10.48047/jocaaa.2026.35.01.39

Reducing Database Footprint: The Fastest Path to Cost and Performance Gains

Amit Kumar Garg
Independent Researcher, USA

Abstract

When a database becomes slow and expensive to run, solutions include expanding database infrastructure, increasing the instance size, or the number of replicas. However, these only affect database symptoms; they do not address the root cause of the growing database size. The footprint of a database is the total amount of data that's in use (including storage, indexing, replication, backup, and recovery), which often goes unmeasured. Organizations also retain information that has no operational value - old logs, records, and disused application data. Given that deleting data is often costly but storage is inexpensive, the cost of retaining data falls on storage, replication, backups, indexes, and disaster recovery. Reducing the size of the database has multiple benefits. As a side effect of the smaller database, databases are typically scanned more quickly, cached better, and disaster recovery is faster, among other advantages. For industries that have low profit margins, a smaller operational database can directly translate into greater profits. Information lifecycle management allows an organization to classify the data based on how frequently it is accessed and how important it is to the business, moving cold data to less-costly storage while stripping or using caching to increase the performance of business-critical data. Footprint reduction eliminates waste, making it more efficient than scaling. Intentional data residency strategies, such as data classification, archiving, and the expiration of ownerless data, can help with cost-effectiveness, performance, and resilience for organizations over time.

Keywords: Database Footprint Reduction, Information Lifecycle Management, Disaster Recovery Optimization, Cloud Cost Management, Data Retention Strategy

1. Introduction

In practice, when performance decreases or costs increase, operations teams often respond by vertically scaling the database infrastructure: bigger instance sizes, more replicas, a more expensive database platform, and so on. These often only treat symptoms. The single most effective way to lower costs, improve performance, and ensure durability might instead be to reduce the database's footprint, the total volume of data that's persisted, indexed, replicated, backed up, and thereby must also be recovered as part of normal production operations.

The cloud database management systems market has started to shift. Enterprise customers, long happy with raw scaling power, have begun to realize this does not solve their data management problems. According to Gartner, enterprises are moving away from measuring the performance of data resources toward measuring cost efficiency and total cost of ownership [1]. This is in part because it is wellunderstood that operational databases may contain large amounts of data not currently used in active business processes, yet the storage, processing, and communication resources are still used.

Modern databases can group enormous amounts of data, and this is able to collect much information, though not to ascertain its worth. As demonstrated by research into the performance of NewSQL database systems, these weaknesses can be partially resolved with improved architectures, but will weaken performance when the data set is large, involving historical data, deprecated data, or data that is not used. [2] Regardless of the

10.48047/jocaaa.2026.35.01.39

database technology used, the reduction of the working dataset size has a positive impact on almost every aspect of database behavior, often without any architectural change or infrastructure changes.

2. The Hidden Cost of Data That Never Gets Deleted

Some of the most common but rarely-discussed kinds of data bloat consist of data that serves no operational purpose: application logs kept in the main database for a long time, historical events kept without a process to retrieve them afterwards, or old records kept across generations of software systems. This data is kept not because it is needed, but because it is considered dangerous to delete and cheap to store.

A study of current data retention practices found that organizations greatly underestimate the compounded costs associated with retaining data indefinitely. An analysis of enterprise data management practices noted that forgotten or unexamined data amasses hidden costs across multiple dimensions of an organization [3]. Retained rows may have no business utility, but still represent a liability. Database systems are designed to keep replica and primary entries consistent. This incurs costs in terms of storage and replication. The overhead is not limited to storage, as CPU resources are used for index maintenance operations, even for data that is not queried and never shows up in query results.

Another high cost of scaling the database is that backup and restore times scale linearly with the size of the database, even if historical data is not used in normal operation, because the backup system must serialize, compress, and store the data according to retention policies. This effect is most pronounced in disaster recovery scenarios where the time to recover a service has a linear dependency on the amount of data that needs to be copied and checked [4]. An analysis of distributed databases at Google, including F1 and Spanner, has demonstrated that the cost of cross-datacenter replication and consistency scales linearly with the amount of data.

The hidden cost multiplier for index maintenance: Database systems maintain indexes to make query execution faster, but there is a cost associated with each write operation to maintain the indexes. Index maintenance for tables with millions or billions of rows with no current business purposes incurs a cost in write throughput and I/O capacity that otherwise could be used for active workloads [4]. Database overhead for maintaining B-tree structures, gathering statistics for query optimization, and index fragmentation increases as the number of rows increases, regardless of whether or not the table region receives any read traffic. This leads to organizations continuously spending computational resources to have rapid access to data that they never use, which is often a hidden cost that is only highlighted once the system is close to being full.

Cost Dimension	Impact Area	Characteristics
Storage and Replication	Infrastructure expense	Multiplies across primary and replica instances
Index Maintenance	Computational overhead	Scales with total row count regardless of access patterns
Backup Operations	Time and capacity requirements	Processes all data, including unused historical records
Disaster Recovery	Cross-datacenter replication	Bandwidth and consistency costs increase with data volume

Table 1: The Hidden Cost of Data That Never Gets Deleted - Cost Dimensions [3, 4]

10.48047/jocaaa.2026.35.01.39

3. Why Footprint Reduction Has an Outsized Financial Impact

The savings multiply in several ways. The economics of footprint reduction are particularly strong when looking at the overall costs of maintaining data, and the costs multiply when the infrastructure is duplicated to maintain copies of the information in multiple locations.

The most immediate benefits come from infrastructure savings, when the production systems do not have to store data that is no longer needed, or is unlikely to be valuable in the future. AWS offers multiple cost optimization frameworks, noting that cloud infrastructure scales with consumption, with databases being a key area for cost savings [5]. The reduction in storage cost for the primary is proportional to the amount of data deleted, while the reduction for replicas is multiplicative. Production database systems often use multiple replicas for high availability and for scaling reads, and so the price of one gigabyte of unnecessary data in the primary can be several gigabytes in the replicas. The cost of backup storage scales with the size of the database. Since backups are often retained for weeks or months, the footprint of useless data compounds. Disaster recovery using cross-region replication also increases storage and bandwidth costs and is of particular concern for globally distributed systems that may have multiple data footprints.

But the most meaningful business benefit is not the direct infrastructure costs, but the fact that, in business sectors with very thin margins, such as retail, the decrease in operational costs flows directly to businesses' bottom lines. Deloitte states that retail profit margins are under significant pressure and remain in the low single digits for many sectors of retailing across the world. In these circumstances, the relationship between operational cost savings and profit impact would be very different from that between revenue growth and profit impact. In a low-margin business, a dollar saved in operational cost directly translates to increased profit without needing to increase sales per unit or gross margin. In particular, many large retailers, for supporting e-commerce, inventory tracking, or analyzing customer preferences can reduce overhead to improve profitability without the need to increase customer acquisition or market share.

A second class of outsized benefit from footprint reduction is performance benefit. Smaller tables have faster scan performance because there are fewer rows that the database systems must consider to satisfy the search predicates. Index structures also improve performance as the size of their indexed datasets decreases. This can be rationalized in terms of the improved cache utilization that is possible for larger caches, which allows a larger fraction of the working data set to be kept in the cache. Experiments on online transaction processing workloads confirm this trend for sequential scans and index range scans [7]. Performance improvements are most pronounced on tail latencies, the slowest queries that are typically due to cache misses, I/O amplification effects, and are exposed when scaling the dataset size. Avoiding issues in the first place with a smaller database results in better performance regardless of the particular pattern of database accesses.

Impact Category	Benefit Type	Key Characteristics
Direct Infrastructure Savings	Immediate cost reduction	Affects storage, replicas, backups, and replication traffic
Business Impact in Low-Margin Industries	Profit amplification	Operational savings deliver greater impact than equivalent revenue increases
Performance Gains Without Scaling	System efficiency	Faster scans, improved cache utilization, reduced lock contention

10.48047/jocaaa.2026.35.01.39

Tail Latency Improvement	User experience enhancement	Addresses slowest queries affected by cache misses and I/O amplification

Table 2: Financial Impact Dimensions of Footprint Reduction [5, 6]

4. Disaster Recovery: Faster Recovery Starts with Less Data

Database footprint also plays a role in disaster recovery, which is sometimes overlooked. Restoring a database specification from a database backup, a clean state/database image from corrupt or damaged data, or a database image from loss of infrastructure may be limited by how much data needs to be copied, validated, and brought to a consistent state to start a recovery. Although raw processing power and network bandwidth are technically limiting factors, it is the volume of data to be sent and validated that is critical.

An analysis of existing work on transaction processing systems shows that recovery operations are sequences of data transfer and consistency, and reconciliation operations that scale with the size of the data set [7]. A database backup restore requires reading backup files into the target instance, copying data, rebuilding indexes, and validating data. The time taken for each of these steps is related to the size of the database, and on smaller databases, the steps are completed more quickly, so normal operation is resumed more quickly, and the window of vulnerability is smaller. Database size and the length of time required for recovery are both key factors when dependencies, such as secondary failures during the recovery process, are considered.

The size of a database affects not only its restore time, but also its complexity, and risk of other, more subtle corruption issues. As a database increases in size, it also increases the risk that some corruption will go undetected during the integrity-check phase following restoration and validation. When replicas have to be rebuilt after a failure, the time to copy data from the primary to make the replicas consistent increases with the database size. Furthermore, companies operating in strictly regulated sectors may be subject to strict recovery time objectives that become impractical with increasing amounts of data copied to a replica. It is typically accompanied by a reduced database footprint, meaning it contains only the data that the business actually needs to continue operating and meet recovery time commitments.

Nowhere is the problem of excessive database footprints and disaster recovery characteristics more obvious than with old application logs, which are intrinsically voluminous. Normal operations create thousands to millions of lines of log data on systems. Since logs will rarely be viewed more than a small time box and only used for diagnostics during the time shortly after an application event occurs, the easiest storage device is often a transactional database, which is often the first database written against during application development. Experimental studies on large-scale data processing and storage systems show that log access patterns differ from those of transactions. Most log accesses do not access data older than some recent time. Apache Spark and other log processing or analytical systems favor append-only data storage and logical partitioning by time, both patterns that do not reflect optimal transactional data. Keeping several years of logs in production databases provides next to no value for operational querying, but considerably increases backup size, restore time, and disaster recovery complexity. They are often in append-only, cost-efficient stores with well-defined retention policies, rather than in transactional systems designed for low-latency random access and strong immediate consistency.

Recovery Aspect	Relationship to Footprint	Operational Significance
Restoration Time	Linear scaling with data volume	Determines business downtime duration

10.48047/jocaaa.2026.35.01.39

Replica Rebuild Speed	Constrained by data transfer requirements	Affects high availability restoration
Validation Complexity	Increases with dataset size	More data structures require consistency verification
Log Data Characteristics	High volume, low ongoing value	A common contributor to excessive footprint and poor recovery metrics

Table 3: Disaster Recovery Impact Factors [7, 8]

5. A Practical Strategy for Reducing Footprint

Database footprint reduction stresses data residency, not aggressive data deletion, where data is stored on the storage tier, and for the duration that matches its actual usage and business value. A disciplined footprint reduction approach brings great active business benefits and keeps data available to support legitimate business needs.

Access frequency and data value classifications are part of the successful footprint management process. Oracle's information lifecycle management framework also recognizes that data value changes as it ages and moves to a different, more cost-effective storage tier [10]. Hot data, needed for production workloads, is read from and written to constantly and typically needs a high-throughput and low-latency tier. Hot data is usually any near-term data, such as recent transactions, active customer accounts, or the current state of an application, and is accessed frequently during business operations. Warm data tended to be used less often and was stored in cheaper storage with lower data access performance. An example of warm data is relatively recent historical data that is accessed by analytical queries for business intelligence, or for queries made by customer service agents. Data with low access frequency, retained for compliance or infrequent use, can be stored with storage optimized for cost, capacity, and durability.

A related finding is that only hot data should reside in primary transactional databases. Information lifecycle management strategies show that information in different stages of the information support lifecycle should reside in different storage tiers. Storing them in a single tier will either incur high costs or hurt performance [10]. Cold data can be moved off the critical path to lower-cost object storage systems with less performance but more capacity and durability, and higher latencies. Archived data can be stored in point-in-time snapshots rather than rows intended to be queried, and is used for compliance and recovery purposes rather than for random access. Offloading cold data reduces index maintenance operations, which would otherwise keep historical data updated, and it also reduces the footprint of query optimizers, which operate over small indexes that can be held largely in memory.

The easiest way to reduce footprint costs is often the hardest to achieve organizationally: Data without a data owner, a clear and demonstrated business use case, or an access pattern over a sufficiently long duration of time are usually great candidates for removal. However, organizational inertia makes deleting systems a decision, whereas retaining them is not. Data governance policies that specify a default retention period and require having a reason to extend it result in an inverted status quo, where the path of least resistance is to delete systems for which data no longer needs retaining. A general principle of optimizing database systems is that systems with smaller working sets of data have much better performance characteristics than otherwise similar systems with larger working sets of data, even if the latter are running on more powerful machines [7]. Footprint reduction beats scaling. While scaling infrastructure makes managing workloads easier, the inefficiencies of data retention and data management remain. Footprint reduction eliminates these

10.48047/jocaaa.2026.35.01.39

inefficiencies, even as workloads continue to grow. It often renders scaling unnecessary or at least considerably delayed, produces more predictable performance improvements than hardware upgrades, reduces operational risk by easing recovery, and results in lower long-term overall costs for users since reliance on resource allocations is minimized.

Strategy Element	Implementation Approach	Expected Outcome
Data Classification	Hot, warm, cold tiers based on access frequency	Aligns storage costs with business value
Cold Data Migration	Archive to object storage, snapshot-based retention	Removes historical data from the critical path
Ownerless Data Deletion	Governance policies with retention justification requirements	Prevents the accumulation of unused information
Lifecycle Management	Automated tier transitions based on data age and access patterns	Sustained footprint optimization without manual intervention

Table 4: Practical Strategy Components for Footprint Reduction [9, 10]

Conclusion

Data may grow in volume over time to the extent that the costs, risks, and operational consequences manifest themselves only as increased storage costs, degraded performance, or degraded disaster recovery capability. Archived historic logs, obsolete records from defunct systems, and data held at a retention period longer than the business rationale can limit an organization's performance or throughput. Database footprint reduction is a high-leverage engineering practice with secondary benefits in economic efficiency, performance, disaster recovery overhead, and business impact. It is backed by empirical evidence from cloud database management systems, distributed databases, transaction processing, large data systems, and information lifecycle management systems that database performance often degrades as dataset size grows, especially the tail latencies that impact the application quality of experience. However, for companies where the profit margins are already low and competition is high, reducing the infrastructure operating cost is much more profitable than increasing revenue. Cost savings go straight to the bottom line without any additional sales, and the existing database technologies can be scaled to large volumes. However, it is best to apply this scale to a dataset that is worth the investment instead of a dataset that should no longer be retained. Smaller databases have faster and easier backup and restore processes. This means that disaster recovery is easier and recovery time objectives are more likely to be met. Structured information lifecycle management involves classifying data by access frequency and value, placing infrequently accessed 'cold' data in inexpensive storage classes, and monitoring the creation of ownerless data. Continuously lowering storage costs, while improving efficiency, is much more achievable when data storage expenditures are aligned to business needs rather than just the cost of the infrastructure to support the increasing demand for storage. Adopting a systematic approach of classifying, archiving, and deleting data builds the foundation for future-proofing the drives, costs, and other components of the infrastructure as workloads change.

References

[1] Gartner, Inc., "Cloud Database Management Systems Reviews and Ratings," 2025. [Online]. Available: https://www.gartner.com/reviews/market/cloud-database-management-systems

10.48047/jocaaa.2026.35.01.39

[2] Andrew Pavlo and Matthew Aslett, "What's Really New with NewSQL?" SIGMOD Record 2016. [Online]. Available: https://db.cs.cmu.edu/papers/2016/pavlo-newsql-sigmodrec2016.pdf

[3] Jean Pierre Ntayagabiri et al., "Study on the Development and Implementation of Different Big Data Clustering Methods," Open Journal of Applied Sciences, 2023. [Online]. Available: https://www.scirp.org/reference/referencespapers?referenceid=3529801

[4] Jeff Shute et al., "F1: A Distributed SQL Database That Scales," University of Wisconsin-Madison, 2013. [Online]. Available: https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/41344.pdf

[5] Amazon Web Services, "Cost Optimization with AWS." [Online]. Available: https://aws.amazon.com/aws-cost-management/cost-optimization/

[6] Evan Sheehan, "Global Powers of Retailing 2023," Deloitte Global. [Online]. Available: https://www.deloitte.com/global/en/Industries/consumer/analysis/global-powers-of-retailing.html [7] Stavros Harizopoulos et al., "OLTP through the looking glass, and what we found there," SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data, 2008. [Online]. Available: https://dl.acm.org/doi/10.1145/1376616.1376713

[8] John Moore, "The cost of downtime and how businesses can avoid it," TechTarget, 2025. Available: https://www.techtarget.com/searchdatabackup/feature/The-cost-of-downtime-and-how-businesses-canavoid-it

[9] Matei Zaharia et al., "Apache Spark: a unified engine for big data processing," Communications of the ACM, 2016. [Online]. Available: https://dl.acm.org/doi/10.1145/2934664

[10] Oracle Corporation, "Information Lifecycle Management (ILM)." [Online]. Available: https://www.oracle.com/database/advanced-compression/information-lifecycle-management/