Data Systems, Quiz - 1

Answer Key

Correct options are highlighted green

---

1. The size (in bytes) of data or disk blocks for a file
    a. must be 2048 bytes
    b. must be the same fixed number of bytes for all blocks
    c. can be any number of bytes but multiple of OS page size in bytes
    d. must be exactly OS page size in bytes
    e. None of the others

Explanation:

a. “must be 2048 bytes”: Block sizes are not required to be exactly 2048 bytes. Block sizes can vary depending on the system and storage requirements.
b. "Must be the same fixed number of bytes for all blocks": This statement is true in many file systems. It is common for all data blocks in a file to have a fixed and consistent size.
c. “Can be any number of bytes but multiple of OS page size in bytes": This statement is more flexible and reflects a common practice. Block sizes can be any number of bytes as long as they are a multiple of the operating system's page size. This flexibility allows for better alignment with hardware and OS requirements.
d. “Must be exactly OS page size in bytes": This statement is not true. While block sizes can be aligned with the OS page size for efficiency, it's not a strict requirement, and file systems may use different block sizes.

2. An ordered file for storing relation R(K, A1, A2, ... An)
    a. must be ordered only on key attributes
    b. can be ordered on one attribute (key or non-key)
    c. must be ordered on any attribute with NO NULL values
    d. can be ordered on any number of attributes with any number of NULL values for these attributes
    e. None of the others

Explanation:

a. "Must be ordered only on key attributes": This statement suggests that the ordered file should be sorted based only on the key attributes (K) and not on any non-key attributes (A1, A2, ... An). While it's common to sort an ordered file primarily based on key attributes, it is not a requirement. We can choose to include non-key attributes in the sorting order if it makes sense for your database design and query requirements.
b. "Can be ordered on one attribute (key or non-key)": This statement is true. An ordered file can be sorted based on either key attributes or non-key attributes or a combination of both, depending on the database design and query needs.
c. "Must be ordered on any attribute with NO NULL values": This statement is not true. Sorting an ordered file based on an attribute with no NULL values is a common practice, though not an absolute requirement. We can still have an ordered file with sorting based on an attribute that may contain NULL values.
d. "Can be ordered on any number of attributes with any number of NULL values for these attributes": This statement is true. You can design an ordered file to be sorted based on any number of attributes, and these attributes may contain NULL values as well, depending on your database schema and requirements.

3. An unordered file storing a relation R(K, A1, A2, ... An)
    a. the length of the row must be less than data block size
    b. all attributes of the row must not have NULL values
    c. the row length must be fixed size
    d. the rows must not be in a sorted order of values for any subset of attributes.
    e. none of the others

Explanation:

a. "The length of the row must be less than the data block size": This statement is not a strict requirement for an unordered file. The length of rows can vary, and it's common for rows to span multiple data blocks if they are large.
b. "All attributes of the row must not have NULL values": This statement is also not a strict requirement for an unordered file. Rows in an unordered file can contain NULL values in some or all attributes, depending on the database schema and data.
c. "The row length must be a fixed size": This statement is not true for unordered files. Rows in an unordered file can have

varying lengths, and the file can accommodate variable-length rows.

d. "The rows must not be in a sorted order of values for any subset of attributes": This statement is not true for unordered files. An unordered file does not impose any specific sorting order on the rows, whether it's based on key attributes or any subset of attributes.

4. In case of extendible/dynamic hashing to store relation R(K, A1, ..., An)
   a. The hash attributes cannot take float/real values
   b. The hash attribute must be always the key attribute
   c. the directory size of extendible hashing is always greater than size of directory for dynamic hashing
   d. Overflow blocks are needed for extendible hashing
   e. None of the others

   Explanation:

   a. "The hash attributes cannot take float/real values": This statement is not true. In extendible/dynamic hashing, the hash attributes can take any data type, including float/real values. The choice of data type for the hash attribute depends on the specific requirements and design of the database.
   b. "The hash attribute must be always the key attribute": This statement is also not true. While it's common for the hash attribute to be the key attribute, it's not a strict requirement. You can choose any attribute as the hash attribute based on your database design and indexing needs.
   c. "The directory size of extendible hashing is always greater than the size of the directory for dynamic hashing": This statement is false. In extendible hashing, the directory size can increase as the data grows, but it doesn't necessarily have to be larger than the directory size used in dynamic hashing. The directory size in both methods depends on various factors such as the number of records, the desired bucket size, and the specific implementation.
   d. "Overflow blocks are needed for extendible hashing": This statement is not true for extendible hashing.

5. Linear hashing is used to store relation R(K, A1, A2, ..., An)
    a. to access a row it can take exactly one block access
    b. to access a row it can take exactly two block accesses
    c. to access a row it can take more than two block accesses
    d. requires overflow blocks to store the rows
    e. None of the others

Explanation:

a. “To access a row, it can take exactly one block access": This statement is true for linear hashing. Note that it is mentioned we “can” access “a row” not “all rows”
b. "To access a row, it can take exactly two block accesses": Similar to the previous statement, there “can” be conditions when this condition is true.
c. "To access a row, it can take more than two block accesses": This statement is true. In linear hashing, as the data distribution changes and the hash table grows, it may require multiple block accesses to retrieve a specific row, especially when there are collisions or the data is distributed across multiple buckets.
d. "Requires overflow blocks to store the rows": This statement is true for linear hashing.

6. Given a relation R(K, A1, A2, ..., An) is a relation that is stored as an ordered file ordered on attribute A1. The file storing R has b data/disk blocks. -
    a. Number of block access needed to find K=val is on an average b/2
    b. Number of block accesses needed to find all rows of A1=val can be 1
    c. Number of block accesses needed to find all rows of A1=val can be 2
    d. Number of block accesses needed to find all rows of A1=val can be **b**
    e. None of the others

Explanation:

a. "Number of block access needed to find K=val is on an average b/2": This statement implies that, on average, you would need to access approximately half of the blocks to find a specific value of K.

NOTE:

i. *The accuracy of this statement can depend on the specific data distribution within the ordered file. If the values of A1 are*

uniformly distributed throughout the file, then this statement is a reasonable estimate. In such a case, you would need to access about half of the blocks, on average, to find a specific value of K.

ii. *If the data distribution is skewed or clustered, it could require more or fewer block accesses than b/2 on average to find the desired value. So, while statement 1 provides a rough estimate of the average number of block accesses, it may not hold true in all scenarios, and the actual number of block accesses can vary based on the data distribution.*

b. "Number of block accesses needed to find all rows of A1=val can be 1": If the relation R is stored as an ordered file on attribute A1, and you are looking for all rows where A1 has the value "val" it is possible to find all such rows in a single block access if the values are contiguous in the ordered file.
c. "Number of block accesses needed to find all rows of A1=val can be 2": This statement suggests that you may need two block accesses to find all rows where A1=val. This could be true if the values are not contiguous in the ordered file, and you need to access two blocks to retrieve all matching rows.
d. "Number of block accesses needed to find all rows of A1=val can be b": This statement implies that you would need to access all the blocks in the file to find all rows where A1=val. This statement “can” be true when A1 has only one value (*val*).
