Assignment#3
MSIT 630 Database Systems
Total: 50 points
Renee Raven

1. Big Data differs from traditional relational databases on distinct metrics. Explain. (4 points)

Our text mentions three metrics which differ between Big Data and traditional relational databases: volume, velocity, and variety.

Volume refers to the amount of data to be managed, stored, and processed. Big Data can handle much larger amounts of data than traditional relational databases, even if the traditional relational databases use parallel processing. While traditional relational databases are limited to hundred's of machines processing data in tandem, Big Data can mange data across thousands of machines.

Velocity refers to the speed of absorption and processing of new data. Big Data utilizes techniques to process data in real time, as it arrives, something traditional relational databases cannot manage.

Variety refers to the forms of data and the tools used to interact with such data. Traditional relational databases are limited to relation data and query languages. Big Data uses a "a new generation of languages and frameworks" to manage, the many types of data collected today (p. 468).

2. What are two advantages of encrypting data stored in the database? (4 points)

Encryption is a valuable tool which allows transmission of sensitive information while protecting that information from unauthorized persons or entities. Without the decryption key, the information appears as nonsense. Two big advantages of encrypting data stored in the database are the low resource use and the seamless integration with other applications. As our text states: "encryption at the database level has the advantage of requiring relatively low time and space overhead and does not require modification of applications" (p. 450).

Additionally encryption based user authentication, through a challenge-response system, is more secure than allowing passwords to be sniffed out when they are passed over the internet.

3. Identify and discuss (4) key systems for Big Data storage developed over the

past two decades to address data management requirements for related applications. (4 points)

The fours systems developed over the past two decades to address the data management requirements of applications on Big Data are distributed file system, sharding across multiple databases, key value storage systems, and parallel and distributed databases (p. 472).

A distributed file system spreads the data across a huge number of computers using redundant storage to compensate for hardware failures. Advantages to the distributed file system include scalability and allowing a single file-system view even though the data is dispersed across many machines. Examples include the GFS (Google File System) and HDFS (Hadoop File System).

Sharding partitions the data, usually on partitioning attributes or keys, and then storms the partitioned data in multiple databases. This method does require additional tracking of which records are on which database. Like a distributed system, it enjoys high scalability, and it also is easy to implement. Disadvantages include an increased chance of failure as it expands and sluggish performance due to routing of queries and tracking.

Key-value storage systems also partition data across multiple machines, but first the data is divided into a small record in the range of kilobytes to megabytes. To help prevent data loss, records are replicated across multiple machines, and the key-value system makes the same updates to all the copies across al the multiple machines. Different platforms (Amazon S3, Google BigTable, Apache Cassandra, etc) may use different methods of storing the value with the associated key.

Parallel databases run across multiple machines, but do not allow easy scalability. Failures may cause a restart and, since they were designed specifically for smaller scale (the limit is generally hundreds of machines), failures are more likely to occur with a larger system.

4. In the sequential file organization, why is an overflow block used even if there is, at the moment, only one overflow record? (6 points)

A sequential file system stores records in a sorted order based on an attribute or a set of attributes known as a search key. Since it relies on a specific order, inserting and deleting records poses serious challenges. Pointers connect each record to the previous record in the daisy chain like structure. As records are deleted, gaps develop in the blocks. When a record needs to be inserted, it needs to be placed within the sequential file organization.

Most of the time, an inserted record is placed in an over flow block, a block at the end of the file structure, and pointers are placed to indicate the record proceeding and following it. The only time an overflow block is not used is when there is a free record in the block the record should be in (following the sequential order system). That is a rare occurrence, therefore, even if there is only one overflow record, if there is not a free spot in the block it belongs to, it must go into an overflow block. As a side note, every so often this system becomes unwieldy and the entire file system must be reorganized with records and properly nested where they should be.

5. List 2 advantages and two disadvantages of each of the following strategies for storing a relational database (10 points):
a. Store each relation in one file
Advantages include easier updates and backups since it uses the native file system provided by the operating system. Disadvantages include decreased performance due to the slower, simpler file structure.

b. Store multiple relations (perhaps even the entire databases) in one file.
Advantages include increased performance since efficient search algorithms can be applied through the database management system. Disadvantages include increased size, resource use, and complexity.

6. What are the causes of bucket overflow in a hash file organization? What can be done to reduce the occurrence of bucket overflows? (4 points)

A hash file organization uses buckets to store the data. Each record belongs in a certain bucket based on an index. When a record needs to be inserted into the hash file organization the appropriate bucket is determined. Bucket overflow occurs when there is no room in the bucket for the record to be inserted. An inadequate number of buckets for the records is another cause of bucket overflow. Finally, “bucket overflow can also occur if some buckets are assigned more records than are others, resulting in one bucket overflowing even when other buckets still have a lot of free space” (p. 660).

Assigning keys randomly and evenly to buckets will help with bucket overflow issues. Additionally, estimating the number of buckets to be used and then creating some percentage more, say 20 percent, will prevent some bucket flow issues. This is known as static hashing, but it doesn’t account for a growing number of records. Several dynamic hashing techniques are being developed, including linear hashing and extendable hashing, which allow gradual increases in buckets based on the flow of records in a database.

7. What benefit does rigorous two-phase locking require? How does it compare with other forms of two-phase locking? (6 points)

Rigorous two-phase locking requires “all locks be held until the transaction commits,” and therefore it enforces the serializability order. Most importantly, concurrency issues are avoided (p. 842). Rigorous two-phase locking is preferentially used in most contemporary database applications, such as banking. All two-phase locking protocols determine locking rules. Rigorous two-phase allows the growth phase of locking, but eliminates the shrink phase because all locks must be held until the transaction is finalized. Static two-phase locking demands all locks are in place at the onset, but allows conditional release of locks, the shrinking phase. Basic two-phase locking allows both growing and shrinking. It is more vulnerable to issues and not used as often in commercial applications.

8. Identify and explain some of the issues that must be addressed when designing a remote backup system. (6 points)

Remote backup systems are physically separate backup locations which keep data synchronized with the primary site. A remote backup site is vital protection against primary site failure. Several factors need consideration when designing a remote backup system (pp. 932- 934):

1. Failure detection
There should be multiple communication and network links to the primary site to ensure a failure is immediately detected.

2. Seamless transfer of control
Once failure occurs, the backup must take over the duties of the primary, including query management. In effect, the backup becomes the new primary. This transfer can be controlled manually or automatically or, alternatively, a high availability proxy may route all interactions to whatever system is currently acting as the primary.

3. Recovery time
Ideally, recovery time should be as short as possible. Some techniques to decreased delay include periodic processing of the redo log records and developing a hot_spare configuration where the backup continuously processes redo log records. The resources devoted to keeping the backup synchronized with processed redo log

records must be balanced with the benefit of an almost instant recovery.

4. Time to commit
To balance availability and the potential problem of lost transactions, a method must be used to manage when a record is considered committed. Three techniques are discussed in the text. One-safe considers a transaction final as long as it is written to the primary site, however this could allow record loss if the primary site goes down and the backup site doesn't have the newer records. Two-very-safe protects much better against data loss since it requires new records be written to both primary and backup sites, however it decreases the availability of the database in real time. Two-safe mimics two-very-safe when both primary and backup sites are available. But if only the primary site is active it allows the transaction to be committed, thereby allowing greater availability than two-very-safe with a higher degree of reliability than one-safe.

9. Explain the differences between the three storage types: volatile, nonvolatile, and stable – in terms of I/O cost. (6 points)

Volatile storage is unstable and vulnerable to data loss because it is stored in active memory, such as RAM or cache. In terms of Input Output cost, volatile uses the fewest resources and is the least costly, but it allows the fastest access.
Non-volatile storage uses physical media to store data that survives system crashes. In terms of I/O cost, non-volatile storage uses more resources, including the physical media and the algorithms to find the data (and finding data usually takes longer than volatile storage).
Stable storage is built with redundancy to prevent data loss at a theoretical 100%. Stable storage uses multiple non-volatile stores using multiple failure modes. Of the three, stable storage is by far the most costly in terms of I/O cost. The same data is stored on multiple media and many unique algorithms are used to prevent data loss.

10. Describe the basic steps involved in query processing. (4 points)

There are three basic steps of query processing:
1. Parsing & translation
First the validity of the query is verified by checking syntax and the referenced relational names. Then the query is translated into a form the database can use.
2. Optimization
A query will be further transformed into an optimized form that maximizes efficiency.
3. Evaluation

During evaluation, the query is submitted to the database and results are returned. Evaluation often involves breaking the query into subqueries, getting those results, and synthesizing the resulting subqueries into a final result through a process called materialization.

11. A drawback of cost-based optimization is the cost of optimization itself. Optimizers use heuristics to reduce the cost of optimization. Please describe at least three heuristic approaches for transforming relational-algebra queries. (4 points)

Heuristics are rules that usually reduce cost and optimize efficiency by reducing the total number of records under consideration. The first heuristic that is generally followed is "Perform selection operations as early as possible," which reduces the number of tuples and the size of the relations. Next, the heuristic rule "Perform projects early" reduces the number of attributes by reordering the select and project operations to the bottom of the tree structure. Finally, "Avoid cartesian products" ensures the most restrictive select and join operations are done before other operations.

12. Explain the concept of materialized views and include key advantages and challenges. (4 points).

Materialized views are stored snapshots of computed results. If one were to think of an Excel workbook as a database, a materialized view is analogous to a worksheet with stored calculations from the other worksheets. Advantages include reduced resource use since we don't have to run queries to calculate the information in the materialized view again and again. Materialized views also improve a programs performance when a view is used to replace a query that will be run frequently.

Materialized views depend on data, so when that data changes, the materialized views need to be updated. This is a big disadvantage that needs to be addressed. Multiple approaches to ensuring updated data is present in the materialized views range from manually updating data to various automated incremental algorithms. However, within this disadvantage is another disadvantage: all of these methods of updated cost resources, thereby potentially decreasing the benefits of a materialized view.

References

Abraham Silberschatz, Henry F. Korth and S. Sudarshan,. (2020). Database System Concepts, (7th ed.). McGraw Hill Education