Oracle® Database

Oracle True Cache User's Guide

23ai

F77385-01

May 2024

Oracle Database Oracle True Cache User's Guide, 23ai

F77385-01

Primary Authors: J. True, J.-F. VERRIER

Contributors: P. Chandiramani, D. Chen, S. Datta , J. De Lavarene , K. Fekete , M. Girkar , W. Hu, S. Krishnaswamy , A. Lee, B. Lee, Y. Li, R. Mallegowda , M. Malvezzi , C. Ruiz, P. Saini, V. Senapati, A. Srivastava, M. Stalin, J. Stamos , D. Surber , T. Thathachary

Contents

Preface

1 Overview of Oracle True Cache

2 Configuring True Cache

8 Complementary Caching Features

9 Deleting True Cache

A True Cache Error Messages

Preface

This guide provides a single location with everything you need to know about configuring and using True Cache.

- Documentation Accessibility
- Related Documents
  The following documents contain information that may be of use when configuring and using True Cache.
- Conventions

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

The following documents contain information that may be of use when configuring and using True Cache.

- Oracle True Cache Technical Architecture
- *Oracle Data Guard Concepts and Administration*
- *Oracle Database Administrator's Guide*
- Oracle Database Container Images
- Oracle Database Error Messages
- *Oracle Database Performance Tuning Guide*
- *Oracle Database PL/SQL Packages and Types Reference*
- *Oracle Database Reference*
- *Oracle Database JDBC Developer's Guide*
- *Oracle Multitenant Administrator's Guide*
- *Oracle Real Application Clusters Administration and Deployment Guide*

- *Oracle Database SQL Language Reference*
- *Oracle Universal Connection Pool Developer's Guide*
- Java™ Platform, Standard Edition 7 API Specification

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1
Overview of Oracle True Cache

Oracle True Cache is an in-memory, consistent, and automatically managed SQL and key-value (object or JSON) cache for Oracle Database.

For an interactive tour, see Oracle True Cache Technical Architecture.

1.1 About Oracle True Cache

True Cache is an in-memory, read-only cache in front of an Oracle database.

Like Oracle Active Data Guard, True Cache is a fully functional, read-only replica of the primary database, except that it's mostly diskless.

Modern applications often require a high number of connections and fast, low-latency access to the data. A popular approach is to place caches in front of the database because applications typically perform many more reads than updates, and they can read from the cache without affecting the database performance. For example, an airline reservation system reads data frequently as people shop for flights, and it's okay if the data isn't the most current compared to what's in the database. The application only needs to access the most current data when someone reserves a flight.

Unlike conventional caches, True Cache automatically keeps the most frequently accessed data in the cache, and it keeps the cache consistent with the primary database, other objects in the same cache, and other caches. It caches all Oracle Database objects and data types, including JSON.

Because True Cache implements Oracle Database security policies, you can control access to the cache. This enables you to cache sensitive data, like private personal information, that you might not ordinarily cache.

1.2 Benefits of Oracle True Cache

Oracle True Cache provides several business benefits related to application development and performance.

- Improves scalability and performance by offloading queries from the primary database.
- Reduces application response time and network latency by deploying True Cache closer to the application. This especially benefits situations where a database is in a different location than the application due to data residency requirements.
- Creates a large, in-memory storage area by dividing data across multiple True Caches. The total size of the cached data across all True Caches can be much larger than it would be for a single primary database or cache.
- Automatically maintains the cache contents.
- Simplifies development and maintenance by being transparent to the application.

1.3 How True Cache Works

At a high level, here's how Oracle True Cache works.

- An application decides whether to query data from True Cache or the primary database. For details on how this works, see Application Usage Models.
- True Cache satisfies queries by using data that's cached in its memory. When the data isn't in the cache, True Cache fetches the data from the primary database.
- True Cache is empty when it starts up, so it reads large chunks of data to populate the cache. After a block is cached, it's updated automatically through redo apply from the primary database. This is similar to the update mechanism used in Oracle Active Data Guard.
- A query to True Cache returns only committed data, and the data is always consistent.
- Like all caches, the True Cache data might not be the most current data as it exists in the primary database.
- If multiple True Caches exist and serve the same database application service, the listener automatically distributes and load balances sessions to each cache.

1.4 Considerations for Using True Cache

Consider these points when deciding whether to use Oracle True Cache.

- You need to configure enough memory for True Cache so your most frequently accessed data will stay in memory.
- True Cache only needs a small amount of storage for the standby redo log files. Configuration files like control files, the `SPFILE`, and temporary data files are automatically created.
- True Cache is a read-only cache, so you can't directly update the cache, but you can use DML redirection to indirectly update the cache.

Related Topics

- Enabling DML Redirection
  True Cache is a read-only cache, so you can't directly update the cache, but you can use DML redirection to indirectly update the cache.

1.5 Application Usage Models

Applications can use True Cache in two ways.

- The application maintains *two physical connections*: one to the primary database and one to True Cache. Each connection has a database application service, and the application chooses which connection to use based on whether it's reading or writing. You can use this model with any existing client drivers and any programming language.

  The application sends queries that don't need to see the most current data to True Cache through a True Cache database application service. The application sends other queries and updates to the primary database through the primary database application service.

- The application maintains *one logical connection* that uses the database application service for the primary database. The JDBC Thin driver (starting with Oracle Database 23ai) maintains physical connections to the primary database and True Cache. This model only works with Java applications.

  The application switches between the primary database and True Cache without having to specify an instance name. The application uses special calls to flag the logical connection as read-only or read-write. If it's read-only, the query is sent to True Cache. Otherwise, it's sent to the primary database.

Related Topics

- Using Oracle True Cache in Your Applications
  When True Cache is configured, applications must decide whether to query data from True Cache or the primary database.

1.6 Lock-Free Concurrency Control

For applications that work with JSON documents, Oracle extends HTTP entity tag (ETAG) support to implement lock-free, optimistic concurrency control.

A database GET request to True Cache automatically computes the ETAG and inserts it into the returned document. When the modified document is PUT back into the primary database, the following occurs:

1. The database verifies that the underlying document rows still match the ETAG that was computed by the GET request.
2. If the rows match the ETAG, the rows are atomically updated.
3. If there's not a match, another user has changed the data and the PUT request is rejected.
4. The PUT request can be retried using the new data.

2

Configuring True Cache

You can configure True Cache using the Oracle Database Configuration Assistant (Oracle DBCA) or you can configure it manually.

- Overview of the True Cache Configuration
  A True Cache environment can have a uniform or partitioned configuration.
- Configuring True Cache with Oracle DBCA
  The simplest way to configure a True Cache environment is to use Oracle Database Configuration Assistant (Oracle DBCA).
- Configuring True Cache Manually
  Follow these steps to set up a True Cache environment manually if you're not using the Oracle Database Configuration Assistant (Oracle DBCA).
- Deploying True Cache for an Oracle RAC Primary Database
  Follow these additional steps if you're deploying True Cache for a primary database in an Oracle Real Application Clusters (Oracle RAC) environment.
- Verifying the True Cache Configuration
  Verify that True Cache and the database application services are working as expected.
- Enabling DML Redirection
  True Cache is a read-only cache, so you can't directly update the cache, but you can use DML redirection to indirectly update the cache.

2.1 Overview of the True Cache Configuration

A True Cache environment can have a uniform or partitioned configuration.

- Uniform Configuration
  In a uniform configuration, you can deploy multiple, identical True Caches that use the same True Cache database application service.
- Partitioned Configuration
  In a partitioned configuration, the data is divided across multiple True Caches, which each cache a different subset of the data.
- True Cache Configuration Process Overview
  At a high level, creating a True Cache configuration involves the following steps.
- Best Practices for Database Application Services
  Associate each primary database application service with a corresponding True Cache database application service.

2.1.1 Uniform Configuration

In a uniform configuration, you can deploy multiple, identical True Caches that use the same True Cache database application service.

Client sessions are evenly distributed among True Caches, which all cache the same set of data.

See the following technical architecture diagram for details and an example of a uniform configuration:

True Cache Uniform Configuration

2.1.2 Partitioned Configuration

In a partitioned configuration, the data is divided across multiple True Caches, which each cache a different subset of the data.

The total size of the cached data across all True Caches can be much larger than it would be for a single primary database or a single cache in a uniform configuration.

See the following technical architecture diagrams for details and examples of partitioned configurations:

- True Cache Configuration with colocation_tag
- True Cache Partitioned Configuration with Multiple Services

2.1.3 True Cache Configuration Process Overview

At a high level, creating a True Cache configuration involves the following steps.

1. Configure the network for True Cache and the primary database, and optionally create a remote listener for high availability.
2. Create True Cache.
3. Create and start the True Cache database application services.

2.1.4 Best Practices for Database Application Services

Associate each primary database application service with a corresponding True Cache database application service.

To distinguish the True Cache database application services, it's a good practice to use the primary service name followed by `_TC`.

For example, if the primary service is `SALES`, the True Cache service would be `SALES_TC`.

Also, only start the True Cache services on True Caches, and make sure they're read-only.

2.2 Configuring True Cache with Oracle DBCA

The simplest way to configure a True Cache environment is to use Oracle Database Configuration Assistant (Oracle DBCA).

Oracle DBCA is an assistant tool that comes with Oracle Database to simplify the process of creating, configuring, and managing Oracle databases and True Caches.

- Prerequisites
  Before configuring True Cache, complete the following prerequisites.
- True Cache Configuration
  Follow these steps to configure True Cache with Oracle DBCA.

- Oracle DBCA Commands and Parameters for True Cache
  Use the following Oracle DBCA commands and parameters to configure True Cache.

2.2.1 Prerequisites

Before configuring True Cache, complete the following prerequisites.

- Set up a network path to access the primary (source) database with an Easy Connect (EZConnect) string from the True Cache node.
- Install the Oracle Database software on the True Cache nodes.
- Ensure that a primary database is running on the primary node in archive logging (ARCHIVELOG) mode.

  The primary database must be in ARCHIVELOG mode to ship redo log files to the True Cache node. Oracle DBCA verifies that the primary database is in ARCHIVELOG mode.

  If the primary database is not in ARCHIVELOG mode, restart it in mount mode, run the ALTER DATABASE ARCHIVELOG command, and open the primary database again.
- Don't set LOG_ARCHIVE_CONFIG and LOG_ARCHIVE_DEST_*n* on the primary database. True Cache automatically configures these for the primary database.

2.2.2 True Cache Configuration

Follow these steps to configure True Cache with Oracle DBCA.

1. Copy the password file, and optionally the Transparent Data Encryption (TDE) wallet, from the primary database to the True Cache node. You can use either of the following methods:

   - Manually copy the primary database's password file (and optional TDE wallet) to any location on the True Cache node.
   - Run a DBCA command on the primary database to package the password file (and optional TDE wallet) into a configuration BLOB file. Then copy the file to the True Cache node and ensure that the file is owned by the Oracle Home user.

     Use the following command to prepare the configuration BLOB file on the primary database:

     ```
     ORACLE_HOME/bin/dbca -configureDatabase -prepareTrueCacheConfigFile -
     sourceDB primary_db_sid_or_db_unique_name -trueCacheBlobLocation
     primary_db_config_blob_path -silent
     ```

     Example:

     ```
     $ORACLE_HOME/bin/dbca -configureDatabase -prepareTrueCacheConfigFile -
     sourceDB primdb1i -trueCacheBlobLocation /tmp/blob_loc2 -silent
     ```

     For descriptions of the parameters, see configureDatabase.
2. On the True Cache node, run the -createTrueCache command to complete the True Cache configuration and start True Cache.

Choose one of the following command formats, depending on the method that you used to copy the primary database's password file (and optional TDE wallet) to the True Cache node in step 1.

- If you manually copied the password file (and optional TDE wallet), use this command:

  ```
  ORACLE_HOME/bin/dbca -createTrueCache -gdbName
  true_cache_global_name -sid true_cache_sid -
  sourceDBConnectionString primary_db_easy_connect_string -
  passwordFileFromSourceDB password_file_path -sgaTargetInMB
  sga_memory_size -pgaAggregateTargetInMB pga_memory_size -silent
  ```

  Example:

  ```
  $ORACLE_HOME/bin/dbca -createTrueCache -gdbName tcdb1 -sid tcdb1
  -sourceDBConnectionString primary.example.com:1522/
  primdb1i.example.com -passwordFileFromSourceDB /tmp/
  password_loc2/orapwddbmc1 -sgaTargetInMB 20000 -
  pgaAggregateTargetInMB 4000 -silent
  ```

- If you created the configuration BLOB file, use this command:

  ```
  ORACLE_HOME/bin/dbca -createTrueCache -gdbName
  true_cache_global_name -sid true_cache_sid -
  sourceDBConnectionString primary_db_easy_connect_string -
  trueCacheBlobFromSourceDB true_cache_config_blob_path -
  sgaTargetInMB sga_memory_size -pgaAggregateTargetInMB
  pga_memory_size -silent
  ```

  Example:

  ```
  $ORACLE_HOME/bin/dbca -createTrueCache -gdbName tcdb1 -sid tcdb1
  -sourceDBConnectionString primary.example.com:1522/
  primdb1i.example.com -trueCacheBlobFromSourceDB /tmp/blob_loc2/
  blob.tar.gz -sgaTargetInMB 20000 -pgaAggregateTargetInMB 4000 -
  silent
  ```

For descriptions of the parameters, see createTrueCache.

> **Note:**
>
> For Oracle RAC primary databases, set the `-sourceDBConnectionString` parameter to `SCAN:port/service_name`.

3. On the primary database, complete the following steps:

   a. Configure the remote listener.

> **Note:**
>
> Check to see if Valid Node Checking for Registration (VNCR) is configured for the listener. If so, add True Cache to the `REGISTRATION_INVITED_NODES_LISTENER` parameter in the `listener.ora` file of the remote listener.

b. Create and start a primary database application service to use with the True Cache service, and then verify that the service is running.

For single instance primary databases, use the `DBMS_SERVICE` PL/SQL package.

For Oracle RAC primary databases, use the `srvctl` command line utility with the `-true_cache_service` option. Start the service on the primary instance. Use `srvctl` only for clusterware-managed instances.

Here's an example using the `DBMS_SERVICE` package in a non-clusterware environment:

```
BEGIN
   DBMS_SERVICE.CREATE_SERVICE('SALES', 'SALES');
   DBMS_SERVICE.START_SERVICE('SALES');
END;
/
PL/SQL procedure successfully completed.


SELECT service_id, name, true_cache_service FROM v$active_services
WHERE name='SALES';

SERVICE_ID NAME               TRUE_CACHE_SERVICE
---------- ------------------ ------------------
         6 SALES

// TRUE_CACHE_SERVICE will be filled in by this (dbca -
configureTrueCacheInstanceService) command
```

c. Run the `dbca -configureDatabase` command with the `-configureTrueCacheInstanceService` parameter to configure True Cache.

This configures the True Cache database application service on the primary database and then starts the True Cache service on True Cache. Run this command for each True Cache database application services.

> **Note:**
>
> The primary database application service must already exist before you run this command.

```
ORACLE_HOME/bin/dbca -configureDatabase -
configureTrueCacheInstanceService -sourceDB
primary_db_sid_or_db_unique_name -trueCacheConnectString
```

```
true_cache_easy_connect_string -trueCacheServiceName
true_cache_service_name -serviceName primary_db_service_name -
silent
```

Example:

```
$ORACLE_HOME/bin/dbca -configureDatabase -
configureTrueCacheInstanceService -sourceDB primdb1i -
trueCacheConnectString tc.example.com:1522/tcdb1.example.com -
trueCacheServiceName sales_tc -serviceName sales -silent
```

For descriptions of the parameters, see configureDatabase.

Related Topics

- CREATE_SERVICE Procedure

2.2.3 Oracle DBCA Commands and Parameters for True Cache

Use the following Oracle DBCA commands and parameters to configure True Cache.

- configureDatabase
  The `configureDatabase` command configures the primary database for True Cache. Run this command on the primary database.
- createTrueCache
  The `createTrueCache` command configures True Cache. Run this command on the True Cache node.

2.2.3.1 configureDatabase

The `configureDatabase` command configures the primary database for True Cache. Run this command on the primary database.

Parameters

For True Cache, use the `dbca -configureDatabase` command with the following syntax.

> **Note:**
>
> This table lists the `configureDatabase` parameters that are specific to True Cache. For the full syntax and parameters, see configureDatabase in the *Oracle Multitenant Administrator's Guide*.

Table 2-1   configureDatabase Parameters

Parameter	Required/ Optional	Description
`--prepareTrueCacheConfigFile`	Required for True Cache	Use this option to prepare a configuration BLOB file that contains the primary database's password file or wallet. Enter the following additional parameters for this option: • `-sourceDB`: Enter the primary database system identifier (SID) or database unique name (`DB_UNIQUE_NAME`). • `-tdeWalletPassword`: If the primary database uses a Transparent Data Encryption (TDE) wallet, enter the password for the wallet. This parameter is optional. • `-trueCacheBlobLocation`: Enter the path where you want to save the configuration BLOB file on the primary database. This parameter is optional.
`-configureTrueCacheInstanceService`	Required for True Cache	Use this option to configure the True Cache database application service on the primary database and start the service on True Cache. Enter the following additional parameters for this option: • `-serviceName`: Enter the primary database application service name. • `-sourceDB`: Enter the primary database SID or database unique name (`DB_UNIQUE_NAME`). • `-trueCacheConnectString`: Enter the Easy Connect (EZConnect) string to connect to True Cache. Example: `host:port/service_name` • `-trueCacheServiceName`: Enter a name for the True Cache database application service. • `-pdbName`: Enter the primary pluggable database (PDB) name. This parameter is optional.
`-cleanupTrueCacheInstanceService`	Required for True Cache	Use this option to remove the True Cache database application services from the primary database configuration if, for example, you delete True Cache. Enter the following additional parameters for this option: • `-serviceName`: Enter the primary database application service name. • `-sourceDB`: Enter the primary database SID or database unique name (`DB_UNIQUE_NAME`). • `-trueCacheConnectString`: Enter the Easy Connect (EZConnect) string to connect to True Cache. Example: `host:port/service_name`

2.2.3.2 createTrueCache

The `createTrueCache` command configures True Cache. Run this command on the True Cache node.

Syntax and Parameters

Use the `dbca -createTrueCache` command with the following syntax:

```
dbca -createTrueCache
   -dbUniqueName true_cache_unique_name | -gdbName
true_cache_global_name
   -sourceDBConnectionString primary_db_easy_connect_string
   -trueCacheBlobFromSourceDB true_cache_config_blob_path | -
passwordFileFromSourceDB password_file_path
      [-tdeWalletFromSourceDB tde_wallet_path]
   [-createListener new_database_listener]
   [-datafileDestination true_cache_control_file_path
   [-initParams initialization_parameters_list
      [-initParamsEscapeChar
initialization_parameters_escape_character]]
   [-listeners listener_list]
   [-pgaAggregateTargetInMB pga_memory_size]
   [-sgaTargetInMB sga_memory_size]
   [-sid true_cache_sid]
   [-sourceTdeWalletPassword primary_db_wallet_password]
   [-tdeWalletLoginType {PASSWORD | AUTO_LOGIN | LOCAL_AUTO_LOGIN}]
   [-tdeWalletRoot tde_wallet_root_init_parameter]
   [-useWalletForDBCredentials {true | false}
      -dbCredentialsWalletLocation wallet_files_directory
      [-dbCredentialsWalletPassword wallet_account_password]]
```

Table 2-2   createTrueCache Parameters

Parameter	Required/ Optional	Description

`-dbUniqueName`
true_cache_unique_name
or
`-gdbName`
true_cache_global_name
	Required	Enter either the unique name for this True Cache or the global database name.

`-sourceDBConnectionString` *primary_db_easy_connect_string*	Required	Enter the Easy Connect (EZConnect) string to connect to the primary database. Example: `host:port/service_name` **Note:** For Oracle RAC primary databases, set the `–sourceDBConnectionString` parameter to `SCAN:port/service_name`.

Table 2-2 (Cont.) createTrueCache Parameters

Parameter	Required/ Optional	Description

`-trueCacheBlobFromSourceDB`
`true_cache_config_blob_path`
or
`-passwordFileFromSourceDB`
`password_file_path`
	Required	Enter one of the following:
		• The full path and file name for the configuration BLOB file that contains the primary database's password file or wallet. This is the path where the file is located on the True Cache node.
		• The path to the primary database's password file that was copied to the True Cache node.
		If you use `-passwordFileFromSourceDB`, you can also enter the following additional parameter:
		`-tdeWalletFromSourceDB`: Enter the path to the primary database's Transparent Data Encryption (TDE) wallet file that was copied to the True Cache node. You can copy and use the TDE wallet file only if the primary (source) database has TDE enabled. Otherwise, the wallet file isn't required.

`-createListener` *`new_database_listener`*	Optional	Enter a new database listener to be created and to register the database in the form *LISTENER_NAME:PORT*.
`-datafileDestination` *`true_cache_control_file_path`*	Optional	Enter the path to the location of the True Cache control files.
`-initParams` *`initialization_parameters_list`*	Optional	Enter a comma-separated list of `name=value` pairs with additional initialization parameter values for this True Cache. You can also provide the `-initParamsEscapeChar` parameter for using a specific escape character between multiple values of an initialization parameter. If an escape character is not specified, backslash (\) is used as the default escape character.
`-listeners` `listeners_list`	Optional	Enter a comma-separated list of existing listeners that the database can be configured with.
`-pgaAggregateTargetInMB` *`pga_memory_size`*	Optional	Enter a value in MB for the target aggregate Program Global Area (PGA) memory to make available to all server processes that are attached to this True Cache.
`-sgaTargetInMB` *`sga_memory_size`*	Optional	Enter a value in MB for the System Global Area (SGA) memory size for this True Cache.
`-sid` *`true_cache_sid`*	Optional	Enter the system identifier (SID) for this True Cache.
`-sourceTdeWalletPassword` *`primary_db_wallet_password`*	Optional	If the primary database uses a TDE wallet, enter the password for the wallet.

`-tdeWalletLoginType`
`{PASSWORD | AUTO_LOGIN | LOCAL_AUTO_LOGIN}`
	Optional	If the primary database uses a TDE wallet, enter one of the following wallet types:
		• `PASSWORD`: This is the default for single instance databases.
		• `AUTO_LOGIN`: This is the default for Oracle RAC instances.
		• `LOCAL_AUTO_LOGIN`

Table 2-2 (Cont.) createTrueCache Parameters

Parameter	Required/ Optional	Description
-tdeWalletRoot *tde_wallet_root_init_parameter*	Optional	If the primary database uses a TDE wallet, enter the path for the TDE wallet root initialization parameter.
-useWalletForDBCredentials {true | false}	Optional	If the primary database uses Oracle Wallet for database credentials, enter `true`. The default is `false`. If you enter `true`, also enter the following additional parameters: • `-dbCredentialsWalletLocation`: Enter the path of the directory that contains the Oracle Wallet files. • `-dbCredentialsWalletPassword`: Enter the password for the Oracle Wallet account to open the wallet with auto-login disabled. This parameter is optional.

Related Topics

- createTrueCache

2.3 Configuring True Cache Manually

Follow these steps to set up a True Cache environment manually if you're not using the Oracle Database Configuration Assistant (Oracle DBCA).

- Prerequisites
  Before configuring True Cache, complete the following steps.
- Creating True Cache
  Follow these steps to create True Cache.
- Configuring Database Application Services for True Cache
  To use True Cache with the JDBC Thin driver, for each primary database application service that you want to cache, create a corresponding True Cache database application service.

2.3.1 Prerequisites

Before configuring True Cache, complete the following steps.

- Install the Oracle Database software on the True Cache nodes.
- Ensure that a primary database is running on the primary node in archive logging (`ARCHIVELOG)` mode.

  The primary database must be in `ARCHIVELOG` mode to ship redo log files to the True Cache node. To see if the primary database is in `ARCHIVELOG` mode, enter the following SQL query:

  ```
  SELECT log_mode FROM v$database;

  LOG_MODE
  ```

```
------------
ARCHIVELOG
```

If the primary database is not in ARCHIVELOG mode, restart it in mount mode, run the ALTER DATABASE ARCHIVELOG command, and open the primary database again.

- Don't set LOG_ARCHIVE_CONFIG and LOG_ARCHIVE_DEST_*n* on the primary database. True Cache automatically configures these for the primary database.

2.3.2 Creating True Cache

Follow these steps to create True Cache.

- Edit the tnsnames.ora File for True Cache and the Primary Database
  Configure the database network connections by editing the tnsnames.ora file on both True Cache and the primary database.
- Configure and Start the Local Listener on the True Cache Node
  Configure and start the local listener to enable the True Cache node to receive redo from the primary database.
- Copy the Password File or Wallet from the Primary Database to the True Cache Node
  To provide authentication for redo transport sessions, copy the primary database's password file or wallet to the True Cache node.
- Prepare a PFILE for True Cache
  For each True Cache, prepare a PFILE with the following parameters.
- Create and Start True Cache
  Follow these steps to create and start True Cache for the first time.

2.3.2.1 Edit the tnsnames.ora File for True Cache and the Primary Database

Configure the database network connections by editing the tnsnames.ora file on both True Cache and the primary database.

On most Linux platforms, you find the tnsnames.ora file in the $ORACLE_HOME/network/admin directory.

You can create this file manually or by using Oracle Net Configuration Assistant (NETCA).

Add the following information to all primary database and True Cache tnsnames.ora files if the information isn't already present:

- Network names for the primary database and all True Caches. These are needed to send redo to True Cache, and they enable redo shipping to work even if the database application services are not yet created.
- All primary database and True Cache database application service names. One True Cache can support multiple database application services that are started simultaneously.
- All participating listener network aliases, if you're using a remote listener.

Here are examples of the entries in the tnsnames.ora files for True Cache and the primary database:

```
# True Cache network name
```

```
tcdb1i =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = true_cache_host)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SID = tcdb1i)
    )
  )


# True Cache database application service name

sales_tc =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = true_cache_host)(PORT = 1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = sales_tc)
    )
  )


# Primary database network name

primdb1i =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = primary_database_host)(PORT = 
1521))
    )
    (CONNECT_DATA =
      (SID = primdb1i)
    )
  )


# Primary database application service name

sales =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = primary_database_host)(PORT = 
1521))
    )
    (CONNECT_DATA =
      (SERVICE_NAME = sales)
    )
  )


listener_primary =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = primary_database_host)
(PORT=1521))
  )


listener_true_cache =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST = true_cache_host)(PORT=1521))
  )
```

2.3.2.2 Configure and Start the Local Listener on the True Cache Node

Configure and start the local listener to enable the True Cache node to receive redo from the primary database.

1. Create the `listener.ora` file manually or by using NETCA.

   Here's an example of `listener.ora` on a True Cache node:

   ```
   SID_LIST_LISTENER =
     (SID_LIST =
       (SID_DESC =
         (GLOBAL_DBNAME = tcdb1i)
         (ORACLE_HOME = local_oracle_home_on_true_cache)
       (SID_NAME = tcdb1i)
       )
     )

   LISTENER =
     (DESCRIPTION =
       (ADDRESS = (PROTOCOL = TCP)(HOST = true_cache_host)(PORT = 1521))
     )
   ```

2. In a command window, ensure that the `ORACLE_HOME` and `ORACLE_SID` environment variables are set properly.

   See Configuring the Operating System Environment Variables.

3. Ensure that the `$ORACLE_HOME/bin` directory is in your `PATH` environment variable.

4. Start the listener and verify that it started correctly.

   ```
   lsnrctl start
   lsnrctl status
   ```

2.3.2.3 Copy the Password File or Wallet from the Primary Database to the True Cache Node

To provide authentication for redo transport sessions, copy the primary database's password file or wallet to the True Cache node.

True Cache uses Oracle Net to transport redo data and control messages between the members of a True Cache configuration. These redo transport sessions are authenticated using a password file, so every True Cache in the configuration needs an up-to-date copy of the password file from the primary database.

Copy the primary database’s password file to the appropriate directory on the True Cache node. You can find the password file in the `V$PASSWORDFILE_INFO` view, and on Linux the file is usually in `$ORACLE_HOME/dbs/orapw`*`SID`*.

In the examples in this documentation, the primary database's password file is `$ORACLE_HOME/dbs/orapwprimdb1i` and would be copied to a True Cache node and renamed to `$ORACLE_HOME/dbs/orapwtcdb1i`.

You need to recopy the password file in the following situations:

- Whenever an administrative privilege (`SYSDG`, `SYSOPER`, `SYSDBA`, and so on) is granted or revoked.
- After the password of any user with administrative privileges is changed.

Copy the wallet instead of the password file in the following situations:

- If the primary database uses TLS certificate authentication instead of a password file.
- If the primary database configures Transparent Data Encryption (TDE). Include only the `ROOT` container's master key in the copy for True Cache, which is used to decrypt encrypted redo that's shipped from the primary database.

2.3.2.4 Prepare a PFILE for True Cache

For each True Cache, prepare a `PFILE` with the following parameters.

Table 2-3 Initialization Parameters

Parameter	Description
`TRUE_CACHE=TRUE`	This tells the server that this is a True Cache.
`DB_NAME`	Enter the same value as the one for the primary database.
`DB_UNIQUE_NAME`	Specify a unique name for this True Cache.
`DB_FILES`	Enter the same value as the one for the primary database.
`SGA_TARGET`	Enter the SGA memory size of this True Cache.

Table 2-3 (Cont.) Initialization Parameters

Parameter	Description
REMOTE_LISTENER	Enter the primary database listener. This is not needed for the CREATE TRUE CACHE command, but it's needed to use JDBC. **Note:** For Oracle RAC primary databases, set the REMOTE_LISTENER parameter to SCAN:port. **Note:** Check to see if Valid Node Checking for Registration (VNCR) is configured for the listener. If so, add True Cache to the REGISTRATION_INVITED_NODES_LISTENER parameter in the listener.ora file of the remote listener. (In the example following this table, this is the primary database's listener.ora file.) **Note:** To simplify configuration and avoid connection issues, you can use the LISTENER_NETWORKS parameter instead of specifying REMOTE_LISTENER and LOCAL_LISTENER separately. All listeners within the same network_name will cross-register. For example: `LISTENER_NETWORKS='((NAME=network_name) (LOCAL_LISTENER=(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP) (HOST=true_cache_host) (PORT=1521)))) (REMOTE_LISTENER=primary_database_host:1521))'` For more information, see Best Practices for Load Balancing in a Uniform Configuration.

Table 2-3 (Cont.) Initialization Parameters

Parameter	Description
FAL_SERVER	Enter the network name for the primary database. **Note:** For Oracle RAC primary databases, set the FAL_SERVER parameter to SCAN:port/service_name. If you have standby databases that could become a primary after a primary-standby switchover, True Cache automatically adds the existing standby databases to the FAL_SERVER list. However, if you add a new standby database after configuring True Cache, you need to append the new standby database to the FAL_SERVER parameter for True Cache. To do this, use the following command with a comma-separated list of databases: `ALTER SYSTEM SET FAL_SERVER='existing_databases,new_standby_database' SCOPE=BOTH`
FAL_CLIENT	Enter the True Cache network name. When you start True Cache, it automatically adds that network name to the corresponding primary database LOG_ARCHIVE_DEST_*n* initialization parameter so that the primary database knows where to send online redo log blocks.
DB_CREATE_FILE_DEST	Enter a directory to store the internal True Cache files (for example, control files and temp files).
DB_DOMAIN	This might be necessary if the primary database also sets DB_DOMAIN. Set this to the domain where the True Cache node resides.
WALLET_ROOT and TDE_CONFIGURATION	If the primary database configures TDE with the recommended WALLET_ROOT and TDE_CONFIGURATION parameters, configure them accordingly on this True Cache.

For example, the following PFILE is named init_tcdb1i.ora and stored in the $ORACLE_HOME/dbs/ directory on the True Cache node:

```
true_cache=true
db_name=primaryd
db_unique_name=tcdb1
db_files=200
sga_target=20G
fal_server=primdb1i
fal_client=tcdb1i
instance_name=tcdb1i
```

```
db_create_file_dest=local_directory_on_true_cache_node
local_listener=listener
remote_listener=listener_primary
```

The preceding example assumes that the primary database's `DB_UNIQUE_NAME` initialization parameter is set to `primarydb` and `DB_FILES` is set to `200`.

Related Topics

- TRUE_CACHE
- Initialization Parameters

2.3.2.5 Create and Start True Cache

Follow these steps to create and start True Cache for the first time.

1. Start up True Cache in `NOMOUNT` mode using SQL*Plus.

   ```
   export ORACLE_SID=tcdb1i


   sqlplus / as SYSDBA


   STARTUP NOMOUNT PFILE=$ORACLE_HOME/dbs/init_tcdb1i.ora;
   ```

2. Verify that the password file points to the right file.

   ```
   SELECT file_name FROM v$passwordfile_info;

   FILE_NAME
   ----------------------------------------------------
   /u01/example/oracle/product/main/db_1/dbs/orapwtcdb1i
   ```

3. Create and start True Cache.

   ```
   CREATE TRUE CACHE;
   ```

4. To verify that True Cache is working, see Verifying That True Cache Is Working as Expected.

When you create True Cache, it automatically creates standby redo logs with the appropriate size, control files, temp files (when needed), and an internal `SPFILE` based on the first `PFILE` that you created. This `SPFILE` is used automatically on later `STARTUP` commands without the need to specify any `PFILE` values.

At this point, your applications can use separate physical connections to the primary database and True Cache and choose which connection to use based on whether it's reading or writing.

To use True Cache with the JDBC Thin driver, continue to the next topic to configure the database application services for True Cache.

2.3.3 Configuring Database Application Services for True Cache

To use True Cache with the JDBC Thin driver, for each primary database application service that you want to cache, create a corresponding True Cache database application service.

This makes it easy for applications to switch an existing JDBC connection from a primary database to True Cache without having to change the JDBC connection URL. This is possible with the 23ai JDBC Thin driver by setting the `ReadOnly` parameter of the connection to `TRUE` or `FALSE`.

- Create Database Application Services on the Primary Database
  From the primary database, create or modify the True Cache and primary database services using one of the following methods.
- Verify That the Database Application Services Are Created
  After you create services on the primary database, True Cache automatically inherits their definitions. Using SQL*Plus, make sure that you can see the same results on both the primary database and True Cache.
- Start the Database Application Services
  Using SQL*Plus, start the database application services on both the primary database and True Cache.

2.3.3.1 Create Database Application Services on the Primary Database

From the primary database, create or modify the True Cache and primary database services using one of the following methods.

- For True Cache and single instance primary databases, use the `DBMS_SERVICE` PL/SQL package.
- For Oracle RAC primary databases, use the `srvctl` command line utility with the `–true_cache_service` option. Start the service on the primary instance. Use `srvctl` only for clusterware-managed instances.

You can associate one primary database service with one True Cache service.

Here's an example of creating these two services using the `DBMS_SERVICE` package in a non-clusterware environment:

```
DECLARE
    db_params dbms_service.svc_parameter_array;
BEGIN
   -- create a database application service for True Cache called
SALES_TC using SALES_TC tnsname
   DBMS_SERVICE.CREATE_SERVICE('SALES_TC', 'SALES_TC', db_params);

   -- create a database application service SALES for primary database
using SALES tnsname and associate it to the SALES_TC service name
using TRUE_CACHE_SERVICE attribute
   db_params('true_cache_service') := 'SALES_TC';
   DBMS_SERVICE.CREATE_SERVICE('SALES', 'SALES', db_params);

   -- or, modify an already existing primary database application
service called SALES to associate it to the SALES_TC service name
```

```
using the TRUE_CACHE_SERVICE attribute
   db_params('true_cache_service') := 'SALES_TC';
   DBMS_SERVICE.MODIFY_SERVICE('SALES', db_params);

END;
```

Related Topics

- CREATE_SERVICE Procedure
- MODIFY_SERVICE Procedure
- Server Control Utility Reference

2.3.3.2 Verify That the Database Application Services Are Created

After you create services on the primary database, True Cache automatically inherits their definitions. Using SQL*Plus, make sure that you can see the same results on both the primary database and True Cache.

For example:

```
connect / as SYSDBA


SELECT name, true_cache_service FROM DBA_SERVICES WHERE name='SALES' or
name='SALES_TC';

NAME        TRUE_CACHE_SERVICE
--------    ------------------
SALES_TC
SALES       SALES_TC
```

2.3.3.3 Start the Database Application Services

Using SQL*Plus, start the database application services on both the primary database and True Cache.

Example 2-1   True Cache

```
connect / as SYSDBA


SELECT database_role FROM v$database;

DATABASE_ROLE
-------------
TRUE CACHE


EXEC DBMS_SERVICE.START_SERVICE('SALES_TC');


SELECT service_id, name FROM v$active_services WHERE name='SALES_TC';
```

```
SERVICE_ID  NAME
----------  ---------
        28  SALES_TC
```

Example 2-2   Primary Database

```
connect / as SYSDBA


SELECT database_role FROM v$database;

DATABASE_ROLE
-------------
PRIMARY


EXEC DBMS_SERVICE.START_SERVICE('SALES');


SELECT service_id, name, true_cache_service FROM v$active_services
WHERE name='SALES';

SERVICE_ID   NAME    TRUE_CACHE_SERVICE
----------   ------  ----------------
        29   SALES   SALES_TC
```

Related Topics

- Verify That the Database Application Services Are Created
  After you create services on the primary database, True Cache automatically inherits their definitions. Using SQL*Plus, make sure that you can see the same results on both the primary database and True Cache.
- START_SERVICE Procedure
- V$ACTIVE_SERVICES
- V$DATABASE

2.4 Deploying True Cache for an Oracle RAC Primary Database

Follow these additional steps if you're deploying True Cache for a primary database in an Oracle Real Application Clusters (Oracle RAC) environment.

True Cache needs to be able to access each instance in the Oracle RAC environment. To do this, use the single client access name (SCAN) listener when configuring True Cache.

- If you're configuring True Cache with DBCA, set the `-sourceDBConnectionString` parameter to `SCAN:port/service_name`.
- If you're manually configuring True Cache, use the following settings when preparing the `PFILE` for True Cache:

  - Set the REMOTE_LISTENER parameter to SCAN:port.
  - Set the FAL_SERVER parameter to SCAN:port/service_name.
- Add the True Cache node to the invited node list for the SCAN listener by using the Server Control Utility (srvctl). (Don't manually edit the grid owner in the listener.ora file.)

  For example:

  ```
  srvctl modify scan_listener -invitednodes <true_cache_host> -endpoints
  TCP:<port>
  ```

- When creating the database application services on the primary database, use the srvctl command line utility with the -true_cache_service option. Start the service on the primary instance.

Related Topics

- srvctl modify scan_listener

2.5 Verifying the True Cache Configuration

Verify that True Cache and the database application services are working as expected.

- Verifying That True Cache Is Working as Expected
  To verify that True Cache is applying redo and making progress, check the following queries on True Cache.
- Verifying the True Cache and Primary Database Application Services
  Verify that all database application services are active on both the primary database and True Cache.

2.5.1 Verifying That True Cache Is Working as Expected

To verify that True Cache is applying redo and making progress, check the following queries on True Cache.

1. Run SQL*Plus on the True Cache.

   ```
   sqlplus / as SYSDBA
   ```

2. Enter the following query:

   ```
   SELECT database_role, open_mode FROM v$database;
   ```

   The output should look like this:

   ```
   DATABASE_ROLE    OPEN_MODE
   ---------------- --------------------
   TRUE CACHE       READ ONLY WITH APPLY
   ```

   If OPEN_MODE is READ ONLY WITH APPLY, it means that the True Cache redo apply is actively working.

3. Enter the following query multiple times:

   ```
   SELECT current_scn FROM v$database;
   ```

   If `CURRENT_SCN` is advancing over time, it means that True Cache is moving forward as expected.

4. To find the size of each log file, enter the following query:

   ```
   SELECT THREAD#, SEQUENCE#, BYTES FROM v$standby_log;
   ```

2.5.2 Verifying the True Cache and Primary Database Application Services

Verify that all database application services are active on both the primary database and True Cache.

True Cache

1. Run SQL*Plus on True Cache.

   ```
   sqlplus / as SYSDBA
   ```

2. Enter the following query:

   ```
   SELECT service_id, name FROM v$active_services WHERE
   name='true_cache_service_name';
   ```

   For example:

   ```
   SELECT service_id, name FROM v$active_services WHERE
   name='SALES_TC';

   SERVICE_ID  NAME
   ----------  ---------
           28  SALES_TC
   ```

Primary Database

1. Run SQL*Plus on the primary database.

   ```
   sqlplus / as SYSDBA
   ```

2. Enter the following query:

   ```
   SELECT service_id, name, true_cache_service FROM v$active_services
   WHERE name='primary_db_service_name';
   ```

For example:

```
SELECT service_id, name, true_cache_service FROM v$active_services WHERE
name='SALES';

SERVICE_ID   NAME     TRUE_CACHE_SERVICE
----------   ------   ----------------
        29   SALES    SALES_TC
```

Related Topics

- Configuring Database Application Services for True Cache
  To use True Cache with the JDBC Thin driver, for each primary database application service that you want to cache, create a corresponding True Cache database application service.
- V$ACTIVE_SERVICES

2.6 Enabling DML Redirection

True Cache is a read-only cache, so you can't directly update the cache, but you can use DML redirection to indirectly update the cache.

DML redirection writes data to the primary database and then that data is automatically updated in the cache. This is similar to how Oracle Active Data Guard works. Because DML redirection uses more resources, it's not recommended for update-intensive applications.

To enable and use DML redirection, set the `ADG_REDIRECT_DML` initialization parameter to `TRUE` on True Cache.

> **Note:**
>
> If the user logs in to True Cache with OS authentication, the user can't enable `ADG_REDIRECT_DML`. This is because the True Cache database administrator might not actually have the `SYSDBA` password but is allowed to use OS authentication to manage True Cache as `SYSDBA`. If the user can set `ADG_REDIRECT_DML`, then that user can update primary data as `SYSDBA` without knowing the `SYSDBA` password.

For more on enabling DML redirection, see Managing Physical and Snapshot Standby Databases.

3

Deploying True Cache in Containers

Containers can simplify the deployment of True Cache configurations.

You can use the scripts that are provided in the True Cache container image to automatically create a True Cache configuration. The scripts complete the following configuration tasks:

- Create the primary database and True Cache containers.
- Create and start True Cache.
- Create and start the database application services for the primary database and True Cache.

See the complete instructions for downloading the images and creating the containers in the Oracle Database Container Images README.md.

Create a separate container for each True Cache. Assign different container names, Oracle SIDs, and database application service names, as needed.

4

Deploying True Cache on Oracle Database Free

You can deploy one True Cache on Oracle Database Free.

- Licensing Restrictions for True Cache
  True Cache has the following licensing restrictions.
- Installing and Configuring True Cache on Oracle Database Free
  To install and configure True Cache for Database Free, see the Database Free installation guide for your platform.

4.1 Licensing Restrictions for True Cache

True Cache has the following licensing restrictions.

- You can deploy at most one Oracle Database Free True Cache for an Oracle Database Free primary database. Otherwise, the editions of True Cache and the primary database must match. For example, an Enterprise Edition True Cache can only be deployed with an Enterprise Edition primary database.
- A Free True Cache must comply with all Oracle Database Free licensing restrictions for memory, CPUs, and user data, with the following exceptions:
  - Redo shipping from a Free primary database or to a Free True Cache is not subject to limitations.
  - A Free True Cache will match the `MAX_PDBS` initialization parameter on the primary database.

Related Topics

- Oracle Database Free Licensing Restrictions (Linux)

4.2 Installing and Configuring True Cache on Oracle Database Free

To install and configure True Cache for Database Free, see the Database Free installation guide for your platform.

Configuring True Cache on Oracle Database Free (Linux)

5

Shutting Down and Starting True Cache

You can shut down and start an existing True Cache.

- Shutting Down True Cache
  To shut down True Cache, use the SHUTDOWN command.
- Starting True Cache
  To start an existing True Cache, use the STARTUP command.

5.1 Shutting Down True Cache

To shut down True Cache, use the SHUTDOWN command.

```
connect / as SYSDBA;


SHUTDOWN;
```

> **Note:**
>
> When you shut down True Cache, the cached content is lost. When you restart True Cache, it fetches the content again from the primary database.

5.2 Starting True Cache

To start an existing True Cache, use the STARTUP command.

```
connect / as SYSDBA;


STARTUP;
```

Work is automatically routed to True Cache when it declares that it's ready to support the True Cache service.

6 Monitoring True Cache

You can monitor True Cache with the V$TRUE_CACHE view, and you can use the Automatic Workload Repository (AWR) to gather performance statistics for True Cache.

- Monitoring True Cache with the V$TRUE_CACHE View
  Use the V$TRUE_CACHE view to monitor the overall relationship and health of a True Cache configuration.
- Using the Automatic Workload Repository for True Cache
  You can use the Automatic Workload Repository (AWR) to gather performance statistics for True Cache.

6.1 Monitoring True Cache with the V$TRUE_CACHE View

Use the V$TRUE_CACHE view to monitor the overall relationship and health of a True Cache configuration.

You can query the V$TRUE_CACHE view on the primary database and True Cache. On True Cache, the view displays a single row for the primary database that it connects to. On the primary database, the view shows one row for each True Cache that's connected to the primary database. Each row displays the status for the True Cache.

To use this view, enter the following query:

```
SELECT * FROM v$true_cache;
```

The following example shows output on True Cache:

```
MY_DG_ID   REMOTE_DG_ID DEST_ID TRUE_CACHE_NAME PRIMARY_NAME STATUS
REMOTE_VERSION CON_ID
---------- ------------ ------- --------------- ------------ -------
-------------- ------
2976625076    626116455       0 TCDB1           PRIMARYD     HEALTHY
23.0.0.0.0          0
```

The following example shows output on a primary database:

```
MY_DG_ID  REMOTE_DG_ID DEST_ID TRUE_CACHE_NAME PRIMARY_NAME STATUS
REMOTE_VERSION  CON_ID
--------- ------------ ------- --------------- ------------ -------
--------------  ------
626116455   2976625076       2 TCDB1           PRIMARYD     HEALTHY
23.0.0.0.0           0
626116455   2499211322       3 TCDB2           PRIMARYD     HEALTHY
23.0.0.0.0           0
```

- V$TRUE_CACHE Columns
  The following table describes the columns in the V$TRUE_CACHE view.

6.1.1 V$TRUE_CACHE Columns

The following table describes the columns in the V$TRUE_CACHE view.

Table 6-1 V$TRUE_CACHE Columns

Column	Description
MY_DG_ID	The Data Guard ID (DGID) that uniquely identifies the True Cache or primary database in the True Cache configuration.
REMOTE_DG_ID	On True Cache, this is the primary database's DGID. On the primary database, it's the True Cache DGID.
DEST_ID	On True Cache, this is always 0. On the primary database, it's the corresponding destination ID (*n*) of the LOG_ARCHIVE_DEST_*n* initialization parameter. The LOG_ARCHIVE_DEST_*n* initialization parameter specifies the network name of the True Cache so that the primary database knows where to send online redo log blocks.
TRUE_CACHE_NAME	The unique name of the True Cache (DB_UNIQUE_NAME in the PFILE).
PRIMARY_NAME	The unique name of the primary database (DB_UNIQUE_NAME in the PFILE).
STATUS	Describes the current status of the True Cache. If no issues are found, it displays HEALTHY. Otherwise, it displays an error description. For example: ORA-01034: The Oracle instance is not available for use. Start the instance.
REMOTE_VERSION	On True Cache, this is the database version of the primary database. On the primary database, this is the database version of the True Cache.
CON_ID	The root container ID.

Related Topics

- V$TRUE_CACHE

6.2 Using the Automatic Workload Repository for True Cache

You can use the Automatic Workload Repository (AWR) to gather performance statistics for True Cache.

You can run these steps on the primary database or on True Cache.

1. Enable the AWR snapshot service for True Cache by entering the following SQL commands:

   ```
   alter session set container = CDB$ROOT;


   BEGIN
     dbms_workload_repository.enable_snapshot_service(
       dbid => NULL,
       snap_type => 'WORKLOAD',
       snap_action => 'CREATE');
   END;
   ```

2. Generate the AWR snapshot report by going to the `$ORACLE_HOME` directory and running the following script:

   ```
   @?/rdbms/admin/awrrpti.sql
   ```

3. Specify whether you want an HTML or a text report. For example:

   ```
   Enter value for report_type: text
   ```

   The output displays a list of available database identifiers and instance numbers.

4. Enter the database identifier (`dbid)` and instance number (`inst_num)` for True Cache. For example:

   ```
   Enter value for dbid: 3309173529
   Using 3309173529 for database Id
   Enter value for inst_num: 1
   ```

5. Specify the number of days for which you want to list snapshot IDs. For example:

   ```
   Enter value for num_days: 2
   ```

   The output displays a list of existing snapshots for the specified time range.

6. Specify beginning and ending snapshot IDs for the workload repository report. For example:

   ```
   Enter value for begin_snap: 150
   Enter value for end_snap: 160
   ```

7. Enter a report name or accept the default report name. For example:

   ```
   Enter value for report_name:
   Using the report name awrrpt_1_150_160
   ```

Related Topics

- Automatic Workload Repository (AWR)
- Generating an AWR Report Using the Command-Line Interface
- DBMS_WORKLOAD_REPOSITORY

7
Using Oracle True Cache in Your Applications

When True Cache is configured, applications must decide whether to query data from True Cache or the primary database.

7.1 Methods for Connecting to True Cache

Applications can use True Cache in two ways.

- The application maintains *two physical connections*: one to the primary database and one to True Cache. Each connection has a database application service, and the application chooses which connection to use based on whether it's reading or writing. You can use this model with any existing client drivers and any programming language.

  The application sends queries that don't need to see the most current data to True Cache through a True Cache database application service. The application sends other queries and updates to the primary database through the primary database application service.

- The application maintains *one logical connection* that uses the database application service for the primary database. The JDBC Thin driver (starting with Oracle Database 23ai) maintains physical connections to the primary database and True Cache. This model only works with Java applications.

  The application switches between the primary database and True Cache without having to specify an instance name. The application uses special calls to flag the logical connection as read-only or read-write. If it's read-only, the query is sent to True Cache. Otherwise, it's sent to the primary database.

  To enable the True Cache functionality for Java applications, set the value of the `oracle.jdbc.useTrueCacheDriverConnection` property to `true`. After you enable True Cache, the JDBC Thin driver uses the standard `java.sql.Connection.setReadOnly(boolean)` and `java.sql.Connection.isReadOnly()` methods to mark a connection as read-only. By default, the read-only mode is `false` for a connection.

See the following technical architecture diagram for details and examples:

True Cache Application

Related Topics

- Support for Oracle True Cache
- setReadOnly(boolean) method
- isReadOnly() method

7.2 Sample Java Code Using the JDBC Thin Driver

The JDBC method is illustrated by the following Java code using the JDBC Thin driver.

> **WARNING:**
>
> Do not use this code in a production environment.

The core of this basic code is in the `test1` function, which does the following:

1. Creates a connection to the primary database by using the JDBC Thin driver with either a default user name and password or values that are specified on the command line when invoking the program.
2. Prints some basic information about the primary database connection to verify that the application is connected to the database.
3. Changes the `isReadOnly` connection parameter to `TRUE` to automatically switch the connection to True Cache and verify that it's connected to it.

```
import java.sql.*;
import java.io.*;
import java.util.*;
import java.util.logging.FileHandler;
import java.util.logging.Level;
import java.util.logging.Logger;

import oracle.jdbc.driver.OracleLog;
import oracle.jdbc.pool.OracleDataSource;

public class TrueCache {
  static String url_primary =
"jdbc:oracle:thin:@primary_database_host:1521/SALES";
  static String user = "username";
  static String password = "password";

  public static void main(String args[]) {
    try {
      TrueCache t = new TrueCache();
      if(args != null && args.length >0 ) {
        url_primary = args[0];
        user = args[1];
        password = args[2];
      }
      t.test1();
    } catch (Exception e) {
      e.printStackTrace();
    }
  }

  public void test1() throws SQLException {
    show("Basic test to connect to primary and True Cache");
    try {
      show("Get Connection from True Cache instance using primary db and 
properties");
      OracleDataSource ods1 = new OracleDataSource();
      show("URL = " + url_primary);
      ods1.setURL(url_primary);
      ods1.setUser(user);
      ods1.setPassword(password);
      ods1.setConnectionProperty("oracle.jdbc.useTrueCacheDriverConnection", 
"true");

      Connection conn1 = ods1.getConnection();

      show("isReadonly " + conn1.isReadOnly());
      verifyConnection(conn1);  // This is connected to Primary

      conn1.setReadOnly(true);
      show("isReadonly " + conn1.isReadOnly());
      verifyConnection(conn1); // This is connected to True Cache

      conn1.close();
    } catch (SQLException sqex) {
      show("test1 -- failed" + sqex.getMessage()+":"+sqex.getCause());
      sqex.printStackTrace();
    }
    show("The end");
  }

  public void verifyConnection(Connection conn) {
    try {
      Statement statement = conn.createStatement();
      ResultSet rs = statement.executeQuery("SELECT database_role from 
v$database");
      ResultSetMetaData rsmd = rs.getMetaData();
      int columnsNumber = rsmd.getColumnCount();
      rs.next();
      show("Database role : " + rs.getString(1));
      rs.close();
      ResultSet resultSet = statement.executeQuery("SELECT 
SYS_CONTEXT('userenv', 'instance_name') as instance_name"
          + ", SYS_CONTEXT('userenv', 'server_host')" + " as server_host" + 
", SYS_CONTEXT('userenv', 'service_name')"
          + " as service_name" + ", SYS_CONTEXT('USERENV','db_unique_name')" 
+ " as db_unique_name" + " from sys.dual");
      resultSet.next();
      show("instance_name : " + resultSet.getString("instance_name"));
      show("server_host : " + resultSet.getString("server_host"));
      show("service_name : " + resultSet.getString("service_name"));
```

```
        show("db_unique_name : " +
resultSet.getString("db_unique_name"));
        resultSet.close();
        statement.close();
      } catch (SQLException sqex) {
        show("verifyConnection failed " + sqex.getMessage());
      }
    }

    public void show(String msg) {
      System.out.println(msg);
    }
}
```

7.3 Sample Java Code for Applications Using a Native Connection to the Database

The following Java code illustrates how to use True Cache with applications that use an object-relational mapping (ORM) framework.

This is another example of maintaining only one logical connection to a database application service running on the primary database and letting the driver handle the underlying physical connections. It shows how to get the underlying connections from the framework so you can set the necessary parameter.

> **WARNING:**
>
> Do not use this code in a production environment.

1. Add the `oracle.jdbc.useTrueCacheDriverConnection=true` connection property to the connection URL in the `persistence.xml` file.
2. In the methods that need to use a True Cache connection, unwrap the session, use the `doWork` method to retrieve the underlying JDBC connection, and set the necessary read-only parameter for the connection. For example:

```
Session session = em.unwrap(Session.class);
session.doWork(new Work() {

   @Override
   public void execute(Connection con) throws SQLException {
      // if it is connected to primary, I want the connection to
switch to true_cache now
      if(!con.isReadOnly()) {
         // The below code demonstrates the connection is to
primary now
         System.out.println("Query getting executed on:");
         Statement statement = con.createStatement();
         ResultSet rs = statement.executeQuery("SELECT
database_role from v$database");
         rs.next();
```

```
                  System.out.println("Database role : " + rs.getString(1));
                  rs.close();

                  // Now switching the connections to true cache
                  con.setReadOnly(true);
                  // The below code demonstrates the connection is to true cache
now
                  System.out.println("After set read only true, Query getting
executed on:");
                  Statement statement2 = con.createStatement();
                  ResultSet rs2 = null;

                  rs2 = statement2.executeQuery("SELECT database_role from
v$database");
                  rs2.next();
                  System.out.println("Database role : " +
rs2.getString(1));
                  rs2.close()
             }
       }
});
```

7.4 Best Practices for Load Balancing in a Uniform Configuration

If multiple True Caches exist and serve the same database application service, the listener automatically distributes and load balances sessions to each cache. In a uniform configuration, the listener picks a True Cache randomly or based on load, and all True Caches cache the same data.

A uniform configuration uses runtime connection load balancing (CLB), which routes requests to the True Cache that offers the best performance. For this to work, all True Caches' REMOTE_LISTENER parameters should point to the same listener, which is also the primary database's listener (whether it's single instance primary database's local listener or an Oracle RAC primary database's SCAN listener).

The application's JDBC URL should point to the primary database. Then the JDBC Thin driver creates one logical connection and multiple physical connections to the primary database and to each True Cache. Setting the setReadOnly(true) flag in the Java code reroutes connections to the True Caches automatically. The True Caches are registered with the primary database listener, and they also send their load statistics to the primary database listener. You should see an even distribution between True Caches.

To simplify configuration and avoid connection issues, consider using the LISTENER_NETWORKS initialization parameter instead of specifying REMOTE_LISTENER and LOCAL_LISTENER separately. With LISTENER_NETWORKS, all listeners within the same network name will cross-register. For example:

```
listener_networks='((NAME=net1)
(LOCAL_LISTENER=(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)(HOST=tc1.example.com)
(PORT=1521))))(REMOTE_LISTENER=scan_primary:1521))',
     '((NAME=net2)(LOCAL_LISTENER=(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP)
(HOST=tc1.example.com)(PORT=1521))))(REMOTE_LISTENER=scan_standby:1521))'
```

For more about configuring remote listeners and listener networks, see Prepare a PFILE for True Cache.

Related Topics

- About Run-Time Connection Load Balancing

8
Complementary Caching Features

True Cache can be combined with other Oracle Database features to improve performance.

- Server-Side Result Set Cache
  The server-side result cache is used to cache the results of the current queries, query fragments, and PL/SQL functions in memory. The cached results are used in future uses of the query, query fragment, or PL/SQL function.
- KEEP Buffer Pool
  You can use the `KEEP` buffer pool to keep frequently accessed tables persistent in the buffer pool.
- Database Smart Flash Cache
  If the data that you want to cache doesn't fit in memory, you can expand the capacity of True Cache by adding flash devices. You do this by configuring Database Smart Flash Cache.

8.1 Server-Side Result Set Cache

The server-side result cache is used to cache the results of the current queries, query fragments, and PL/SQL functions in memory. The cached results are used in future uses of the query, query fragment, or PL/SQL function.

The cached results stay in the result cache memory part of the SGA. A cached result is automatically invalidated whenever a database object used in its creation is successfully changed. See Tuning the Result Cache for more information about configuring the server-side result set cache.

To enable the server-side result cache on True Cache, issue the following command on the primary database:

```
ALTER TABLE table_name RESULT_CACHE (STANDBY ENABLE);
```

Support for server-side result set caching is available for both JDBC Thin and JDBC Oracle Call Interface (JDBC OCI) drivers.

Related Topics

- Using the Result Cache on Physical Standby Databases

8.2 KEEP Buffer Pool

You can use the `KEEP` buffer pool to keep frequently accessed tables persistent in the buffer pool.

- Overview of Using the KEEP Buffer Pool with True Cache
  You can assign different objects to the `KEEP` buffer pool on different True Caches.

- Configuring DB_KEEP_CACHE_SIZE on True Cache
  To configure the KEEP buffer pool on True Cache, set the DB_KEEP_CACHE_SIZE initialization parameter to a large size (such as 10 GB).
- Assigning Objects to the KEEP Buffer Pool for True Cache
  After configuring the KEEP buffer pool, assign the object to the KEEP buffer pool for True Cache.
- Removing an Object's KEEP Buffer Pool Assignment for True Cache
  You can use DBMS_CACHEUTIL.TRUE_CACHE_UNKEEP() to remove objects from the KEEP buffer pool.
- Viewing a List of KEEP Objects on True Cache
  Use the V$TRUE_CACHE_KEEPOBJ view to see which objects are assigned to the KEEP buffer cache for True Cache.

8.2.1 Overview of Using the KEEP Buffer Pool with True Cache

You can assign different objects to the KEEP buffer pool on different True Caches.

At a high level, this involves the following steps:

1. Configure DB_KEEP_CACHE_SIZE on True Cache.
2. Assign objects on True Cache to the KEEP buffer pool.

As with the primary database, when an object is assigned to the KEEP buffer pool on True Cache, the object's blocks are loaded for a query and then kept in the KEEP buffer pool. When new data is inserted into the object on the primary database, that new data is automatically propagated to the KEEP buffer pool on True Cache through the redo apply mechanism.

To propagate direct load data to True Cache, specify the LOGGING clause in the corresponding DDL or DML, or alter the corresponding object property to LOGGING. The LOGGING clause lets you specify whether certain operations will be logged in the redo log file (LOGGING) or not (NOLOGGING).

- How True Cache Works with the Primary Database Buffer Cache
  By default, marking something KEEP on the primary database also marks it KEEP on True Cache.

Related Topics

- Configuring the KEEP Pool
- logging_clause

8.2.1.1 How True Cache Works with the Primary Database Buffer Cache

By default, marking something KEEP on the primary database also marks it KEEP on True Cache.

To prevent objects that are intended to be KEEP objects only on the primary database from filling up the True Cache KEEP buffer pool, you can use the DBMS_CACHEUTIL.TRUE_CACHE_KEEP procedure to mark objects as KEEP on True Cache. DBMS_CACHEUTIL.TRUE_CACHE_KEEP takes precedence and overrides objects that are marked as KEEP on the primary database.

Also consider the following points:

- `ALTER TABLE KEEP` assignments on the primary database are persistent because `ALTER TABLE` is a DDL. The `DBMS_CACHEUTIL.TRUE_CACHE_KEEP` procedure isn't persistent when True Cache restarts.
- You can't use the `DBMS_CACHEUTIL.TRUE_CACHE_UNKEEP` procedure to unkeep a primary `ALTER TABLE KEEP` assignment. Instead, either don't configure `DB_KEEP_CACHE_SIZE` on True Cache or assign a different object with `DBMS_CACHEUTIL.TRUE_CACHE_KEEP` on True Cache.
- All scans on True Cache use the `CACHE` path instead of the direct (`NOCACHE`) path, except temporary tables that are local to True Cache.

8.2.2 Configuring DB_KEEP_CACHE_SIZE on True Cache

To configure the `KEEP` buffer pool on True Cache, set the `DB_KEEP_CACHE_SIZE` initialization parameter to a large size (such as 10 GB).

For example:

```
ALTER SYSTEM SET DB_KEEP_CACHE_SIZE=size SCOPE=BOTH;
```

Related Topics

- DB_KEEP_CACHE_SIZE
- Setting or Changing Initialization Parameter Values

8.2.3 Assigning Objects to the KEEP Buffer Pool for True Cache

After configuring the `KEEP` buffer pool, assign the object to the `KEEP` buffer pool for True Cache.

To do this, call the `DBMS_CACHEUTIL.TRUE_CACHE_KEEP()` procedure on True Cache.

Example: Nonpartitioned Object

```
EXECUTE DBMS_CACHEUTIL.TRUE_CACHE_KEEP('SYS', 'TABLE1');
```

Example: Partition of a Partitioned Object

```
EXECUTE DBMS_CACHEUTIL.TRUE_CACHE_KEEP('SYS', 'TABLE2', 'TABLE2_PART1');
```

- TRUE_CACHE_KEEP Procedure
  When you call this procedure on True Cache, it assigns the object to the `KEEP` buffer pool on that cache.

8.2.3.1 TRUE_CACHE_KEEP Procedure

When you call this procedure on True Cache, it assigns the object to the `KEEP` buffer pool on that cache.

To use this procedure, the `DB_KEEP_CACHE_SIZE` initialization parameter must be configured on True Cache.

Syntax

```
DBMS_CACHEUTIL.TRUE_CACHE_KEEP(
   schema                  IN VARCHAR2,
   obj                     IN VARCHAR2,
   partition               IN VARCHAR2 := NULL);
```

Parameters

Table 8-1 TRUE_CACHE_KEEP Procedure Parameters

Parameter	Description
schema	The name of the schema for the object.
obj	The name of the object.
partition	• If the object is not partitioned, then this is NULL. • If the object is partitioned, use the name of the partition segment. • If it's a composited partitioned object, use the name of the subpartition segment.

Related Topics

- TRUE_CACHE_KEEP Procedure

8.2.4 Removing an Object's KEEP Buffer Pool Assignment for True Cache

You can use DBMS_CACHEUTIL.TRUE_CACHE_UNKEEP() to remove objects from the KEEP buffer pool.

Note that the block is not removed immediately. Instead, it will be naturally aged out as new blocks for other objects are brought into the KEEP buffer pool. Also, the KEEP buffer pool assignment on True Cache is only remembered while True Cache is up.

Example: Nonpartitioned Object

```
EXECUTE DBMS_CACHEUTIL.TRUE_CACHE_UNKEEP('SYS', 'TABLE1');
```

Example: Partition of a Partitioned Object

```
EXECUTE DBMS_CACHEUTIL.TRUE_CACHE_UNKEEP('SYS', 'TABLE2',
'TABLE2_PART1');
```

- TRUE_CACHE_UNKEEP Procedure
  When an object on True Cache no longer needs to be in the KEEP buffer pool, use this procedure to remove the object's KEEP assignment.

8.2.4.1 TRUE_CACHE_UNKEEP Procedure

When an object on True Cache no longer needs to be in the KEEP buffer pool, use this procedure to remove the object's KEEP assignment.

Syntax

```
DBMS_CACHEUTIL.TRUE_CACHE_UNKEEP(
   schema                  IN VARCHAR2,
   obj                     IN VARCHAR2,
   partition               IN VARCHAR2 := NULL);
```

Parameters

Table 8-2   TRUE_CACHE_UNKEEP Procedure Parameters

Parameter	Description
schema	The name of the schema for the object.
obj	The name of the object.
partition	• If the object is not partitioned, then this is NULL. • If the object is partitioned, use the name of the partition segment. • If it's a composited partitioned object, use the name of the subpartition segment.

Related Topics

- TRUE_CACHE_UNKEEP Procedure

8.2.5 Viewing a List of KEEP Objects on True Cache

Use the V$TRUE_CACHE_KEEPOBJ view to see which objects are assigned to the KEEP buffer cache for True Cache.

Example 1

```
SELECT * FROM v$true_cache_keepobj;

TS_NUMBER  DATA_OBJECT_ID  CON_ID
---------- --------------  ------
         5          72948       3
         5          72950       3
```

Example 2

```
SELECT owner as schema, object_name as keepobj, subobject_name as partition,
o.data_object_id
FROM dba_objects o, sys_objects so, v$true_cache_keepobj vtck
WHERE o.data_object_id = so.object_id
  AND vtck.con_id = sys_context('USERENV', 'CON_ID')
  AND so.ts_number=vtck.ts_number
  AND o.data_object_id = vtck.data_object_id;
```

```
SCHEMA  KEEPOBJ  PARTITION     DATA_OBJECT_ID
------  -------  ------------  --------------
SYS     TABLE1                          72948
SYS     TABLE2   TABLE2_PART1           72950
```

- V$TRUE_CACHE_KEEPOBJ Columns
  The following table describes the columns in the V$TRUE_CACHE_KEEPOBJ view.

8.2.5.1 V$TRUE_CACHE_KEEPOBJ Columns

The following table describes the columns in the V$TRUE_CACHE_KEEPOBJ view.

Column	Description
TS_NUMBER	The tablespace number.
DATA_OBJECT_ID	The dictionary object number of the segment that contains the object.
CON_ID	The root container ID.

8.3 Database Smart Flash Cache

If the data that you want to cache doesn't fit in memory, you can expand the capacity of True Cache by adding flash devices. You do this by configuring Database Smart Flash Cache.

To enable Database Smart Flash Cache, see Configuring Database Smart Flash Cache.

> **Note:**
>
> True Cache is not currently supported for primary databases with Transparent Data Encryption (TDE) encrypted tablespaces.

9

Deleting True Cache

You can delete True Cache with Oracle DBCA or manually.

- Using Oracle DBCA to Delete True Cache
  You can use Oracle DBCA to clean up the True Cache services and delete True Cache.
- Deleting True Cache Manually
  To delete True Cache manually, use the following commands.

9.1 Using Oracle DBCA to Delete True Cache

You can use Oracle DBCA to clean up the True Cache services and delete True Cache.

- Cleaning Up the True Cache Services from the Primary Database
  To delete True Cache, you need to clean up its corresponding services on the primary database.
- Deleting True Cache from the True Cache Node
  Run the following command to delete True Cache from the True Cache node.

9.1.1 Cleaning Up the True Cache Services from the Primary Database

To delete True Cache, you need to clean up its corresponding services on the primary database.

Run the following command on the primary database to delete all the services that correspond to the True Cache that you're planning to delete.

```
ORACLE_HOME/bin/dbca -configureDatabase -cleanupTrueCacheInstanceService -
sourceDB primary_sid_or_db_unique_name -trueCacheConnectString
true_cache_easy_connect_string -serviceName primary_service_name -silent
```

Example:

```
$ORACLE_HOME/bin/dbca -configureDatabase -cleanupTrueCacheInstanceService -
sourceDB primdb1i -trueCacheConnectString tc.example.com:1522/
tcdb1.example.com -serviceName sales -silent
```

For descriptions of the parameters, see configureDatabase.

9.1.2 Deleting True Cache from the True Cache Node

Run the following command to delete True Cache from the True Cache node.

```
ORACLE_HOME/bin/dbca -deleteDatabase -sourceDB primary_sid_or_db_unique_name
-silent
```

Example:

```
$ORACLE_HOME/bin/dbca -deleteDatabase -sourceDB tcdb1 -silent
```

9.2 Deleting True Cache Manually

To delete True Cache manually, use the following commands.

```
connect / as SYSDBA;


shutdown immediate;


startup mount exclusive restrict;


drop database;
```

> **Note:**
>
> This does *not* drop the primary database. It deletes the True Cache SPFILE; drops the True Cache control, standby log, and temporary files; and removes LOG_ARCHIVE_DEST entries from the primary database.

A

True Cache Error Messages

View error messages that you might encounter when using True Cache.

Select an error message to read the full description in the Error Help Portal.

- ORA-61850, Oracle version mismatch between primary database and True Cache.
- ORA-61851, True Cache instance does not allow string operation.
- ORA-61852, CREATE TRUE CACHE failed.
- ORA-61853 Cannot start True Cache.
- ORA-61854 Cannot start True Cache with non True Cache control file.
- ORA-61855 Invalid TRUE_CACHE_CONFIG attribute.
- ORA-61856 True Cache requires a valid parameter value for string.
- ORA-61857 Communication with the primary failed.
- ORA-61858 Unable to translate FAL_SERVER initialization parameter string.
- ORA-61859 There was an error when executing a remote procedure call (op: string-string-string) at primary server string.
- ORA-61860 True Cache instance is being shutdown because it has been disconnected from the primary for too long.
- ORA-61861 Unable to receive redo since standby redo log is not available.
- ORA-61862 String is not configured as a multitenant database.
- ORA-61863 True Cache could not be started because there was a configuration parameter mismatch with the primary. Restart the database.
- ORA-61864 True Cache failed to read metadata from the primary.
- ORA-61865 The TNS alias string could not be translated.
- ORA-61866 Some static parameters (string) differed from the value in the primary database. These parameters have been fixed and the database will be shut down.
- ORA-61867 Oracle True Cache failed to restart. Shutting down.
- ORA-61868 True Cache entry corresponding to the hash ID string not found in a LOG_ARCHIVE_DEST_n parameter.
- ORA-61869 Unable to connect to True Cache.
- ORA-61870 True Cache requires ARCHIVELOG mode to be enabled at the primary database.
- ORA-61871 Cannot fetch data block from the primary database (file # string, block # string).
- ORA-61872 The string command in pluggable database string failed because some process is keeping a lock on it for too long. True cache will be restarted.

- ORA-61873 True Cache could not be started because the system parameter 'string' is different in the PRIMARY database. Remove the parameter from the True Cache initialization file.
- ORA-61874 Cannot fetch data block from the primary database (file # string, block # string).
- ORA-61875 There was a failure adding the system parameter LOG_ARCHIVE_DEST_n at the primary database. True Cache database cannot be opened without it.
- ORA-61876 Failed to open True Cache because the primary database is configured for shared server.
- ORA-61877 True Cache recovery lags too far behind, exceeding string seconds.
- ORA-61878 PDB (con_id:string) is not open on the primary database.
- ORA-61879 db_keep_cache_size is not configured on True Cache.
- ORA-61880 This operation (string) is only allowed on True Cache instance.
- ORA-61881 Failed to open True Cache because there has been an error reading metadata (string) from the primary database string.