Data Archiving Essentials — What Every Administrator Needs to Know

Dr. Rolf Gersbacher and Helmut Stefani

Dr. Rolf Gersbacher
Data Archiving Development and Coordination
SAP AG

Helmut Stefani
Data Archiving Development and Coordination
SAP AG

(complete bios appear on page 28)

You undoubtedly will recognize this scenario: A user wants to know the sales figures for a specific customer for the last quarter so that she can reassess her current sales strategy. She runs the appropriate report, taking the utmost care of selection parameters. System performance is generally good, but the report seems to take forever.

In all likelihood, the report needs to process an unnecessarily large volume of data because the database contains an enormous number of records belonging to closed or no longer relevant business transactions. Processing larger volumes takes a performance toll on the database and application servers. Upgrading these servers can prevent data bottlenecks only up to a point, since system performance is subject to certain physical limitations and hardware is also only scalable within certain boundaries.

A program running a selection across a large data volume can lead to low system performance.[1] Even in the case of access using a qualified database index, as is normally the case, the program still has to search a highly nested index. The situation becomes critical if the data of the corresponding database index table is distributed across several database blocks, all of which have to be read.

[1] A large data volume does not necessarily imply poor performance, since the database access time is scaled in a logarithmic, rather than linear, fashion according to the formula **T~ln** (data volume). In addition, other factors, such as the following, may influence response time:

- Load times for programs and screens
- Time required to roll in and roll out user contexts on the application servers
- ABAP processing time
- Time to transport the data across the network

As the saying goes, "an ounce of prevention is worth a pound of cure." Administrators are well advised to archive data before ailing performance becomes a characteristic of their SAP environment. Most SAP shops don't need to be convinced of the merits of archiving, but for the few nonbelievers out there, or for those of you who deem archiving to be a low priority, we offer some advice. Experience shows that change-mode access to data for closed business transactions is required in only the rarest of cases.

Furthermore, making changes to financial documents for a closed posting period is tantamount to data manipulation, and is therefore forbidden. Application-specific reports hardly ever require access to data from closed business transactions, since the business process to which the data belongs is no longer of any importance for daily business. Leaving this data in the database only increases the data volume, which leads to degradation of system performance, increased IT overhead, and extra hardware provisioning. Simply deleting data is not a viable option if read access to individual documents is required from time to time or legal requirements dictate that the information be retained. In cases such as these, you need a solution where the relevant data can be relocated from the database to external storage media in such a way that it can be read again at a later date. SAP Data Archiving was designed expressly for this purpose.[2]

SAP Data Archiving enables you to keep the size of your database under control over the long term. This consequently ensures improved system availability and performance, and more efficient use of IT resources such as hard disks, CPUs, networks, and so on.

Archiving has become such an important concern to so many IT departments that we consider it beneficial to revisit (in abridged form) some of the archiving fundamentals we introduced in a white paper a while back, entitled "Managing SAP Data Archiving Projects."[3] This article's descriptions of how SAP's Archive Development Kit (ADK) works, the ways in which you interact with it, and how to plan for and execute archiving and post-archiving activities will bring newcomers up to speed very quickly. And there's plenty of good information here for those of you who are already well versed in the fundamentals. Threaded throughout every section are suggestions and tips about how best to wield SAP ADK technology,[4] particularly with regard to FI_DOCUMNT, which is one of the most commonly used and therefore most important archiving objects.

At the Heart of the SAP Data Archiving Solution — The Archive Development Kit (ADK)

Archiving basically boils down to two runtime activities:

- A write program saves the data for archiving to archive files and stores them in the file system. (From here, you can transfer them to other storage media.)
- A delete program deletes from the database the data that was archived by the write program. (For security reasons, the archived data is not deleted from the database until it has been successfully stored in the file system or in the connected storage system. This ensures that data that has not been successfully stored in the archive is not deleted.)

[2] SAP Data Archiving is the only method approved by SAP for removing data from the database while at the same time maintaining the technical and business integrity of the application data. (Comprehensive consistency checks at the application level ensure that only data relating to closed business transactions is archived.)

[3] This document is available on the SAP Data Archiving Web site at **http://service.sap.com/data-archiving**. From here, choose **Media Center → Literature**.

[4] This article focuses on the ADK technology that is available in Releases 4.0, 4.5, and 4.6.

Know Your Archiving Vocabulary!

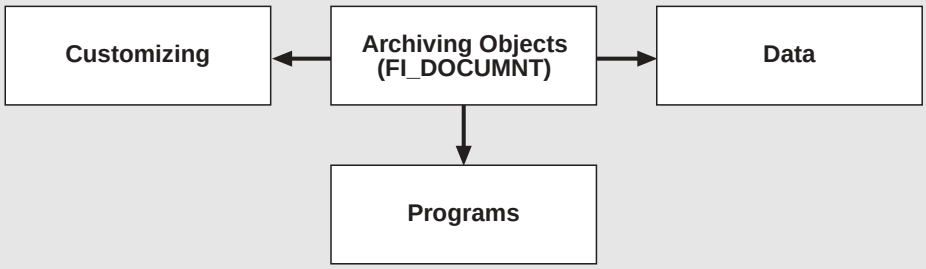
Since the term “archiving” tends to be used indiscriminately in IT, it is important to distinguish between “reorganization” and “optical archiving” on the one hand, and “data archiving” on the other.

Reorganization: Within SAP, this term is often mistakenly confused with data archiving. In the applications, the term refers predominantly to the physical deletion of application data from the database. Its actual meaning refers to the reorganization of the database, which involves a physical redistribution of scattered data. The aim of reorganization is to consolidate the data into context-relevant storage areas (defragmentation).

Optical archiving: This term is also confused with data archiving. Optical archiving refers to the electronic storage and administration of documents (such as scanned original documents, print lists, and outbound documents) in a storage system that is based on optical media, such as CDs or WORMs. This concept would be better served by the terms “imaging” or “document storage.” Instead of being stored in the database, this digitized data is stored and administered in a document management system (DMS). The SAP system only retains a link to the storage location of the document in question.

Residence time: The residence time denotes the period between creation of the data and the point from which, according to logical checks, the data can be considered archivable. The residence time is used primarily as a criterion for archivability for transaction data. You can specify the residence time in application-specific Customizing. In the case of financial accounting documents, residence times can be selected according to account type and document type.[5]

Archiving object: The archiving object is a central element in SAP Data Archiving, and represents the smallest entity in the SAP system that can be wholly archived and deleted from the database. An archiving object describes how to access the appropriate tables in order to archive a business object fully. For example, archiving object FI_DOCUMNT “knows” exactly the tables from which data records must be read to remove a specific financial accounting document completely from the database. In addition to this logic component, an archiving object also contains the required archiving programs, such as write, delete, or pre-processing programs, and a Customizing component that enables you to set various archiving parameters. The diagram below shows an overview of the archiving object components.



[5] In FI_DOCUMNT, a document is classified according to document type. The document type is specified in the document header. For each document type, properties are set up that control how the document is created.

(continued on next page)

(continued from previous page)

Data object: The data object is the application-specific counterpart of the archiving object. It includes all application data (this can be master data or transaction data) necessary to archive a business object completely and consistently. Examples of master data objects are material masters, company masters, and bills of material; examples of transaction data objects are financial accounting documents, billing documents, and deliveries. A typical data object is composed of:

- **Header** — This contains all general information identifying the data object.
- **Items** — These contain the actual application data.

In a financial accounting document, for example, the header might contain general information that is valid for all items, such as the document currency, company code, or posting date, while the item data might record individual posting items, including amounts, payment conditions, debtor and creditor, and so on. The exact storage location of this data in the database is defined in the data model for the financial accounting document. Information in the header is stored in table BKPF, the item data in table BSEG, and so on. Descriptions, scanned customer letters, etc. can be linked to the financial accounting document. If the document is changed, change documents (which record the exact nature of the changes) are also written. During archiving, these various types of data must be relocated from the database as one entity, thereby ensuring that the financial accounting document remains intact.

In terms of SAP Data Archiving, the data object is the smallest business unit, and has one unique key in all SAP systems. A data object of this kind is persistently stored in the database and the data that it contains is stored in various relational database tables. This means that data is spread over several database tables rather than being stored in a single, object-specific table. The tables involved and how they are interconnected is defined in the underlying application data model. The tables are not usually linked by foreign key relationships or views. Therefore, the underlying database management system (DBMS) has no knowledge of the grouping effects and tables that identify a business data object. For data archiving purposes, this information is held solely by the archiving programs.

✓ *Tip*

Once they have been created, archive files can be stored automatically. To trigger this, choose ***Customizing → Archiving Object-Specific Customizing → File Storage to Storage System*** *(from the Archive Administration initial screen, shown in Figure 1).*[6] *Select the relevant content repository and mark "Start automatically."*

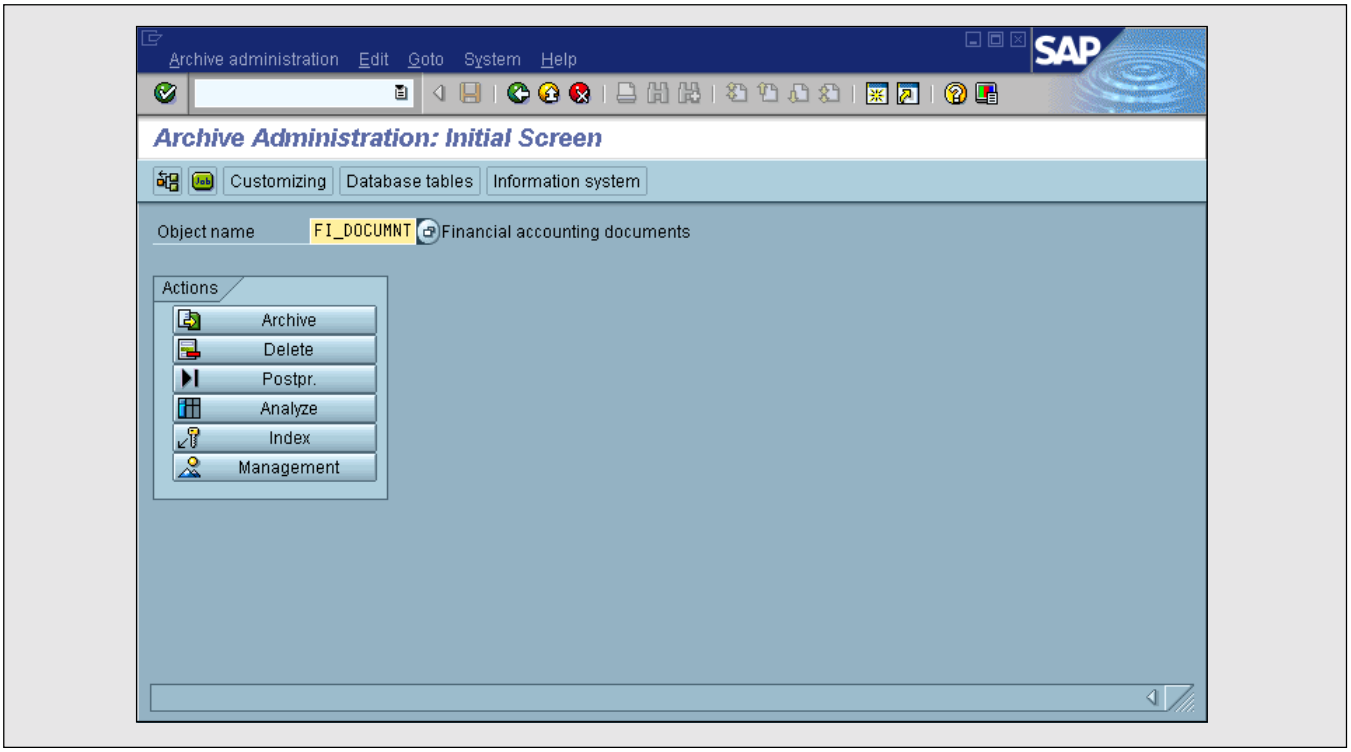
Both the write and delete programs are part of the runtime environment of the Archive Development Kit, which lies at the heart of the SAP Data Archiving solution. The ADK is an independent software layer linking the SAP application with the storage medium where the archive files are stored.

The ADK conducts all activities relating to archive files, including creation, opening, writing, reading, and closing. Furthermore, the ADK offers the following functions:

- Control and parameterization of the archiving session
- A central repository containing the definitions of archiving objects
- Administration of archiving sessions and archive files
- Control of batch processes when accessing the file system

[6] This function is available as of Release 4.6C. Prior to this release, this function was called "Connection to archive system" and was also available in Customizing.

Figure 1 Initial Screen of Archive Administration (Transaction SARA)



- Control of communication with external storage systems

You interact with the ADK through its user interface (transaction SARA).[7]

The ADK User Interface

Transaction SARA accesses the user interface for Archive Administration, which is the central starting point for the use of all archiving programs. It is the point from which you can:

- Schedule archiving sessions and delete jobs
- Read and analyze archive files
- Make technical settings (Customizing)
- Store or retrieve archive files
- Reload archive files
- Administer archiving sessions
- Check the status of archiving sessions and archive files
- Display dependencies between individual archiving objects

> **✓ *Tip***
>
> *If you access Archive Administration directly from the application menu (which is possible in most applications), the relevant parameters are set automatically. For example, to trigger the archiving of financial accounting documents, choose **Accounting → Financial Accounting → General Ledger → Periodic Processing → Archiving → Documents**. The name of the relevant archiving object, FI_DOCUMNT, is already set, and does not have to be entered manually.*

Figure 1 shows the initial screen of the Archive Administration transaction. Only the functions that

[7] The ADK also provides an *application programming interface* (API) that customers can use to develop their own archiving solutions.

are available for the relevant archiving object (in this case, FI_DOCUMNT) are displayed.

The ADK Runtime System

The ADK runtime system, which includes the user interface for Archive Administration (transaction SARA) and CCMS (Computing Center Management System) job management for running archiving sessions, provides all the functions required for archiving data. These functions include the archiving programs, such as the write program (which creates the archive files), the delete program (which subsequently deletes the data from the database), and various read and analysis programs (such as the SAP Archive Information System) for accessing archived data. Furthermore, the ADK runtime system offers monitoring tools for determining the appropriate archiving objects and for displaying the storage space in the underlying database tables. The physical storage locations of archive files are also stored in the ADK runtime system.

In addition to these baseline functions, the ADK also has a number of special functions related primarily to accessing archived data.

> ### ✓ *Tip*
>
> *Some archiving objects also have a reload function that enables you to reload archived data into the live database. This function is, however, to be treated with extreme caution, as historical data can be copied into a current database context. This can lead to serious data inconsistencies. Consequently, the reload function should be used only in an emergency — for example, if after an archiving session you realize that you have inadvertently selected too much or incorrect data for archiving. You should only reload data immediately after the archiving session.*

For example, when accessing the archive on a read-only basis, it carries out temporary *conversion* of the data to ensure that it can still be displayed in the current system, even after hardware and software upgrades. This guarantees that the data remains release and platform independent. The conversion does not have any effect on the data in the archive, since it only occurs temporarily in the read program.

How does the ADK conduct this conversion? The system stores metadata along with application data in the archive file. The metadata includes information such as:

- Schema of a database table at the time of archiving
- Data type and length of a table column
- Code page used (ASCII, EBCDIC)
- Number format used (for example, integer representation on various hardware platforms)

The ADK can use this information to read archive files long after their creation, even after release upgrades. If database structures in an application have changed to such an extent that the ADK cannot cope (for example, if table fields have been moved between tables, or one table has been redistributed across several tables), a program is supplied within the application that temporarily converts the read archive files at runtime.

When data is archived, it is automatically *compressed*. Depending on the table type, it is possible to compress data by up to a factor of 5. Data that is stored in cluster tables cannot be further compressed.

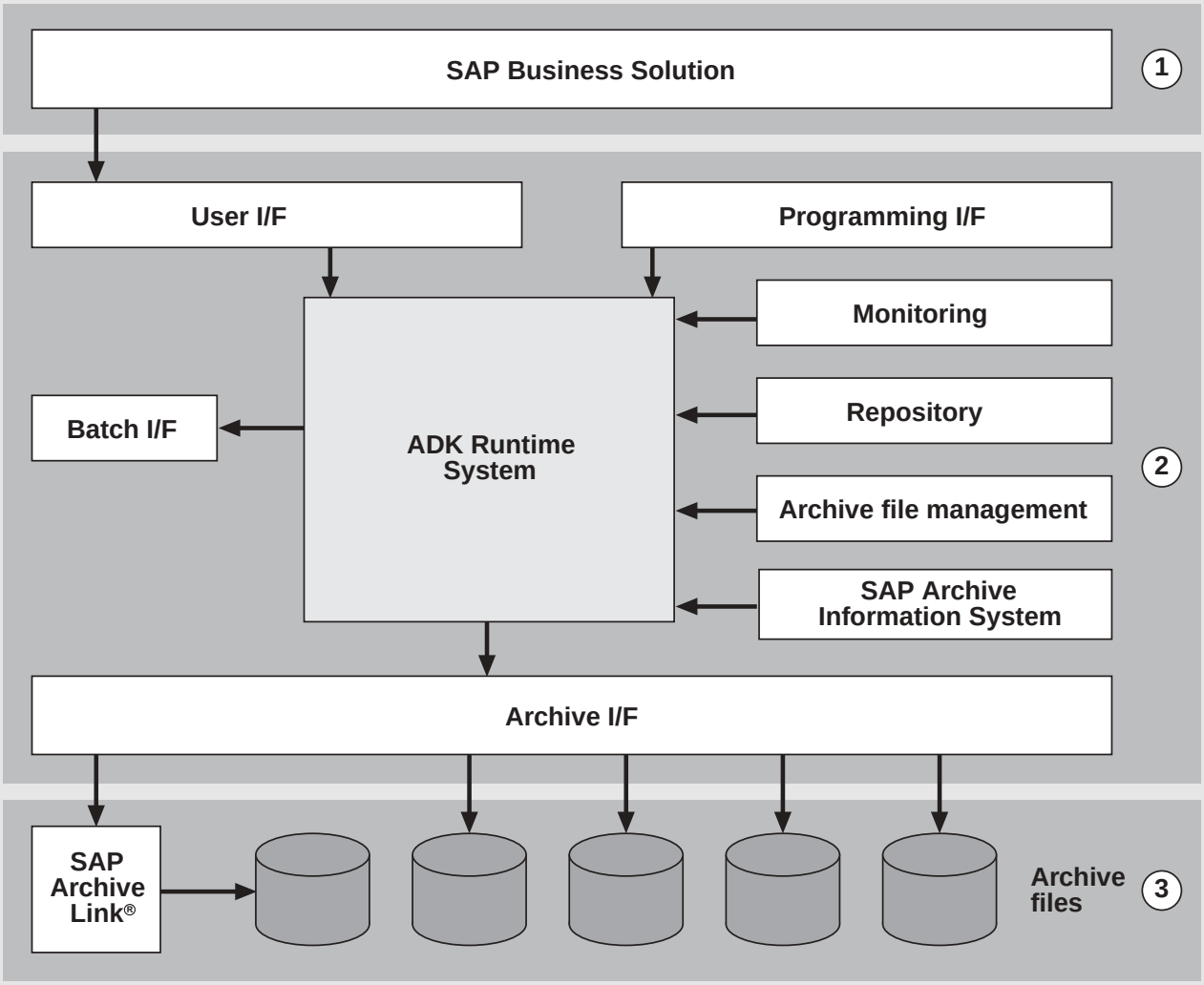
Data Archiving Repository

Transaction AOBJ accesses the central repository for archiving objects, which is used for setting up and customizing archiving objects.

Integration of ADK in the System Landscape

The ADK is embedded in the system context at three levels. The highest level (marked “1” in the diagram below) is the SAP Business Solution (application) from which the data is archived. Archived data objects are also accessed from the application. The individual archive files are read using defined interfaces in the ADK. The next level down (“2”) is the Archive Administration, with interfaces to the application and physical archive files. It provides the link between the tertiary storage media and the application. At the lowest level (“3”) are the archive files that are managed by Archive Administration.

The diagram below illustrates this relationship.



> ✓ ***Tip***
>
> *AOBJ is a cross-client transaction. Any change that is made is applied system-wide, and includes other clients, so consider the implications of your changes. Due to the logical dependencies between client-specific Customizing data and application data on the one hand, and cross-client Customizing data on the other, changing or deleting data can lead to inconsistencies.*

In the detail view for an archiving object, you can set the basic parameters for the programs and functions involved. Here, you specify the work area (such as FI) and the application components for the archiving object, and define which programs the archiving object is to have. You can also store additional useful information on the individual programs here.

Transaction AOBJ also offers several Customizing views that enable you to make the following settings:

- Structure definition (that is, the physical design of the data model)
- Tables from which only deletions are carried out during archiving
- Dependencies between individual archiving objects (network graphic)
- Read programs used
- Customizing settings, such as identification of archiving sessions and archive files, size of an archive file, point at which the storage is to take place, etc.

Access to Archived Data

During archiving, the data objects are stored in such a way that read access is possible for the SAP system at any time. Each archiving object has (at least) one read program that enables the data to be displayed within the SAP system. Users can restrict their search for data objects by specifying business-related selection criteria.

The following two sections briefly explain the two basic methods for accessing archived data:

- Sequential reading
- Direct access

Sequential Reading

This method is the simplest form of access to archived data. The read program positions the file pointer at the start of the archive file and processes the data bit by bit. The read process is completed when the pointer reaches the end of the file. The display program decompresses the data and displays it in line with the user's selection criteria.

Sequential read access is mainly used to analyze archived datasets and is available for most archiving objects. This enables users to display in list form all the data objects, for example, for a specific posting period or for a document number range. There is a direct, linear relationship between the time behavior of the read program and the size of the archive file to be read.

Direct Access

This method makes it possible to access individual data objects in the archive file, such as a billing document or a sales order, directly. In this case, the read program positions the file pointer at the start of the relevant data object within the archive file, and only reads the data object that was specified in the selection screen.

To access a data object directly, the read program must know the name and the path of the archive file

and the position of the data object within the archive file. This is only possible with the aid of an index, which can be created using the SAP Archive Information System (SAP AS) and stored in a transparent database table.

Archiving Procedures

Now that you are familiar with the technical background of data archiving, we shall examine in detail how you prepare for archiving and actually use the ADK, and then discuss what you do once archive files have been created. The technical processes of data archiving can be roughly divided into the following phases:

- Steps prior to archiving
- Writing data objects to archive files, and deleting those data objects from the database
- Storing archive files
- Steps following archiving

Steps Prior to Archiving

This phase is concerned with identifying the archiving objects that will give you the best results in reducing the database volume. It is particularly important to select the right archiving object(s) if a database table has several archiving objects associated with it. You should also discuss this with the relevant user department so that you can more accurately restrict these objects, and so that you can establish precise selection criteria.

For archiving object FI_DOCUMNT, this means, for example, determining the company code, the document number range, and the archivable fiscal years. This also includes an evaluation of the display and reporting functions.

You can use transaction DB02 to determine the largest tables in the database. Based on the historical growth of the database, you can estimate its future storage requirements. This transaction also supplies important database statistics. In the case of Oracle databases, for example, this includes table spaces, the number and size of the assigned database index tables, and the average utilized capacity of the extents.

In addition to transaction DB02, you can obtain further indicators that are useful for data archiving by using the SAPDBA tool[8] and the performance monitor (transaction ST03). In many cases, the SQL trace report has also proved beneficial.

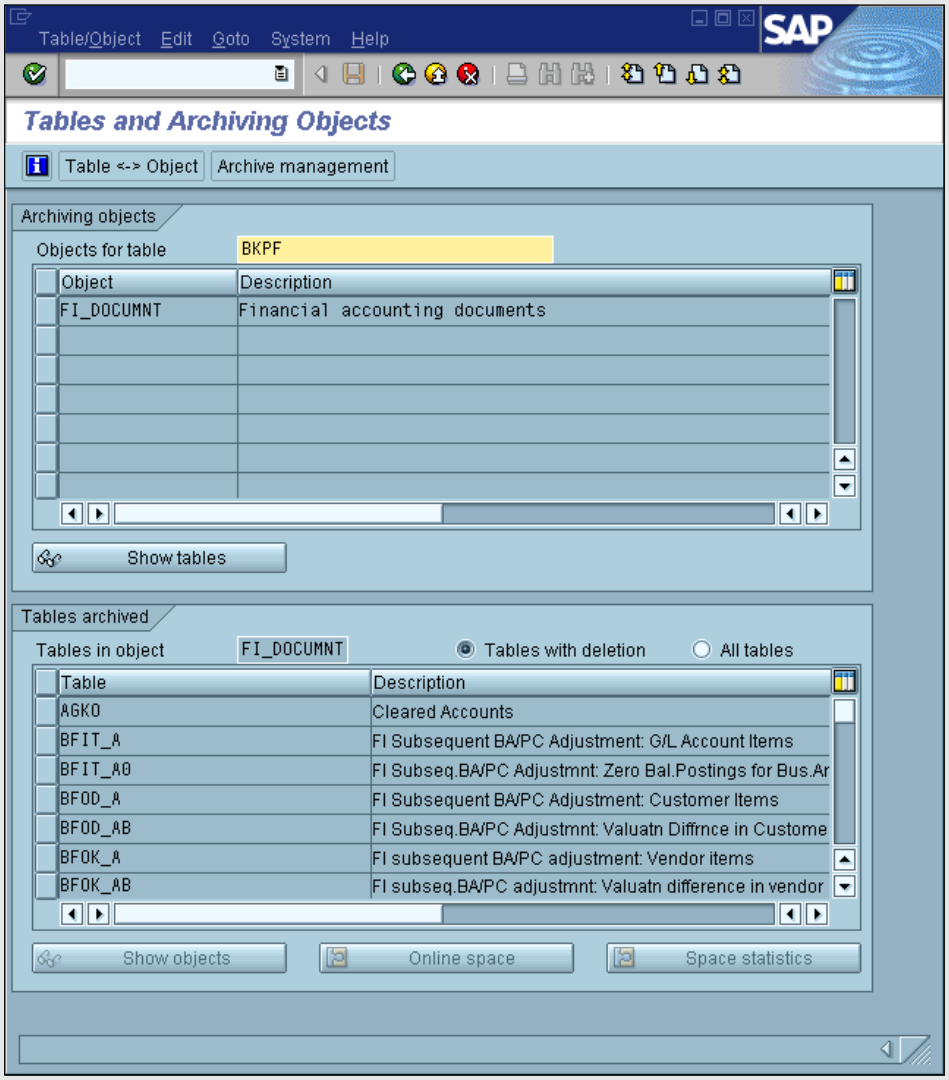
These tools help you to establish whether, from the point of view of the database, data archiving is necessary. Provided you have your database and database server ideally configured, the following symptoms may indicate that data archiving is necessary:

- Problems arise with the database administration (such as backup) due to large data volumes.
- Response times become critical, even when searches are conducted with a qualified database index.
- Database response time constitutes the most significant part of the total response time.
- The underlying tables are disproportionately large.

> ✓ ***Tip***
>
> *To check whether the database access is being optimally supported by means of a qualified index, you can carry out an SQL trace (transaction ST05) and analyze the result. For more information on this function, refer to the ST05 application help* ***(Help → Application Help)****.*

[8] SAPDBA runs outside the R/3 system, and is only available for Oracle and Informix database systems.

Figure 2 *Transaction DB15*



Once you have identified the relevant tables, you can determine the required archiving objects using transaction DB15, as shown in **Figure 2**.

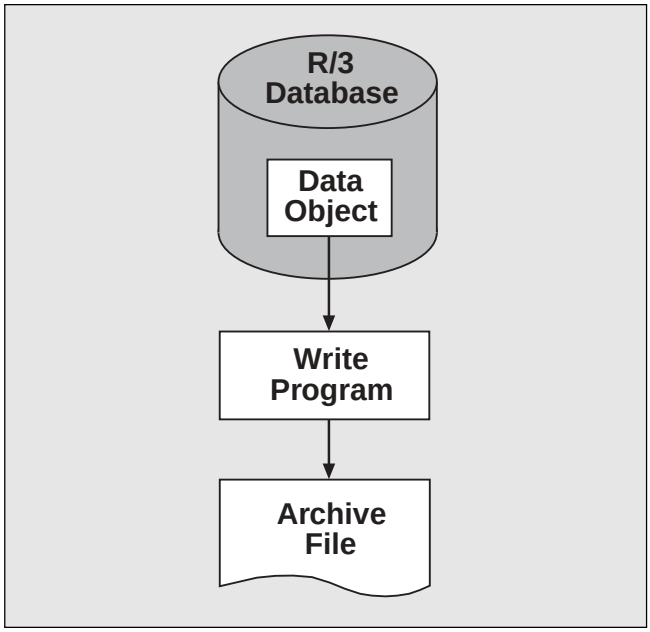
Writing Data Objects to Archive Files

First of all, the write program selects the data objects that are to be archived from the database and checks whether, from a business point of view, the data object is archivable. FI_DOCUMNT checks, for example, whether:

- Residence times have been completed at the header and item level
- The document has been settled (balance = 0)

The data object can only be archived once these checks have been performed satisfactorily for each data object. The write program initially creates the archive file and opens it for writing. In the next step, it writes the individual objects sequentially to the archive file(s). This is shown in **Figure 3**. You can

Figure 3 Using the Write Program to Create an Archive File



specify at what point the archive file is to be closed in Customizing — either once a specific file size has been reached, or when a specific number of data objects have been written to the archive file.

As soon as all the data objects have been copied to archive files, the program closes the archive files and updates the relevant management data in the SAP system's archive management.

> ✓ ***Tip***
>
> *The tried and trusted archive file size is between 20 and 100 MB. It must not be larger than 2 GB, otherwise no archive index can be built, thereby prohibiting single document access.*

Deleting Data Objects from the Database

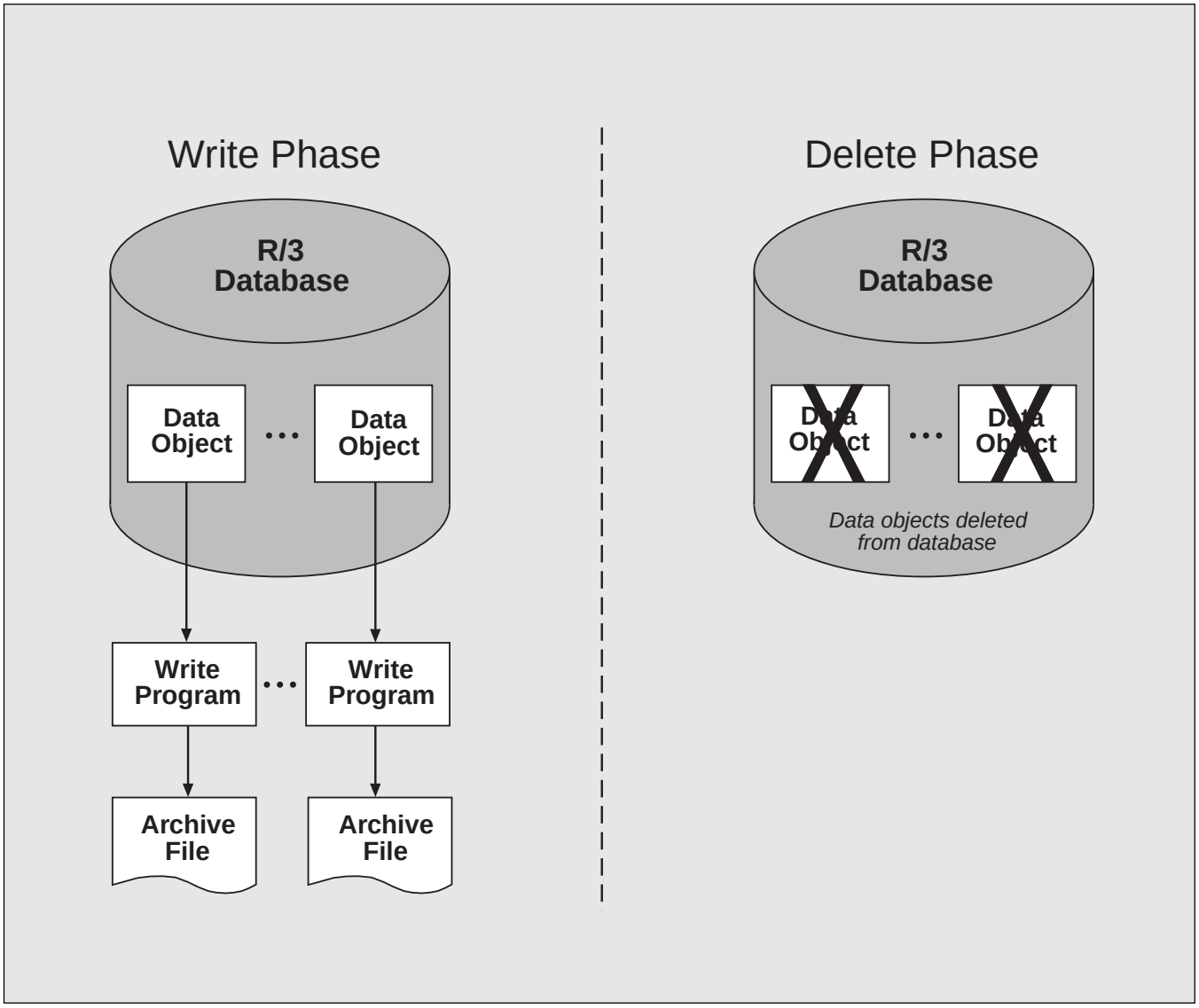
Once all the selected data objects have been copied to archive files, the data objects must still be deleted from the database. The delete program first reads the previously created archive files and then deletes the successfully read data objects. This procedure ensures that no objects are deleted from the database before they have been correctly stored in an archive file. The delete program variants are client-specific, so you should adjust your variant settings for each client that you want to use for archiving sessions.

> ✓ ***Tip***
>
> *As of Release 4.6C, you can, optionally, schedule the delete phase to run after the archive files have been stored in an external storage system. You can make the required setting in archiving object-specific Customizing.*
>
> *Typically, when scheduling the archiving programs for the first time, the adminstrator forgets to maintain completely both test and productive runs for the delete phase. Note particularly that if the productive run variant is not available, no data is deleted from the database when you run the delete program in productive mode.*
>
> *When deleting, remember that if you have set the "Start automatically" option for the delete program in archiving object-specific Customizing, and you select a productive variant for the write program, the delete program follows automatically with its productive variant — which means that the data is deleted from the database following archiving.*

There are two possible delete procedures:

- **Delete jobs are performed sequentially:** In this case, the delete job is scheduled separately after all archive files have been written. The write job is completed as soon as the write program has written all data objects to archive

Figure 4 Archiving Session with Sequential Delete Jobs

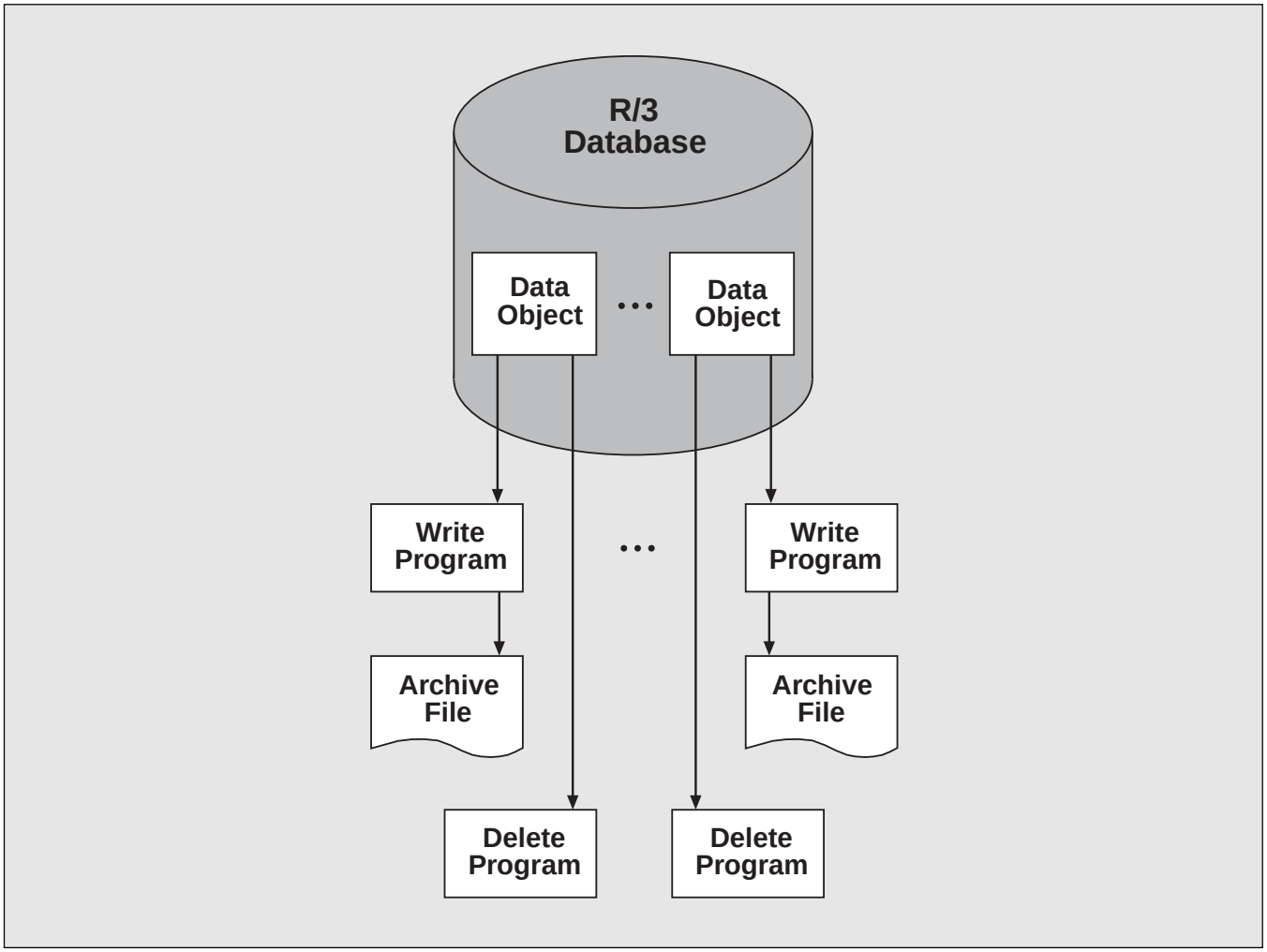


files and has closed the last-created archive file. In the second step, the delete program is run. Here, the previously created archive files are read, and the relevant data objects are deleted from the database. This is illustrated in **Figure 4**.

- **Delete jobs run in parallel:** As soon as the write program has closed the first archive file,[9] the delete program reads the data objects that have just been copied and then deletes from the database those that it could read successfully. The Archive Development Kit runtime system then automatically schedules the relevant delete jobs. Since deleting data usually takes longer than writing individual archive files, there are usually several delete programs active at the same time. As a result, the throughput of data objects is raised and the duration of the archiving session is reduced. **Figure 5** demonstrates how a delete

[9] While the write program continues to write the next archive file.

Figure 5 Archiving Session with Parallel Delete Jobs



job is run although the relevant archive file has been closed, thereby resulting in parallel processing.

Storing Archive Files

In this phase, the archive files, which were created previously by the write program for long-term storage, are stored. In addition to being stored locally on the file system or copied to magnetic tape, it is usually also possible to store them in an HSM (Hierarchical Storage Management) system or in an external storage system. We provide more information on this topic in the section “Decision-Making Criteria for the Storage of Archive Files.”

Steps After Archiving

Once you have archived your data successfully, you should check whether subsequent tasks are required, such as reorganizing the database, updating database statistics, and, in some cases, backing up archive files.

Let’s start with reorganizing the database. Database systems create data records in blocks with a

predefined length. When initially created, data is spread evenly across these blocks. By deleting and updating data records (updating can also change the length of a record), storage space may be freed up that can be used for the storage of new data records. Since data records are generally of differing length, it is possible that the available physical storage space is not being fully used. In time, more and more gaps of varying sizes occur. This process is described as fragmentation. The lost space can be reclaimed by reorganization.

Fragmentation not only affects the blocks in which application data is stored, it also affects database blocks in which index tables are stored. A high level of fragmentation means that more storage space is required (and therefore also more blocks). In some circumstances, this can also affect the time behavior of database operations, since a greater number of blocks must be processed.

In the SAP system, for each application table, there is at least one index table that holds the primary

The Most Commonly Used Archiving Object: FI_DOCUMNT

Financial accounting documents are archived using the archiving object FI_DOCUMNT. This is one of the most commonly used archiving objects in the SAP system.

To ensure that only documents that are no longer required by the online database are archived, the documents must fulfill a series of conditions. The write program checks the documents for archivability at both the header and line item level. Financial accounting documents can only be archived in their entirety — in other words, including all their line items and all assigned change documents and long texts.

Data Model

The FI_DOCUMNT data model provides the basis for an understanding of this archiving object. The physical data model includes the following tables that have been defined in the ABAP Dictionary (transaction SE11):

- BKPF: Contains the document header data
- BSEG: Contains the line item data, RFBLG[10]
- BSET: Contains the document's control data, RFBLG
- BSEC: Contains the suspense account[11] data for each item, RFBLG
- BSED: Contains the document segments for bill of exchange fields, RFBLG
- Tables in which the change documents and long texts are stored
- BSIS, BSAS, BSAD, BSAK, BSIM, BSIP: Secondary indexes for storing document data
- ARCH_IDX: Contains the precise position of a document in the archive

The diagram to the right shows the data model for the archiving object FI_DOCUMNT.

The tables BKPF, BSIS, BSAS, BSAD, BSAK, BSIM, and ARCH_IDX are transparent tables whereas BSEG, BSET, BSEC, and BSED are cluster tables. Since these terms are crucial for an understanding of FI_DOCUMNT, let's have a closer look at what they represent.

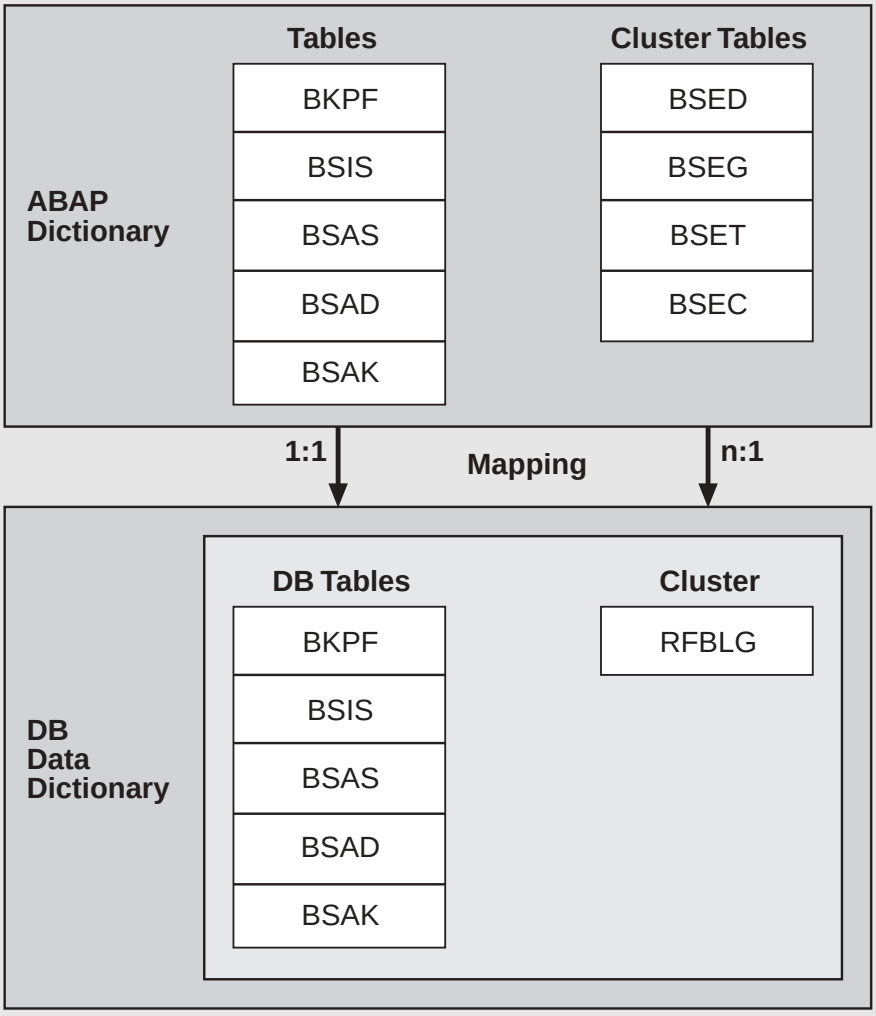
[10] The cluster to which this cluster table belongs.

[11] Collective account for incoming bank transfers that are not for bank customers, and which do not contain any bank details.

key for the application table. Access to a data record usually occurs via a qualified index. For this reason, a high level of fragmentation in the application data usually has no noticeable effect on response times. The fragmentation of index tables is of even greater significance — the greater the fragmentation, the greater the number of blocks that need to be read. This can have a negative effect on time behavior.

Data archiving leads to fragmentation in that gaps occur in the database when mass data is deleted from different areas of the database.

> **✓ *Tip***
>
> *We usually only recommend reorganization if the index tables are badly fragmented. How best to reorganize your database depends on the type of database in use. Before carrying out reorganization, consult your database literature.*



(continued on next page)

To improve data access, many database systems implement a Cost-Based Optimizer (CBO). This stores statistical information on the distribution of data in individual tables. When access is required to a data object, the CBO first assesses the statistical data and then determines the optimal access path. Of course, this presupposes that the CBO's statistics are kept up-to-date — especially following data archiving.

Updating database statistics requires you to do two things:

1. **Check for obsolete or incorrect statistics.** Run this check regularly — ideally, every week. If this is not possible, then run the check after each archiving session. The duration of the check depends on the number and size of the database tables and can take several hours. This check should therefore be carried out at a period of low system load. The results of the check are entered in a control table that serves as the basis for the next step.

2. **Create the statistics.** In this step, you analyze the control table and create new statistics for the tables determined. To do this, you can use either the database administration calendar (transaction DB13) or the tool SAPDBA.

(continued from previous page)

In addition to the ABAP Dictionary, the SAP system also has its own database interface that can be used to access the underlying database. In this context, the term "table" has two meanings:

- A collection of data records sharing the same data structure that is stored in its own database table in the underlying database system.
- In the SAP system, a collection of similar structures that are stored in the database, but not persistently stored in their own database tables.

To distinguish between these, we refer in the first case to transparent tables, and in the second to pool or cluster tables.

Transparent tables are mapped directly on a physical database table of the same name. Their structure in the database is the same as its definition in the ABAP Dictionary. Transparent tables have the following characteristics:

- Direct 1:1 mapping on the underlying database
- External analysis of table contents is possible
- No where-clause restrictions when accessing data using SQL commands
- Definition of views possible
- Multiple database indexes supported

Multiple tables that are defined in the ABAP Dictionary with the same (cluster) key and are mapped on one database table are referred to as **cluster tables**. Records that are stored in a cluster table are stored per data object and have the same cluster key. Cluster tables offer the following advantages:

- Reduced load on database cache due to the lower number of tables
- Reduced load on SAP cursor cache due to the lower number of various SQL instructions
- Compressed storage means less network traffic and lower space requirements in the database

Lastly, there is the issue of backing up archive files. Remember, the archive files are no longer governed by the DBMS of the underlying database. An additional specific administration concept, which includes data backup, must therefore be developed for these files. This concept is not dependent on the type of tertiary storage medium used to store the data. We recommend the following procedure:

1. Replicate the data objects to the archive.
2. Create a backup copy.
3. Run delete jobs in the SAP system.

Factors Affecting the Archiving Runtime

It is not possible to give specific figures regarding the runtime of an archiving session. The runtime depends, however, on the following factors:

- Hardware used
- Configuration and parameterization of the database
- Checks run for archivability
- Number of data objects for archiving
- Sequence in which archiving objects are used
- Database size
- System load at time of archiving

- Storage by object in the database = object-based buffering
- Fewer inquiries made to the database system than when using transparent tables
- Improved I/O behavior due to the storage of the data per data object

Cluster tables also have disadvantages:

- Not possible to have further database indexes in addition to the index using primary keys
- No selectivity within the database except with parts of the cluster key
- No transparency for external programs or database tools

In the case of the cluster tables BSEG, BSET, BSEC, and BSED, the data records are stored in a single cluster table, RFBLG, which has a business key that is composed of the document number, company code, and fiscal year.

Making a selection across non-key fields in the cluster RFBLG, such as when searching for all documents relating to a particular vendor, leads to massive performance problems (full table scan), because the “vendor” field is not contained in the database index for cluster table RFBLG. To avoid this, the data for a financial accounting document is stored in additional transparent database tables known as secondary indexes. For example, the table BSAK contains vendor data, the (secondary index) table BSAD contains customer data, and additionally, the secondary index BSAS contains information on general ledger accounts. For each secondary index — which is nothing more than another transparent table — there is at least one database index that can be used to access the data records in the secondary indexes. It is therefore important to remember that secondary indexes are not the same as database indexes.

When a financial accounting document is archived, the name of the archive file created and the location of the document within the archive file is stored in the transparent database table ARCH_IDX, which is also termed an archive index.

In summary, it can be said that the term “index,” in the context of document archiving in financial accounting, can have three meanings: **secondary index**, **database index**, and **archive index**.

(continued on next page)

- Access path to archive files (local, NFS, LAN, WAN)

Which application server you use for your write programs will depend on the release status of your SAP system. Up to and including Release 4.0B, it is not possible to make an explicit server selection. The scheduler only checks that the database server has an R/3 instance that is configured with background processes, and schedules the write programs there, provided the background processes are free. If there is no R/3 instance on the database server with free

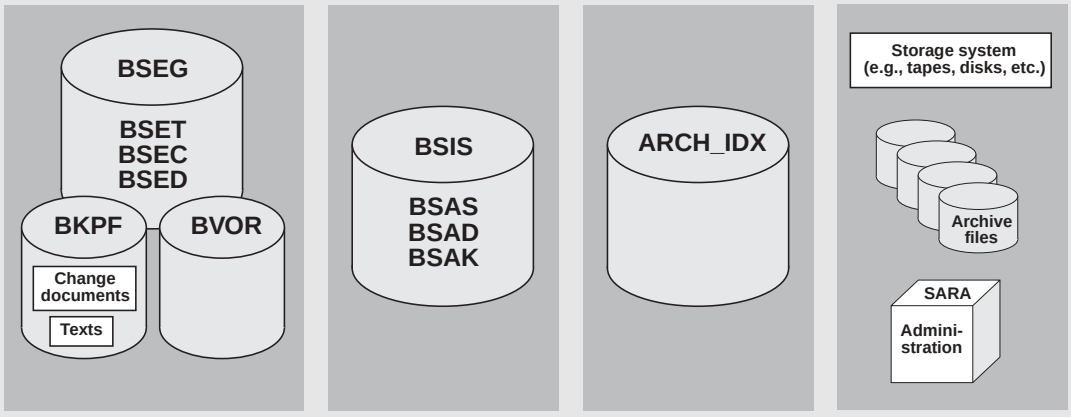
(continued from previous page)

Archiving Financial Accounting Documents

Archiving financial accounting documents means removing and deleting entries from the primary tables BKPF and BVOR and the aforementioned cluster tables. The entries in the tables BKPF, BVOR, and RFBLG are physically deleted in the database. Even when you archive the financial accounting documents themselves, you can still leave the secondary indexes and the archive index in the database. The secondary indexes can be deleted or rebuilt separately. If they are rebuilt, archived data can also be accessed.

Building and deleting secondary indexes is carried out by a post-processing program that you can run directly after archiving. This is shown graphically in the following diagrams.

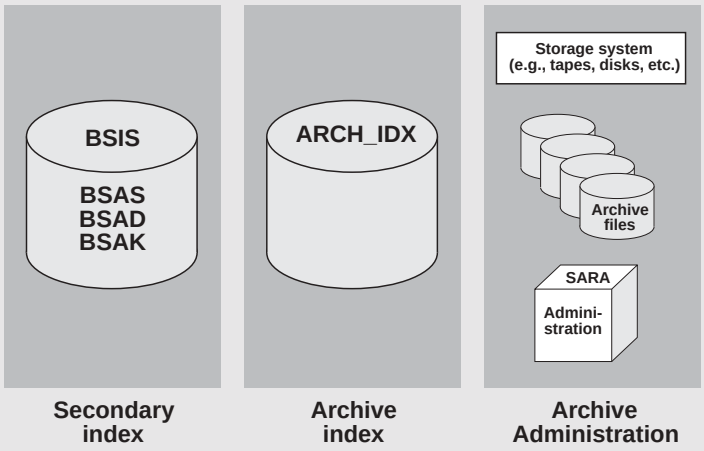
The first of these, shown below, shows the status before archiving. You can see that there are entries in both primary table BKPF and the cluster tables as well as in the secondary tables. The secondary indexes are created automatically when a document is posted or manually by running report SAPF048S. Data from the secondary indexes is not written to the archive file. The archive index ARCH_IDX is not filled.



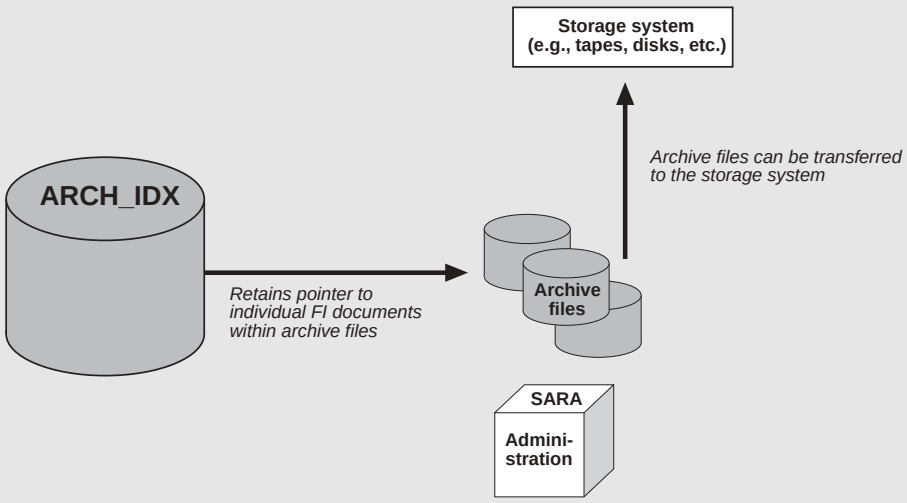
After archiving, as shown in the next diagram at the right, all entries from the primary tables, including the change documents and long texts, are deleted. The corresponding data is now in archive files. During archiving, the archive index is automatically built in the database. The secondary indexes are not deleted — they remain in the database and can be used for reporting.

background processes, the write programs are scheduled on (any) application server. It is therefore important that all application servers on which background processes are configured have access to the directory in which the archive files were created. The delete programs can, as a rule, be scheduled on all application servers on which background processes are run.

From Release 4.5A, you can set up a dedicated application server for the archiving programs. You make these settings in Customizing.



In transaction SARA, you can schedule the post-processing program to delete the secondary indexes automatically (according to the current Customizing settings). This runs the index build program SAPF048I, which, optionally, you can also use to delete the archive index. The diagram below shows the post-processing status.



(continued on next page)

> ✓ *Tip*
>
> *If the database server has sufficient free resources, you should configure an R/3 instance on it with background processes for data archiving. This has the advantage of reducing the LAN network load, and leads to better utilization of resources on the database server.*

Decision-Making Criteria for the Storage of Archive Files

Once the application data has been removed from the database and saved in archive files, the archive files can be stored in a local file system, a Hierarchical Storage Management (HSM) system, or an external storage system. In this section we examine the pros and cons of each approach.

(continued from previous page)

Checks/Customizing

The write program checks the archivability of a financial accounting document at both the document header and line item level. If any one of the archivability criteria is not fulfilled, the whole document is regarded as unarchivable.

The following archivability criteria apply to the **document header**:

- The document type life must have been exceeded.
- The document must have been in the system for at least the minimum number of days.

In addition to the document type life and account type life, a document must also have exceeded the minimum retention period so that it can be archived. This means that the period between the creation or change data of a document and the entered archiving key date must be greater than the minimum retention period.

The screen shot at the right shows the Customizing interface for setting the document type life. It shows the minimum number of days that the document has to be in the database since posting. At the archiving stage, the system checks whether this value has been reached for each document. On this screen, you can also enter the archive index life for document types. This specifies how long the archive indexes are to be kept in the database.

The following archivability criteria apply to the **line items**:

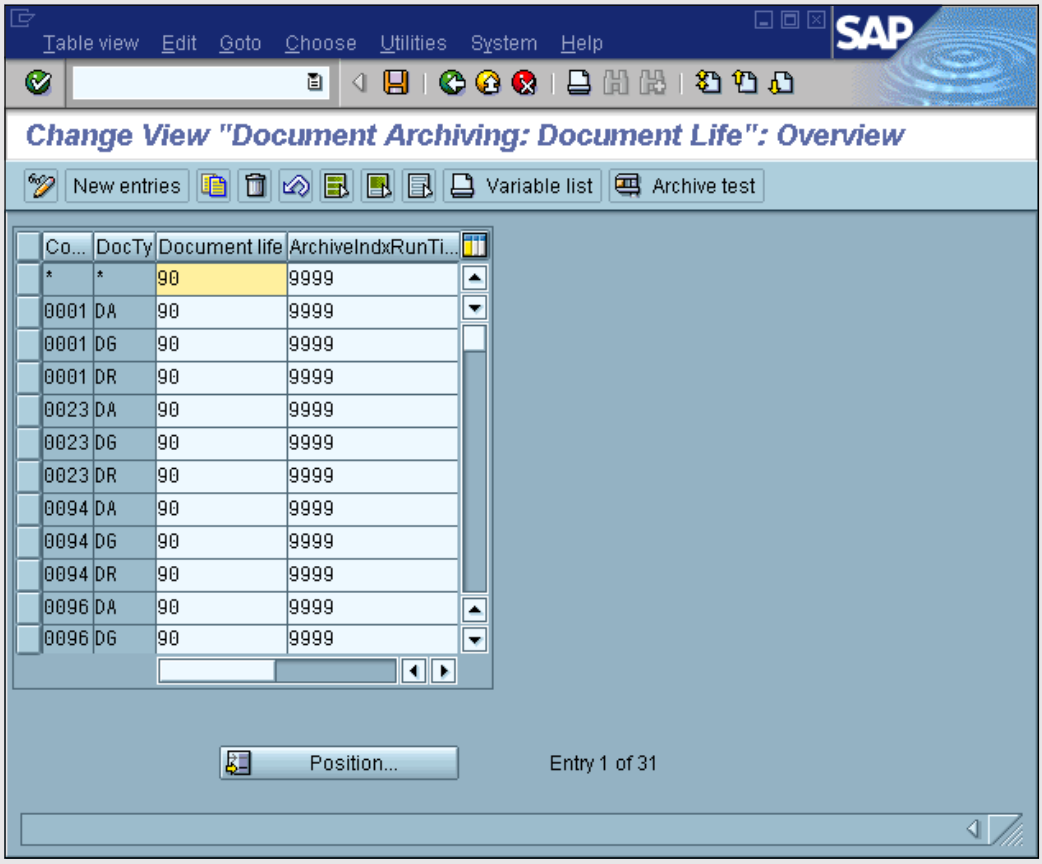
- The document must not have any open items. The system only considers settled items or those without open item management.
- The account type life must be exceeded. You can specify a minimum life in days in the system with reference to a specific company code, account type, or account number range. When you archive documents, the system checks that the minimum life has been exceeded.

Storage in a File System

When using this type of storage, you must ensure that all application servers have read access to the file system. Application servers that run write programs also require the relevant write authorization to create and write the archive files.

Storage in a file system is advisable if no storage system or HSM system is (or will be) in use. When you consider that the compression factor for archive files is between 2 and 5, and that hard drive storage is reasonably inexpensive, this type of storage certainly represents good value for the money, plus it offers the fastest access times. Simply by adding new file systems, you can have an infinitely scalable solution.

The downside to this option is that files in the file system are no longer governed by the DBMS. Consequently, you must elaborate a specific concept for the administration of your archive files.



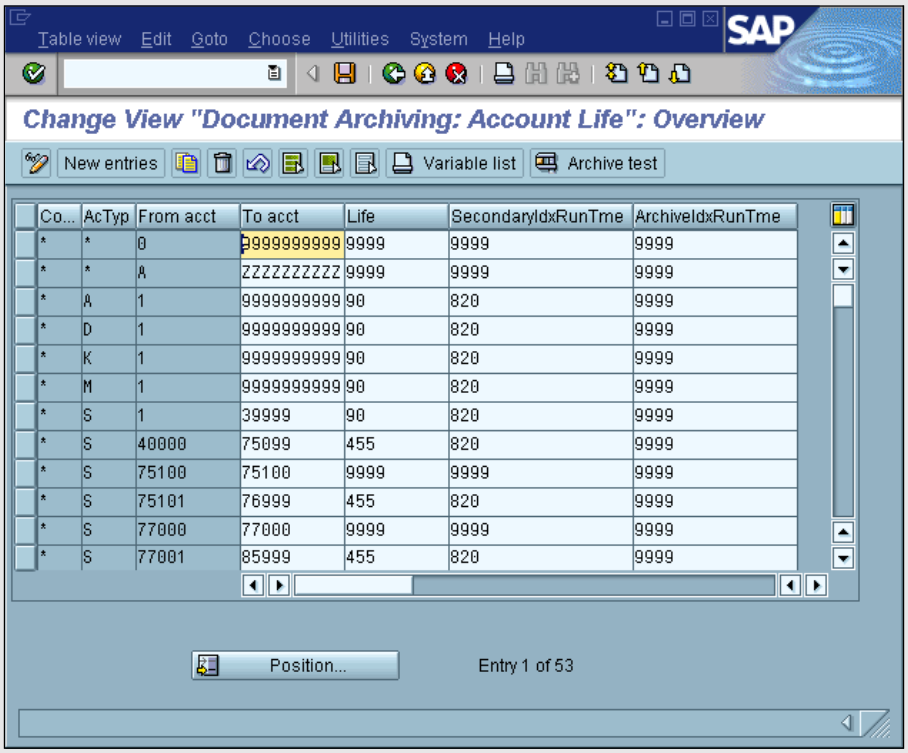
(continued on next page)

Storage Using an HSM System

Hierarchical Storage Management (HSM) systems provide a further option for storing archive files. These offer a complete solution comprising hardware and software that automatically migrates files in line with a specified storage hierarchy from fast but expensive storage media (such as hard disks) to slower, less expensive storage media (such as tape). Conversely, if the data is accessed more frequently at a later stage, the system will move the data to a higher level in the hierarchy, which provides faster access.

(continued from previous page)

The screen shot below shows the Customizing interface for setting the account type life.



In this step, you specify the minimum life for accounts in days. When you archive documents, the system checks whether this minimum life has been exceeded.

You can also define the index life. The system contains the following indexes for financial accounting:

- Secondary indexes
- Archive indexes

The secondary indexes for general ledger, customer, and vendor accounts contain information on the document in parallel with the actual document data. You can now use the secondary index life to define how many days a secondary index is to remain in the system with reference to the clearing date and archiving key date. This means that the secondary indexes are not deleted immediately after archiving, but at the end of the period you have specified. You can run the delete program automatically after the archiving session or manually as a post-processing program.

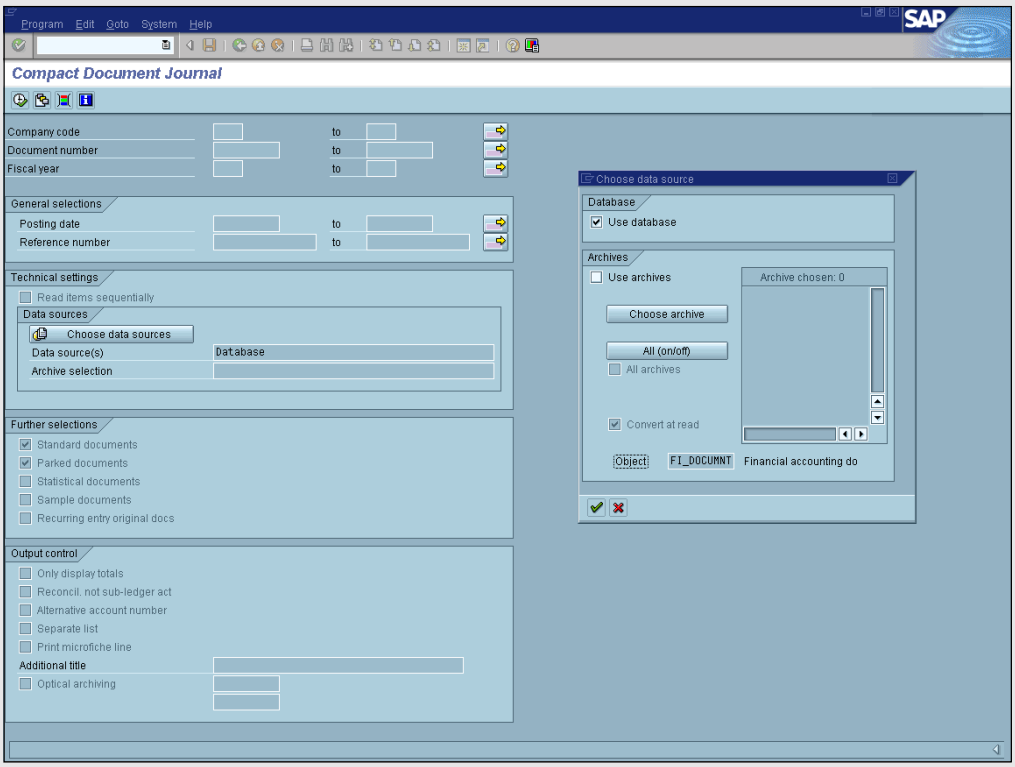
For the migration of files to various storage levels, the system usually uses an LRU (least recently used) algorithm. This means that files that were not accessed over a long period are relegated to the lowest level in the storage hierarchy.

From the application viewpoint, this offers a transparent, "infinitely" large file system to which the files for storage can be transferred simply. The principal advantage for the user is that files, after storage in a file system of this kind, can always be recalled with the same name. Although the file may have

For example, let's say the archiving key date is 1 April 1996, and the secondary index life is 60 days. The posting date for document A is 15 March 1996, and for document B it is 15 January 1996. The secondary index for document A is not deleted at the time of archiving, whereas the secondary index for document B is deleted.

Access to Archived Financial Accounting Documents

In order that you can access archived financial accounting documents, a range of programs is available, enabling you to access and analyze both online and archived documents. A prime example is the Compact Document Journal (RFBELJ00), shown in the screen shot below, which lists the most important header and item data for selected documents. The Compact Document Journal is based on the read program BRF (logical database), which is also used by other read functions. Archived documents can also be displayed in the original display transaction FB03. In this transaction, the archived data is formatted temporarily and displayed in the same way as in the online database.



(continued on next page)

been moved to tape (because, for example, the file was not accessed over a long period), the name of the file remains the same, ensuring that the file can be accessed at any time. Access times are slightly poorer than with a local file system, even if the HSM system has a read-ahead mechanism.

(continued from previous page)

You can determine the data source in the dialog box shown in the previous screen shot. To call up this dialog box from the Compact Document Journal main screen, click "Choose data sources" in the frame "Technical settings." You are now able to choose particular archive files for processing, or select/deselect all archive files.

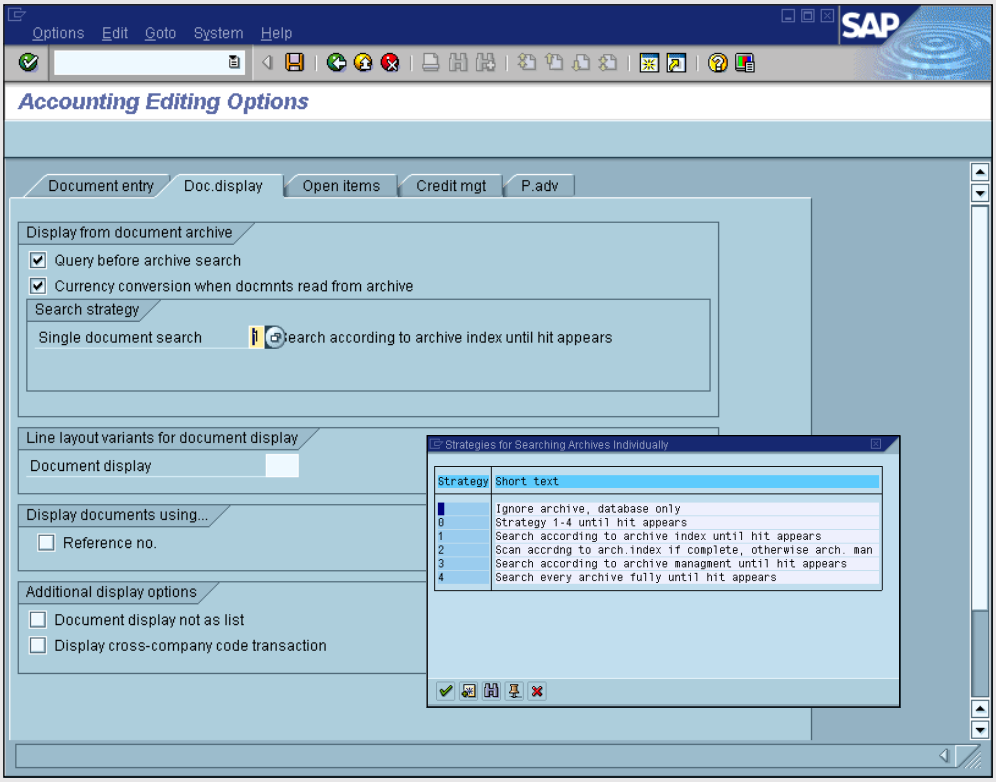
Search Strategy for Archived Financial Accounting Documents

You can search for archived financial accounting documents according to various search strategies, which you can set up in Customizing transaction FB00 (see the screen shot below):

- Search database only, not in archive (no strategy set up)
- Search in archive according to specific rules (strategy 0, 1, 2, or 3 set up)
- Search all archives (strategy 4 set up)

✓ Tip

If the search for an archived financial accounting document takes too long, you may have selected the wrong search strategy in Customizing. The program may be carrying out a sequential search across all archives (strategy 4). Check, and, if necessary, amend your search strategy.



The downside to this option is that HSM systems are relatively expensive on account of the range of technology that they use. In what circumstances is the additional investment worthwhile?

You should consider a storage solution of this kind if, in addition to your archive files, you have to manage other large data volumes. Using an HSM system as your only data storage medium is only advisable if multiple SAP systems are in operation, and the volume of data for storage exceeds 1 TB.

Storage in an External Storage System

Archive files that were created using SAP Data Archiving can be transferred to a connected storage system,[12] while the ADK controls the data transfer. This means that the administrator has a better overview of the storage jobs. It also offers a greater degree of control over the entire storage phase. For administration purposes, the properties of the storage medium used remain completely transparent. Conversion to a different storage medium is possible at any time. Data that was written to logical archives using SAP ArchiveLink® also remains easily accessible. The logical archives that — before Release 4.6C — could only be assigned indirectly to an archiving object using the document type, can now be accessed directly using content repositories of the same name. From Release 4.0, access times for direct access are similar to those when using an HSM system.

Virtually all the external storage systems available on the market are document management systems that you can also use for storing your archive files. However, we only recommend that you use a system of this kind if you already use one for document archiving (imaging).

Since Release 4.6C, SAP offers an integrated Content Management Service (CMS) for the storage of archive files. The CMS is a part of mySAP Technology's cross-media knowledge infrastructure, and is compatible with a broad range of storage systems (including various storage media).

[12] Storage systems are sometimes also referred to as content management systems.

> ✓ ***Tip***
>
> *To store the archive files in the correct content repository, you only have to enter the name in archiving-object-specific Customizing. Here, you can also specify whether the files are to be stored automatically or manually, and whether storage occurs before or after the delete phase.*

Archive files are stored and managed independently of the archiving functions in the SAP system. This means you are free to choose any storage system on the market that suits your needs. However, there are suppliers of storage systems that offer an SAP-certified connection to the SAP system. You can find a complete list of SAP-certified solutions in the SAP Service Marketplace at **http://service.sap.com/CSP** (Complementary Software Program).

Conclusion

Archiving application data is an important part of the administration of rapidly growing data volumes. It provides a means for restricting data growth in the long term and addresses the need for long-term data retention. Rather than tackling problems that result from mass data growth as they arise, SAP Data Archiving ensures the continued smooth running of the SAP system.

SAP Data Archiving is to be seen as an integral part of a cross-system data prevention concept. Data should be archived only after all other means, such as data prevention and data summarization, have been exhausted.

Figure 6 *Data Archiving and Data Prevention*

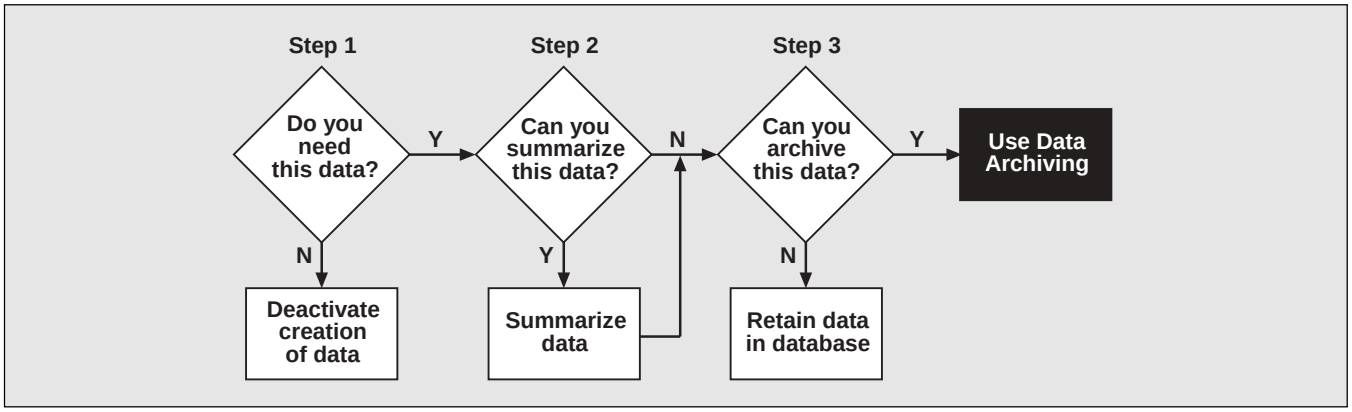


Figure 6 illustrates the decision-making process. It shows that data archiving is not the only way to limit data growth. Before archiving application data, check whether there are more appropriate methods. If you know that the system generates data that is not required, you can deactivate it — e.g., spool data, runtime statistics, or batch input administration data — and if it has already been created, you can delete it. In some cases, you can summarize information at a higher level, for example, by totals formation. Provided this does not lead to the loss of crucial information, this is also a means of keeping data volumes under control over a longer period.

Unlike backup and recovery, the data archiving process is closely related to the application, and has immediate repercussions on an enterprise's business processes. When data is archived, it is no longer available from the online database. Thus, archived data can be accessed, but not changed. Only data that the application no longer requires should be archived. Storing application data in archive files is not an alternative to keeping data in the database. All data that is still required by the application should remain in the database. If, for example, all billing documents for a specific fiscal year are required for rebate processing, the data should not be archived until the rebate has been calculated and established.

As more business processes are handled electronically, resulting in ever-increasing data volumes, data prevention and archiving must be tackled.

Dr. Rolf Gersbacher joined SAP in 1994, where he was engaged in the implementation of business process reengineering projects using workflow technology. Since 1998 he has been a member of SAP's Performance, Benchmark and Data Archiving (PBA) department, where he is responsible for conducting extensive analysis and has co-written projects in the area of data archiving. Together with Helmut Stefani, he developed the best practices guide "Managing SAP Data Archiving Projects," which has become the vade mecum *for data archiving issues. Rolf can be reached at rolf.gersbacher@sap.com.*

Helmut Stefani joined SAP in 1997 as a member of the Data Archiving Coordination team (now a part of PBA), which coordinates the development of archiving solutions across applications. He holds responsibility for information development in this area, and is co-author of the best practices guide "Managing SAP Data Archiving Projects."

Prior to joining SAP, Helmut was a software localization specialist at Computervision (now PTC), a major producer of CAD/CAM software. He can be reached at helmut.stefani@sap.com.

