TAFJ-Read Only Database

R14/R15

10/3/2015

Temenos

Amendment History:

Revision	Date Amended	Name	Description
1	18/12/2013	TAFJ team	Initial version
2	12/2/2914	R. Vincent	R14GA Review
6	10th March 2015	H. Aubert	R15 AMR review
7	1 April 2015	R. Vincent	Add $DIM documentation

Copyright

Errata and Comments

If you have any comments regarding this manual or wish to report any errors in the documentation, please document them and send them to the address below:
Technology Department

Temenos Headquarters SA
2 Rue de l'Ecole-de-Chimie,
CH - 1205 Geneva,
Switzerland

Tel SB: +41 (0) 22 708 1150
Fax: +41 (0) 22 708 1160

Please include your name, company, address, and telephone and fax numbers, and email address if applicable. TAFJdev@temenos.com

Table of Contents

Introduction

TAFJ supports a read-only database feature where certain tables will be archived and stored in a separate database instance. This will enable the logical and physical separation of historical data from the 'live' data hence minimizing the amount of data retained in the live data area. Minimizing the live data will improve the general performance and reduce the non-functional operational overheads required to support the banks daily transactional activities. As well as "lessening the load", a read-only dimensional database can act as a staging area for data warehousing purposes without impact to the transactional database.

TAFJ Readonly files are suffixed with $RO. Dimensional files are suffixed with $DIM

Prerequisites

A read only database must be created and loaded before TAFJ read-only functionality can work.   Likewise, a dimensional database needs to be prepared beforehand. Generally, a set of scripts is run for the typical tables that will be added to the read-only database. This process is not covered in this document, nor is the preparation of a dimensional database.

From a TAFJ perspective, it means that the TAFJ_VOC must be defined correctly. See the section "Underneath the covers."

A database link from the T24 transactional database to each read-only database must exist (created on the transactional database).

TAFJ Properties

Standalone connection

The following TAFJ properties will invoke another two standalone connections, one for the read-only DB TESTRODB and one for the dimensional database TESTDIM. Multiple values are separated by a comma. (Oracle is shown here as an example). The database link name must match the one defined in the database.

```
temn.tafj.jdbc.ro.urls=jdbc:oracle:thin:@localhost:1521/TESTRODB,
jdbc:oracle:thin:@localhost:1521/TESTDIM

temn.tafj.jdbc.ro.drivers=oracle.jdbc.driver.OracleDriver, oracle.jdbc.driver.OracleDriver

temn.tafj.jdbc.ro.usernames=myuser,myuser2

temn.tafj.jdbc.ro.passwords=mypassord,mypassword2
```

temn.tafj.jdbc.ro.dataSourceNames=

temn.tafj.jdbc.ro.dataBaseLinks=TESTRODB,TESTDIMLINK

Even though dimensional databases don't need a database link, one must be specified above in order for the connections to be properly parsed. Above TESTDIMLINK need not actually exist. Any string will suffice. Note that $DIM files DO NOT require a physical database link, but one must be filled out because TAFJ cannot determine what kind of database it is connecting to. It only knows this through the file type ($RO or $DIM).

Application server connection

The following TAFJ properties will invoke another connection from a defined datasource. The datasource below (t24RODataSource) is defined as a reference in the TAFJ EAR file. **If more than one is needed, it might require modifications to the TAFJEAR file. Note that database link again is required, but will only be used for $RO files.**

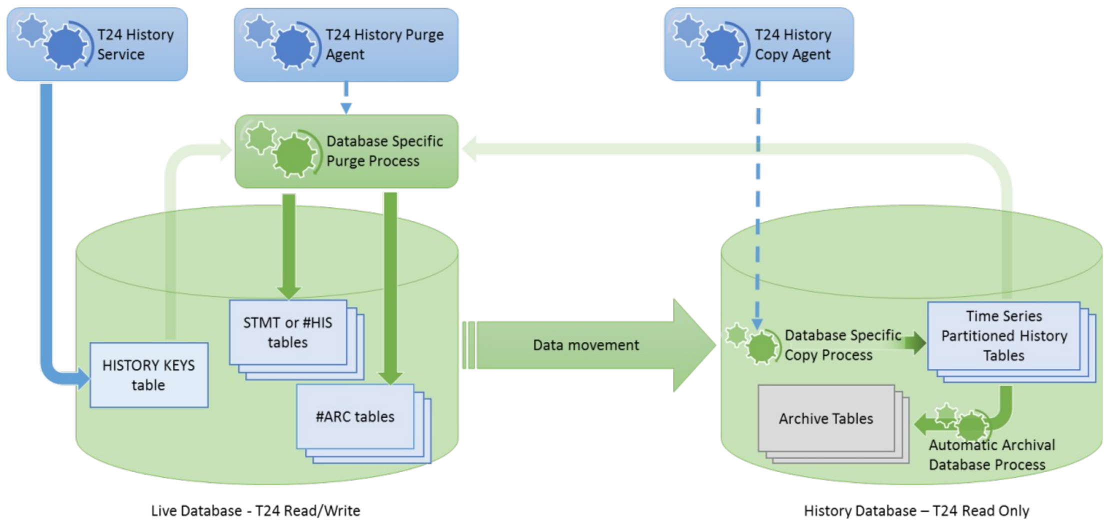
temn.tafj.jdbc.ro.dataSourceNames= java:comp/env/jdbc/t24RODataSource

temn.tafj.jdbc.ro.dataBaseLinks=TESTRODB

When TAFJ is invoked within an application server context, a JNDI lookup will occur on the datasource defined. Therefore, the datasource must also be defined within the particular application server.

Underneath the Covers

The design of the solution appears below from a T24 perspective.


T24 History Service
T24 History Purge Agent
T24 History Copy Agent
Database Specific Purge Process
STMT or #HIS tables
HISTORY KEYS table
#ARC tables
Data movement
Database Specific Copy Process
Time Series Partitioned History Tables
Archive Tables
Automatic Archival Database Process
Live Database - T24 Read/Write
History Database – T24 Read Only

Particular to TAFJ, is the TAFJ_VOC that is a mapping table of T24 internal names to physical tables (not all fields shown). The highlighted fields are particular to a read-only table pair (one exists in the transactional database, one exists in the read-only database)

```
SQL> desc tafj_voc
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 RECID                                     NOT NULL VARCHAR2(255)
 XMLRECORD                                          CLOB
 ISREADONLY                                         VARCHAR2(1)
 ASSOCIATED                                         VARCHAR2(70)
 PDATEALIAS                                         VARCHAR2(70)
```

An example follows:

T24 table F_TABLE is a table that exists in both the transactional and read-only database. A row would exist in the TAFJ_VOC for it, AND the associated read-only table named DB.SCHEMA.F_TABLE#RO (which again exists in the separate read-only table). DB specifies the database in which it is located (local or remote). SCHEMA specifies the schema in which it is located (local or remote)

RECID	TABLENAME	ASSOCIATED	ISREADONLY	PDATEALIAS
F.TABLE	F_TABLE	F.TABLE$RO	N	MYDATE
F.TABLE$RO	DB.SCHEMA. F_TABLE#RO		Y	

A synonym for F_TABLE#RO would have been created in the transactional database for the remote table F_TABLE#RO. This way UNION clauses can be constructed to form an "all rows" queries.

On the read-only database side, F_TABLE#RO would exist with one extra field called PDATE. This field exists as the partition key for very large tables. Since this field doesn't exist on the transactional database, queries that want rows from the read-only side must be rewritten to swap in the PDATE field on the other side. For example:

SELECT RECID FROM F_TABLE WHERE MYDATE = '13-DEC-12' UNION SELECT RECID FROM F_TABLE#RO WHERE PDATE = '13-DEC-12'

Above PDATE would replace MYDATE for performance reasons on the second half of the UNION clause which hits the remote database.

For dimensional files ($DIM) files, they would appear like this:

RECID	TABLENAME	ASSOCIATED	ISREADONLY	PDATEALIAS
F.TABLE$DIM	DB.SCHEMA. F_TABLE#DIM		Y	

$DIM files have no association. They exist on their own. Queries on a $DIM file will use the connection setup to the dimensional database directly.

The workflow for $RO is:

T24 App → READ $RO → TAFJ → Check Live DB → Not Found → Check RO DB → Found → Read

It is the below for $DIM

T24 App → READ $DIM → TAFJ→ Check DIM DB → Found → Read

There are no need for UNION queries with $DIM file either, since they are assumed to exist in a different database anyway.

Creating new tables with the CREATE-FILE command

Two commands are needed to create new T24 archive tables.

```
CREATE-FILE F.TESTROFILE TYPE=XML ASSOCIATE="YES"
```

The command above creates the table in the transactional database and sets it up for a "read-only" pair.

```
CREATE-FILE F.TESTROFILE$RO TYPE=XML DATABASE="TESTRODB"
SCHEMA="TAFJRO" READONLY="YES"
```

The command above finishes the sequence such the two tables are now linked. Assuming the database link has been set up correctly, a synonym would be created on the transactional database to the read-only database. The database targeted is TESTRODB with schema TAFJRO. Ie, a URL like:

```
temn.tafj.jdbc.ro.urls=jdbc:oracle:thin:@localhost:1521/TESTRODB
```

and user like:

```
temn.tafj.jdbc.ro.usernames= TAFJRO
```

must exist in the tafj.properties file for this to work (or a datasource correctly defined if in an application server).

For $DIM files the CREATE-FILE would be as follows:

```
CREATE-FILE F.TESTROFILE$DIM TYPE=XML DATABASE="TESTDIMDB"
SCHEMA="TAFJRO" READONLY="YES"
```

This will create the F.TESTROFILE$DIM table on the database TESTDIMDB in the schema TAFJRO. The view to the table will exist in TESTDIMDB. TAFJRO as well.