Database Archiving: Managing Data for Long Retention Periods

Craig Mullins,
Corporate Technologist, *NEON Enterprise Software, Inc.*

white paper

NEON
Enterprise Software, Inc.

Table of Contents

white paper

Introduction

Organizations are generating and keeping more data now than at any time in history. There are many reasons why this is so. First of all, the amount of data in general is growing. According to industry analysts, enterprise databases are growing at the rate of 125% annually. Even more interesting is that as much as 80% of the information in those databases is not actively used (in other words, it is ready to be archived).

But why are we producing so much data? Advances in technology have better enabled our ability to capture and store data. But technology alone is not sufficient to account for the current rate of data growth.

Data may need to be retained for both internal and external reasons. Internal reasons are driven by company needs. Simply stated, if an organization requires the data to conduct business and make money, then that data will be retained. Today's modern organizations are storing more data for longer periods of time for many internal reasons. Typically, data is stored longer than it used to be to enable analytical processes to be conducted on the data. Data warehousing, data mining, OLAP, and similar technologies have delivered more and better techniques for extracting information out of data. So businesses are inclined to keep the data around for longer periods of time.

But external reasons, typically driven by the mandate to comply with legal and governmental regulations are another significant factor driving the need to store more data.

Legal Requirements to Archive

The corporate accounting scandals of the past few years have caused an onslaught of new laws to be written. These laws place regulations on how businesses are to treat their sensitive, business-critical data. Additionally, older laws that have been on the books are being enforced more rigorously than in the past. Basically, government regulations are being adopted to ensure that corporations are "doing the right thing" with their data.

The number one driver of data management initiatives is government regulations. The growing number of regulations and the need for organizations to be in compliance is driving data retention and extending the length of time that data must be retained. Regulations such as the Sarbanes-Oxley Act, HIPAA and BASEL II are some of the laws governing how long data must be retained. But this is just the tip of the iceberg. Industry analysts have estimated that there are over 150 federal and state laws that dictate how long data must be retained.

Many of these laws greatly expand the duration over which data must be retained. Until recently, most organizations dealt with mandatory retention periods of only a few years for important business data. And this data was kept around longer because of business reasons, not legal requirements. But the situation has changed due to the bevy of new regulations at the federal, state, and local levels. Depending on the industry, what was once five or seven year retention periods is now expanding to 20, 30, even 70 years. Today, retention periods are determined almost exclusively by government regulations and not from business needs.

To comply with these laws corporations must re-evaluate their established methods and policies for managing and retaining data. What worked in the past to retain data for a few years will no longer be sufficient over a much longer period.

Perhaps the most significant piece of legislation impacting data governance is the Sarbanes-Oxley Act. Section 802 of this act defines penalties for altering or deleting important business data and documents. Additionally, this legislation supports the records preservation rule defined in the Securities and Exchange Act of 1934 (Rule 240.17a-4). This means that electronic storage media must preserve the records in a non-rewritable, non-erasable format. Clearly, Sarbanes-Oxley requires organizations to implement a robust data retention solution.

But, of course, Sarbanes-Oxley is not the only legislation driving data retention requirements. Indeed, the Association of Corporate Counsel publishes its "Model Corporate Records Retention Plan" which establishes guidelines for developing and implementing a records retention program. Additionally, it provides copious examples of the types of data that must be retained. This information is documented in *Electronic Evidence and Discovery,* by Lange & Nimsger.

Of course, the exact legal requirements for data retention will vary for each organization based on its business and location. The only overarching truism that can be stated is that more and more data is mandated to be retained for longer and longer durations. And businesses must be capable of retaining and accessing that data when it is required.

The ability to produce retained data upon request is frequently driven by lawsuits. You probably can recall examples of courtroom showdowns on television where truckloads of paper documents were required during the discovery process of the lawsuit. But times have changed. Increasingly, the data required during the discovery process is electronic, not written.

According to published research from Gartner, "Electronically stored documents are becoming the predominant form of evidence presented in courts of law." As such, we need to take great care with the data stored in our computer systems to preserve and maintain that data for electronic discovery purposes. Gartner goes on to state "Because more litigation is based on evidence that is stored and managed electronically, correct and swift production of that evidence is an important business process."

To wit, the U.S. Supreme Court has approved amendments to the Federal Rules of Civil Procedure (FRCP) concerning discovery of electronically stored information. FRCP Rule 34b states that "A party who produces documents for inspection shall produce them . . . as they are kept in the usual course of business . . . " So, if the data is stored electronically during the usual course of business, it must be produced electronically.

As we begin to comply with laws requiring long-term data retention, data archival will become more pervasive. We can't just keep decades upon decades of data in our production databases. Research conducted by Enterprise Strategy Group indicates that digital archive capacity will increase nearly tenfold between 2005 and 2010. Total worldwide digital archive capacity in the commercial and government sectors will grow from about 2,500 petabytes in 2005 to more than 27,000 petabytes by 2010. And they state that the major factors driving this growth will be regulatory compliance, corporate governance, litigation support, records management, and data management initiatives.

Clearly, organizations will be retaining more data over longer periods of time. And this will create the need for new policies, procedures, methodologies, and software to support storage, management and access of archived data.

The Lifecycle of Data

So how can we determine when data needs to be archived? In order to accurately answer that question we need to understand the different states of data as it progresses through its lifespan. The diagram in Figure 1 delineates the various states of data over its useful life.

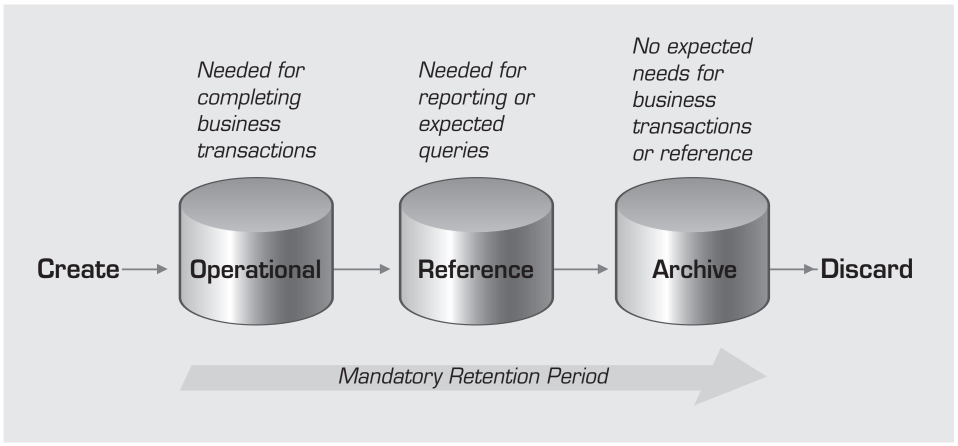
Needed for completing business transactions
Needed for reporting or expected queries
No expected needs for business transactions or reference
Create
Operational
Reference
Archive
Discard
Mandatory Retention Period

Figure 1. The Lifecycle of Data

Data is created at some point, usually by means of a transaction: a product is released, an order is processed, a deposit is made, etc. For a period of time after creation, the data enters it first state: it is operational. That is, the data is needed to complete on-going business transactions. This is where it serves it primary business purpose. Transactions are enacted upon data in this state.

The operational state is followed by the reference state. This is the time during which the data is still needed for reporting and query purposes. It could be to produce internal reports, external statements, or simply exist in case a customer asks about it.

Then, after some additional period of time, the data moves into an area where it is no longer needed for completing business transactions and the chance of it being needed for querying and reporting is small to none. However, the data still needs to be saved for regulatory compliance and other legal purposes, particularly if it pertains to a financial transaction. This is the archive state. It is the requirements for data in this state which this white paper addresses.

Finally, after a designated period of time in the archive, the data is no longer needed at all and it can be discarded. This actually should be emphasized much stronger: the data must be discarded. In most cases the only reason older data is being kept at all is to comply with regulations, many of which help to enable lawsuits. When there is no legal requirement to maintain such data, it is only right and proper for organizations to demand that it be destroyed – why enable anyone to sue you if it is not a legal requirement to do so?

Don't think in terms of databases or technologies that you already know when considering these data states. The data could be in three separate databases, a single database, or any combination thereof. Furthermore, don't think about data warehousing in this context – here we are talking about the single, official store of data – and its production lifecycle.

From here-on out we will use the data lifecycle terms introduced in this section, with the emphasis being on archiving database data and the issues arising from doing so.

What is Database Archiving?

Database Archiving is part of a larger topic, namely Data Archiving. Data exists in many formats and for many purposes, and only a small percentage of it is actually in a database. Physical documents, electronic documents, computer files and data sets, e-mail, and multimedia files are all examples of data that may reasonably need to be archived at some point. Refer to Figure 2. Each of these "things" needs to be archived to fulfill regulatory, legal, and business requirements.

But each type of data requires different archival processing requirements due to its form and nature. What works to archive e-mail is not sufficient for archiving database data, and so on. In other words, type of data may need to command its own technology. This is most certainly true for database data. Why?

Well, data stored in a database is different than other types of data in many ways. The main advantage of using a DBMS is to impose a logical, structured organization on the data. A DBMS provides a layer of independence between the data and the applications that use the data. In other words, applications are insulated from how data is structured and stored. The interface to the data is through the DBMS data language, whether it is SQL for relational databases, DL/1 for IMS, or even XQuery for XML databases. So the archival of data from a database requires knowledge of, and operation in conjunction with, the mechanisms and interfaces of the DBMS.

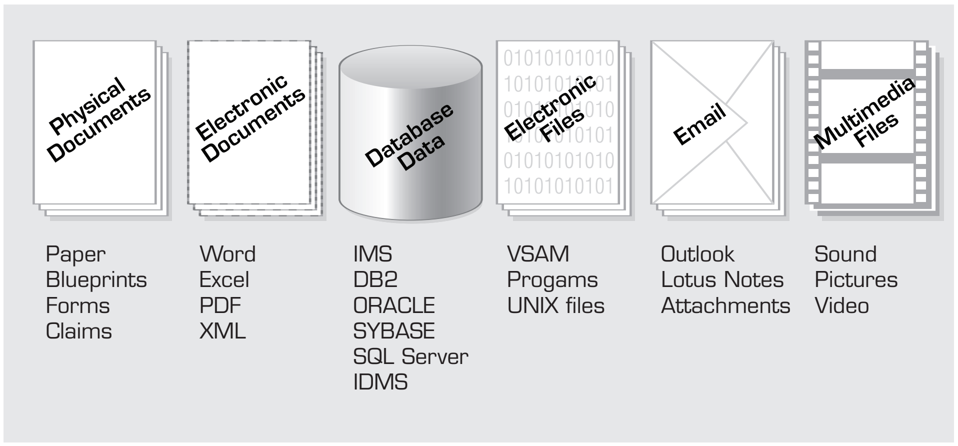

Figure 2. All Types of Data Need to be Archived

OK, if we now accept that database archiving is a subset of data archiving, let's define exactly what we mean by the term. *Database Archiving* is the process of removing selected data records from operational databases that are not expected to be referenced again and storing them in an archive data store where they can be retrieved if needed.

Let's examine each of the major components of that last sentence. We say *removing* because the data is deleted from the operational database when it is moved to the data archive. Recall our earlier discussion of the data lifecycle. When data moves into the archive state, query and access is no longer anticipated to be required, so it no longer needs to reside in the operational database.

Next, we say *selected records*. This is important because we do not want to archive database data at the file level. We need only those specific pieces of data that are no longer needed for operational and reference purposes by the business. This means that the archive needs to be able to selectively choose particular pieces of related data for archival…not the whole database, not an entire table or segment, and not even a specific row. Instead, all of the data that represents a business object is archived at the same time. For example, if we choose to archive order data, we would also want to archive the specifics about each item on that order. This data likely spans multiple constructs within the database (tables for DB2 or Oracle; segments and/or databases for IMS).

The next interesting piece of the definition is this: *and storing them (the data) in an archive data store.* This implies that the data is stored separately from the operational database and does not require either the DBMS or the operational applications any longer. Archived data is separate and independent from the production systems from which it was moved.

The final component of the definition that warrants clarification is…*where they can be retrieved if needed.* The whole purpose of archiving is to maintain the data in case it is required for some purpose. The purpose may be external, in the form of a lawsuit or to support a governmental regulation; or the purpose may be internal, in the form of a new business practice or requirement. At any rate, the data needs to be readily accessible in a reasonable timeframe without requiring a lot of manual manipulation. I mean, let's face it, anyone can archive data if they don't have to worry about how to query it later, right?

Current "Solutions"

Oftentimes organizations believe they are archiving their data for long-term retention, but the "solutions" that are being employed to do so actually are falling short of the mark. This situation typically occurs when insufficient attention is paid to the query requirements against the archived data. Let's examine some of these "solutions" in a bit more depth.

File and database backups are sometimes viewed as a solution to data archival. The idea proffered is that backups need to be done for recoverability anyway so why not use them for the archive? This is an incorrect assumption for several reasons. Why?

The first problem is that backup copies are not in a format that can be queried. In order to access the data it needs to be recovered back to a database format. Conducting such a recovery to enable queries against data no longer in the database is problematic, though. Do you recover the data directly into the existing operational database? This can cause data integrity problems because the backup data contains not just the data no longer in the database, but perhaps also data currently in the database but in a different state. In other words, although you might put the old data back in the database, you are likely also to corrupt existing data in the database.

Of course, you could use an unload utility to unload the data from the image copy backup (if that technology is available to you). But then you would have to go through that data to determine what is and what isn't needed before you load it into the production database.

Keep in mind, though, that none of the above has even taken into account that the database schema might have changed since the backup was made. So you will have to consider that when re-loading the data from the backup. And is the data actually "the same" if it is loaded into a different schema definition? It might seem to be, but technically it is not – and that might be a problem from a legal perspective.

And think about the work involved when you run into an audit situation, where you are asked to produce transaction records spanning multiple years. This requires the very labor-intensive, manual process of slogging through the data and matching it against every database change that was made during the timeframe in question (which could be several decades). It might be (somewhat) feasible for time spans of 2 to 3 years perhaps, but not for archived data that spans 10, 20 or more years.

A similar approach to using backups is to use UNLOAD files. But, once again, this has all of the same problems as using image copy backups, with the additional problem that the UNLOAD files have to be created in addition to the backups.

Another approach might be to create a "reference" copy of the operational database to house the data. But, of course, we still have to deal with the issue of changing database schemata. Furthermore, if the data is maintained in a traditional database system then it is not protected from modification by database language commands such as SQL INSERT, UPDATE, and DELETE statements. And even if you use database security commands to prohibit modification this will not stop super users (e.g. SYSADM) from changing data.

Sometimes organizations choose just to leave the data resident in the operational database. This has all of the problems of the "reference" database approach, along with the additional problem of negatively impacting the performance of production applications and queries against the operational database.

white paper

These “solutions” are really no solution to the problem of archiving data for long-term retention. None of them are really useful in today’s environment. Indeed, your long-term data retention plan needs to account for the following:

- Archival of logically selected data instead of entire files or database structures
- Policy-based archival rules that automatically trigger data retention based on business and regulatory mandates
- Deletion of the data from the operational database when it is archived
- The ability to query directly from the archive to avoid the problems of rebuilding data to support audit requests

Database Archiving Requirements

OK, we have defined database archiving, including what it is and what it is not. And that, in and of itself, is a big accomplishment. But there are many additional aspects that need to be considered for a true database archiving solution. And most of these issues are not thought of until sometime during an actual database archiving project. But by then it may be too late to address them.

Perhaps the most important consideration is that the archived data must be *hardware and software independent.* In other words, the archive cannot rely on the operational platform or applications in order to be useful. It must exist independently and be accessible without interacting with the production systems.

Why is independence so important? Consider the duration over which the archived data must exist. With a lifespan of decades it is more than likely that the production system from which the data was archived will no longer exist – at least not in the same form, and perhaps not at all. What changes have your production applications gone through over the course of the past ten years? twenty years? fifty years? No, it is completely unreasonable to expect that the operational environment will exist to enable access to the archived data.

Because the operational system and data structures will change over such long retention periods, the archive needs to be independent from the operational system. We constantly change our databases. And the archive must be able to support multiple variations of the application as it changes. This includes keeping up-to-date with the metadata changes within each variation of the system for which we are archiving data.

The archive solution also must be able to storage a *large amount of data.* This is so because, as we discussed at the beginning of this white paper, current levels of data storage are higher than ever before. And when we combine this with increasing rates of data growth and long data retention periods we have an explosive combination.

The archive must be able to *manage data for very long time periods.* As we have discussed already, many data retention requirements are stated in decades. This means that the archive will outlive the applications, DBMSes, operating systems, and so on, that generated them. Yes, there may be a DB2 thirty-five years from now but it will look a lot different than the DB2 we use today. It also means that the archive will outlive the people who designed and managed the operational systems. And the archive will outlive the media we store it on so we will need to manage the data over time and re-platform it.

And that brings up another interesting issue. No media is guaranteed to exist over long periods of time. Indeed, consider the lifespan of tape: most are not guaranteed more than 7 years. This poses a problem for archived data. The archive solution must offer management capabilities to periodically move the archive data from one type of storage media to another. It must also enable vigilant tracking of all media used by the archive and trigger corrective actions when the media is reaching the end of its useful life.

Data, once archived, must remain unchanged. So the archive must be able to *protect against data modification.* Only read access should be available to the archived data, with the exception of periodic administration (such as making backups). The archived data must be guaranteed to be authentic. In other words, what was archived from the operational systems, perhaps years ago, is forever unchanged in the archive. And mechanisms to prevent surreptitious modification are necessary, too.

The archive requires *metadata* to be useful. Gartner corroborates this assertion saying "Metadata must be preserved when files are requested for evidentiary purposes." But there are two types of archive metadata. The metadata defining the archived data itself is needed in order to enable access to the data. The archive must be able to store multiple versions of this metadata for the same "logical" archived object. As the operational schema changes the archive metadata changes, but it is the same "logical" object. Because data cannot change once it is stored in the archive, the metadata must remain unchanged for all prior versions of the archived data. And the query mechanism must handle the differences in metadata across the like "logical" business objects.

The second type of metadata is the archive metadata that controls when the data is archived, which data is archived, from where, and any transformations that occur. This is the metadata that drives and defines the archive itself. Both types of metadata are needed for the archive to operate.

Taking all of these considerations into account, then, a secure, durable archive data store must be used to retain data that is no longer needed for operational purposes, and it must enable query retrieval of the archived data in a meaningful format until it is discarded.

Components of a Database Archive Solution

The issues discussed in this white paper are some of the many that we at NEON Enterprise Software, Inc. have discovered as we built our Titan Archive solution. Titan Archive offers a legal archive that is locked down from change once in the archive.

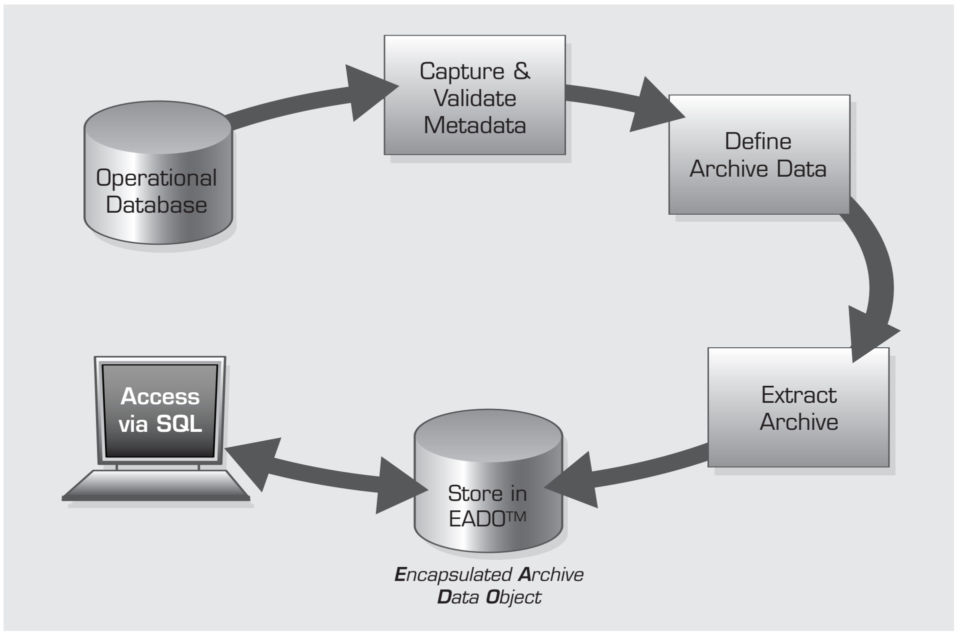

Figure 3. NEON Enterprise Software's Titan Archive solution

The diagram in Figure 3 shows a high-level view of the functions you would perform using NEON Enterprise Software's Titan Archive solution. To build your archive, you would need to capture and enhance the metadata for the archive. It doesn't really matter what you archive if you don't know what it is. Therefore, the first step is to capture your database metadata, validate it, and even possibly enhance it.

Why might enhancement be needed? Perhaps you have not defined all of your relationships using referential integrity in your relational databases. Or maybe you want to transform codes into more meaningful definitions in the archive. Furthermore, consider the work that might be needed for non-relational databases. Studies have shown that as many as 80% of COBOL copybooks do not accurately map to IMS data. So the archive user will need to obtain real data to test against the metadata to make sure it accurately describes the data and make necessary changes.

After capturing and validating your metadata, it can be used to help define and drive the data that you want to archive. At this point you can select exactly what data to archive, specify reference information needed by the archive for context, create rules for archive triggers, and even integrate the archive with your change control process.

After capturing, validating, defining, and testing your metadata and data, you will need to set up the jobs for extracting the data from your operational database. Because the extract must run against the operational database it is important for this process to be as non-disruptive as possible. Generally, you will want to schedule it to run in batch using your job scheduling product, but online archiving is supported, too, for one-off and simpler archiving needs.

Although it is important that strict controls be placed on extraction, flexibility is also important. One very important option is simulation. A simulation will allow you to "test out" your archive process. It lets you see the size of your archive, and even create an archive that you can test to ensure that you are archiving what you want to archive.

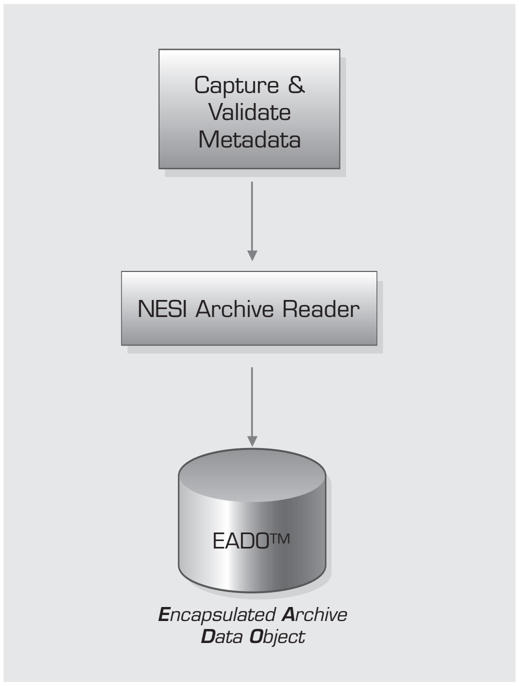

Figure 4. Querying the Archive Using Standard Industry Tools

Keep in mind, too, that the extract will be gathering the appropriate data and moving it to the archive, but it will also be deleting data from the operational database. Efficient, high-speed processes are needed to ensure that the extract is as non-disruptive as possible.

The final piece of the process is the ability to access the data in the archive using a standard interface regardless of data source. Refer to Figure 4. Titan Archive provides the ability to query archived data using SQL and accepted industry standards. This means you can use query tools such as Crystal Reports, Cognos, and Business Objects.

Archive Administration

Of course, as with any important data store, the archive will require on-going care and feeding. Archive administration tasks need to be designed into your archive solution in order to efficiently facilitate:

- Job management
- Security management
- Audit trail management
- Media management

The archive solution needs to manage jobs, and interoperate with job scheduling systems, in order to create and execute extracts from operational databases. Furthermore, the archive environment mandates a configurable security management system that is separate from your original database security. The people who work on your operational database will not be the same set of people you want to access the archived data. And even if they are the same, there will be different security requirements for data that is retained over long periods of time in the archive.

Furthermore, the archive processes themselves need to be instrumented. This is necessary to track who is doing what to which set of archived data. A robust set of audit data showing when archives are created, who created them, the history of the jobs that created them, when they were backed up, and so on, is imperative to support archived data.

Finally, management of the storage media is required to ensure that archived data is in tact and accessible over time. An effective archive solution will provide mechanism for re-platforming archived data as needed due to media deterioration, changing standards, new system software, and so on.

Summary

As regulatory burdens continue to increase and organizations find themselves faced with ever more onerous data retention requirements, a robust, enterprise-level database archiving solution will become a required component of a comprehensive data management policy. Be sure to completely review and understand all of the requirements for long-term data retention before implementing a database archiving solution.

About NESI

NEON Enterprise Software is the technology leader in enterprise data availability software and services. In a world where every second counts, our tools maximize database performance and availability and minimize business risk. Founded in 1995, NEON Enterprise Software is headquartered in Sugar Land, Texas and serves customers worldwide with its dedicated team of industry experts.

For more information, visit www.neonesoft.com or call 281.491.6366 or 888.338.6366.