Solution-Soft®

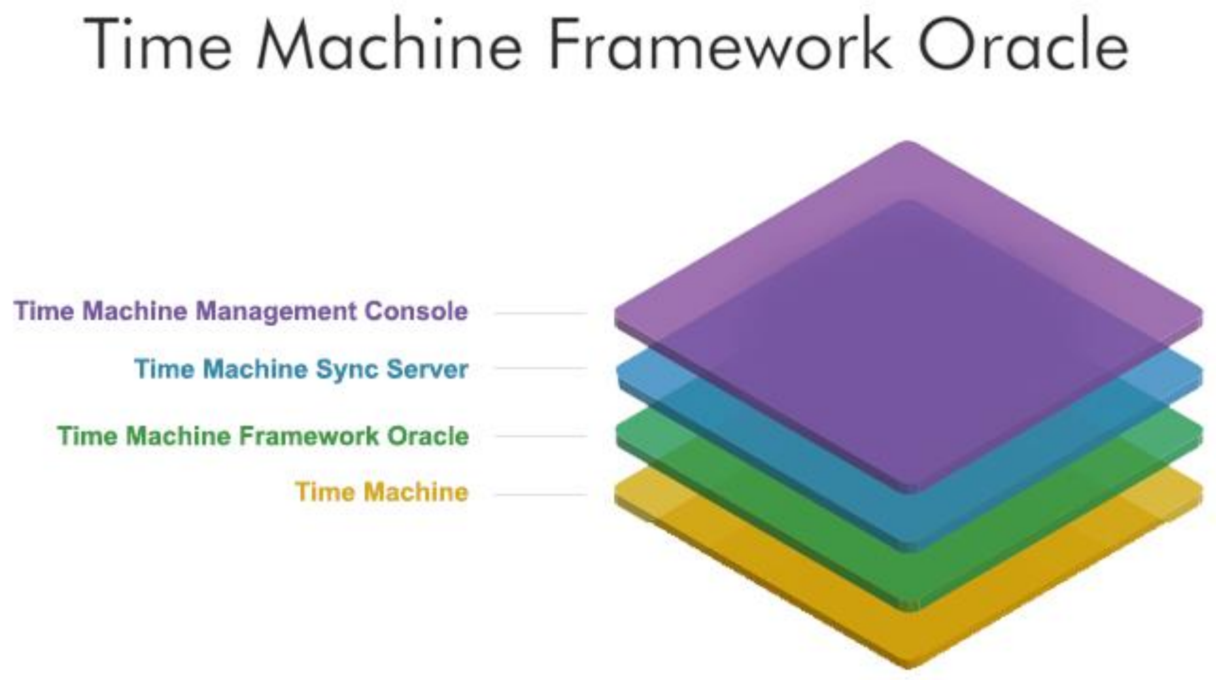
WHITE PAPER

TIME MACHINE FRAMEWORK FOR ORACLE (TMFO) MULTITENANT DATABASES

TIME MACHINE FRAMEWORK FOR ORACLE (TMFO)

Time Machine Framework for Oracle (TMFO) is part of the Time Machine stack that provides more granularity of time travel within an Oracle instance, so multiple different virtual clocks can co-exist. TMFO enables customers to monitor all connections to an Oracle instance and to time travel specific connection with its own virtual clock with the console. Customers can also define, enable or disable rules, so virtual clock will be automatically set upon connection if the rule matches the filters on database user, host, or program.

TMFO has unique feature to time travel separate tenants including CDB and PDBs of a single Oracle instance independently thus providing more flexibility and independence in QA testing of applications which use separate tenant databases in a single QA Oracle instance. TMFO supports all Oracle Multi-Tenant architecture releases from 12c up to 19c and up.


Time Machine Framework Oracle
Time Machine Management Console
Time Machine Sync Server
Time Machine Framework Oracle
Time Machine

GRAPHIC 1

TM BENEFITS & KEY FEATURES:

- Enables multiple virtual clocks with the same Oracle instance
- Time travel by connection, program or database user within an Oracle instance
- Automate date and time sensitive testing by defining simple rules
- Intuitive and easy to use GUI
- Supports Linux, Windows, Solaris, AIX, HP-UX
- Supports all Oracle Multitenant Architecture releases from 12c and up
- Supports independent time travel of CDB and PDB tenants
- Support of cloud Oracle deployments in AWS, Oracle Cloud, etc
- Supports PL/SQL API for easy scripting

This white paper provides detailed information about how to install and configure TMFO for separate CDB and PDB tenants in the latest Oracle19c release and time travel tenants to different virtual times.

TMFO REQUIREMENTS

Before installing TMFO in Oracle19c tenants, the following requirements must be satisfied:

- Time Machine (TM) must be installed on a host operating system before you can proceed to TMFO installation. You must ensure that Time Machine is successfully activated with valid license codes and the Time Machine service runs without any issues.
- Ensure that Time Machine installation directory is contained in PATH environment variable.
- The following TM versions are supported:

    - AIX 64 bit - Time Machine version 10 or later
    - HPUX Itanium - Time Machine version 10 or later
    - Linux x64 - Time Machine version 11.1R9 or later
    - Solaris SPARC - Time Machine version 10 or later
    - Windows x64 - Time Machine version 11.1R12 or later

TMFO INSTALLATION IN ORACLE 19C TENANT DATABASES

TMFO installation in Oracle19c tenants includes three major steps:

- Install TMFO modules in the host where Oracle19c database is running
- Install TMFO database objects in a first Oracle19c tenant (CDB or PDB)
- Install TMFO database objects in subsequent Oracle19c tenants if needed

TMFO installation differs for the first tenant and all subsequent tenants within a single Oracle19c instance. This white paper outlines steps how to install TMFO in Oracle19c CDB and PDB tenants.

TMFO MODULES INSTALLATION IN DATABASE HOST OPERATING SYSTEM

First, you must install TMFO modules in an operating system where Oracle19c is running:

- Login to Oracle database server host as database software owner user. For Unix/Linux platforms, this user is called usually oracle.
- Create a new directory owned by Oracle database software owner user. This directory will be used as installation and working directory of TMFO product. This directory will be referred to as TMFO_DIR. For example, on Linux create *tmfo* folder by the command:

  ```
  $ mkdir /home/oracle/tmfo
  ```

- Extract the TMFO distribution archive to the TMFO_DIR folder.
- Go to TMFO_DIR folder, open *tmWorkPck.sql* script file in any editor program and configure the following required parameters:
  - *tmOwner* - name of the Oracle tenant database schema which will be used to store necessary database objects and TMFO PL/SQL API. This parameter should be set to the schema name (in upper case). In most cases, TMFO schema name is TM. However, for CDB database you must name the Oracle user as **C##TM** because user names in CDB container must have C## prefix if the INIT.ORA parameter **COMMON_USER_PREFIX** is set to its default value C##.

    For example,

    ```
    tmOwner            varchar2(100) := 'C##TM';
    ```

  - *tmUserProgram* - absolute path to Time Machine *tmuser* program executable;

    For example, on Unix/Linux platforms:

    ```
    tmUserProgram      varchar2(1000) := '/etc/ssstm/tmuser';
    ```

  - *tmWorkDir* - absolute path to TMFO_DIR folder.

    For example, on Unix/Linux platforms:

    ```
    tmWorkDir          varchar2(1000) := '/home/oracle/tmfo';
    ```

- Proceed to TMFO installation in an Oracle19c CDB tenant database.

INITIAL TMFO INSTALLATION IN FIRST TENANT

After you installed TMFO modules in the operating system, you must install TMFO database objects in each Oracle19c tenant which you are planning to time travel.

Note that the installation process is divided into 2 parts. First part is initial installing of TMFO in a first Oracle19c tenant (CDB or PDB). Second part is installing TMFO in additional tenants (refer to the topic “TMFO Installation in Additional Tenant”)

Note that TMFO installation is identical for CDB or PDB tenant. The only difference might be the name of TMFO database user in CDB tenant as shown in the example below.

The steps below describe the process of the initial TMFO installation in a CDB tenant:

- Go to the TMFO_DIR folder.
- Run SQL*Plus and connect to Oracle19c CDB tenant database as SYS user.
- Create a new schema **C##TM** which will be used to store necessary database objects and TMFO PL/SQL API. The schema name must be set to the name configured in *tmOwner* parameter *in tmWorkPck.sql* script. For example, run SQL statements as SYS user in the CDB tenant database:

```
$ sqlplus sys as sysdba

-- connect to CDB Oracle19c tenant database

SQL> CREATE USER C##TM IDENTIFIED BY pwd;
User created.

SQL> GRANT CONNECT,RESOURCE,UNLIMITED TABLESPACE to tm;
Grant succeeded.
```

- Run *install.sql* script in the CDB tenant:

```
  SQL> show con_name

  CON_NAME
------------------------------
CDB$ROOT

SQL> @install.sql
```

- When prompted, enter the name of TMFO schema name C##TM configured by the parameter *tmOwner*.
- Wait until script execution completes and verify that no errors occurred in the log file *install.text* created in the TMFO_DIR.
- If no errors were generated, TMFO framework is installed successfully and ready to use in the CDB tenant database.
- Optionally, you can grant EXECUTE privilege on TMWORK PL/SQL package to other database users who will use TMFO product.

For example, login in SQL*Plus to the tenant database under the TMFO user and execute the following SQL statement to grant EXECUTE privilege to a database user C##USER1:

```
$ sqlplus c##tm/pwd
```

```
SQL> show con_name

CON_NAME
------------------------------
CDB$ROOT

SQL> grant execute on tmwork to c##user1;
Grant succeeded.
```

TMFO INSTALLATION IN ADDITIONAL TENANT

After successful TMFO installation in the first tenant (CDB as in the example above), you can optionally install TMFO in additional tenants available in your Oracle19c database. Below steps provide installation example for a PDB tenant:

- Go to the TMFO_DIR folder.
- Run SQL*Plus and connect to a target Oracle19c tenant database as SYS user.
- Create a new schema which will be used to store necessary database objects and TMFO PL/SQL API in this tenant. The schema name must be identical to the name configured by *tmOwner* parameter in *tmWorkPck.sql* script. Notice that if you previously set this *tmOwner* parameter to C##TM user name for the initial TMFO installation in CDB tenant, then you **must update** this parameter to the user name that will be used for TMFO in this tenant. For example, run SQL statements as SYS user in the PDB tenant database:

```
$ sqlplus sys as sysdba

-- connect to the Oracle19c tenant database PDB

SQL> ALTER SESSION SET CONTAINER = PDB;
Session altered.

SQL> CREATE USER TM IDENTIFIED BY pwd;
User created.

SQL> GRANT CONNECT,RESOURCE,UNLIMITED TABLESPACE to tm;
Grant succeeded.
```

- Run addDB.sql script in the tenant database PDB:

```
SQL> show con_name

CON_NAME
------------------------------
PDB
```

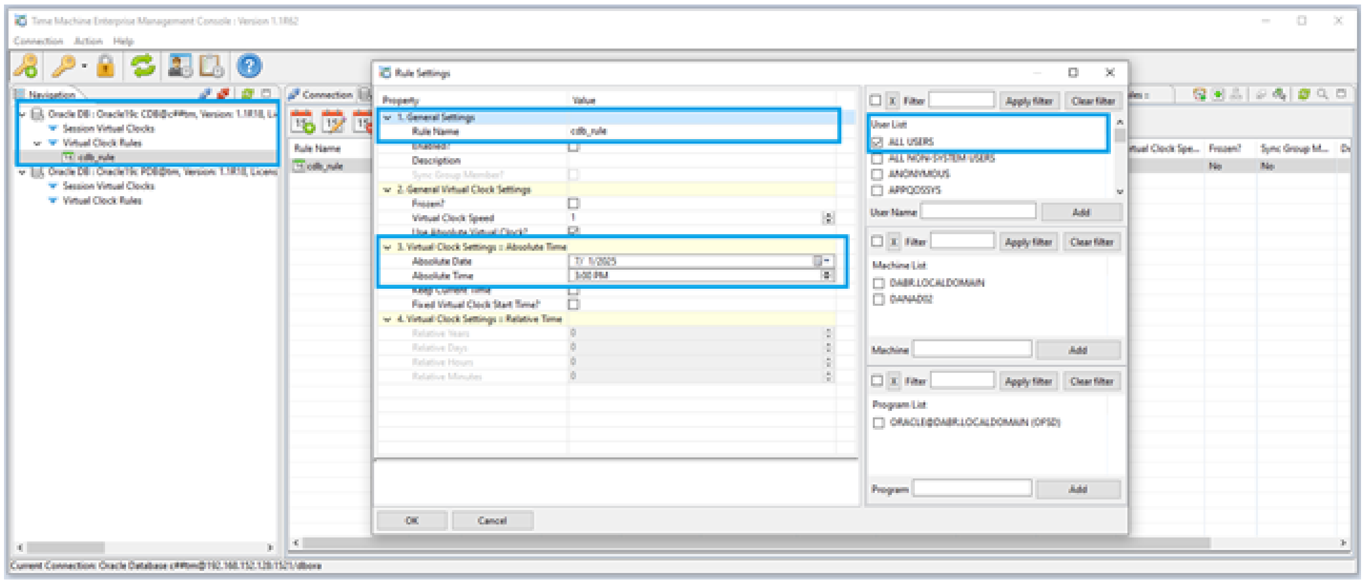
```
SQL> @addDB.sql
```

- When prompted, enter the name of TMFO schema name configured by the parameter *tmOwner* in the script *tmWorkPck.sql.* In this example, it must be TM.
- Wait until script execution completes and verify that no errors occurred in the log file *addDB.text* created in the TMFO_DIR.
- If no errors were generated, TMFO framework is installed successfully and ready to use in the tenant database.

TIME TRAVEL ORACLE19C TENANT

TMFO allows to setup different virtual times for different Oracle19c tenants where TMFO is installed. TMFO supports time travelling of CDB and PDB tenants. This feature enables customers to easily perform independent time shift QA testing in cloud based databases. Let's see below how to configure different virtual times for 2 Oracle 19c tenants CDB and PDB using TMFO and TMConsole Enterprise products.

- In TMConsole Enterprise connect to the TMFO repository installed in the tenant database CDB.
- In the "Oracle Database Virtual Clock Rules" configure a new rule "cdb_rule" to enable virtual time for the tenant CDB:



Time Machine Enterprise Management Console : Version 1.1862

Property	Value
1. General Settings	
Rule Name	cdb_rule
Enabled?	☐
Description	
Sync Group Member?	☐
2. General Virtual Clock Settings	
Frozen?	☐
Virtual Clock Speed	1
Use Absolute Virtual Clock?	✓
3. Virtual Clock Settings :: Absolute Time	
Absolute Date	7/ 1/2025
Absolute Time	3:00 PM
Keep Current Time	☐
Fixed Virtual Clock Start Time?	☐
4. Virtual Clock Settings :: Relative Time	
Relative Years	0
Relative Days	0
Relative Hours	0
Relative Minutes	0

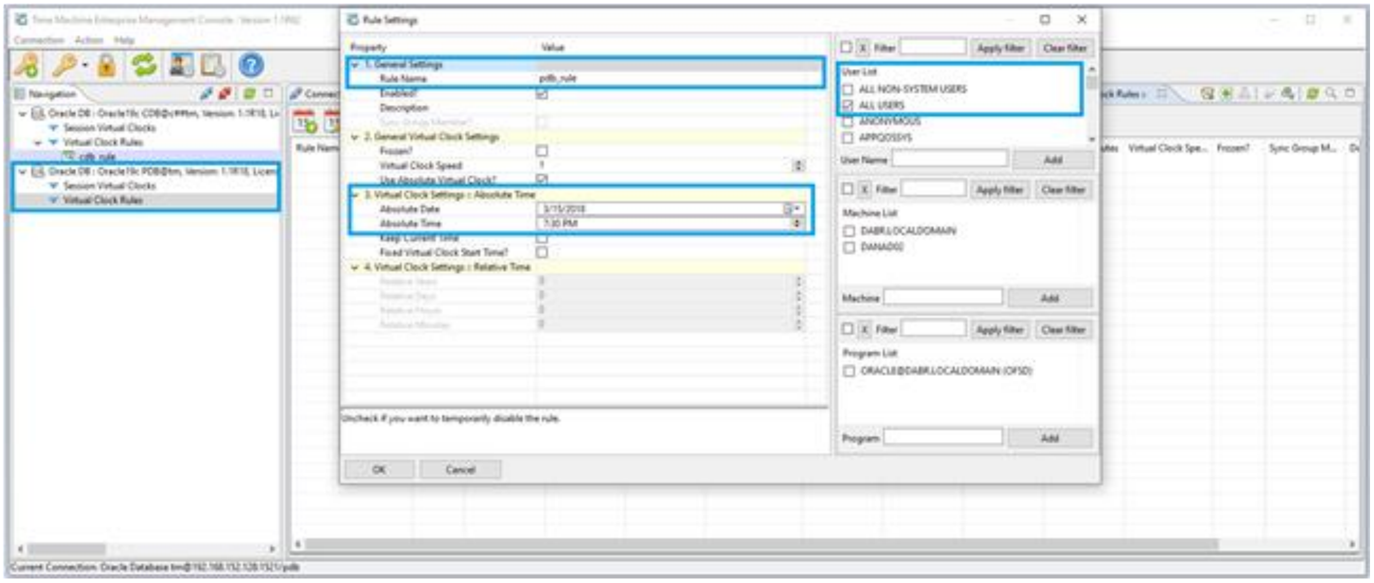
User List: ✓ ALL USERS; ☐ ALL NON-SYSTEM USERS; ☐ ANONYMOUS; ☐ APPQOSSYS

Machine List: ☐ DABR.LOCALDOMAIN; ☐ DAWADS2

Program List: ☐ ORACLE@DABR.LOCALDOMAIN (OFSD)

GRAPHIC 2

- In the "User List" you can either check "ALL USERS" to enable virtual clock for all tenant database users or select particular users which are specific for your application.
- You can optionally configure additional rule filters based on client host names and application names.
- Check "Enabled?" check box and press OK to activate virtual clock for the tenant CDB.
- Connect to the tenant PDB and configure a rule named *pdb_rule* for this tenant:



Rule Settings

Property	Value
1. General Settings	
Rule Name	pdb_rule
Enabled?	✓
Description	
2. General Virtual Clock Settings	
Frozen?	☐
Virtual Clock Speed	1
Use Absolute Virtual Clock?	✓
3. Virtual Clock Settings :: Absolute Time	
Absolute Date	3/15/2018
Absolute Time	7:30 PM
Keep Current Time	☐
Fixed Virtual Clock Start Time?	☐
4. Virtual Clock Settings :: Relative Time	

Uncheck if you want to temporarily disable the rule.

User List: ☐ ALL NON-SYSTEM USERS, ✓ ALL USERS, ☐ ANONYMOUS, ☐ APPQOSSYS

Machine List: ☐ DABR.LOCALDOMAIN, ☐ DANADOO

Program List: ☐ ORACLE@DABR.LOCALDOMAIN (OFSD)

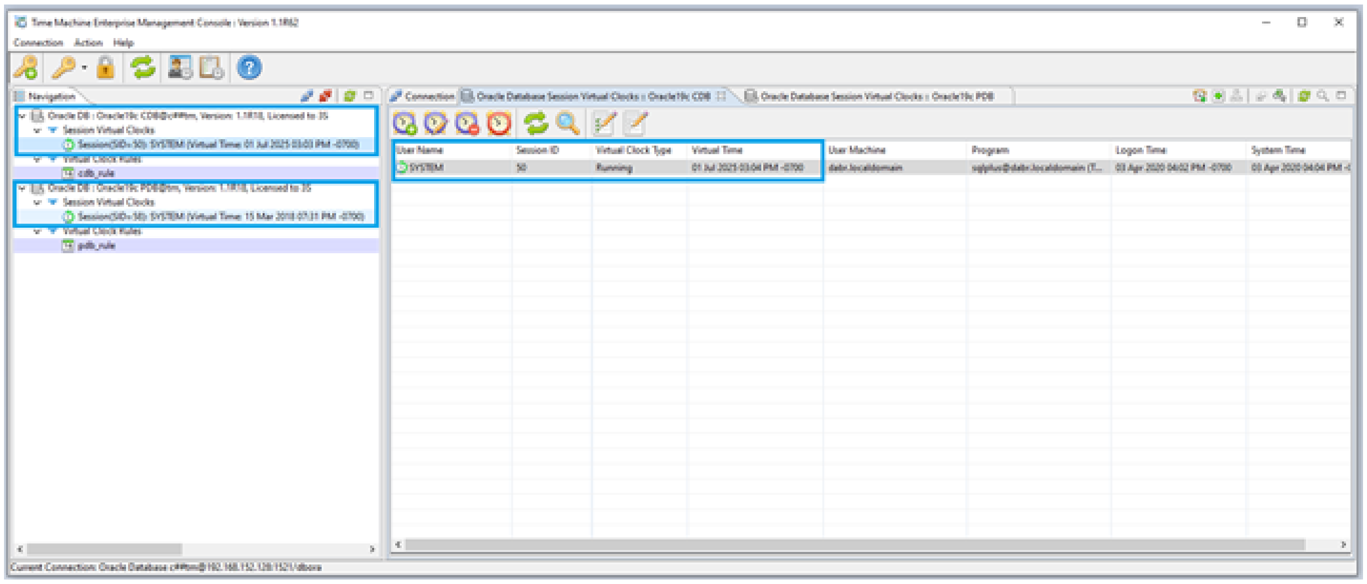
Current Connection: Oracle Database tm@192.168.152.128:1521/pdb

GRAPHIC 3

- You can optionally configure additional rule filters based on client host names and application names. Save your changes.
- Save your changes.

.

Now, you have configured different virtual times for two tenant databases CDB and PDB. Let's connect to both tenants and verify if applications see correct virtual times in every tenant. In TMConsole Enterprise, it is also possible to verify actual virtual times for all user sessions in each tenant database:



Time Machine Enterprise Management Console : Version 1.1R62

Oracle DB : Oracle19c CDB@c##tm, Version: 1.1R18, Licensed to 35
Session Virtual Clocks
Session(SID=50): SYSTEM (Virtual Time: 01 Jul 2025 03:03 PM -0700)
Virtual Clock Rules
cdb_rule
Oracle DB : Oracle19c PDB@tm, Version: 1.1R18, Licensed to 35
Session Virtual Clocks
Session(SID=50): SYSTEM (Virtual Time: 15 Mar 2018 07:31 PM -0700)
Virtual Clock Rules
pdb_rule

User Name	Session ID	Virtual Clock Type	Virtual Time	User Machine	Program	Logon Time	System Time
SYSTEM	50	Running	01 Jul 2025 03:04 PM -0700	dabr.localdomain	sqlplus@dabr.localdomain (T...	03 Apr 2020 04:02 PM -0700	03 Apr 2020 04:04 PM -

Current Connection: Oracle Database c##tm@192.168.152.128:1521/dbora

GRAPHIC 4

Oracle Cloud Ready

The Time Machine Framework for Oracle was developed and tested in the Oracle Database Cloud Service and is listed in the Oracle Cloud Marketplace.

Visit our page in the *Oracle Cloud Marketplace*

SolutionSoft Systems, Inc.
2350 Mission College Blvd., Suite #777
Santa Clara, CA 95054, U.S.A.
Phone: 1.408.346.1400
Sales: 1.408.346.1415

Europe
SolutionSoft Systems, Inc.
Trnska 8, Suite 7
Belgrade, Serbia, 11000
Phone: +1381 11 403 1523

www.solution-soft.com

www.facebook.com/solution-soft
www.twitter.com/solution-soft
www.linkedin.com/solution-soft