Content-wise, the tuples of G will always make each query run the longest possible and do the most page accesses. Always consider what the worst-case *content/values* for each column can be.
- Order-wise, the tuples of G can be in any order. Keep this in mind when computing *minimum* versus *maximum* number of pages that the DBMS will potentially have to read. Think of all possible orderings, but always for the worst-case content of data

(a) Consider the following query:

```
SELECT MAX(player_count) FROM G
    WHERE dev_id = 15445 ;
```

i. **[5 points]** Suppose the DBMS uses the N-ary storage model (NSM). How many pages will the DBMS potentially have to read from disk to answer this query?
Be sure to keep in mind the assumption about the contents of G.
☐ 1-100 ☐ 101-200 ☐ 201-300 ☐ 301-400 ☐ > 400 ☐ Not possible to determine

ii. **[5 points]** Suppose the DBMS uses the decomposition storage model (DSM) with implicit offsets. How many pages will the DBMS potentially have to read from disk to answer this query?
Be sure to keep in mind the assumption about the contents of G.
☐ 1-100 ☐ 101-200 ☐ 201-300 ☐ 301-400 ☐ > 400 ☐ Not possible to determine

(b) Now consider the following query:

```
SELECT player_count, dev_id FROM G
    WHERE game_id = 1 OR game_id = 999
```

i. Suppose the DBMS uses the N-ary storage model (NSM).

α) **[5 points]** What is the *minimum* number of pages that the DBMS will potentially have to read from disk to answer this query?
☐ 1 ☐ 2-9 ☐ 10-100 ☐ 101-200 ☐ 201-300 ☐ 301-400
☐ > 400 ☐ Not possible to determine

β) **[5 points]** What is the *maximum* number of pages that the DBMS will potentially have to read from disk to answer this query?
☐ 1 ☐ 2-9 ☐ 10-100 ☐ 101-200 ☐ 201-300 ☐ 301-400
☐ > 400 ☐ Not possible to determine

ii. Suppose the DBMS uses the decomposition storage model (DSM) with implicit offsets.

α) **[5 points]** What is the *minimum* number of pages that the DBMS will potentially have to read from disk to answer this query?
☐ 1 ☐ 2-9 ☐ 10-100 ☐ 101-200 ☐ 201-300 ☐ 301-400
☐ > 400 ☐ Not possible to determine

β) **[5 points]** What is the *maximum* number of pages that the DBMS will potentially have to read from disk to answer this query?
☐ 1 ☐ 2-9 ☐ 10-100 ☐ 101-200 ☐ 201-300 ☐ 301-400
☐ > 400 ☐ Not possible to determine

(c) Finally consider the following query:

```
SELECT MIN(game_id) FROM G
    WHERE player_count = (SELECT MIN(player_count) FROM G);
```

Suppose the DBMS uses the decomposition storage model (DSM) with implicit offsets.

i. **[5 points]** What is the *minimum* number of pages that the DBMS will potentially have to **read from disk** to answer this query?
☐ 1 ☐ 2-9 ☐ 10-100 ☐ 101-200 ☐ 201-300 ☐ 301-400 ☐ > 400
☐ Not possible to determine

Question 3: Database Compression............................[35 points]

(a) **[5 points]** Suppose that the DBMS has a `VARCHAR` column storing the following values:

```
[Gates-Hillman Complex, Porter Hall, Doherty Hall, Wean
Hall, Hunt Library]
```

Which of the following are valid encodings (`uint32`) for this column under dictionary compression as discussed in lecture that will support both point queries and range queries? Select **all** the valid encodings.

☐ `[1,2,3,4,5]`
☐ `[2,4,1,5,3]`
☐ `[0,12,32,33,31]`
☐ `[50,40,20,10,30]`
☐ `[10,34,8,67,17]`
☐ `[49,9,29,19,39]`

(b) **[15 points]** Suppose the DBMS wants to compresses a table `R(a)` using columnar compression. Which of the following compression schemes **will benefit** (when considering space efficiency) from sorting the table before compressing column `a`? Select **all** that apply.

Hint: "Benefit" means that the efficacy of the compression scheme may improve on sorted data. You should not make any assumptions about the column type or its distribution of values.

☐ Bit-packing Encoding
☐ Run-length Encoding
☐ Mostly Encoding
☐ Bitmap Encoding
☐ Dictionary Encoding
☐ Delta Encoding
☐ None of the above will benefit.

(c) **[15 points]** A colleague approaches with a list of true and false statements about run-length encoding, delta encoding, bitmap encoding, and dictionary encoding. The colleague wants your assistance in identifying the true statements. Select **all** that apply.

☐ Run-length Encoding is effective for compressing a low cardinality integer column.
☐ Delta Encoding is good at compressing large text values.
☐ For *point lookup-only* workload, order-preserving dictionary encoding is required.
☐ If dictionary codes are order-preserving and you insert a new distinct value that falls between two existing values, you may need to reassign codes.
☐ Run Length Encoding is most effective on unsorted uniformly random data.
☐ None of the above.