Data Archiving for Adabas

Data Selection Methods

Version 1.7.1

November 2016

This document applies to Data Archiving for Adabas Version 1.7.1.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 2008-2016 Software AG, Darmstadt, Germany and/or Software AG USA, Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA, Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at http://softwareag.com/licenses.

Use of this software is subject to adherence to Software AG's licensing conditions and terms. These terms are part of the product documentation, located at http://softwareag.com/licenses/ and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at http://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

Document ID: ADR-EXSYN-171-20161124

Table of Contents

Data Selection Methods

The first step in any archival process is to identify what determines stale or inactive data. This is often a matter of subjective interpretation of business rules and is therefore best performed by the business lines whose data is under scrutiny.

The next step is to define the collection of records, perhaps spread across multiple databases and files, which relate to the inactive data. Data Archiving for Adabas describes such a collection of records as a business object.

Once such business objects can be identified, the next step is to choose the most efficient means of selecting these objects. Data Archiving for Adabas offers three different data selection methods:

- Extraction Syntax
- User Generated Archive/Transfer Lists (To-Do Lists)
- Record Selection Plug-Ins (User Library)

Extraction Syntax

This method uses a simple scripting language to allow you to select business objects and to perform the archiving/transfer actions to be taken on those objects.

For detailed information refer to the *ADRExtractionSyntaxReference.pdf* document located in the product's sdk/doc directory.

Some simple introductory examples on how to use the syntax can be seen here *Extraction Syntax Examples*.

User Generated Archive/Transfer Lists (To-Do Lists)

This method allows you to select business objects with your own custom applications (written in Natural or C programming language) using the Data Archiving for Adabas *Application Programming Interface* . The application programming interface enables the ability to ‘stage’ selected business objects in a repository file, an Action is then run which references these staged business objects and performs the appropriate archiving /transfer operations. Refer to *Data Selection Mode* for information on how to define an Action to run in To-Do list mode.

Record Selection Plug-Ins (User Library)

This method allows you to create a C programming language plug-in which, in conjunction with the Data Archiving for Adabas *Application Programming Interface*, provides business objects at runtime when requested by the Data Archiving Service Extractor process. Refer to *Data Selection Mode* for information on how to define an Action to run in User Library mode.

1 Extraction Syntax Examples

Example 1: Selecting all records from a single Source File

```
EXTRACT PEOPLE(*)
{
    ARCHIVE MOVE PEOPLE [*];
}
```

In the above example all PEOPLE records are archived.

Example 2: Selecting records from a single Source File

```
EXTRACT PEOPLE(PEOPLE.CITY == "DERBY")
{
    ARCHIVE MOVE PEOPLE [*];
}
```

In the above example all PEOPLE records where the CITY field equals DERBY are archived.

Example 3: Selecting records from a single Source File based on current date

```
EXTRACT PEOPLE( PEOPLE.AH == [TODAY(-50Y,DATX)..TODAY(-40Y,DATX)] )
{
    ARCHIVE MOVE PEOPLE [*];
}
```

In the above example all PEOPLE records which are in the age range 40 – 50 years old are archived. Note the use of the DATX format in the *TODAY()* function to indicate field AH in the PEOPLE file contains the date of birth in Natural D-format.

```
EXTRACT PEOPLE( PEOPLE.AH < TODAY(-50Y,”YYYY-MM-DD”) )
{
    ARCHIVE MOVE PEOPLE [*];
}
```

In the above example all PEOPLE records which are over 50 years old are archived. Note the use of a user defined format to indicate field AH in the PEOPLE file contains the date of birth in a 10 byte Alpha field of format YYYY-MM-DD.

Example 4: Selecting records from multiple related Source Files

```
EXTRACT PEOPLE(PEOPLE.CITY == ["DERBY", "PARIS", "DETROIT"])
{
    ARCHIVE MOVE PEOPLE[*];

    EXTRACT CARS(CARS.PERSONNEL_ID == PEOPLE.PERSONNEL_ID)
    {
        ARCHIVE MOVE CARS[*];
    }

    EXTRACT OTHER(OTHER.PERSONNEL_ID== PEOPLE.PERSONNEL_ID)
    {
        ARCHIVE MOVE OTHER[*];
    }
}
```

In the above example all PEOPLE records where the CITY field equals DERBY, PARIS or DETROIT are archived. In addition, related records (based on PERSONNEL_ID) from the CARS file and OTHER file are also archived.