# Contents

About This Guide ............................................................................................................................... 7  
About Database Users .......................................................................................................................... 7  
Migrating Databases and Database Schema ....................................................................................... 9  
   Using the Migrate Database Wizard .............................................................................................. 9  
      Running a Schema Validation on the Database ........................................................................ 9  
      Running the Data Pump Utility Using the Wizard .................................................................. 10  
      Running the Data Pump Utility From the Command Line ..................................................... 11  
Migrating Database Schema ............................................................................................................... 12  
   Prerequisites for Schema Migration ......................................................................................... 12  
   Migrating P6 EPPM Schema to P6 Professional with Oracle .................................................. 12  
   Migrating P6 EPPM Schema to P6 Professional with Microsoft SQL Server .......................... 13  
Creating and Updating P6 EPPM Configurations ......................................................................... 15  
   Creating P6 EPPM Configurations ............................................................................................... 15  
   Updating P6 EPPM Configurations .............................................................................................. 16  
Setting Up the P6 EPPM Database .................................................................................................. 17  
   Installing a P6 EPPM Database with Oracle ............................................................................. 17  
   Installing a P6 EPPM Database with Microsoft SQL Server ..................................................... 18  
   Upgrading a P6 EPPM Database with Oracle Database ............................................................ 19  
   Upgrading a P6 EPPM Database with Microsoft SQL Server ................................................... 20  
Changing the Database Base Currency .......................................................................................... 21  
   The Base Currency ...................................................................................................................... 21  
   Reviewing Currency Choices ..................................................................................................... 21  
   Changing the Base Currency ...................................................................................................... 21  
Private Database Credentials for P6 EPPM .................................................................................. 22  
   Resetting Private Database Passwords ..................................................................................... 22  
   Adding Private Database Logins for P6 EPPM ...................................................................... 23  
   Modifying Private Database Logins for P6 EPPM ..................................................................... 23  
   Deleting Private Database Logins for P6 EPPM .................................................................... 24  
Setting Up Cloud Sync ..................................................................................................................... 27  
   Synchronizing data from P6 EPPM to Oracle Prime Projects ................................................ 27  
   Synchronizing data from Oracle Prime Projects to P6 EPPM ............................................... 33  
Configuring Background Processes and Jobs ................................................................................ 39  
   RDBMS Scheduler Configuration .............................................................................................. 39  
   Database Settings Table ............................................................................................................. 39  
   Reading Setting Values .............................................................................................................. 40  
      Using Code to Read Setting Values with Oracle ................................................................. 40
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using Code to Read Setting Values with Microsoft SQL Server</td>
<td>41</td>
</tr>
<tr>
<td>Writing Setting Values</td>
<td>41</td>
</tr>
<tr>
<td>Using Code to Write Setting Values with Oracle</td>
<td>41</td>
</tr>
<tr>
<td>Using Code to Write Setting Values with Microsoft SQL Server</td>
<td>41</td>
</tr>
<tr>
<td>Tracking Background Job Execution</td>
<td>42</td>
</tr>
<tr>
<td>High Level Status Settings</td>
<td>42</td>
</tr>
<tr>
<td>The BGPLOG Table</td>
<td>42</td>
</tr>
<tr>
<td>Monitoring Processes and Procedures</td>
<td>45</td>
</tr>
<tr>
<td>PAUDIT Auditing</td>
<td>45</td>
</tr>
<tr>
<td>Auditing Level Configuration</td>
<td>45</td>
</tr>
<tr>
<td>Simple Configuration</td>
<td>45</td>
</tr>
<tr>
<td>Detailed Configuration</td>
<td>46</td>
</tr>
<tr>
<td>Auditing Status</td>
<td>46</td>
</tr>
<tr>
<td>Options Setting</td>
<td>47</td>
</tr>
<tr>
<td>SETTINGS_WRITE_STRING Procedure</td>
<td>48</td>
</tr>
<tr>
<td>The Audit Table</td>
<td>48</td>
</tr>
<tr>
<td>Session Auditing</td>
<td>50</td>
</tr>
<tr>
<td>Column Audit Data</td>
<td>50</td>
</tr>
<tr>
<td>Tuning the P6 EPPM Database</td>
<td>51</td>
</tr>
<tr>
<td>Oracle Database Tuning</td>
<td>51</td>
</tr>
<tr>
<td>Partitioning Oracle Database Tables for P6 EPPM Schema</td>
<td>51</td>
</tr>
<tr>
<td>Gathering Statistics for Cost Based Optimizations</td>
<td>53</td>
</tr>
<tr>
<td>Viewing the USESSION Table for GET_SAFETY_DATE</td>
<td>53</td>
</tr>
<tr>
<td>Rebuilding the P6 EPPM Index Table</td>
<td>53</td>
</tr>
<tr>
<td>Where to Find Additional Oracle Database Tuning Information</td>
<td>54</td>
</tr>
<tr>
<td>Microsoft SQL Server Database Tuning</td>
<td>54</td>
</tr>
<tr>
<td>Where to Find Additional Microsoft SQL Server Database Tuning Information</td>
<td>54</td>
</tr>
<tr>
<td>Isolating Snapshots</td>
<td>55</td>
</tr>
<tr>
<td>Rebuilding the P6 EPPM Index</td>
<td>55</td>
</tr>
<tr>
<td>Gathering Statistics</td>
<td>55</td>
</tr>
<tr>
<td>General Tuning</td>
<td>57</td>
</tr>
<tr>
<td>Background Processes and Clean Up in P6 EPPM</td>
<td>57</td>
</tr>
<tr>
<td>SYMON (System Monitor) Procedures</td>
<td>57</td>
</tr>
<tr>
<td>OBSPROJ_PROCESS_QUEUE Procedure</td>
<td>58</td>
</tr>
<tr>
<td>USESSION_CLEANUP_EXPIRED Procedure</td>
<td>58</td>
</tr>
<tr>
<td>Tracking Concurrent Usage of P6 EPPM</td>
<td>60</td>
</tr>
<tr>
<td>DAMON (Data Monitor) Procedures</td>
<td>61</td>
</tr>
<tr>
<td>BGPLOG_CLEANUP Procedure</td>
<td>61</td>
</tr>
<tr>
<td>REFRDEL_CLEANUP Procedure</td>
<td>62</td>
</tr>
<tr>
<td>REFRDEL Bypass Procedure</td>
<td>63</td>
</tr>
<tr>
<td>CLEANUP_PRMQUEUE Procedure</td>
<td>63</td>
</tr>
<tr>
<td>USESSION_CLEAR_LOGICAL_DELETES Procedure</td>
<td>64</td>
</tr>
<tr>
<td>CLEANUP_LOGICAL_DELETES Procedure</td>
<td>64</td>
</tr>
</tbody>
</table>
About This Guide

Scope
This guide describes how to:

- Migrate your P6 EPPM databases between Microsoft SQL Server database and Oracle database or your database schema between P6 EPPM and P6 Professional.
- Create or update configurations for the Primavera P6 Administrator using dbconfig.
- Configure database settings and the scheduler for background processes and jobs.
- Monitor processes and procedures using PAUDIT audit, BGPLOG table, or high level statuses.
- Perform database maintenance tasks using the SYMON and DAMON utilities.

Audience
Database administrators should use this guide.

Using This Guide
This guide assumes you can perform common database administration procedures and have experience using the command line.

About Database Users

Schema User (Oracle database only)
Default: ADMUSER
Description: The administrative user owns most database objects within the schema including tables, indexes and constraints, procedures, triggers, and functions, and is primarily used during database creation and upgrades.

Privileged User
Default: PRIVUSER
Description: The following is true for the privileged user:

- Owns views to most of the schema user tables that filter out logically deleted data.
- Has SELECT, INSERT, UPDATE, and DELETE privileges on all of the tables owned by the schema user.
- Has EXECUTE privileges on all schema procedures.
- Is used to facilitate communication between P6 EPPM and P6 Professional.
- Owns the synonyms for objects that do not have a view, which means that references to tables that are owned by the schema owner do not have to be fully qualified (Oracle database only).
**Public User**  
Default: PUBUSER  
Description: The public user has few permissions in the P6 EPPM database. It has SELECT privileges on the PUBUSER table, which is used to connect a P6 EPPM to the database.

**Background Job User (Oracle database only)**  
Default: BGJOBUSER  
Description: The background job user initiates and runs P6 EPPM background jobs.

**Reporting User**  
Default: PXRPTUSER  
Description: The reporting user owns views to the P6 EPPM extended schema for enterprise reporting.

**Read-Only Administrative User (Oracle database only)**  
Default: ROADMUSER  
Description: The read-only user has read-only access to the tables owned by the schema user. The read-only user has the same views as reporting user.
Migrating Databases and Database Schema

Migrate.bat is a java-based tool that enables you to migrate data between Microsoft SQL Server database and Oracle database as well as database schema from P6 EPPM to P6 Professional. Migrate.bat launches the Migrate Database wizard.

For more information about migrating data between databases, see Using the Migrate Database Wizard (on page 9).

For more information about migrating database schema from P6 EPPM to P6 Professional, see Migrating Database Schema (on page 12).

In This Section

Using the Migrate Database Wizard ................................................................. 9
Migrating Database Schema ................................................................. 12

Using the Migrate Database Wizard

The Migrate Database wizard is a java-based tool that enables you to migrate data from a Microsoft SQL Server database to an Oracle database.

Use the Migrate Database wizard to:

- Import data into a newly created database.
- Repair a damaged database to correct database object issues (such as constraints or views).

Do not use the Migrate Database wizard to:

- Convert the database type from EPPM to PPM or from PPM to EPPM.
- Upgrade from one database version to another.
- Import data into an existing database.
- Correct data-specific issues.
- Correct schema-related issues. It is not ideal for large databases (more than 10 GB) due to the performance of the wizard. For larger databases, try first to manually resolve schema-related issues.

Before running the Migrate Database wizard, you must first run a schema validation on the database. See Running a Schema Validation on the Database (on page 9) for details.

Running a Schema Validation on the Database

The schema validation utility finds missing, extra, and modified schema objects for a P6 database.

To run a schema validation, complete the following steps:

1) In the software download, locate the utility in the Database Download folder.
2) Edit the utility similar to the following:

```bash
@echo off
REM --------------------------------------
REM Run the Primavera Schema Validation Tool
REM --------------------------------------

SET JAR_FILE_DIR=lib
SET JAR_FILE_NAME=dbmt.jar
SET DB_SCHEMA=ppm

SET JVM_PROPERTIES=-Ddbmt.dbschema.package.location=%JAR_FILE_DIR%
-Dbprimavera.db.schema=%DB_SCHEMA%
-Dcom.primavera.database.logfile=SchemaValidation.log
SET DBMT_ACTION=application/ppmschemaval
SET DBMT_COMMAND=schemavalpm
set JAVA_HOME=C:\Program Files\Java\jre6
IF NOT EXIST "%JAVA_HOME%\bin\java.exe" (echo JAVA_HOME IS NOT SET pause goto :EXIT)
```

3) Run validate.bat (with Windows) or validate.sh (with UNIX or Linux).

4) With an Oracle database in an Enterprise environment, enter values for the following:
   - **Username**: Use the appropriate ADMUSER username and password.
   - **Database host address**: Database server name or IP address.
   - **Database host port**: The port your Oracle listener is listening on. The default port is 1521.
   - **Database name (SID)**: Enter the Oracle SID for your database.

   With a Microsoft SQL Server database, enter values for the following:
   - **Database host address**: Database server name or IP address.
   - **Database host port**: The SQL port.
   - **Database name**: pmd$primavera (default)

   **Note**: When running for an Enterprise version of the database, you will be prompted for the appropriate privileged and public usernames.

5) Review the results of the schema validation utility that display in the browser.

**Running the Data Pump Utility Using the Wizard**

To run the data pump utility using the wizard, complete the following steps:

   **Note**: Skip the first two steps if you are using a Microsoft SQL Server Database.

1) From the command line, execute the following and provide the password when prompted:

   ```sql
   sqlplus sys@<db_tns_names_entry> as sysdba
   ```
2) Go to p6suite\database\scripts\install\PM_<release_level>, and execute manual_script_before_install.sql.

3) Go to the <P6_EPPM_Home>\database folder.

4) Run migrate.bat (with Windows) or migrate.sh (with UNIX or Linux).

5) Follow the prompts on each screen to provide connection information for the source database and the target database.

**Note:** When using an Oracle schema, the migration utility allows you to create new tables or use existing tables in the target instance, but new schema users must be created through the migration process. The ability to map to existing schema users is not currently available through the migration process.

---

### Running the Data Pump Utility From the Command Line

**Note:** Do not use the data pump utility to convert the database type from EPPM to PPM or from PPM to EPPM.

To run the data pump utility from the command line, complete the following steps:

1) Open a new command line console.

2) Go to \install\database.

3) Run the command with the following parameters to define the migration:

    **With Windows**

    ```
migrate.bat -source sa/sa@sqlserver:rcgsrv:1433:vader_pmdb -target system/manager@oracle:rcgsrv:1521:rcg02 -db PM
    ```

    **With Windows (Creating Non-Default Oracle Users)**

    ```
migrate.bat -source sa/sa@sqlserver:1433:vader_pmdb -target system/manager@oracle:1521:rcg02 -db PM -<admuser> <admuser password> -<privuser> <privuser password> -<pubuser> <pubuser password>
    ```

    **With UNIX or Linux**

    ```
    sh migrate.sh -source sa/sa@sqlserver:saumverm-lap\primavera:1433:PMDB -dattbsp <Name of data tblspc at target> -ndxbsp <Name of index tblspc at target> -pxtbsp <Name of px tblspc at target> -lobtbsp <Name of LOB tblspc at target> -<admuser> <admuser password> -<privuser> <privuser password> -<pubuser> <pubuser password> -<pxrptuser> <pxrptuser password> -<bgjobuser> <bgjobuser password> -<bgjobuser password at target>
    ```

    For example:

Migrating Database Schema

The schema migration process creates a copy of an existing source P6 EPPM or P6 Professional database and then modifies the database objects for the target database. Because the source database is not being modified, you do not need to backup your database before attempting to migrate your schema.

Tip: You should migrate your schema at a time when no database transactions occur.

Prerequisites for Schema Migration

In order to migrate schema between P6 EPPM and P6 Professional, you must be on the same version of both applications.

If you are using an Oracle database, ensure that you have completed the following prerequisites before attempting to migrate your schema:

- Create an empty Oracle database for your target database.
- Set the `open_cursors` system parameter to a value of 1000 or greater depending on the size of your source database.
- Run the `manual_script_before_install.sql` script from the P6 EPPM or P6 Professional installation folder on your target database. Alternatively, if you are migrating your schema to P6 Professional, you can run the following script on the P6 Professional database:

```sql
GRANT SELECT ON sys.DBA_EXTENTS TO SYSTEM with grant option;
```

If you are using Microsoft SQL Server, you must enable FILESTREAM.

Migrating P6 EPPM Schema to P6 Professional with Oracle

To migrate your P6 EPPM schema to P6 Professional:

1) Open the Migrate Database wizard by completing the following:

   a. Navigate to the database folder of your P6 EPPM installation. For example, `C:\P6EPPM_1\database\` (with Windows) or `/u01/P6EPPM_1/database/` (with UNIX or Linux).
   b. Run `migrate.bat` (with Windows) or `migrate.sh` (with UNIX or Linux).
2) On the Select Migration Type screen, select Schema Migration.
3) On the Select Source screen, select Oracle.
4) On the Connection Information screen, enter the connection details for the source database and private database user.
5) On the Select Target screen, select Oracle.
6) On the Connection Information screen, enter the connection details for the target database and system database user.
7) On the Configure Oracle Tablespaces screen, complete one of the following steps:
   - If you want to use the existing P6 EPPM tablespaces for P6 Professional, select Use existing tablespaces and then select the tablespace names from the lists.
   - If you want to create new tablespaces for P6 Professional, enter the tablespace names and sizes in the tablespace name and tablespace size (M) fields.
   
   Note: If you chose to use an existing tablespace, skip the next step.
8) On the Specify Oracle Tablespace Location screen, enter the location at which you want to create each tablespace and then click Create.
9) On the Create Oracle Users screen, in the User Name, Password, and Confirm Password fields, enter the credentials for each database user.
10) On the Migrate Options screen, enter the size of the batch that will be used for the SQL insert of P6 data into the P6 Professional database.

**Migrating P6 EPPM Schema to P6 Professional with Microsoft SQL Server**

To migrate your P6 EPPM schema to P6 Professional:

1) Open the Migrate Database wizard by completing the following:
   a. Navigate to the database folder of your P6 EPPM installation. For example, C:\P6EPPM_1\database\ (with Windows) or /u01/P6EPPM_1/database/ (with UNIX or Linux).
   b. Run migrate.bat (with Windows) or migrate.sh (with UNIX or Linux).

   Note: After each step, click Next.
2) On the Select Migration Type screen, select Schema Migration.
3) On the Select Source screen, select Microsoft SQL Server.
4) On the Connection Information screen, enter the connection details for the source SQL Server database and the administrative database user.
5) On the Select Target screen, select Microsoft SQL Server.
6) On the Connection Information screen, enter the connection details for the target SQL Server database and the sys admin database user.
7) On the Configure Database Users screen, complete one of the following steps:
If you want to use the existing P6 EPPM users for P6 Professional, complete the following:

a. Select the **Use Existing** check box.
b. In each **User Name** list, select the username of the database user.
c. In each **Password** field, enter the password for each database user.

If you want to create database users for P6 Professional, enter the login credentials for each database user in the **User Name**, **Password**, and **Confirm Password** fields.

8) On the **Configure Microsoft SQL Server/SQL Express Database** screen, enter the required information in the **Database name**, **Data file**, **Log file**, and **Database code page** for the P6 Professional schema.

9) On the **Migrate Options** screen, enter the size of the batch that will be used for the SQL insert of P6 data into the P6 Professional database.
Creating and Updating P6 EPPM Configurations

The utility that you can use to create or update the configuration settings for P6 EPPM in Primavera P6 Administrator is the Database Configuration Wizard.

When either creating or updating P6 EPPM configurations, the Database Configuration Wizard updates the admin_config table in the P6 EPPM database and modifies the BREBootStrap.xml file.

The Database Configuration Wizard can be accessed by running dbconfigpv.bat (with Windows) dbconfigpv.sh (with UNIX or Linux) at <P6_EPPM_Home>/p6.

Where: <P6_EPPM_Home> is the P6 EPPM home directory that was set during installation.

For more information about creating configurations, refer to Creating P6 EPPM Configurations (on page 15).

For more information about updating configurations, refer to Updating P6 EPPM Configurations (on page 16).

In This Section

Creating P6 EPPM Configurations ................................................................. 15
Updating P6 EPPM Configurations............................................................... 16

Creating P6 EPPM Configurations

You would want to create a configuration if you are manually installing and configuring P6 EPPM for the first time, or if you want to create additional configurations other than the one created by the P6 EPPM Configuration Wizard. The default name of the configuration created by the P6 EPPM Configuration Wizard is "Primavera P6 Configuration".

To create a P6 EPPM configuration:

1) Run dbconfigpv.bat (with Windows) dbconfigpv.sh (with UNIX or Linux) at <P6_EPPM_Home>/p6. The Database Configuration Wizard opens.

   Where: <P6_EPPM_Home> is the P6 EPPM home directory that was set during installation.

   Note: Click Next after each of the following steps.

2) On the Setup and Configuration of the Primavera P6 Database screen, select Microsoft SQL Server.

3) On the Please enter the following information... screen, enter the public user credentials and the connection details for your database. The default public username is pubuser.
4) On the **What would you like to do?** screen, select **Create a new configuration** and enter a name for the configuration in the field.

5) On the **Configuration of the Primavera P6 database completed successfully** screen, click **OK**.

### Updating P6 EPPM Configurations

You would want to update your P6 EPPM configurations using the Database Configuration Wizard if you are manually upgrading P6 EPPM from an earlier release to the current release.

To update an existing P6 EPPM configuration:

1) Run `dbconfigpv.bat` (with Windows) `dbconfigpv.sh` (with UNIX or Linux) at `<P6_EPPM_Home>/p6`. The **Database Configuration Wizard** opens.

   **Where:** `<P6_EPPM_Home>` is the P6 EPPM home directory that was set during installation.

   **Note:** Click **Next** after each of the following steps.

2) On the **Setup and Configuration of the Primavera P6 Database** screen, select your database.

3) On the **Please enter the following information...** screen, enter the public user credentials and the connection details for your database. The default public username is pubuser.

4) On the **What would you like to do?** screen, select **Use an Existing configuration** and then select a configuration from the drop down list.

5) On the **Configuration of the Primavera P6 database completed successfully** screen, click **OK**.

6) Repeat the previous steps to update other configurations to the current release.
Setting Up the P6 EPPM Database

Use the Primavera Database Setup Wizard to install or upgrade the P6 EPPM database.

For more information about installing a P6 EPPM database, see *Installing a P6 EPPM Database with Oracle* (on page 17) or *Installing a P6 EPPM Database with Microsoft SQL Server* (on page 18).

For more information about upgrading a P6 EPPM database, see *Upgrading a P6 EPPM Database with Oracle Database* (on page 19) or *Upgrading a P6 EPPM Database with Microsoft SQL Server* (on page 20).

In This Section

- Installing a P6 EPPM Database with Oracle ............................................................ 17
- Installing a P6 EPPM Database with Microsoft SQL Server ..................................... 18
- Upgrading a P6 EPPM Database with Oracle Database ............................................. 19
- Upgrading a P6 EPPM Database with Microsoft SQL Server .................................. 20

Installing a P6 EPPM Database with Oracle

If you want to manually install P6 EPPM, you should install a P6 EPPM database using the Primavera Database Setup Wizard.

To install a P6 EPPM database using the Primavera Database Setup Wizard:

1) To open the Primavera Database Setup Wizard, run `dbsetup.bat` (with Windows) `dbsetup.sh` (with UNIX or Linux) at `<P6_EPPM_Home>/database` where `<P6_EPPM_Home>` is the P6 EPPM home directory that was set during installation.

   **Note:** Click **Next** after each of the following steps.

2) On the Primavera P6 screen, complete the following:
   a. Select **Install a new database**.
   b. Select **Oracle**.

   **Note:** Create Read Only User is only intended for use with cloud installations of the application.

3) On the **Connection Information** screen, enter the connection details for an Oracle database using system credentials.

4) On the **Create New Keystore** screen, complete one of the following:
   - If you want to use an existing keystore, enter the password in the **Existing Keystore Password** field.
   - If you are creating a new keystore, do the following:
     a. Select **Create New Keystore**.
b. In the **Enter Keystore Password** field, enter a password for the new keystore.

c. In the **Confirm Keystore Password** field, enter the new password again for verification.

   - If you do not want to create a keystore and you do not have an existing keystore, ensure all fields are clear.

5) On the **Configure Oracle Tablespaces** screen, enter a name for each tablespace and modify the tablespace size (M) if necessary.

6) On the **Specify Oracle Tablespace Location** screen, enter the location at which you want to create each tablespace and then click **Create**.

7) On the **Create Oracle Users** screen in the **User Name**, **Password**, and **Confirm Password** fields, modify the database user names and enter the credentials for each database user.

8) On the **Configuration Options** screen, complete the following:

   a. Enter a name and password for an administrative application user.
   
   b. Choose whether or not you want to load sample data.
   
   c. Select your currency from the **Currency** list.
   
   d. If synchronizing P6 EPPM and Oracle Prime Projects select **Use Oracle Database Partitioning Option (separately licensed)** to improve synchronization performance.

   e. Click **Install**.

---

### Installing a P6 EPPM Database with Microsoft SQL Server

If you want to manually install P6 EPPM, you should install a P6 EPPM database using the Primavera Database Setup Wizard.

To install a P6 EPPM database using the Primavera Database Setup Wizard:

1) To open the Primavera Database Setup Wizard, run `dbsetup.bat` (with Windows) `dbsetup.sh` (with UNIX or Linux) at `<P6_EPPM_Home>/database` where `<P6_EPPM_Home>` is the P6 EPPM home directory that was set during installation.

   **Note:** Click **Next** after each of the following steps.

2) On the **Primavera P6** screen, complete the following:

   a. Select **Install a new database**.
   b. Select **Microsoft SQL Server**.

3) On the **Connection Information** screen, enter the connection details for a Microsoft SQL Server database and the administrative database user.

4) On the **Configure Microsoft SQL Server** screen, complete the following:

   a. Enter the name of the P6 EPPM database.
   b. Enter the location of the Data and Log files.
   c. Select the database code page. The default is Database default.
   d. Enter the location of the keystore.

5) On the **Create New Keystore** screen, complete one of the following:

   - If you want to use an existing keystore, enter the password in the **Existing Keystore Password** field.
If you are creating a new keystore, do the following:
   a. Select **Create New Keystore**.
   b. In the **Enter Keystore Password** field, enter a password for the new keystore.
   c. In the **Confirm Keystore Password** field, enter the new password again for verification.

If you do not want to create a keystore and you do not have an existing keystore, ensure all fields are clear.

6) On the **Create SQL Server Users** screen, complete one of the following steps:
   a. If you want to use the existing P6 EPPM users for P6 Professional, complete the following:
      a. Select the **Use Existing** check box.
      b. In each **User Name** list, select the username of the database user.
      c. In each **Password** field, enter the password for each database user.
   b. If you want to create database users for P6 Professional, enter the login credentials for each database user in the **User Name**, **Password**, and **Confirm Password** fields.

7) On the **Configuration Options** screen, complete the following:
   a. Enter a name and password for an administrative application user.
   b. Choose whether or not you want to load sample data.
   c. Select your currency from the **Currency** list.
   d. Click **Install**.

---

**Upgrading a P6 EPPM Database with Oracle Database**

If you want to manually upgrade P6 EPPM, you should upgrade your P6 EPPM database using the Primavera Database Setup Wizard.

To upgrade a P6 EPPM database using the Primavera Database Setup Wizard:

1) To open the Primavera Database Setup Wizard, run **dbsetup.bat** (with Windows) **dbsetup.sh** (with UNIX or Linux) at `<P6_EPPM_Home>/database` where `<P6_EPPM_Home>` is the P6 EPPM home directory that was set during installation.

   **Note**: Click **Next** after each of the following steps.

2) On the **Primavera P6** screen, complete the following:
   a. Select **Upgrade an existing database**.
   b. Select **Oracle**.

3) On the **Connection Information** screen, enter the connection details for an Oracle database using system credentials.

4) On the **Create New Keystore** screen, do one of the following:
   a. If you want to use an existing keystore, enter the password in the **Existing Keystore Password** field.
   b. If you are creating a new keystore, do the following:
      a. Select **Create New Keystore**.
      b. In the **Enter Keystore Password** field, enter a password for the new keystore.
c. In the **Confirm Keystore Password** field, enter the new password again for verification.
   - If you do not want to create a keystore and you do not have an existing keystore, ensure all fields are clear.

5) On the **Configure Oracle Tablespaces** screen, enter the credentials for the administrative user. The default is admuser.

6) On the **Upgrade Options** screen, enter the credentials for the privileged, public, and background user. The defaults are privuser, pubuser, and bgjobuser.

7) On the **Ready to Begin Upgrading Data** screen, select **Yes, upgrade my database**.

### Upgrading a P6 EPPM Database with Microsoft SQL Server

If you want to manually upgrade P6 EPPM, you should upgrade your P6 EPPM database using the Primavera Database Setup Wizard.

To upgrade a P6 EPPM database using the Primavera Database Setup Wizard:

1) To open the Primavera Database Setup Wizard, run `dbsetup.bat` (with Windows) `dbsetup.sh` (with UNIX or Linux) at `<P6_EPPM_Home>/database` where `<P6_EPPM_Home>` is the P6 EPPM home directory that was set during installation.

   **Note:** Click **Next** after each of the following steps.

2) On the **Primavera P6** screen, complete the following:
   a. Select **Upgrade an existing database**.
   b. Select **Microsoft SQL Server**.

3) On the **Connection Information** screen, enter the connection details for a Microsoft SQL Server database and the administrative database user.

4) On the **Create New Keystore** screen, do one of the following:
   - If you want to use an existing keystore, enter the password in the **Existing Keystore Password** field.
   - If you are creating a new keystore, do the following:
     a. Select **Create New Keystore**.
     b. In the **Enter Keystore Password** field, enter a password for the new keystore.
     c. In the **Confirm Keystore Password** field, enter the new password again for verification.
   - If you do not want to create a keystore and you do not have an existing keystore, ensure all fields are clear.

5) On the **Upgrade Option** screen, enter the credentials for the Px Reporting user. The default is pxrptuser.

6) On the **Ready to Begin Upgrading Data** screen, select **Yes, upgrade my database**.
Changing the Database Base Currency

Caution: You cannot change the base currency once projects begin.

After manually creating and configuring the P6 EPPM database, you must change the base currency if you do not want the databases to use US dollars ($) as the base currency.

The Base Currency

The base currency is the monetary unit used to store cost data for all projects in the database and is controlled by a global administrative setting. The default base currency for P6 EPPM is US dollars ($). The view currency is the monetary unit used to display cost data in P6 EPPM and is controlled by a user preference.

The exchange rate for the base currency is always 1.0. When a user selects a different currency than the base currency to view cost data, the base currency value is multiplied times the current exchange rate for the view currency to calculate the values displayed in cost and price fields.

For example, if the base currency is US Dollars, the view currency is Euros, and the exchange rate for Euros is $1 = €0.75, a value of $10 stored in the database is displayed as €7.5 in cost and price fields. Similarly, if you enter €7.5 in a cost or price field, it is stored in the database as $10.

When data is displayed in a view currency that is different than the base currency, some cost and price values can vary slightly (e.g., due to rounding). As long as the correct base currency is selected during database installation, a user can view completely accurate cost and price data by changing the view currency to match the base currency.

Reviewing Currency Choices

To change the base currency you need to edit and run the P6 EPPM script provided. By default, US dollars is the base currency, and USD is the short name used in the script. To know which short name to use, review the list of available short names for P6 EPPM by running the following query on the P6 EPPM database:

```
select curr_type, curr_short_name from currtype;
```

Changing the Base Currency

To change the base currency:

1) On the P6 EPPM physical media or download:
   a. Browse to \Database\scripts\common.
   b. Copy this script to a local drive:
      ```
      or_set_currency.sql
      ```
2) If you copied the script from the physical media, turn off the script's read-only attribute.
Since files on physical media are read-only, this attribute turns on when you copy a file from a CD or DVD.

a. In Windows Explorer, right-click the file.
b. Choose Properties.
c. Clear the Read-Only option.

3) Open the script for editing and locate the line containing `v_new_base_currency: = 'USD'`
4) Replace USD with the currency short name of your choice.
5) Save your changes and run the modified script.

**Private Database Credentials for P6 EPPM**

The P6 server and P6 Professional components obtain their run-time database connection credentials from a credential configuration table in the P6 EPPM database. The P6 run-time database credentials (known as privuser or P6 private database login) are stored in an encrypted format in this special P6 configuration table. Any time that you change or rotate the privuser password credentials in your Oracle or MS SQL Server database, you must re-synchronize the stored credentials in the P6 credential table by using the Database Login tool.

Because encryption algorithms are often enhanced in newer releases, Oracle highly recommends that you reset these stored privuser credentials when you perform a major version upgrade of P6 EPPM. By resetting the stored credentials, the new encryption algorithm can be applied to other stored credentials (for example, pubuser) in the P6 EPPM credential table. For information about resetting private database passwords, see **Resetting Private Database Passwords** (on page 22).

*Note:* This tool does not reset database user logins or passwords. Administrators should use SQL Developer or other DBA consoles to set or reset database user passwords.

**Resetting Private Database Passwords**

Password encryption algorithms are frequently improved in new releases of P6 Professional and P6 Professional. You should reset private database passwords in order to use improved password encryption algorithms.

To reset private database passwords to use the new encryption algorithm:

1) Go to **P6 EPPM <release_level>\database** or **P6 Professional <release_level>\database**.
2) Run **databaselogins.bat** (with Windows) **databaselogins.sh** (with UNIX or Linux).
3) In the **Database Connection** dialog box:
   a. Select the database.
   b. Type the user name and password of a privileged database user (for example, privuser). This login should have administrative rights on the database.
c. Enter the host address, host port, and database/instance name specific to your installation. The Port field displays the default port for the database type you selected.

d. Click Next.

4) In the Private Database Logins dialog box:
   a. Select the private database user name that you wish to reset.
   b. Highlight the password and change it (or re-enter the existing password).
   c. Click Update Password.
   d. Click Save.
   e. Click OK.

---

**Adding Private Database Logins for P6 EPPM**

You can add private database users to your P6 EPPM database.

To add private database logins for P6 EPPM:

1) Go to P6 EPPM <release_level>/database and run databaselogins.bat (with Windows) databaselogins.sh (with UNIX or Linux).

2) On the Database Connection dialog box:
   a. Select the database.
   b. Type the user name and password of a privileged database user (for example, privuser). This login should have administrative rights on the database.
   c. Enter the host address, host port, and database/instance name specific to your installation. The Port field displays the default port for the database type you selected.
   d. Click Next.

3) On the Private Database Logins dialog box:
   a. Click Add.
   b. Enter a user name.
   c. Enter a password.

   **Note:** To reverse a change, click Undo. Undo will reverse any changes made during the current session.

   d. Click Save.
   e. Click OK to exit.

---

**Modifying Private Database Logins for P6 EPPM**

You can update the passwords and usernames of the private database users on your P6 EPPM database.

To modify private database logins:

1) Go to P6 EPPM <release_level>/database and run databaselogins.bat (with Windows) databaselogins.sh (with UNIX or Linux).

2) On the Database Connection dialog box:
a. Select the database.
b. Type the user name and password of a privileged database user (for example, privuser). This login should have administrative rights on the database.
c. Enter the host address, host port, and database/instance name specific to your installation. The Port field displays the default port for the database type you selected.
d. Click **Next**.

3) On the **Private Database Logins** dialog box:
   a. Select the private database user name that you wish to modify.
   b. Enter a new user name.
   c. Highlight the password, and change it.
   d. Click the **Update Password** button.

   **Note:** To reverse a change, click **Undo**. Undo will reverse any changes made during the current session.

   e. Click **Save**.
   f. Click **OK** to exit the Database Logins tool.

### Deleting Private Database Logins for P6 EPPM

If you no longer need your private database users you can delete them.

To delete private database logins for P6 EPPM:

1) Go to **P6 EPPM <release_level>/database** and run **databaselogins.bat** (with Windows) **databaselogins.sh** (with UNIX or Linux).

2) On the **Database Connection** dialog box:
   a. Select the database.
   b. Type the user name and password of a privileged database user (for example, privuser). This login should have administrative rights on the database.
   c. Enter the host address, host port, and database/instance name specific to your installation. The Port field displays the default port for the database type you selected.
   d. Click **Next**.

3) On the **Private Database Logins** dialog box:
   a. Select the private database user name that you wish to remove.

   **Note:** You must have at least one private user name for the P6 EPPM database at all times.

   b. Click **Delete**.

   **Note:** To reverse a change, click **Undo**. Undo will reverse any changes made during the current session.

   c. Click **Save**.
   d. Click **OK** to exit the Database Logins tool.
Setting Up Cloud Sync

You can setup Cloud Sync to share project data between your P6 EPPM and Oracle Prime Projects applications. P6 EPPM and Oracle Prime Projects utilize the REST protocol to transfer data to and from local installations of P6 EPPM. Enable project table auditing to track changes committed to projects across synchronization. Contact your Oracle Prime Projects administrators to ensure Oracle Prime Projects is configured for Cloud Sync.

For more information on establishing Cloud Sync, see **Synchronizing data from P6 EPPM to Oracle Prime Projects** (on page 27).

For more information on project table auditing, see **PAUDIT Auditing** (on page 45). Alternatively the application administrator can establish project table auditing using the Audit Page. For more information on the Audit Page, see the "Audit Page" topic in the *Application Administration Guide*.

**Note:** If you are using an Oracle database, select **Use Oracle Database Partitioning Option (separately licensed)** during database installation to improve synchronization performance.

In This Section

Synchronizing data from P6 EPPM to Oracle Prime Projects ................................... 27
Synchronizing data from Oracle Prime Projects to P6 EPPM ................................... 33

Synchronizing data from P6 EPPM to Oracle Prime Projects

You can send project data from local installations of P6 EPPM to cloud based installations of Oracle Prime Projects using Cloud Sync. To configure synchronization between your products, you must modify synchronization properties. Refer to the **Synchronization Properties** table for descriptions of synchronization property fields and values.

To setup your P6 EPPM database for synchronization with Oracle Prime Projects:

1) Navigate to the folder in which P6 EPPM is installed, then navigate to the `/cloudsync` folder.
2) Open **client_send.properties** in a text editor of your choice and modify the following properties:
   
   ```properties
   Sync Properties
   sync.mode=CLIENT
   sync.type=SENDER
   sync.name=<Sync name>
   sync.number=<Sync number>
   ```
**Note:** Contact your cloud administrator to request appropriate values for the `sync.number` property.

**Source Database Properties**
- `source.db.host=<P6 EPPM database host address>`
- `source.db.port=<P6 EPPM database port number>`
- `source.db.service=<P6 EPPM database service name>`
- `source.db.type=P6`

**Source Credential Properties**
- `source.db.syncuser=<P6 EPPM database sync user>`
- `source.db.schema=<P6 EPPM database administrative user>`
- `source.db.sysuser=<P6 EPPM database SYSDBA user>`

**Note:** The database sync user is automatically created upon running the `syncsetup` script.

**Server Properties**
- `server.base_url=http://<HOST>:<PORT>/sync/rest`
- `server.connectuser=<USER>`
- `optin.project_mode=PRIME`
- `optin.project_value=Y`

**Notes:**
- Contact your cloud administrator to request appropriate values for the `server.base_url` and `server.connectuser` properties.
- Leave password properties fields, such as `source.db.syncuserpwd` blank. The value of these properties are populated and encrypted when you run the `syncsetup` script.

3) From the command line, navigate to the `/prime-sync` folder and run the `syncsetup` script:

**Note:** If you wish to sync Oracle Prime Projects data back to P6 EPPM complete the steps in *Synchronizing data from Oracle Prime Projects to P6 EPPM* (on page 33) before running the `syncsetup` script.

On Windows, run:

```bash
syncsetup.cmd client_send.properties
```

On UNIX, run:

```bash
./syncsetup.sh client_send.properties
```

4) When prompted, enter the **password** for the database sync manager, database administrative manager, database SYS user, web user, and syncadmin.
5) From the command line, run the *startserver* script to initialize synchronization:
   On Windows, run:
   
   `startserver.cmd`
   
   On UNIX, run:
   
   `./startserver.sh`

6) Once the server has started, run the *commander* script to load the configuration:
   On Windows, run:
   
   `commander.cmd LOAD client_send.properties`
   
   On UNIX, run:
   
   `./commander.sh LOAD client_send.properties`

### Synchronization Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Value</th>
<th>Example Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>sync.mode</td>
<td>The application’s role in the configured synchronization. The application sending data acts as a client, the other application acts as a server.</td>
<td>CLIENT</td>
<td>CLIENT</td>
</tr>
<tr>
<td>sync.type</td>
<td>Specifies whether the application is sending or receiving synchronization requests. The client application typically sends requests, the server application typically receives requests.</td>
<td>SENDER</td>
<td>SENDER</td>
</tr>
<tr>
<td>sync.name</td>
<td>The name used to identify the sync process.</td>
<td>Sync Name</td>
<td>receive1</td>
</tr>
<tr>
<td>sync.number</td>
<td>The unique synchronization number to connect to the hosted server application. Contact your cloud administrator to request the appropriate value for this property.</td>
<td>Sync Number</td>
<td>1</td>
</tr>
<tr>
<td>source.db.host</td>
<td>The host name of the database that acts as the source of data in the synchronization. Data from the application's source database is shared to the target database of the other application.</td>
<td>Database Host Name</td>
<td>localhost</td>
</tr>
<tr>
<td>source.db.port</td>
<td>The port number of the database that acts as the source of data in the synchronization. Data from the application's source database is shared to the target database of the other application.</td>
<td>Database Port Number</td>
<td>1521</td>
</tr>
<tr>
<td>source.db.service</td>
<td>The service name of the database that acts as the source of data in the synchronization. Data from the application's source database is shared to the target database of the other application.</td>
<td>Database Service Name</td>
<td>pmdb</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td>Application Name</td>
<td>Description</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>source.db.type</td>
<td>The name of the application related to the source database. For example, P6 or PRIME.</td>
<td></td>
<td>P6</td>
</tr>
<tr>
<td>source.db.syncuser</td>
<td>The name of the synchronization user of the specified source database. This user is created upon running the syncsetup script.</td>
<td></td>
<td>syncmgr</td>
</tr>
<tr>
<td>source.db.schema</td>
<td>The name of the administrative user of the source database. The default administrative user for P6 EPPM installations is admuser.</td>
<td></td>
<td>admuser</td>
</tr>
<tr>
<td>source.db.sysuser</td>
<td>The name of the system user of the source database. The default SYSDBA user on an Oracle database is SYS.</td>
<td></td>
<td>SYS</td>
</tr>
<tr>
<td>source.db.userts</td>
<td>The tablespace in the source database in which user data is stored. The default value for P6 EPPM installations is PMDB_DAT1.</td>
<td></td>
<td>PMDB_DAT1</td>
</tr>
<tr>
<td>source.db.tempts</td>
<td>The tablespace in the source database in which temporary data is stored. The default value for P6 EPPM installations is TEMP.</td>
<td></td>
<td>TEMP</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
<td>Value</td>
<td></td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------------------------</td>
<td>------------------------</td>
<td></td>
</tr>
<tr>
<td>server.base_url</td>
<td>The URL used to establish connection to the target server application for synchronization. Specified host and port values vary depending on your setup. Contact your cloud administrator to request the appropriate value for this property.</td>
<td><a href="http://localhost:4000/p6sync/sync">http://localhost:4000/p6sync/sync</a></td>
<td></td>
</tr>
<tr>
<td>server.connectuser</td>
<td>The user name of the user used to connect to the hosted server application. Contact your cloud administrator to request the appropriate value for this property.</td>
<td>WebLogic</td>
<td></td>
</tr>
<tr>
<td>file.size</td>
<td>The maximum file size in bytes which can be sent across the synchronization.</td>
<td>100M</td>
<td></td>
</tr>
<tr>
<td>file.send.timeout</td>
<td>The amount of time in seconds after which attempts at sending data across the synchronization will timeout.</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>optin.project_mode</td>
<td>The method the applications uses to determine what project data will be synchronized. For P6 EPPM-Oracle Prime Projects synchronizations this value should only be PRIME.</td>
<td>PRIME</td>
<td></td>
</tr>
</tbody>
</table>
Synchronizing data from Oracle Prime Projects to P6 EPPM

You can send project data from cloud installations of Oracle Prime Projects to local installations of P6 EPPM using Cloud Sync. To configure synchronization between your products, you must modify synchronization properties. Refer to the Synchronization Properties table for descriptions of synchronization property fields and values.

To synchronize data from Oracle Prime Projects to P6 EPPM:

1) Navigate to the folder in which P6 EPPM is installed, then navigate to the /cloudsync folder.
2) Open client_send.properties in a text editor of your choice and modify the following properties:

   **Sync Properties**
   - sync.mode=CLIENT
   - sync.type=RECEIVER
   - sync.name=<Sync Name>
   - sync.number=<Sync Number>

   **Note:** Contact your cloud administrator to request the appropriate value for the sync.number property.

   **Target Database Properties**
   - target.db.host=<P6 EPPM database schema host>
   - target.db.port=<P6 EPPM database port number>
   - target.db.service=<P6 EPPM database service>
   - target.db.type=P6

   **Target Database User Properties**
   - target.db.syncuser=<P6 EPPM database sync user>
   - target.db.schema=<P6 EPPM database administrative user>
   - target.db.sysuser=<P6 EPPM database SYSDBA user>

   **Note:** The database sync user is automatically created upon running the syncsetup script.

   **Server Properties**
   - server.base_url=http://<HOST>:\<PORT>/p6sync/sync
   - server.connectuser=<USER>
Notes:
- Contact your cloud administrator to request appropriate values for the server.base_url and server.connectuser properties.
- Leave password properties fields, such as source.db.syncuserpwd blank. The values of these properties are set when you run the syncsetup script.

3) From the command line, navigate to the /prime-sync folder and run the syncsetup script:
   On Windows, run:
   ```
syncsetup.cmd client_send.properties
   ```
   On UNIX, run:
   ```
./syncsetup.sh client_send.properties
   ```

4) When prompted, enter the password for the database sync manager, database administrative manager, database SYS user, web user, and syncadmin.

5) From the command line, run the startserver script to initialize synchronization:
   On Windows, run:
   ```
startserver.cmd
   ```
   On UNIX, run:
   ```
./startserver.sh
   ```

6) Once the server has started, run the startserver script to load the configuration:
   On Windows, run:
   ```
commander.cmd LOAD client_send.properties
   ```
   On UNIX, run:
   ```
./commander.sh LOAD client_send.properties
   ```

Synchronization Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
<th>Value</th>
<th>Example Value</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Setting Up Cloud Sync</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>sync.mode</strong></td>
</tr>
<tr>
<td>CLIENT</td>
</tr>
<tr>
<td><strong>sync.type</strong></td>
</tr>
<tr>
<td>RECEIVER</td>
</tr>
<tr>
<td><strong>sync.name</strong></td>
</tr>
<tr>
<td>Sync Name</td>
</tr>
<tr>
<td><strong>sync.number</strong></td>
</tr>
<tr>
<td>Sync Number</td>
</tr>
<tr>
<td><strong>target.db.host</strong></td>
</tr>
<tr>
<td>Database Host Name</td>
</tr>
<tr>
<td>target.db.port</td>
</tr>
<tr>
<td>----------------------</td>
</tr>
<tr>
<td>target.db.service</td>
</tr>
<tr>
<td>target.db.type</td>
</tr>
<tr>
<td>target.db.syncuser</td>
</tr>
<tr>
<td>target.db.schema</td>
</tr>
<tr>
<td>Setting Up Cloud Sync</td>
</tr>
<tr>
<td>-----------------------</td>
</tr>
<tr>
<td><strong>target.db.sysuser</strong></td>
</tr>
<tr>
<td><strong>Database System User</strong></td>
</tr>
<tr>
<td><strong>target.db.userts</strong></td>
</tr>
<tr>
<td><strong>Database User Tablespace Name</strong></td>
</tr>
<tr>
<td><strong>target.db.tempts</strong></td>
</tr>
<tr>
<td><strong>Database Temporary Tablespace Name</strong></td>
</tr>
<tr>
<td><strong>server.base_url</strong></td>
</tr>
<tr>
<td><strong>Server Application URL</strong></td>
</tr>
<tr>
<td><strong>server.connectuser</strong></td>
</tr>
<tr>
<td><strong>Server Connection User Name</strong></td>
</tr>
<tr>
<td>optin.project_mode</td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>optin.project_value</td>
</tr>
</tbody>
</table>
Configuring Background Processes and Jobs

In This Section

RDBMS Scheduler Configuration ................................................................. 39
Database Settings Table ............................................................................... 39
Reading Setting Values .............................................................................. 40
Writing Setting Values ................................................................................ 41
Tracking Background Job Execution .......................................................... 42

RDBMS Scheduler Configuration

Since background jobs are initiated by the job scheduler supplied by the RDBMS, you need to ensure that the scheduler for your specific RDBMS is properly configured.

With Oracle: P6 EPPM uses DBMS_SCHEDULER to schedule background jobs with Oracle.

With Microsoft SQL Server: P6 EPPM uses the SQL Agent service to automatically schedule background job execution with Microsoft SQL Server. Background jobs handle the maintenance of the utilities tables (for example, USESSION and REFRDEL). As a part of background jobs, the agent runs the SYMON and DAMON stored procedures at specific intervals.

Database Settings Table

Settings Table Overview

The settings table contains name-value pairs that configure the behavior of the background processes.

Namespace

The namespace component is a dot-notation string representing a formal path to the parameter.

Setting Name

The setting name identifies the name of the setting.

Value

Values in the SETTINGS table are case-sensitive. The value portion of the pair can be:

- **String**: The string data type is a free text value. The most common string sub-type is interval which represents an interval of time by combining a numeric portion with a unit portion as depicted in the table below.

  Interval Subtype Table
### Numeric: The numeric data type consists of any number.

### Boolean: The Boolean data type can have one of two values: true or false, where zero represents false and any non-zero number represents true.

### Date: The date data type consists of dates.

#### Setting Example

The following is an example of a setting:

- Namespace: `database.cleanup.Usession`
- Setting Name: `ExpiredSessionTimeout`
- Value: `2h` (two hour interval)

#### Reading Setting Values

You can configure settings through the Settings API Procedures. These procedures are similar to registry or INI file procedure calls.

#### Reading Settings Values

Use the following `SETTINGS_READ_*` procedures to determine the current value of specific settings:

- `SETTINGS_READ_STRING(ret_val,namespace,settings_name,default)`
- `SETTINGS_READ_DATE(ret_val,namespace,settings_name,default)`
- `SETTINGS_READ_NUMBER(ret_val,namespace,settings_name,default)`
- `SETTINGS_READ_BOOL(ret_val,namespace,settings_name,default)`

#### Using Code to Read Setting Values with Oracle

The following code snippets demonstrate how the `SETTINGS_READ_*` procedures read the setting values with Oracle.

To retrieve the value of the `KeepInterval` setting:

1. Use the following code:

   ```sql
   SQL> variable vset varchar2(255)
   SQL> exec settings_read_string(:vset,'database.cleanup.Usession', 'ExpiredSessionTimeout');
   ```

2. The following message should appear:
PL/SQL procedure successfully completed.
SQL> print vset

Using Code to Read Setting Values with Microsoft SQL Server

The following code snippets demonstrate how the SETTINGS_READ_* procedures read the setting values with Microsoft SQL Server.

To retrieve the value of the KeepInterval setting:

1) Use the following code:

```plsql
declare @vset varchar(255)
exec settings_read_string @vset
OUTPUT,'database.cleanup.Usession','ExpiredSessionTimeout'
print @vset
```

2) The following message should appear:

PL/SQL procedure successfully completed.
SQL> print vset

Writing Setting Values

Use the SETTINGS_WRITE_STRING procedure to set the value of a specific setting:

```
SETTINGS_WRITE_STRING(new value,namespace,settings_name);
```

Using Code to Write Setting Values with Oracle

The following code snippets demonstrate how the SETTINGS_WRITE_STRING procedure sets the value of the ExpiredSessionTimeout setting to 12 hours with Oracle.

To set the value of the ExpiredSessionTimeout setting to 12 hours:

1) Log in to SQL*Plus with your privuser credentials.
2) Run the following statement:

```sql
SQL > exec SETTINGS_WRITE_STRING ('12h','database.cleanup.Usession','ExpiredSessionTimeout');
```

Using Code to Write Setting Values with Microsoft SQL Server

The following code snippets demonstrate how the SETTINGS_WRITE_STRING procedure sets the value of the ExpiredSessionTimeout setting to 12 hours with Microsoft SQL Server.

To set the value of the ExpiredSessionTimeout setting to 12 hours:

1) Open the Query Analyzer/SSMS and connect as privuser.
2) Select the P6 EPPM database, then run the following statement (using 12 hours as an example):

exec SETTINGS_WRITE_STRING '12h', 'database.cleanup.Usession', 'ExpiredSessionTimeout'

### Tracking Background Job Execution

You can track the execution of background jobs by monitoring the high level status settings or by inspecting the BGPLOG table.

#### High Level Status Settings

Each time a job runs, it will update the SETTINGS table for the setting_name = 'HeartBeatTime.' The job can update this value multiple times during the execution. You can monitor the maximum difference between this time and the current date to ensure that the job is running promptly. Refer to the High Level Status Settings table below for information about the HeartBeatTime setting.

<table>
<thead>
<tr>
<th>High Level Status Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Last date and time background job SYMON ran.</strong></td>
</tr>
<tr>
<td>Namespace</td>
</tr>
<tr>
<td>Setting Name</td>
</tr>
<tr>
<td>Default Setting</td>
</tr>
</tbody>
</table>

| **Last date and time background job DAMON ran.** |
| Namespace | database.background.Damon |
| Setting Name | HeartBeatTime |
| Default Setting | N/A |

#### The BGPLOG Table

You can also track the execution of background jobs by inspecting the BGPLOG table. The BGPLOG table holds detailed entries from the background processes including informational, elapsed time, and error entries. Refer to the BGPLOG Table Descriptions for information about what this table contains.
<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log_time</td>
<td>Time when background process made a log entry</td>
<td>Datetime</td>
</tr>
<tr>
<td>Source</td>
<td>Program generating log entry</td>
<td>&quot;system_monitor&quot;, &quot;data_monitor&quot;</td>
</tr>
<tr>
<td>Type</td>
<td>Type of message</td>
<td>INFORMATION, ELAPSED TIME, ERROR</td>
</tr>
<tr>
<td>Description</td>
<td>Message from the background process</td>
<td>A variable message followed by a number in parenthesis that represents the number of rows that processed. As an example, the message &quot;Complete BGPLOG (40)&quot; indicates that forty rows processed.</td>
</tr>
</tbody>
</table>
PAUDIT Auditing

PAUDIT auditing permits you to log the edits, additions, and deletions made by users of P6 EPPM applications. When users make changes, they create a Data Manipulation Language (DML) INSERT, UPDATE, or DELETE statement. PAUDIT auditing uses the Data Manipulation Language (DML) INSERT, UPDATE, or DELETE statement being executed against tables in the database schema. Since every application table in the schema has its own auditing trigger, you can log changes made to each table regardless of who made the change or when the change was made. The database schema owner owns the auditing trigger: you can bypass trigger execution.

Auditing Level Configuration

You can adjust the amount of information that is logged by adjusting the audit level for each table. You can refine the audit further by setting the audit level individually for insert, updates, and deletes within each table.

Auditing Levels

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level 0</td>
<td>No audit.</td>
</tr>
<tr>
<td>Level 1</td>
<td>Row-level audit. Audit only the operation without column details</td>
</tr>
<tr>
<td>Level 2</td>
<td>Column-level Audit without blobs. Audit changes to the data at the column level but without blob changes</td>
</tr>
<tr>
<td>Level 3</td>
<td>Full Audit. Audit changes to the data at the column level. With Oracle, column level changes to blobs are audited. With Microsoft SQL Server, column level changes to blobs are not included.</td>
</tr>
</tbody>
</table>

Simple Configuration

You can use two configuration procedures to provide simple control of the auditing feature:

- auditing_enable(table_name, level)
You can set the audit level on an individual table or the same audit level for all of the tables. However, the simple configuration procedures do not allow for setting individual auditing levels for insert, update, or delete operations within a table.

**Oracle Database Examples:**
Use the following examples as a guide to use the simple audit configuration procedures to control the auditing feature.

- The following code snippet enables full auditing on all tables:

  ```sql
  exec auditing_enable(null,3);
  ```

- The following code snippet enables level one auditing on the task table:

  ```sql
  exec auditing_enable('TASK',1);
  ```

- The following code snippet disables auditing on PROJWBS:

  ```sql
  exec auditing_disable('PROJWBS');
  ```

- The following code snippet completely disables auditing across the entire database:

  ```sql
  exec auditing_disable(null);
  ```

**Detailed Configuration**
You can configure auditing trigger behavior by changing values in the settings table that either enable or disable:

- The auditing feature itself
- The auditing of specific tables
- The auditing of table insert, update, or delete operations within each table

**Auditing Status**
You can enable or disable the auditing feature by using the database.audit.Enable setting. Use the settings_write_bool procedure to enable/disable the auditing feature.

**Oracle Example:**
To enable the auditing feature in Oracle, use the following code:

```sql
exec settings_write_bool(1,'database.audit','Enabled');
```

**Microsoft SQL Server Example:**
To enable the auditing feature in Microsoft SQL Server, use the following code:

```sql
exec settings_write_bool 1,'database.audit','Enabled'
```
**Options Setting**

Each table’s auditing settings are controlled by the Options setting in each table’s auditing namespace (for example, database.audit.TASK). The Options setting is a three character string with a numeric value in each character position representing the audit level for insert, update, and delete.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Insert</th>
<th>Update</th>
<th>Delete</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level 0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>No audit.</td>
</tr>
<tr>
<td>Level 1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Row-level audit. Audit only the operation without column details.</td>
</tr>
<tr>
<td>Level 2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>Column-level audit without blobs. Audit changes to the data at the column level but without blob changes.</td>
</tr>
<tr>
<td>Level 3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>Full Audit. Audit changes to the data at the column level. With Oracle, column level changes to blobs are audited. With Microsoft SQL Server, column level changes to blobs are not included.</td>
</tr>
</tbody>
</table>

The following table provides examples of the options setting:

<table>
<thead>
<tr>
<th>Namespace</th>
<th>Setting</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>database.audit.TASK</td>
<td>Options</td>
<td>330</td>
<td>Fully audit any insert and update operations. Do not audit any delete operations.</td>
</tr>
</tbody>
</table>
### SETTINGS_WRITE_STRING Procedure

You can change table audit settings using the `settings_write_string` procedure.

**Oracle Example:**

To set the table settings to fully audit insert and update operations but ignore any delete operations, use the following code with Oracle:

```sql
exec settings_write_string('330','database.audit.TASK','Options');
```

**Microsoft SQL Server Example:**

To set the table settings to fully audit insert and update operations but ignore any delete operations, use the following code with Microsoft SQL Server:

```sql
exec settings_write_string '330','database.audit.TASK','Options'
```

**Note:** Changes to auditing settings will not appear immediately in the application. The program will need to close the database connection and then reconnect to the database to get the new settings.

### The Audit Table

Audit records are inserted into the PAUDIT table. One record is inserted into the audit table for each row changed in the database.

<table>
<thead>
<tr>
<th>Column</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUDIT_TS</td>
<td>TIMESTAMP(6)</td>
<td>Date and time of change</td>
</tr>
<tr>
<td>TABLE_NAME</td>
<td>STRING(30)</td>
<td>Table Name</td>
</tr>
<tr>
<td>PK1, PK2, PK3, PK4</td>
<td>STRING(255)</td>
<td>Primary key values for audited record</td>
</tr>
<tr>
<td>PROJ_ID</td>
<td>NUMBER</td>
<td>Unique object id of the audited project.</td>
</tr>
<tr>
<td>OPER</td>
<td>STRING(1)</td>
<td>I=Insert, U=Update, D=Delete</td>
</tr>
</tbody>
</table>
### Monitoring Processes and Procedures

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRM_USER_NAME</td>
<td>STRING(32)</td>
<td>P6 EPPM user name if the change was made in P6 EPPM applications</td>
</tr>
<tr>
<td>AUDIT_OLD</td>
<td>STRING(4000)</td>
<td>Column changes up to 4000 characters (Level 2 and 3 only)</td>
</tr>
<tr>
<td>AUDIT_NEW</td>
<td>STRING(4000)</td>
<td>Column changes up to 4000 characters (Level 2 and 3 only)</td>
</tr>
<tr>
<td>AUDIT_EXT_OLD</td>
<td>BLOB</td>
<td>Blob changes and overflow from audit_old (Level 2 and 3 only)</td>
</tr>
<tr>
<td>AUDIT_EXT_NEW</td>
<td>BLOB</td>
<td>Blob changes and overflow from audit_new (Level 2 and 3 only)</td>
</tr>
<tr>
<td>LOGICAL_DELETE_FLAG</td>
<td>STRING(1)</td>
<td>Flag for deletes that are logical (marked) rather than a physical delete</td>
</tr>
<tr>
<td>RDBMS_USER_NAME*</td>
<td>STRING(255)</td>
<td>Database user name (usually privuser)</td>
</tr>
<tr>
<td>OS_USER_NAME*</td>
<td>STRING(255)</td>
<td>Operating system user name of connected session</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Note:</strong> When auditing is enabled against a SQL Server database, PAUDIT.os_user_name will always return a null value. This is because the value for os_user_name in PAUDIT table is taken from the nt_username column in master.sys.sysprocesses view and the nt_username value in master.sys.sysprocesses view is empty for the P6 Professional program because P6 is connected to SQL DB using SQL authentication. The nt_username field is filled only when we connect to SQL DB using Windows authentication.</td>
</tr>
<tr>
<td>PROGRAM*</td>
<td>STRING(255)</td>
<td>Name of program connecting to the database</td>
</tr>
<tr>
<td>HOST_NAME*</td>
<td>STRING(255)</td>
<td>Computer name of connected session</td>
</tr>
<tr>
<td>APP_NAME*</td>
<td>STRING(25)</td>
<td>Name of application connected to the database</td>
</tr>
<tr>
<td>------------</td>
<td>------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>NETADDRESS*</td>
<td>STRING(24)</td>
<td>IP or MAC address of connected session</td>
</tr>
</tbody>
</table>

**Note**: Grant select privileges to the administrative user on `V_$SESSION` to ensure correct values for several auditing table values.

**Session Auditing**

Activity for the USESSION table is audited with its own trigger and table. When an application user logs out of the system they logically delete or mark their session record in the USESSION table. One record is written to the USESSAUD table for each logout. The format of the USESSAUD table mirrors that of the USESSION table. This audit can be enabled using the usessaud_enable procedure and disabled using the usessaud_disable procedure.

**Column Audit Data**

The data changes for each audit are stored in the audit_info and audit_info_extended columns. The audit_info column contains all the row changes as long as they do not exceed 4000 characters. Changes over 4000 characters or any edit to a blob will be written to the audit_info_extended BLOB column.

Data in the two audit_info columns has a specific format. Each column audit within the data begins with either ":O" (old data) or ":N" (new data) to distinguish between the audit of the previous (old) or the changed (new) value (for BLOB columns the data starts with :BLOBO or :BLOBN). Directly after this is the name of the column in lowercase. Following the column name is the length of the audited value in a fixed four character field. Finally, the actual data is placed in the audit record. Updates will have both an old and new value for each change. Inserts will have only a new value and deletes only an old value.

The following is an example of the audit record for TASK to change the task_code from 'A1010' to 'B102':

```sql
audit_info => :O task_code: 5:A1010 :N task_code: 4:B102
```
The performance of the P6 EPPM database depends on an effective configuration of physical design structures, such as indexes or hints, in the database.

Both Oracle and SQL Server have different tools which can be used to diagnose the need for database design performance tuning and is discussed in detail below.

In This Section

Oracle Database Tuning .......................................................................................... 51
Microsoft SQL Server Database Tuning .................................................................. 54
General Tuning ....................................................................................................... 57

Oracle Database Tuning

The responsibility for maintaining performance of the Oracle database is a task of the Oracle database administrator (DBA). This section describes how to tune an Oracle database for the P6 EPPM database.

Partitioning Oracle Database Tables for P6 EPPM Schema

Partitioning addresses key issues in supporting very large tables and indexes by letting you decompose them into smaller and more manageable pieces called partitions. SQL queries and DML statements do not need to be modified in order to access partitioned tables. However, after partitions are defined, DDL statements can access and manipulate individuals partitions rather than entire tables or indexes. This is how partitioning can simplify the manageability of large database objects. Also, partitioning is entirely transparent to applications.

Each partition of a table or index must have the same logical attributes (for example, column names, datatypes, and constraints); however, each partition can have separate physical attributes such as pctfree, pctused, and tablespaces.

Partitioning is useful for applications that manage large volumes of data.

Table partitioning is a manual process that can be performed after you set up the P6 EPPM Database. Oracle Primavera does not provide any tools or utilities to configure table partitioning. Partitioning of an Oracle table within a schema should be performed by the Oracle DBA.

For more information about Oracle Database partitioning, refer to the Database Performance Tuning Guide and Database Administrator's Guide.

For more information about partitioning a non-partitioned table, refer to How to Partition a Non-partitioned / Regular / Normal Table (Doc ID 1070693.6) on My Oracle Support.

Supported Tables for Partitioning

The following tables are supported for partitioning:
Supported Partitioning Type

The **LIST** and **RANGE** partitioning types are supported. List partitioning enables you to explicitly control how rows map to partitions by specifying a list of discrete values for the partitioning key in the description for each partition. Range partitioning enables you to specify a range of values for the partitioning key in the description of each partition. Rows with values matching the specified partition range map to the partition.

Table Partition Keys

The partitioning key is comprised of one or more columns that determine the partition where each row will be stored. Oracle automatically directs insert, update, and delete operations to the appropriate partition through the use of the partitioning key. The following list includes the partition key for each table:

- The partition key for **PAUDIT** is **AUSIT_TS**. The partition is by range **INTERVAL** of **(1, 'day')**.
- The partition key for **REFERDEL** is **DELETE_DATE**. The partition is by range **INTERVAL** of **(1, 'day')**.
- The partition key for **PROJWBS** is **PROJ_NODE_FLAG**. The partition is by list **Y** or **N**.
- The partition key for **PROJECT** is **ORIG_PROJ_ID**. The partition is by list null or default (not null).
- The partition key for **UDFVALUE** is **TABLE_NAME**.

**TABLE_NAME** is a de-normalized column and duplicates values from the **UDFTYPE** table. You can create a trigger to sync **UDFVALUE.TABLE_NAME** values with **UDFTYPE.TABLE_NAME**. The following script can be used to create a new trigger when partitioning **UDFVALUE**:

```sql
-- new trigger
CREATE OR REPLACE TRIGGER "DN_TABLE_NAME_UDFVALUE"
BEFORE INSERT OR UPDATE OF UDF_TYPE_ID ON UDFVALUE
FOR EACH ROW
DECLARE
BEGIN
select TABLE_NAME into :new.TABLE_NAME from UDFTYPE where udf_type_id = :new.udf_type_id;
end;
```

You can also enhance the login process and open project queries by changing the **SETTINGS.SETTING_VALUE** from **N** to **Y** in the **SETTINGS** table where **SETTINGS SETTINGS_NAME='UDFVALUE_DENORM'**.
Gathering Statistics for Cost Based Optimizations

Oracle Databases only support cost-based optimization, which relies on accurate statistics to determine the optimal access path for a query. To gather the appropriate statistics for the optimizer, which will improve database performance, run the ORPM_STATS_GATHER.SQL script located in the \Database\scripts\common folder of the P6 EPPM physical media or download.

Viewing the USESSION Table for GET_SAFETY_DATE

V_$TRANSACTION is a system view in Oracle that lists the active transactions in the system. GET_SAFETY_DATE (a procedure in the Project Management schema) accesses this view to get the oldest start time of an active transaction. If the schema owner does not have privileges to this view, then it returns a safety date using the USESSION table. V_$Transaction tunes performance during a refresh action. USESSION records the login time of the logged in user, whereas data in the V_$Transaction view is recorded at a system level. The logged in user could be logged in for more than an hour (as seen from the USESSION table), but the V_$Transaction view has the current transaction datetime, regardless of the time the user logged in.

Note: Access to the V_$TRANSACTION view was built into the procedure to tune performance with refreshing operations; however, third party functions could impact performance with database refresh operations that use an older time in the V_$TRANSACTION view.

To grant access to this view, connect to Oracle as SYS. Run the RUN_AS_SYS.SQL script located in the \Database\scripts\common folder of the P6 EPPM physical media or download.

Rebuilding the P6 EPPM Index Table

Indexes can become skewed if you frequently access parts of the index and not others. As a result, disk contention may occur and create a bottleneck in SQL performance. To prevent this performance degradation, you should monitor your P6 EPPM indexes and rebuild if necessary.

You can use the analyze_P6EPPM_indexes.sql script to compute statistics on the P6 EPPM index, validate the index structure, and return a report that includes the following information:

- The number of values in the index that exceed 100.
- Deleted entries in the index that represents 20% or more of the current entries or the index depth is more than four levels based on index_stats result

The indexes that are returned from this report should be considered for a rebuild as they could represent a skewed tree structure and can lead to unnecessary database block reads of the index.

To run analyze_P6EPPM_indexes.sql:

2) Save the script to your local database server.
3) Open a command prompt and then change the directory to the location of the script.
4) From the command line, execute the following scripts the following and provide the password when prompted:

```sql
sqlplus <admuser>@<db_tns_names_entry>
    @analyze_P6EPPM_indexes.sql
```

If you need to rebuild an index, you can run the following script:

```sql
ALTER INDEX <Index_Name> REBUILD ONLINE;
```

**Where:** `<Index_Name>` is the name of the index returned in the report.

---

### Where to Find Additional Oracle Database Tuning Information

The *Oracle Database 2 Day + Performance Tuning Guide* is intended for Oracle database administrators (DBAs) who want to tune and optimize the performance of their Oracle database. It is meant as a quick start guide that teaches you how to perform day-to-day database performance tuning tasks using features provided by Oracle Diagnostics Pack, Oracle Tuning Pack, and Oracle Enterprise Manager (Enterprise Manager). In particular, this document is targeted toward the following groups of users:

- Oracle DBAs who want to acquire database performance tuning skills.
- DBAs who are new to Oracle Database.

The *Oracle Database Performance Tuning Guide* is an aid for people responsible for the operation, maintenance, and performance of Oracle database. This book describes detailed ways to enhance Oracle database performance by using performance tools, and optimizing instance performance. It also explains how to create an initial database for good performance and includes performance-related reference information. This book could be useful for database administrators, application designers, and programmers.

---

### Microsoft SQL Server Database Tuning

The responsibility for maintaining performance of the Microsoft SQL Server Database is a task of the SQL administrator (sa). This section describes how to tune a Microsoft SQL Server database for the P6 EPPM database.

---

### Where to Find Additional Microsoft SQL Server Database Tuning Information

Monitoring the performance of a database requires a periodic review of the performance of processes that the database uses. As you collect data, you can isolate processes that cause problems or you can track performance trends.

For more information about how to effectively monitor a Microsoft SQL Server database for performance, refer to the following topics in the Microsoft documentation library:
Isolating Snapshots

When you enable snapshots and create a transaction in your SQL Server Database, a transaction is created (with a unique sequence number before your intended transaction) in tempdb. Your transaction then relies on all of the rows that have a lower sequence number than the rows in tempdb which ensures that your transaction is run against your database as it appeared at the time the query was performed. This prevents rows that were created after a transaction begins from being utilized by current transactions.

For more information about snapshot isolation as well as instructions to enable it, refer to the Snapshot Isolation in SQL Server topic in the Microsoft documentation library.

Rebuilding the P6 EPPM Index

Indexes become fragmented if you frequently create, modify, or delete data within a database. When indexes become fragmented, performance degrades. You can prevent the performance of your database from degrading as a result of fragmentation by monitoring your P6 EPPM indexes and rebuilding them if necessary.

For more information about detecting fragmentation within your database and rebuilding your indexes, refer to the Reorganize and Rebuild Indexes topic in the Microsoft documentation library.

Gathering Statistics

Statistics are an option that you can enable for your Microsoft SQL Server Database that automatically collects information about data distribution within tables and indexes, query patterns, query results, and the like. You should gather statistics so that you can optimize your interactions with your P6 EPPM database.

For example, gathering statistics can help you recognize and resolve the following issues:

- Loading data that hangs at 98%
- Performance issues
- Projects that open slowly
- Windows that disappear after loading data

For more information about enabling statistics gathering in your Microsoft SQL Server Database, refer to Introduction to the Statistics in SQL Server article in the Microsoft documentation library.

To gather statistics for all of the P6 EPPM tables:

1) Log in to your Microsoft SQL Server Database as a system administrator (sa) user.
2) Run the following query:

```sql
exec sp_updatestats
```

To gather statistics for the PROJECTS and TASKS tables:

1) Log in to your Microsoft SQL Server Database as a system administrator (sa) user.
2) Run the following queries:

```sql
update statistics project with fullscan ,all
update statistics task with fullscan ,all
```

To determine the last time statistics were gathered per index and table:

1) Log in to your Microsoft SQL Server Database as a system administrator (sa) user.
2) Run the following query:

```sql
SELECT OBJECT_NAME(A.object_id) AS Object_Name, A.name AS index_name,
       STATS_DATE(A.OBJECT_ID, index_id) AS StatsUpdated,
       DATEDIFF(d,STATS_DATE(A.OBJECT_ID, index_id),getdate()) DaysOld
FROM sys.indexes A
INNER JOIN sys.tables B ON A.object_id = B.object_id
WHERE A.name IS NOT NULL
ORDER BY DATEDIFF(d,STATS_DATE(A.OBJECT_ID, index_id),getdate()) DESC
```
In This Section

Background Processes and Clean Up in P6 EPPM ................................................. 57
PX Tables Clean Up Procedure ............................................................................... 68
Safe Deletes............................................................................................................ 68

Background Processes and Clean Up in P6 EPPM

Because clean up tasks can be resource intensive and time consuming, these tasks are initiated in P6 EPPM by two background jobs that run on the database server using the background job processes user name:

- SYMON (System Monitor), responsible for running procedures that take less than a few seconds to complete.
- DAMON (Data Monitor), responsible for running procedures that take longer than a few seconds to complete.

Both of these jobs are pre-configured with default settings. Since the default settings are optimal for most environments, you generally do not need to tune them. However, if you need to optimize your background process further, you can use the background job processes user to change the settings to tune the behavior of the background jobs for specific environments.

The background process will:

- Update the settings table with a record each time a job runs.
- Update the settings table with a HeartBeatTime record for both the SYMON and DAMON processes. The background processes regularly refresh the record to indicate that they are still running.

When you initiate background jobs:

- Execute the stored procedure INITIALIZE_BACKGROUND_PROCS.
- Locate the logs of background process activity in BPLOGS and SETTINGS tables within the Primavera database.

SYM ON (System Monitor) Procedures

SYM ON runs simple P6 EPPM tasks on a quick schedule. By default, the job runs every minute; the tasks assigned to this job should take only a few seconds to complete on each run. Do not change the scheduled run time (every minute) for this procedure.

Procedures performed by SYMON

The procedures run by SYMON perform these tasks:

- Processing the PRMQUEUE entries for Project Security by queuing OBSPROJ updates to the PRMQUEUE table.
- Marking expired USESSION records as logically deleted.
**Note:** You can manually run queries to assist you with tracking concurrent usage of P6 EPPM.

**OBSPROJ_PROCESS_QUEUE Procedure**

OBSPROJ_PROCESS_QUEUE processes the PRMQUEUE entries for Project Security. It defers processing of OBSPROJ updates by queuing the updates to the PRMQUEUE table.

Refer to the following table for information about the settings associated with the OBSPROJ_PROCESS_QUEUE procedure.

<table>
<thead>
<tr>
<th><strong>OBSPROJ_PROCESS_QUEUE Settings</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting Description: Maximum project-level queue records to process on each run.</td>
</tr>
<tr>
<td>Namespace</td>
</tr>
<tr>
<td>Setting Name</td>
</tr>
<tr>
<td>Default Setting</td>
</tr>
<tr>
<td>Type</td>
</tr>
</tbody>
</table>

| Setting Description: Maximum EPS-level queue records to process on each run. |
| Namespace | database.obsproj.queue |
| Setting Name | MaxEpsUpdate |
| Default Setting | 25 |
| Type | Numeric |

| Setting Description: Maximum times to re-process a failed entry before marking it as an error. |
| Namespace | database.obsproj.queue |
| Setting Name | MaxRetries |
| Default Setting | 50 |
| Type | Numeric |

**USESSION_CLEANUP_EXPIRED Procedure**

USESSION_CLEANUP_EXPIRED logically deletes USESSION records that have not updated their last_active_time based on the Expired Session settings. Marking expired USESSION records as logically deleted maximizes the number of module access logins that are available. Since it is not cleaning up the underlying data (physically deleting rows), the task completes quickly.
Values in the SETTINGS table control the clean up of expired sessions. By default, although the clean up of expired sessions occurs every two hours, the SETTINGS table does not contain a value for this setting. Use the SETTINGS_WRITE_STRING (value, namespace, setting) stored procedure to change the default clean up value.

For example, setting the value to "2d" deletes expired sessions older than two days.

Note: Oracle recommends that you set the ExpiredLongSessionTimeout sessions to at least one hour longer than your longest job. For example, if your longest job is a summarizer job that usually takes 12 hours, you should set the value in the SETTINGS table to at least 13.

Refer to the table below for information about the USESSION_CLEANUP_EXPIRED Settings.

**USESSION CLEANUP EXPIRED Settings**

<table>
<thead>
<tr>
<th>Setting Description: ExpiredSessionTimeout determines how long an inactive user session will remain in the records before it is marked deleted. User sessions are created when a P6 user logs into P6.</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespace</td>
<td>database.cleanup.Usession</td>
</tr>
<tr>
<td>Setting Name</td>
<td>ExpiredSessionTimeout</td>
</tr>
<tr>
<td>Default Setting</td>
<td>2h</td>
</tr>
<tr>
<td>Type</td>
<td>Interval</td>
</tr>
</tbody>
</table>

Setting Description: ExpiredLongSessionTimeout determines how long a session that is running a job-like operation (that is still processing) will remain in the records before it is marked deleted. Job-like operations are processed by job services and some are performed by logged in P6 users. Operations that are considered job-like are:

- Scheduling
- Leveling
- Apply Actuals
- Update Progress
- Copy/Paste Project
- Create Project from Template
- Maintain Baselines (create new baseline)
- Approve Timesheets
- Summarize
- PX Publish
- Export
- Import

<table>
<thead>
<tr>
<th>Setting Name</th>
<th>ExpiredLongSessionTimeout</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespace</td>
<td>database.cleanup.Usession</td>
</tr>
</tbody>
</table>
Default Setting  |  12h
---|---
Type  |  Interval

### Tracking Concurrent Usage of P6 EPPM

To track concurrent usage of P6 EPPM, you can run queries against the USESSION and USESSAUD tables to perform self-audits. See sample queries below.

**Note:** See *[DAMON (Data Monitor) Procedures](on page 61)* for information on how to set up the USESSAUD procedure. To ensure accuracy of these queries, run them before physically deleting remaining USESSION records and cleaning up the USESSAUD table.

- Against the USESSION table, run the following query to determine how many users are logged in at a given time:
  ```sql
  select count(*) from usession where delete_session_id is null
  ```
- Against the USESSION table, run the following query to determine how many users are logged into a specific P6 EPPM product at a given time:
  ```sql
  select count(*) from usession where delete_session_id is null and app_name='P6 EPPM product name'
  ```
  where *P6 EPPM product name* is the application abbreviation.

**Note:** You can view all available application abbreviations by running the following query as an administrative database user:
```sql
select distinct(db_engine_type) from usereng
```

- Against the USESSAUD table, run a query similar to the following to determine how many users logged into P6 EPPM on a specific date during a specified time range. You can alter the date, time range, and P6 EPPM product as needed. The following example will search for all users who logged into P6 Professional on February 17, 2010 between 9am and 10am:

  **For an Oracle database:**
  ```sql
  select * from usessaud where login_date between to_date('17-FEB-10 09:00:00','DD-MON-YY HH:MI:SS') and to_date('17-FEB-10 10:00:00','DD-MON-YY HH:MI:SS') and app_name='Project Management'
  ```

  **For a Microsoft SQL Server database:**
  ```sql
  select * from usessaud where login_date between '2010-02-17 09:00' and '2011-02-17 10:00' and app_name='Project Management'
  ```

**Tips**

See "Counting Users" in the *[P6 EPPM System Administration Guide]* for information on counting users and how to view the total number of licenses assigned for each module.
DAMON (Data Monitor) Procedures

The second database job is the DAMON data monitor job. The DAMON job runs the majority of the background processing and is responsible for running background clean up processes required by the application that can potentially take a relatively long time to run.

Oracle and DAMON

By default, DAMON runs every Saturday. It uses the Oracle DBMS_SCHEDULER package to schedule the jobs. An Interval setting controls the schedule and accepts the same parameters as the DBMS_SCHEDULER interval. For more information, refer to your Oracle database documentation.

Microsoft SQL Server and DAMON

By default, DAMON runs every Saturday. You can set it to run every two weeks or on a specific day. To run DAMON every two weeks, use the following command to set the interval: `-eg 2W`

To set DAMON to run on a specific day, use the following setting under namespace: `Database.background.Damon DayOfWeek`

DAMON Procedures

DAMON cleans the following:
- BGPLOG table containing the background logs
- REFRDEL table
- PRMQUEUE table
- Remaining USESSION records
- Logically deleted records
- PAUDIT table
- USESSION audit table (USESSAUD)

You can also dynamically extend DAMON functionality via the user-defined procedure, USER_DEFINED_BACKGROUND.

BGPLOG_CLEANUP Procedure

This procedure keeps the BGPLOG table at a reasonable size. The default cleanup interval is 5 days which will result in a table size of about 54,000 records.

Refer to the following table for information about the settings associated with the BGPLOG_CLEANUP procedure.

**BGPLOG_CLEANUP Settings**

<table>
<thead>
<tr>
<th>Setting Description: The oldest records to keep in the BGPLOG table.</th>
<th>Setting Name</th>
<th>Default Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespace database.cleanup.BackGroundProcessLog</td>
<td>KeepInterval</td>
<td>5d</td>
</tr>
</tbody>
</table>
Referrer Cleanup Procedure

This procedure deletes records from the REFRDEL table. The REFRDEL CLEANUP procedure runs based on the frequency of data_monitor_job, which has a default frequency of one week. Alternatively, you can run REFRDEL_CLEANUP by itself if needed.

Refer to the following table for information about the settings associated with the REFRDEL_CLEANUP procedure:

### REFRDEL_CLEANUP Settings

<table>
<thead>
<tr>
<th>Setting Description</th>
<th>Namespace</th>
<th>Setting Name</th>
<th>Default Setting</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identifies the number of days that records are stored in the REFRDEL table before they can be removed. For example, the default setting keeps the REFRDEL records from the last five days.</td>
<td>database.cleanup.Refrdel</td>
<td>KeepInterval</td>
<td>5d</td>
<td>Interval</td>
</tr>
<tr>
<td>Identifies the number of days that are set to be removed from the REFRDEL table starting with the oldest record in the table.</td>
<td>database.cleanup.Refrdel</td>
<td>DaysToDelete</td>
<td>1</td>
<td>Numeric</td>
</tr>
<tr>
<td>Determines the intervals of time (in minutes) in which data is grouped and removed from the REFRDEL table. The number of IntervalSteps is equal to DaysToDelete (in minutes) divided by IntervalStep.</td>
<td>database.cleanup.Refrdel</td>
<td>IntervalStep</td>
<td>15</td>
<td>Numeric</td>
</tr>
</tbody>
</table>
**REFRDEL Bypass Procedure**

The `REFRDEL` table maintains a list of deleted records from P6 database tables. However, when an entire project is deleted, a large amount of detailed delete records can be inserted into the `REFRDEL` table resulting in the potential for downstream performance degradation when joins are made to the `REFRDEL` table.

The `REFRDEL` Bypass procedure is an alternative to inserting `REFRDEL` records for tracking delete records on a large scale. This procedure bypasses the `REFRDEL` table and simply adds a single delete record for a deleted project and project ID.

By default, the `REFRDEL` BYPASS procedure is set to 0 (zero).

To enable `REFRDEL` Bypass, execute the following procedure and set to a non-zero value:

```
SQL> exec SET_REFRDEL_PROJECT_BYPASS(1);
```

Database triggers check for the value of the `REFRDEL` BYPASS value and process accordingly.

**Note:** The bypass procedure is meant only for a PROJECT DELETE operations.

---

**CLEANUP_PRMQUEUE Procedure**

This procedure deletes records from the `PRMQUEUE` table based on the value of the `KeepInterval` setting. The remaining settings are similar to the `REFRDEL_CLEANUP`.

Refer to the following table for information about the settings associated with the `CLEANUP_PRMQUEUE` procedure:

### CLEANUP_PRMQUEUE Settings

<table>
<thead>
<tr>
<th>Setting Description: The oldest records to keep in the <code>PRMQUEUE</code> table. Default is five days.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespace</td>
</tr>
<tr>
<td>Setting Name</td>
</tr>
<tr>
<td>Default Setting</td>
</tr>
<tr>
<td>Type</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Setting Description: Determines whether the procedure will delete all of the <code>PRMQUEUE</code> records possible on each pass.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespace</td>
</tr>
<tr>
<td>Setting Name</td>
</tr>
<tr>
<td>Default Setting</td>
</tr>
<tr>
<td>Type</td>
</tr>
<tr>
<td>---------------</td>
</tr>
<tr>
<td>Setting Description: Determines whether all of the records are cleaned. If the total record count is less than this number then all the records are cleaned.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Namespace</th>
<th>database.cleanup.Prmqueue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting Name</td>
<td>DeleteAllThreshold</td>
</tr>
<tr>
<td>Default Setting</td>
<td>1,000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Numeric</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting Description: Percentage of records to delete on each pass.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Namespace</th>
<th>database.cleanup.Prmqueue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting Name</td>
<td>DeletePercentage</td>
</tr>
<tr>
<td>Default Setting</td>
<td>10(%)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Numeric</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting Description: Maximum rows to delete on each pass.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Namespace</th>
<th>database.cleanup.Prmqueue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting Name</td>
<td>MaxRowsToDelete</td>
</tr>
<tr>
<td>Default Setting</td>
<td>10,000</td>
</tr>
</tbody>
</table>

**USESESSION_CLEAR_LOGICAL_DELETES Procedure**

This procedure physically deletes all logically deleted USESSION records. This procedure does not have settings associated with it: All logically deleted USESSION records are cleared.

**CLEANUP_LOGICAL_DELETES Procedure**

This procedure removes logically deleted rows based on the value of the KeepInterval setting. Records in the database can be marked as deleted (logically deleted) by setting the DELETE_SESSION_ID column to a non-null value. By default, records that were deleted more than 5 days ago will be deleted by this procedure.

**Notes:**

- The CLEANUP_LOGICAL_DELETES procedure will not delete
records whose DELETE_SESSION_ID column is set to a negative value.

- This procedure will not delete records older than the earliest user session in USESSION, as determined by the minimum value in the login_date column.

Refer to the following table for information about the settings associated with the CLEANUP_LOGICAL_DELETES procedure:

**CLEANUP_LOGICAL_DELETES Settings**

<table>
<thead>
<tr>
<th>Setting Description: The oldest logically deleted records to keep in tables.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespace</td>
</tr>
<tr>
<td>Setting Name</td>
</tr>
<tr>
<td>Default Setting</td>
</tr>
<tr>
<td>Type</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Setting Description: Determines whether the procedure will delete all of the logically deleted records possible on each pass.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespace</td>
</tr>
<tr>
<td>Setting Name</td>
</tr>
<tr>
<td>Default Setting</td>
</tr>
<tr>
<td>Type</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Setting Description: Maximum rows to delete on each pass.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespace</td>
</tr>
<tr>
<td>Setting Name</td>
</tr>
<tr>
<td>Default Setting</td>
</tr>
<tr>
<td>Type</td>
</tr>
</tbody>
</table>

**PAUDIT_CLEANUP Procedure**

If the auditing feature is enabled, this procedure will physically delete records from the table based on the value of the KeepInterval setting.

Refer to the following table for information about the settings associated with the PAUDIT_CLEANUP procedure:
PAUDIT_CLEANUP Settings

Setting Description: Should the procedure attempt PAUDIT records cleanup.

<table>
<thead>
<tr>
<th>Namespace</th>
<th>database.cleanup.auditing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting Name</td>
<td>Enabled</td>
</tr>
<tr>
<td>Default Setting</td>
<td>1 (true)</td>
</tr>
<tr>
<td>Type</td>
<td>Boolean</td>
</tr>
</tbody>
</table>

Setting Description: The oldest audit records to keep in PAUDIT.

<table>
<thead>
<tr>
<th>Namespace</th>
<th>database.cleanup.auditing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting Name</td>
<td>KeepInterval</td>
</tr>
<tr>
<td>Default Setting</td>
<td>30d</td>
</tr>
<tr>
<td>Type</td>
<td>Interval</td>
</tr>
</tbody>
</table>

CLEANUP_USESSAUD Procedure

The CLEANUP_USESSAUD procedure deletes records from the USESSAUD table based on the KeepInterval parameter setting. All other settings are similar to the REFRDEL_CLEANUP procedure. The following table describes the settings associated with the CLEANUP_USESSAUD procedure.

CLEANUP_USESSAUD Settings

Setting Description: Determines the oldest records to keep in the USESSAUD table. Based on the logout time, data beyond the KeepInterval parameter value will not be deleted. The KeepInterval parameter setting overrides all other CLEANUP_USESSAUD settings.

<table>
<thead>
<tr>
<th>Namespace</th>
<th>database.cleanup.Usessaud</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting Name</td>
<td>KeepInterval</td>
</tr>
<tr>
<td>Default Setting</td>
<td>5d</td>
</tr>
<tr>
<td>Type</td>
<td>Interval</td>
</tr>
</tbody>
</table>

Setting Description: Deletes all the REFRDEL records that satisfy the KeepInterval setting on each pass. The DeleteAll parameter setting overrides the settings of DeleteAllThreshold, DeletePercentage, and MaxRowsToDelete parameters.

<table>
<thead>
<tr>
<th>Namespace</th>
<th>database.cleanup.Usessaud</th>
</tr>
</thead>
</table>
### Setting Name: DeleteAll

**Default Setting:** 0 (false)

**Type:** Boolean

**Setting Description:** Determines the minimum number of records to delete after satisfying the KeepInterval parameter setting. By default, a minimum of 1000 records are deleted. If the total record count is less than this setting, all records are deleted. The DeleteAllThreshold parameter setting overrides the settings of the DeletePercentage and MaxRowsToDelete parameters.

**Namespace:** database.cleanup.Usessaud

### Setting Name: DeleteAllThreshold

**Default Setting:** 1,000

**Type:** Numeric

**Setting Description:** Determines the maximum rows to delete on each pass after satisfying the KeepInterval parameter setting. The MaxRowsToDelete parameter setting overrides the DeletePercentage parameter setting.

**Namespace:** database.cleanup.Usessaud

### Setting Name: MaxRowsToDelete

**Default Setting:** 10,000

**Type:** Numeric

**Setting Description:** Determines the percentage of records to delete on each pass after satisfying the DeleteAllThreshold and MaxRowsToDelete settings. However, the percentage of records deleted is limited to the default value of the MaxRowsToDelete setting.

**Namespace:** database.cleanup.Usessaud

### Setting Name: DeletePercentage

**Default Setting:** 10 (%)

**Type:** Numeric

**Setting Description:**

**Namespace:** database.cleanup.Usessaud

---

**USERDEFINEDBACKGROUND Procedure**

This procedure is an optional customer procedure that DAMON runs. This procedure does not have settings associated with it.
PX Tables Clean Up Procedure

To avoid operational problems for P6, PX table entries are not automatically deleted when you delete data from P6 tables. You can use the following clean up procedure to clear the obsolete entries in PX tables.

1) Run the CLEAN_PX_DELETE stored procedure to purge obsolete entries in the PX tables:

   With Oracle:

   BEGIN
   CLEAN_PX_DELETE();
   END;

   With Microsoft SQL Server:

   USE [ddname]
   GO
   DECLARE @return_value int
   EXEC @return_value = [dbo].[CLEAN_PX_DELETE]
   SELECT 'Return Value' = @return_value
   GO

   where ddname is the is your admin user name.

   **Note**: You can run this procedure any time you want. You can also run it as a database background job and schedule a specific time for it to run.

2) To delete batch size and max delete from each table in a single run, run the following queries:

   **Note**: You can specify the delete batch size and max delete from each table in a single run. Delete batch size is the number of rows deleted before a commit. Max delete is the max number of rows that will be deleted from a table in a single run. They have default values if you do not provide your own. They’re used to avoid resource problems with the P6 operations.

   ```
   - insert into settings (namespace, setting_name, setting_value) values ('PxService.Cleanup','DeleteMaxSize','100000');
   - insert into settings (namespace, setting_name, setting_value) values ('PxService.Cleanup','DeleteBatchSize','5000');
   ```

Safe Deletes

The P6 EPPM database normally handles restoring select deleted data using a safe delete setting. While using P6 Professional, the Undo command (Edit, Undo) allows users to restore certain types of data that have been deleted. Deleted data remains in the P6 EPPM database until the CLEANUP_LOGICAL_DELETES procedure clears it (after 5 days, by default).

See the *P6 Professional Help* for more information about using undo.
Turning Off Safe Deletes

You can turn off safe deletes to save storage space. Turning off safe deletes disables undo functionality and instantly clears deleted data from the P6 EPPM database.

To turn off safe deletes:

1) Verify the current state of your safe deletes setting. In the database, if the table ADMIN_CONFIG has the following row, a CONFIG_VALUE of 'N' means turn off safe deletes.

   CONFIG_NAME = 'SAFEDELETE.ACTIVE' and CONFIG_TYPE = 'SETTINGS'

   **Note:** This is only loaded at startup. If you change CONFIG_VALUE while a user is running P6 Professional, the setting will not apply until the user restarts the P6 Professional session.

2) Once you have determined the current state of your safe deletes setting, run one of the following statements.

   ▶ To turn off safe deletes for the first time:
     
     ```sql
     INSERT INTO ADMIN_CONFIG (CONFIG_NAME, CONFIG_TYPE, CONFIG_VALUE)
     VALUES ('SAFEDELETE.ACTIVE', 'SETTINGS', 'N')
     ```

   ▶ To turn on safe deletes after it has been turned off:
     
     ```sql
     UPDATE ADMIN_CONFIG SET CONFIG_VALUE = 'Y' WHERE CONFIG_NAME =
     'SAFEDELETE.ACTIVE' AND CONFIG_TYPE = 'SETTINGS'
     ```

   ▶ To turn off safe deletes after it has been turned on:
     
     ```sql
     UPDATE ADMIN_CONFIG SET CONFIG_VALUE = 'N' WHERE CONFIG_NAME =
     'SAFEDELETE.ACTIVE' AND CONFIG_TYPE = 'SETTINGS'
     ```

3) Restart the P6 server.
Legal Notices

Oracle Primavera P6 EPPM Database Administration Guide for On-Premises

Copyright © 1999, 2018, Oracle and/or its affiliates. All rights reserved.
Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are “commercial computer software” pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate failsafe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

This software or hardware and documentation may provide access to or information on content, products and services from third-parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.