



Grid-Tools
The power of test data

Grid-Tools Data Archive

MANAGE LONG TERM DATA GROWTH
AND DATABASE PERFORMANCE



Database archiving is the only effective long-term solution to data growth that ensures optimal performance.

Contents

- [Data Archive For Oracle](#)
- [Data Archive For DB2](#)
- [Data Archive For MySQL](#)
- [Data Archive For SQL Server](#)
- [Data Archive For Sybase](#)
- [Data Archive Filestore Adapter](#)
- [Data Archive ViewDirect Adapter](#)

Data Archive for Oracle

Data Archive proactively manages accelerated data growth caused by the proliferation of new enterprise applications, new technology and legal requirements. Database archiving is the only effective long-term solution to data growth that ensures optimal performance of your mission-critical databases. Data Archive allows continued access to crucial historical data to meet stringent compliance regulations, and can result in significant storage cost savings.

Effective Data Management

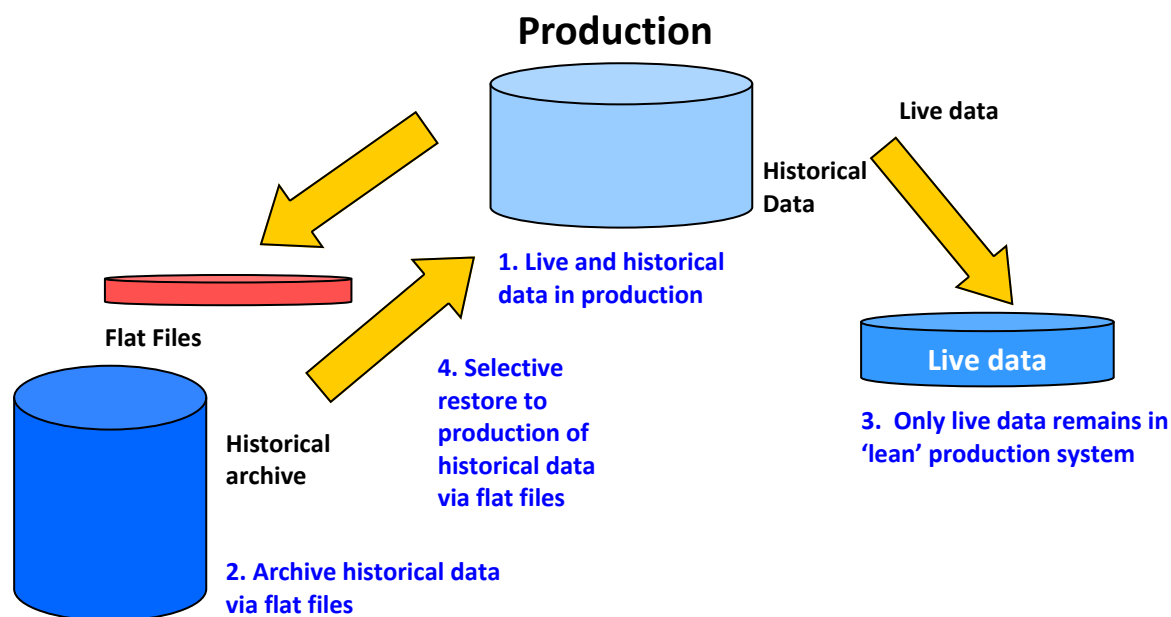
Information Lifecycle Management (ILM) seeks to classify data based on its value to an organisation. Data is assigned to an appropriate storage class, taking into account retention periods, and managed over time as its perceived value diminishes. Data Archive is crucial to any ILM strategy, as it enables databases to be archived with referential integrity intact.

Data Archive for Oracle can be used to:

- **Identify** historical data that fits the criteria for archiving.
- **Archive** inactive or infrequently accessed data to less costly storage media.
- **Maintain** reporting access to archived data, preserving 'transparency' between current and archive data.

Reduce the impact of historical data:

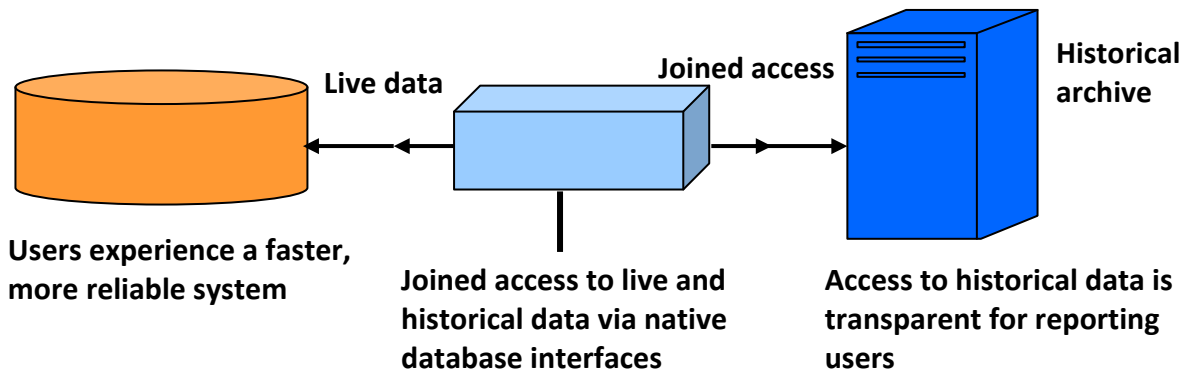
- **Increase** database performance
- **Achieve** savings on storage
- **Improve** enterprise database reliability and availability
- **Remove** redundant data from searches



Product Features

| Feature | Benefit |
|--|--|
| Data cannot be deleted unless the data exists in the history schema. | This ensures no data can be removed by accident. |
| Live and history data can be viewed together. | Reports can use the joined view as required when history data needs to be included. |
| Application transparency | Synonyms, aliases or proxy tables will be created such that applications can be selectively switched to view live and history. Thus allowing reports and inquiry screens to function with no code changes. |
| Data can be restored from history. | If too much data or incorrectly identified data has been moved to history it can be selectively restored. |
| The copy process is separate from the purge. | This allows users to verify the correct data has been archived prior to removal. |
| Data is removed using the joined schema. | This allows for high performance parallel purging of data. |
| Foreign keys relationships dictate which tables are removed first. | Child tables are removed prior to parent tables. |
| The commit frequency of the deletes can be controlled. | This allows performance to be balanced against database resources. |
| If table definitions change in production the same changes are identified and made in history. | This allows the history data to remain at the same release level as production. If data is exported to files and needs to be restored it requires mapping of the old structure onto the production structure every time. |
| The history data is on the same technology stack. | Writing data into a different technology requires that interfaces be maintained from the old to the new to ensure that the history data can be restored at any time. |
| The Data Archive technology can be used as part of your existing archiving strategy. | Data Archive builds command scripts which can easily be incorporated as part of your existing batch processes. |

Transparent access to live and historical data



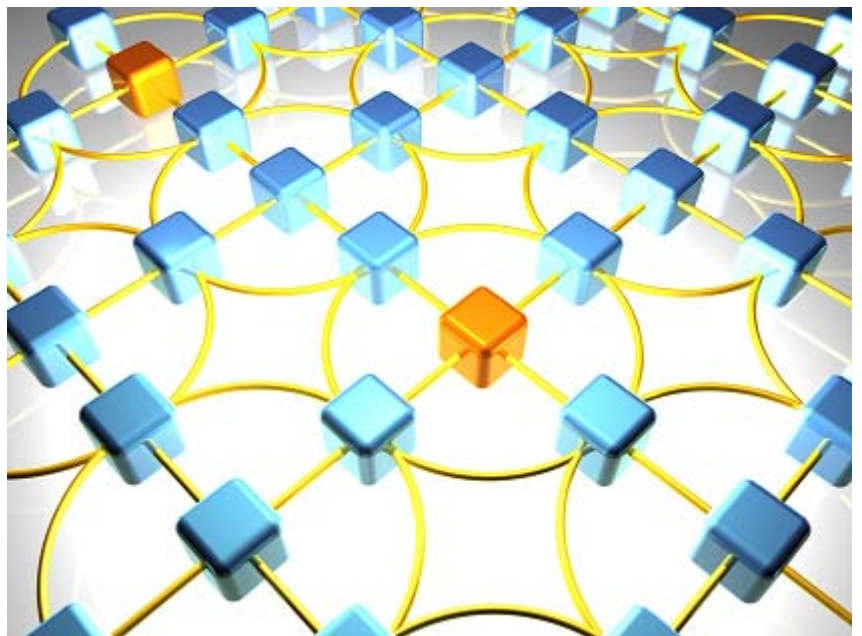
Sensitive to business rules

Data is selected for archiving based on a history of activity and relevant retention policies. To ensure compliance with legal requirements, access to archived data is maintained through a 'joined view' of current and historical data that utilises the native application interfaces already familiar to your users. Archived data can be brought back into the live system, without breaking the integrity of your data, in the unlikely event that this should be required.

System Requirements

Data Archive for Oracle requires:
Oracle version 8i or higher

Supported Environments:
Linux, Unix, and Windows



Data Archive for DB2

Data Archive proactively manages accelerated data growth caused by the proliferation of new enterprise applications, new technology and legal requirements. Database archiving is the only effective long-term solution to data growth that ensures optimal performance of your mission-critical databases. Data Archive allows continued access to crucial historical data to meet stringent compliance regulations, and can result in significant storage cost savings.

Effective Data Management

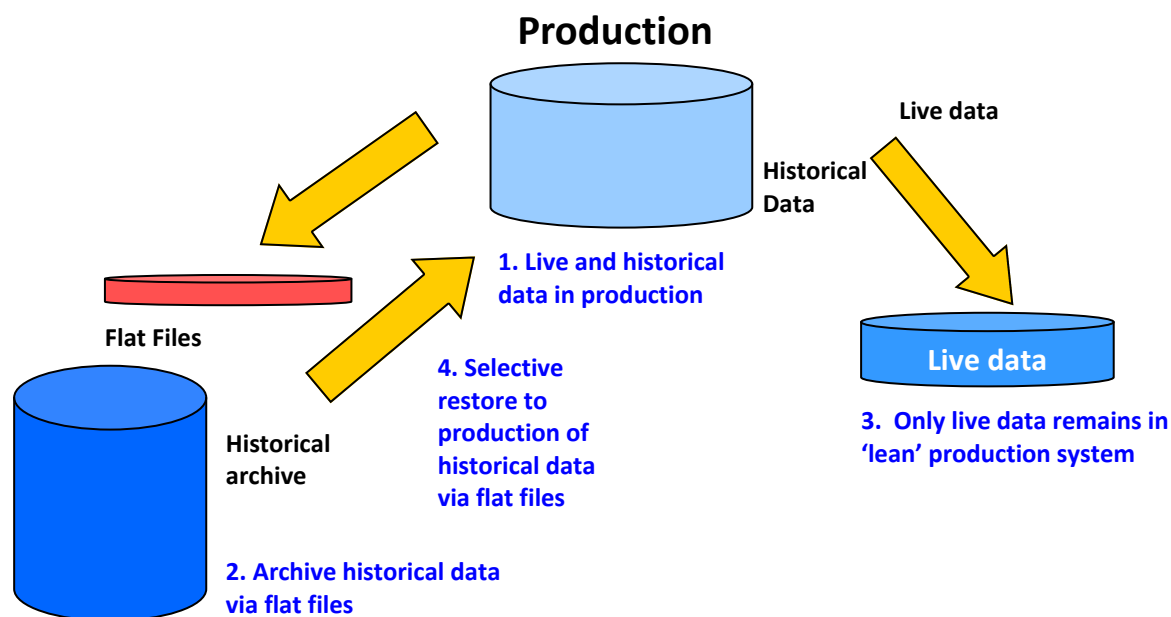
Information Lifecycle Management (ILM) seeks to classify data based on its value to an organisation. Data is assigned to an appropriate storage class, taking into account retention periods, and managed over time as its perceived value diminishes. Data Archive is crucial to any ILM strategy, as it enables databases to be archived with referential integrity intact.

Data Archive for DB2 can be used to:

- **Identify** historical data that fits the criteria for archiving.
- **Archive** inactive or infrequently accessed data to less costly storage media.
- **Maintain** reporting access to archived data, preserving 'transparency' between current and archive data.

Reduce the impact of historical data:

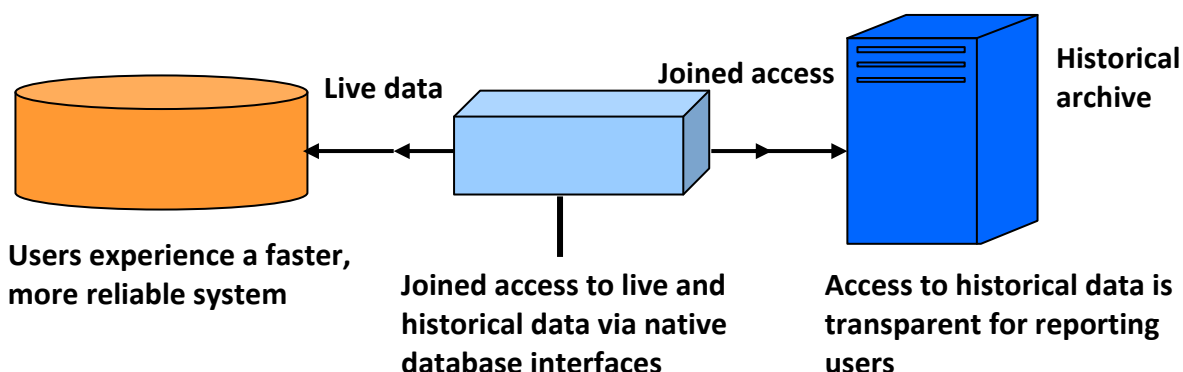
- **Increase** database performance
- **Achieve** savings on storage
- **Improve** enterprise database reliability and availability
- **Remove** redundant data from searches



Product Features

| Feature | Benefit |
|--|--|
| Data cannot be deleted unless the data exists in the history schema. | This ensures no data can be removed by accident. |
| Live and history data can be viewed together. | Reports can use the joined view as required when history data needs to be included. |
| Application transparency | Synonyms, aliases or proxy tables will be created such that applications can be selectively switched to view live and history. Thus allowing reports and inquiry screens to function with no code changes. |
| Data can be restored from history. | If too much data or incorrectly identified data has been moved to history it can be selectively restored. |
| The copy process is separate from the purge. | This allows users to verify the correct data has been archived prior to removal. |
| Data is removed using the joined schema. | This allows for high performance parallel purging of data. |
| Foreign keys relationships dictate which tables are removed first. | Child tables are removed prior to parent tables. |
| The commit frequency of the deletes can be controlled. | This allows performance to be balanced against database resources. |
| If table definitions change in production the same changes are identified and made in history. | This allows the history data to remain at the same release level as production. If data is exported to files and needs to be restored it requires mapping of the old structure onto the production structure every time. |
| The history data is on the same technology stack. | Writing data into a different technology requires that interfaces be maintained from the old to the new to ensure that the history data can be restored at any time. |
| The Data Archive technology can be used as part of your existing archiving strategy. | Data Archive builds command scripts which can easily be incorporated as part of your existing batch processes. |

Transparent access to live and historical data



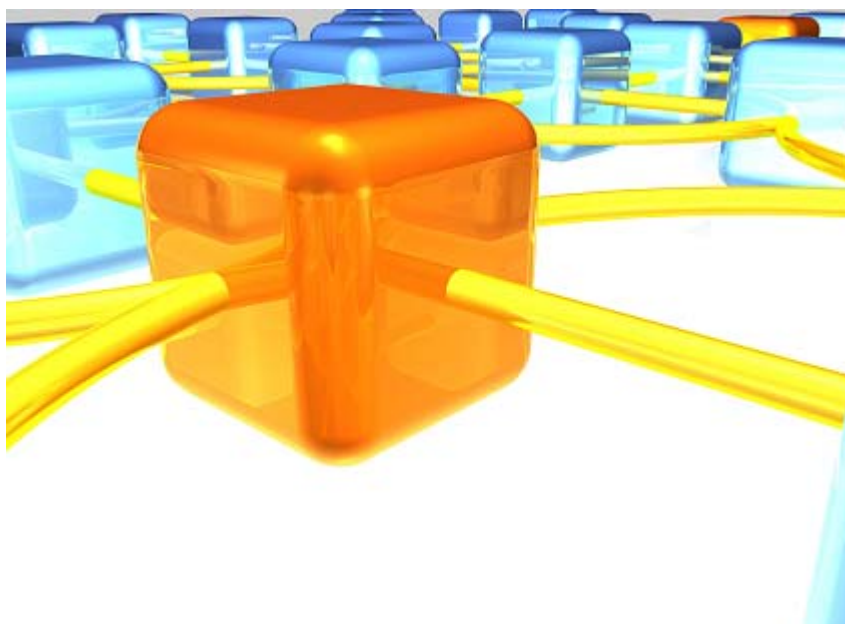
Sensitive to business rules

Data is selected for archiving based on a history of activity and relevant retention policies. To ensure compliance with legal requirements, access to archived data is maintained through a 'joined view' of current and historical data that utilises the native application interfaces already familiar to your users. Archived data can be brought back into the live system, without breaking the integrity of your data, in the unlikely event that this should be required.

System Requirements

Data Archive for DB2 requires:
DB2 version 7 or higher

Supported Environments:
Linux, Unix, and Windows



Data Archive for MySQL

Data Archive proactively manages accelerated data growth caused by the proliferation of new enterprise applications, new technology and legal requirements. Database archiving is the only effective long-term solution to data growth that ensures optimal performance of your mission-critical databases. Data Archive allows continued access to crucial historical data to meet stringent compliance regulations, and can result in significant storage cost savings.

Effective Data Management

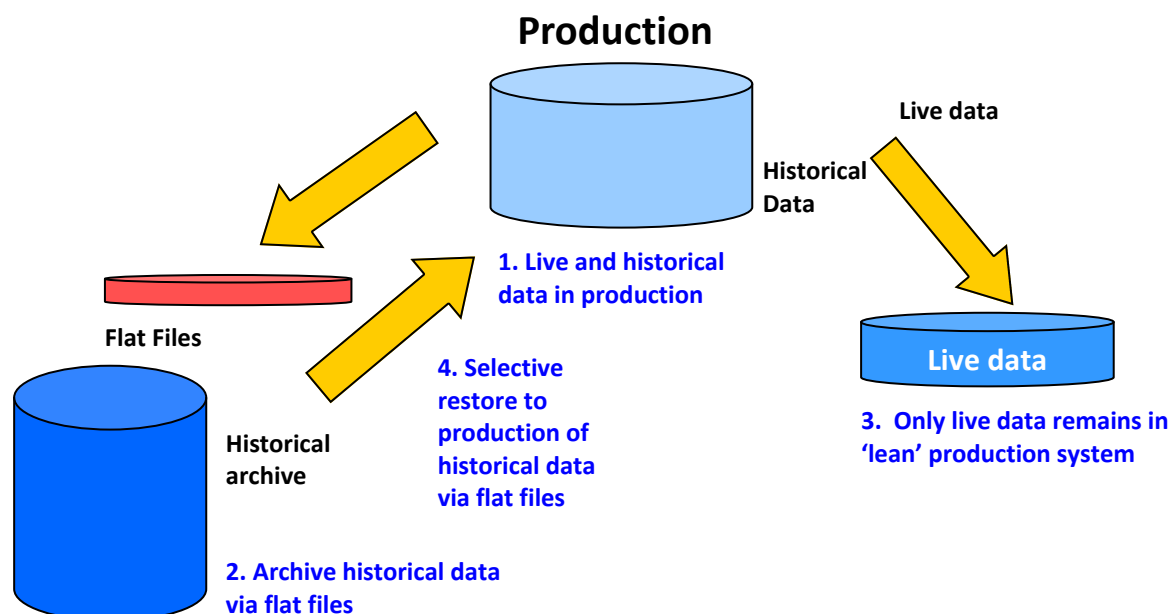
Information Lifecycle Management (ILM) seeks to classify data based on its value to an organisation. Data is assigned to an appropriate storage class, taking into account retention periods, and managed over time as its perceived value diminishes. Data Archive is crucial to any ILM strategy, as it enables databases to be archived with referential integrity intact.

Data Archive for MySQL can be used to:

- **Identify** historical data that fits the criteria for archiving.
- **Archive** inactive or infrequently accessed data to less costly storage media.
- **Maintain** reporting access to archived data, preserving 'transparency' between current and archive data.

Reduce the impact of historical data:

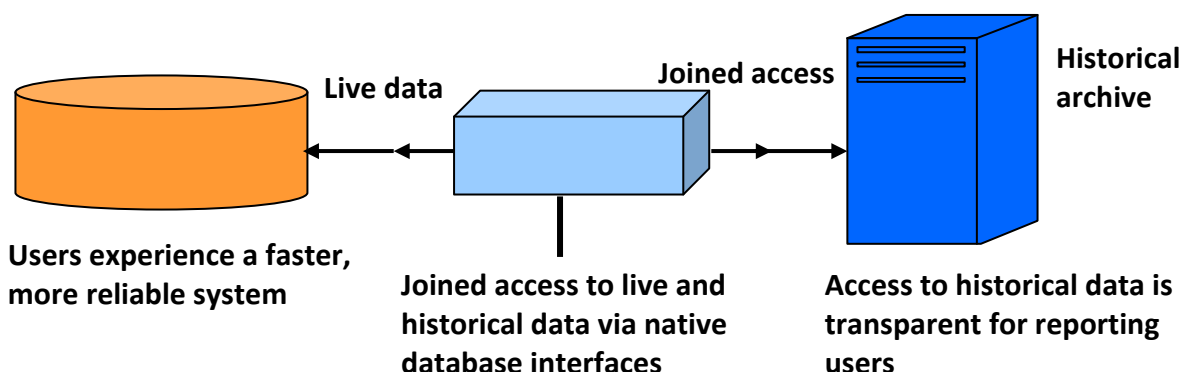
- **Increase** database performance
- **Achieve** savings on storage
- **Improve** enterprise database reliability and availability
- **Remove** redundant data from searches



Product Features

| Feature | Benefit |
|--|--|
| Data cannot be deleted unless the data exists in the history schema. | This ensures no data can be removed by accident. |
| Live and history data can be viewed together. | Reports can use the joined view as required when history data needs to be included. |
| Application transparency | Synonyms, aliases or proxy tables will be created such that applications can be selectively switched to view live and history. Thus allowing reports and inquiry screens to function with no code changes. |
| Data can be restored from history. | If too much data or incorrectly identified data has been moved to history it can be selectively restored. |
| The copy process is separate from the purge. | This allows users to verify the correct data has been archived prior to removal. |
| Data is removed using the joined schema. | This allows for high performance parallel purging of data. |
| Foreign keys relationships dictate which tables are removed first. | Child tables are removed prior to parent tables. |
| The commit frequency of the deletes can be controlled. | This allows performance to be balanced against database resources. |
| If table definitions change in production the same changes are identified and made in history. | This allows the history data to remain at the same release level as production. If data is exported to files and needs to be restored it requires mapping of the old structure onto the production structure every time. |
| The history data is on the same technology stack. | Writing data into a different technology requires that interfaces be maintained from the old to the new to ensure that the history data can be restored at any time. |
| The Data Archive technology can be used as part of your existing archiving strategy. | Data Archive builds command scripts which can easily be incorporated as part of your existing batch processes. |

Transparent access to live and historical data



Sensitive to business rules

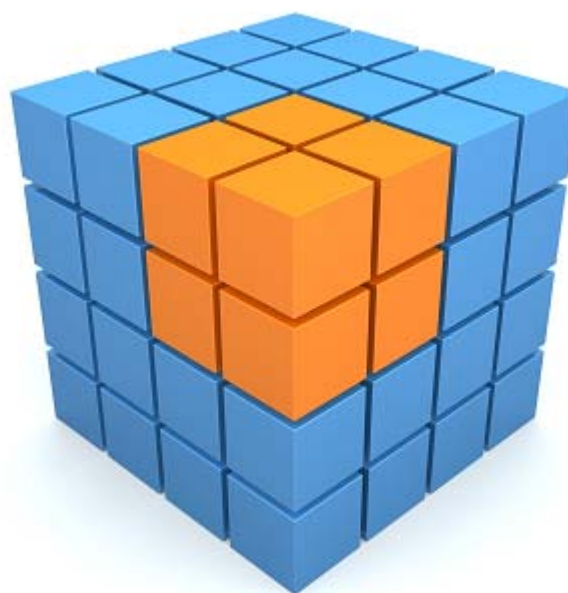
Data is selected for archiving based on a history of activity and relevant retention policies. To ensure compliance with legal requirements, access to archived data is maintained through a 'joined view' of current and historical data that utilises the native application interfaces already familiar to your users. Archived data can be brought back into the live system, without breaking the integrity of your data, in the unlikely event that this should be required.

System Requirements

Data Archive for MySQL requires:
MySQL version 4.1 or higher

Supported Environments:

Linux, Unix, and Windows



Data Archive for SQL

Data Archive proactively manages accelerated data growth caused by the proliferation of new enterprise applications, new technology and legal requirements. Database archiving is the only effective long-term solution to data growth that ensures optimal performance of your mission-critical databases. Data Archive allows continued access to crucial historical data to meet stringent compliance regulations, and can result in significant storage cost savings.

Effective Data Management

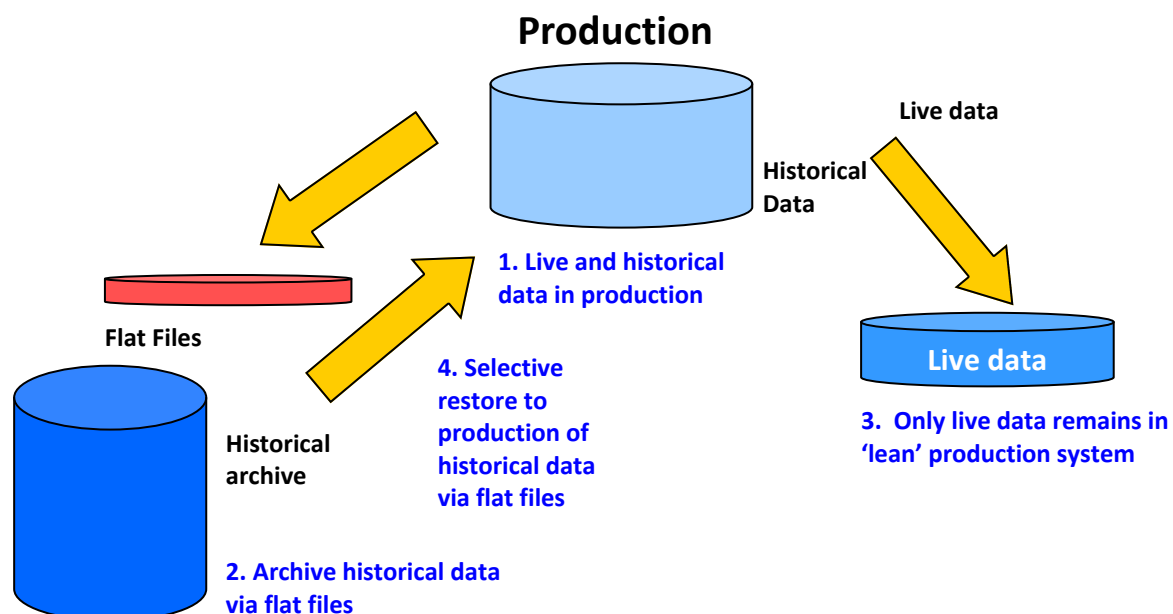
Information Lifecycle Management (ILM) seeks to classify data based on its value to an organisation. Data is assigned to an appropriate storage class, taking into account retention periods, and managed over time as its perceived value diminishes. Data Archive is crucial to any ILM strategy, as it enables databases to be archived with referential integrity intact.

Data Archive for SQL can be used to:

- **Identify** historical data that fits the criteria for archiving.
- **Archive** inactive or infrequently accessed data to less costly storage media.
- **Maintain** reporting access to archived data, preserving 'transparency' between current and archive data.

Reduce the impact of historical data:

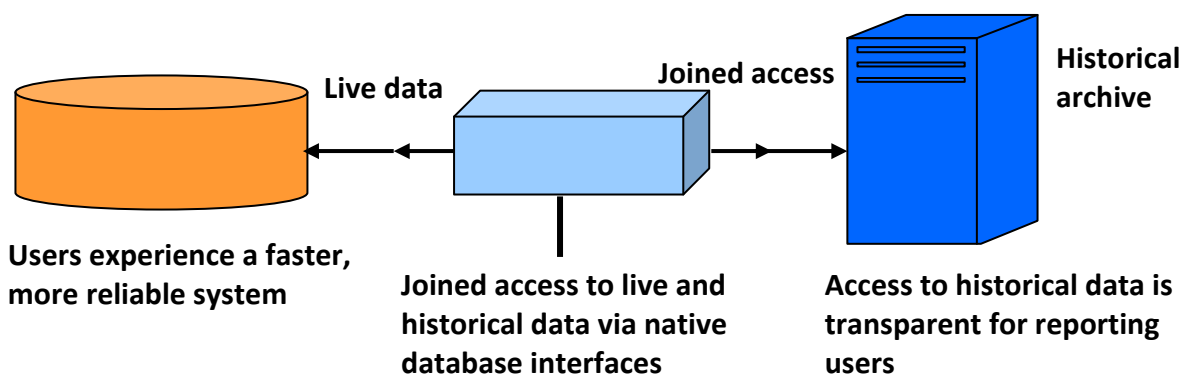
- **Increase** database performance
- **Achieve** savings on storage
- **Improve** enterprise database reliability and availability
- **Remove** redundant data from searches



Product Features

| Feature | Benefit |
|--|--|
| Data cannot be deleted unless the data exists in the history schema. | This ensures no data can be removed by accident. |
| Live and history data can be viewed together. | Reports can use the joined view as required when history data needs to be included. |
| Application transparency | Synonyms, aliases or proxy tables will be created such that applications can be selectively switched to view live and history. Thus allowing reports and inquiry screens to function with no code changes. |
| Data can be restored from history. | If too much data or incorrectly identified data has been moved to history it can be selectively restored. |
| The copy process is separate from the purge. | This allows users to verify the correct data has been archived prior to removal. |
| Data is removed using the joined schema. | This allows for high performance parallel purging of data. |
| Foreign keys relationships dictate which tables are removed first. | Child tables are removed prior to parent tables. |
| The commit frequency of the deletes can be controlled. | This allows performance to be balanced against database resources. |
| If table definitions change in production the same changes are identified and made in history. | This allows the history data to remain at the same release level as production. If data is exported to files and needs to be restored it requires mapping of the old structure onto the production structure every time. |
| The history data is on the same technology stack. | Writing data into a different technology requires that interfaces be maintained from the old to the new to ensure that the history data can be restored at any time. |
| The Data Archive technology can be used as part of your existing archiving strategy. | Data Archive builds command scripts which can easily be incorporated as part of your existing batch processes. |

Transparent access to live and historical data



Sensitive to business rules

Data is selected for archiving based on a history of activity and relevant retention policies. To ensure compliance with legal requirements, access to archived data is maintained through a 'joined view' of current and historical data that utilises the native application interfaces already familiar to your users. Archived data can be brought back into the live system, without breaking the integrity of your data, in the unlikely event that this should be required.

System Requirements

Data Archive for SQL requires:

SQL server 2000 or higher

Supported Environments:

Linux, Unix, and Windows



Data Archive for Sybase

Data Archive proactively manages accelerated data growth caused by the proliferation of new enterprise applications, new technology and legal requirements. Database archiving is the only effective long-term solution to data growth that ensures optimal performance of your mission-critical databases. Data Archive allows continued access to crucial historical data to meet stringent compliance regulations, and can result in significant storage cost savings.

Effective Data Management

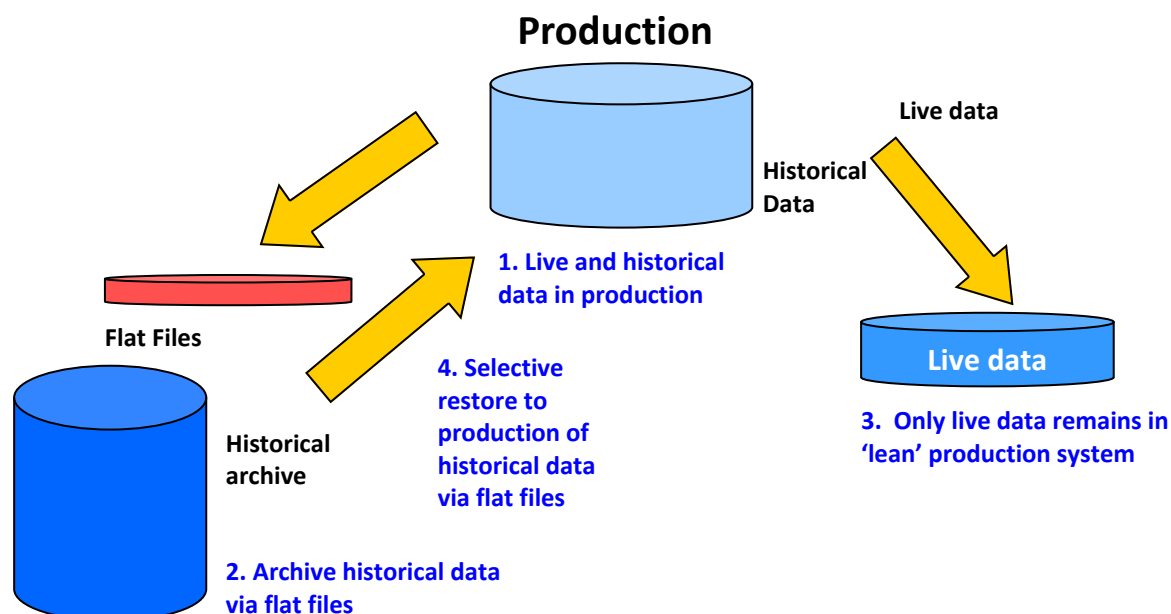
Information Lifecycle Management (ILM) seeks to classify data based on its value to an organisation. Data is assigned to an appropriate storage class, taking into account retention periods, and managed over time as its perceived value diminishes. Data Archive is crucial to any ILM strategy, as it enables databases to be archived with referential integrity intact.

Data Archive for Sybase can be used to:

- **Identify** historical data that fits the criteria for archiving.
- **Archive** inactive or infrequently accessed data to less costly storage media.
- **Maintain** reporting access to archived data, preserving 'transparency' between current and archive data.

Reduce the impact of historical data:

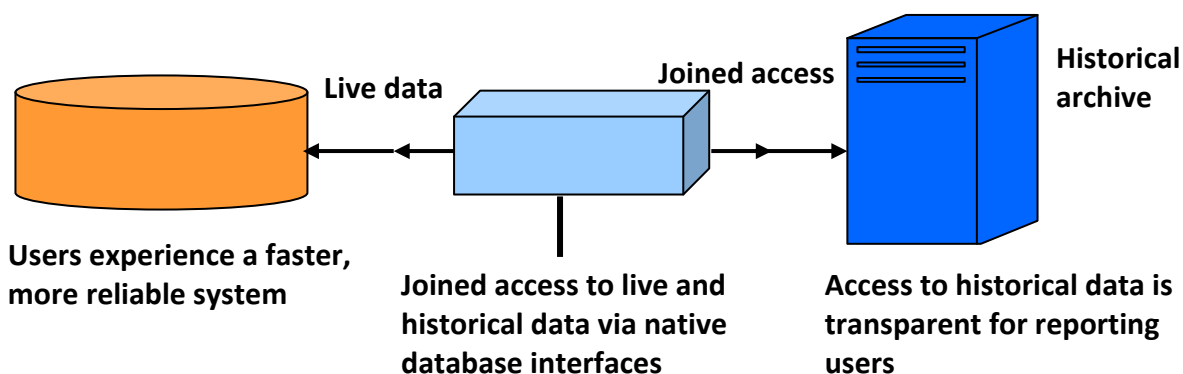
- **Increase** database performance
- **Achieve** savings on storage
- **Improve** enterprise database reliability and availability
- **Remove** redundant data from searches



Product Features

| Feature | Benefit |
|--|--|
| Data cannot be deleted unless the data exists in the history schema. | This ensures no data can be removed by accident. |
| Live and history data can be viewed together. | Reports can use the joined view as required when history data needs to be included. |
| Application transparency | Synonyms, aliases or proxy tables will be created such that applications can be selectively switched to view live and history. Thus allowing reports and inquiry screens to function with no code changes. |
| Data can be restored from history. | If too much data or incorrectly identified data has been moved to history it can be selectively restored. |
| The copy process is separate from the purge. | This allows users to verify the correct data has been archived prior to removal. |
| Data is removed using the joined schema. | This allows for high performance parallel purging of data. |
| Foreign keys relationships dictate which tables are removed first. | Child tables are removed prior to parent tables. |
| The commit frequency of the deletes can be controlled. | This allows performance to be balanced against database resources. |
| If table definitions change in production the same changes are identified and made in history. | This allows the history data to remain at the same release level as production. If data is exported to files and needs to be restored it requires mapping of the old structure onto the production structure every time. |
| The history data is on the same technology stack. | Writing data into a different technology requires that interfaces be maintained from the old to the new to ensure that the history data can be restored at any time. |
| The Data Archive technology can be used as part of your existing archiving strategy. | Data Archive builds command scripts which can easily be incorporated as part of your existing batch processes. |

Transparent access to live and historical data



Sensitive to business rules

Data is selected for archiving based on a history of activity and relevant retention policies. To ensure compliance with legal requirements, access to archived data is maintained through a 'joined view' of current and historical data that utilises the native application interfaces already familiar to your users. Archived data can be brought back into the live system, without breaking the integrity of your data, in the unlikely event that this should be required.

System Requirements

Data Archive for Sybase requires:

Sybase version 10 or higher

Supported Environments:

Linux, Unix, and Windows

Filestore Adapter

The Filestore Adapter enables BridgeHead Software's FileStore technology to manage your database archiving. Database archiving is the only effective long-term solution to data growth that ensures optimal performance of mission critical databases and minimal storage costs, with appropriate levels of access to historical data.

Archiving Data

With The Filestore Adapter, archived data is removed from the database and stored on whichever medium is convenient for the organisation. The data may then be queried for reporting purposes and can, where necessary, be restored to a temporary schema or back into the source database. Referential and business intact data is removed from the database to flat files and into FileStore where BridgeHead's Integrated Storage Management (ISM) platform which allows all the data movement services to share access to all the secondary storage devices across the enterprise. The files are automatically indexed using the The Filestore Adapter software as they are archived allowing data to be easily selected for reporting and automatic restore into an RDBMS.

Retrieving Data

Most database users are concerned with the need to restore archived data. The Filestore Adapter provides the user with the ability to seek data using a searchable local database and once the archived data – from a myriad of archive files – has been found, the data may be browsed or restored. These steps enable end users to quickly find data, display it in the most convenient format and to report on it with ease.

Standard Format

The database data is stored in a standard format that allows multiple RDBMS types to archive data to and from FileStore. Data can be archived from one technology and restored into another. The ability to archive data from legacy systems and then restore into newer open technology provides the IT manager with a powerful new toolset.

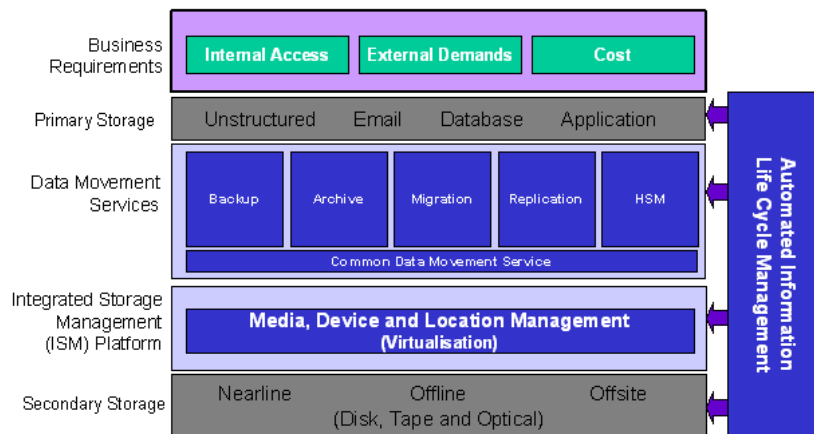
GT Subset Professional Edition: user01@920ipa

Select Schema: USER01 Select Table: CUSTOMER

USER01.CUSTOMER USER01.ORD USER01.ITEM USER01.PRODUCT USER01.PRICE

| PROCID | STDPRICE | MINPRICE | STARTDATE | ENDDATE | END_DATE |
|--------|----------|----------|---------------------|---------------------|----------|
| 100890 | 33.5 | 40.5 | 1995/06/01 00:00:00 | 1995/05/31 00:00:00 | 99C011 |
| 100890 | 53 | 46.4 | 1996/01/01 00:00:00 | | 99C001 |
| 200376 | 2.4 | 1.75 | 1997/11/15 00:00:00 | | |
| 100870 | 2.4 | 1.9 | 1996/01/01 00:00:00 | 1996/12/01 00:00:00 | |
| 100870 | 2 | 2.4 | 1997/01/01 00:00:00 | | |
| 100860 | 30 | 24 | 1996/01/01 00:00:00 | 1996/12/31 00:00:00 | 99C003 |
| 100860 | 23 | 25.4 | 1997/01/01 00:00:00 | 1997/05/31 00:00:00 | 99C002 |
| 100860 | 35 | 26 | 1997/06/01 00:00:00 | | 99C002 |
| 100861 | 39 | 31.2 | 1996/01/01 00:00:00 | 1996/12/31 00:00:00 | J01 |
| 100861 | 42 | 33.6 | 1997/01/01 00:00:00 | 1997/05/31 00:00:00 | J0101 |
| 100861 | 45 | 36 | 1997/06/01 00:00:00 | | 991201 |
| 101060 | 24 | 16 | 1996/02/15 00:00:00 | | |
| 101863 | 12.5 | 9.4 | 1996/02/15 00:00:00 | | |
| 102130 | 2 | 2.8 | 1996/08/18 00:00:00 | | |
| 100871 | 4.8 | 3.2 | 1996/01/01 00:00:00 | 1996/12/01 00:00:00 | J01 |
| 100871 | 5.6 | 4.8 | 1997/01/01 00:00:00 | | |
| 200360 | 4 | 3.2 | 1997/11/15 00:00:00 | | |

Transactions can be encapsulated – all appropriate references to a transaction are included at archive.



Integrated Storage Management (ISM): the basis of real ILM

| Feature | Benefit |
|---|--|
| Data is archived to the appropriate platform according to its value. | Production database performance is enhanced and storage costs are reduced, whilst appropriate access to historical data is maintained. |
| Where necessary archived data can be restored | Specific queries which cannot be handled via ViewDirect can be dealt with by the automatic selective restore of 'encapsulated' transaction data. |
| Archived data is referentially intact | A partial restore, containing only relevant data, is enabled to speed up the process. |
| Archived data can be restored into a new database, which does not need to be of the same type as the original database. | Legacy databases can be retired. |

ISM Platform

The ISM platform which allows all the data movement services to share access to all the secondary storage devices across the enterprise. Its full compliment of integrated data movement services which include backup, file archive, GT Direct database archive, data migration, replication and HSM.

Summary

The Filestore Adapter and FileStore contain all of the functionality that you need to archive from your struggling databases. Ease the data cholesterol in your live systems by removing historical data, with the confidence that you can locate, view, report upon and restore any data in the future – wherever it exists.

ViewDirect Adapter

The ViewDirect Adapter enables the Mobius ViewDirect content management solution to be applied to database archiving. Database archiving is the only effective long-term solution to data growth that ensures optimal performance of mission-critical databases and minimal storage costs, with appropriate levels of access to historical data.

Archiving Data

With The ViewDirect Adapter, archived data is removed from the database and stored on whichever medium is convenient for the organisation. The data may then be queried for reporting purposes and can, where necessary, be restored to a temporary schema or back into the source database. Referential and business intact data is removed from the database to flat files and into Mobius's ViewDirect. The files are automatically indexed using the Mobius EnterpriseIndexing methodology as they are extracted.

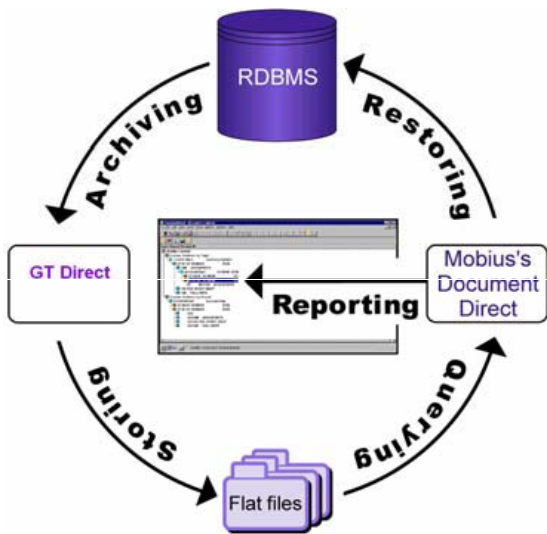


Standard Format

The database data is stored in a standard format that allows multiple RDBMS types to archive data to and from the Mobius Archive. Data can be archived from one technology and restored into another. The ability to archive data from legacy systems and then restore into newer open technology provides the IT manager with a powerful new toolset.

Retrieving Data

Most database users are concerned with the need to restore archived data. ViewDirect from Mobius provides the user with the ability to seek data using the EnterpriseIndexing Technology. Once the archived data – from a myriad of archive files – has been found, the data may be browsed or reported upon using the DocumentDirect interface. The DocumentDirect interface then points to a (number of) archived file(s) containing the relevant database data. Once the file or files have been located, the contents can be viewed, using the DocumentDirect 'viewing policy'. Or restored using GTDirect. These steps enable end users to find data, display it in the most convenient format and to report on it with ease.



Summary

The ViewDirect Adapter and ViewDirect contain all of the functionality that you need to archive from your struggling databases. Ease the data cholesterol in your live systems by removing historical data, with the confidence that you can locate, view, report upon and restore any data in the future – wherever it exists.

| PROCID | STDPRICE | MINPRICE | STARTDATE | ENDDATE | END_DATES |
|--------|----------|----------|---------------------|---------------------|-----------|
| 100890 | 39.5 | 40.5 | 1995/06/01 00:00:00 | 1995/05/31 00:00:00 | 950201 |
| 100890 | 58 | 46.4 | 1996/01/01 00:00:00 | | 990201 |
| 200376 | 2.4 | 1.75 | 1997/11/15 00:00:00 | | |
| 100070 | 2.4 | 1.9 | 1996/01/01 00:00:00 | 1996/12/01 00:00:00 | |
| 100070 | 2 | 2.4 | 1997/01/01 00:00:00 | | |
| 100060 | 30 | 24 | 1996/01/01 00:00:00 | 1996/12/31 00:00:00 | 960203 |
| 100060 | 23 | 25.6 | 1997/01/01 00:00:00 | 1997/05/31 00:00:00 | 990202 |
| 100060 | 35 | 28 | 1997/06/01 00:00:00 | | 920202 |
| 100061 | 39 | 31.2 | 1996/01/01 00:00:00 | 1996/12/31 00:00:00 | 101 |
| 100061 | 42 | 33.6 | 1997/01/01 00:00:00 | 1997/05/31 00:00:00 | 10101 |
| 100061 | 45 | 36 | 1997/06/01 00:00:00 | | 991201 |
| 101060 | 24 | 18 | 1996/02/15 00:00:00 | | |
| 101863 | 12.5 | 9.4 | 1996/02/15 00:00:00 | | |
| 102130 | 2 | 2.8 | 1996/08/18 00:00:00 | | |
| 100071 | 4.8 | 3.2 | 1996/01/01 00:00:00 | 1996/12/01 00:00:00 | 101 |
| 100071 | 5.6 | 4.8 | 1997/01/01 00:00:00 | | |
| 200380 | + | 3.2 | 1997/11/15 00:00:00 | | |

Transactions can be encapsulated – all appropriate references to a transaction are included at archive time.

Restoring Data

Occasionally, a full or partial restore of data files will be necessary. After DocumentDirect has identified the files that the user needs, they can easily be restored into a database for SQL querying, or as a pre-cursor to moving the data back into the live system. All information necessary for such a task is stored within the data files themselves. For example, the DDL (full definitions of the tables involved) is embedded in the flat files so that any differences between the structure of the database tables at the time of archiving and the present can be detected.

Product Features

| Feature | Benefit |
|---|--|
| Data is archived to the appropriate platform according to its value. | Production database performance is enhanced and storage costs are reduced, whilst appropriate access to historical data is maintained. |
| Archived data can be accessed via ViewDirect | Information retrieval is quick and easy. |
| Where necessary archived data can be restored | Specific queries which cannot be handled via ViewDirect can be dealt with by the automatic selective restore of 'encapsulated' transaction data. |
| Archived data is referentially intact. | A partial restore, containing only relevant data, is enabled to speed up the process. |
| Archived data can be restored into a new database, which does not need to be of the same type as the original database. | Legacy databases can be retired. |

