

Oracle® Fusion Cloud EPM

EPM 自動化の操作



F28927-24

ORACLE®

Oracle Fusion Cloud EPM EPM 自動化の操作、

F28927-24

Copyright © 2016, 2025, Oracle and/or its affiliates.

著者: EPM Information Development Team

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, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. 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 fail-safe, 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.

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside 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, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about 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 unless otherwise set forth in an applicable agreement between you and Oracle. 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, except as set forth in an applicable agreement between you and Oracle.

目次

ドキュメントのアクセシビリティについて

ドキュメントのフィードバック

1 EPM 自動化について

EPM 自動化のインストール	1-1
能力およびポートの要件	1-2
サポートされているプラットフォーム	1-2
Java Runtime Environment および EPM 自動化	1-4
OpenJDK の使用	1-4
Windows での手順	1-5
Linux/UNIX/macOS X での手順	1-5
EPM 自動化のコマンドのサーバー側での実行	1-6
EPM 自動化の更新	1-6
EPM 自動化のアンインストール	1-7
EPM 自動化の暗号化レベルの理解	1-7
OCI (Gen2)環境での OAuth 2.0 承認プロトコルの使用	1-7

2 コマンド・リファレンス

EPM 自動化コマンドの実行について	2-1
前提条件	2-1
デフォルトのファイルの場所	2-3
Transport Layer Security プロトコル 1.2 の有効化	2-4
EPM 自動化コマンドの使用	2-4
パラメータに対する複数の値の指定	2-5
日次メンテナンス中の動作	2-5
EPM 自動化の実行	2-6
Windows	2-6
Linux	2-7
EPM 自動化の複数インスタンスの実行	2-8

コマンドの概要	2-10
EPM 自動化コマンド	2-16
addUsers	2-16
addUsersToGroup	2-17
addUsersToTeam	2-18
addUserToGroups	2-19
applicationAdminMode	2-20
applyDataGrants	2-21
archiveTmTransactions	2-21
assignRole	2-23
autoPredict	2-25
calculateModel	2-25
clearCube	2-28
clearDataByPointOfView	2-28
clearDataByProfile	2-29
clearPOV	2-30
cloneEnvironment	2-31
compactCube	2-36
copyDataByPointOfView	2-36
copyDataByProfile	2-37
copyFileFromInstance	2-38
copyFromObjectStorage	2-38
copyFromSFTP	2-40
copyOwnershipDataToNextYear	2-41
copyPOV	2-42
copySnapshotFromInstance	2-43
copyToObjectStorage	2-44
copyToSFTP	2-46
createGroups	2-47
createNRSnapshot	2-47
createReconciliations	2-48
deleteFile	2-48
deleteGroups	2-50
deletePointOfView	2-50
deletePOV	2-51
deployCube	2-52
deployEJTemplates	2-53
deployFormTemplates	2-53
deployTaskManagerTemplate	2-54
dismissIPMInsights	2-55
downloadFile	2-55
enableApp	2-56

enableQueryTracking	2-57
encrypt	2-57
essbaseBlockAnalysisReport	2-58
executeAggregationProcess	2-59
executeBurstDefinition	2-60
executeReportBurstingDefinition	2-61
exportAccessControl	2-61
exportAppAudit	2-62
exportAppSecurity	2-63
exportARApplicationProperties	2-64
exportBackgroundImage	2-65
exportCellLevelSecurity	2-65
exportConsolidationJournals	2-66
exportData	2-66
exportDataManagement	2-67
exportDimension	2-67
exportDimensionMapping	2-68
exportEJournals	2-69
exportEssbaseData	2-71
exportJobConsole	2-72
exportLibraryArtifact	2-74
exportLibraryDocument	2-75
exportLogoImage	2-77
exportMapping	2-77
exportMetadata	2-78
exportOwnershipData	2-78
exportQueryResults	2-79
exportSnapshot	2-81
exportTemplate	2-82
exportTaskManagerAccessControl	2-83
exportValidIntersections	2-84
extractDimension	2-84
extractPackage	2-85
feedback	2-86
getApplicationAdminMode	2-88
getDailyMaintenanceStartTime	2-88
getEssbaseQryGovExecTime	2-89
getIdleSessionTimeout	2-89
getIPAllowlist	2-90
getRestrictedDataAccess	2-91
getSubstVar	2-91
getVirusScanOnFileUploads	2-92

groupAssignmentAuditReport	2-93
help	2-94
importAppAudit	2-94
importAppSecurity	2-95
importARApplicationProperties	2-96
importBackgroundImage	2-96
importBalances	2-97
importCellLevelSecurity	2-97
importConsolidationJournals	2-98
importData	2-99
importDataManagement	2-100
importDimension	2-101
importJobConsole	2-102
importLibraryArtifact	2-102
importLogoImage	2-104
importMapping	2-104
importMetadata	2-105
importOwnershipData	2-107
importPreMappedBalances	2-107
importPreMappedTransactions	2-108
importProfiles	2-109
importRates	2-109
importRCAAttributeValues	2-110
importReconciliationAttributes	2-111
importSnapshot	2-112
importSupplementalCollectionData	2-115
importSupplementalData	2-116
importTemplate	2-117
importTMAAttributeValues	2-117
importTmPremappedTransactions	2-119
importValidIntersections	2-120
invalidLoginReport	2-121
listBackups	2-122
listFiles	2-123
loadData	2-124
loadDimensionViewpoint	2-125
loadDimData	2-126
loadViewpoint	2-127
login	2-128
logout	2-131
maskData	2-132
mergeDataSlices	2-132

mergeSlices	2-133
optimizeASOCube	2-133
programDocumentationReport	2-134
provisionReport	2-135
purgeArchivedTmTransactions	2-137
purgeTmTransactions	2-138
recomputeOwnershipData	2-139
recreate	2-140
refreshCube	2-143
removeUserFromGroups	2-143
removeUsers	2-144
removeUsersFromGroup	2-145
removeUsersFromTeam	2-146
renameSnapshot	2-147
replay	2-148
resetService	2-149
restoreBackup	2-150
restructureCube	2-152
roleAssignmentAuditReport	2-152
roleAssignmentReport	2-154
runAutomatch	2-155
runBatch	2-156
runBusinessRule	2-157
runCalc	2-157
runComplianceReport	2-159
runDailyMaintenance	2-160
runDataRule	2-161
runDMReport	2-163
runIntegration	2-164
runIntercompanyMatchingReport	2-168
runMatchingReport	2-169
runPipeline	2-170
runPlanTypeMap	2-172
runRuleSet	2-172
runSupplementalDataReport	2-173
runTaskManagerReport	2-174
sendMail	2-175
setApplicationAdminMode	2-176
setDailyMaintenanceStartTime	2-177
setDemoDates	2-178
setEJJournalStatus	2-179
setEncryptionKey	2-180

setEssbaseQryGovExecTime	2-181
setIdleSessionTimeout	2-181
setIPAllowlist	2-182
setManualDataAccess	2-183
setPeriodStatus	2-184
setRestrictedDataAccess	2-185
setSubstVars	2-185
setVirusScanOnFileUploads	2-186
simulateConcurrentUsage	2-187
skipUpdate	2-191
snapshotCompareReport	2-193
sortMember	2-194
unassignRole	2-195
updateGuidedLearningSettings	2-197
updateUsers	2-197
upgrade	2-198
uploadFile	2-199
userAuditReport	2-200
userGroupReport	2-201
validateConsolidationMetadata	2-202
validateModel	2-203
終了コード	2-204

3 コマンド実行のサンプル・シナリオ

サンプル・スクリプトのコピーについて	3-1
すべてのサービスのサンプル・シナリオ	3-1
アプリケーション・スナップショットのコンピュータへのバックアップ	3-3
ユーザーへの日次メンテナンス完了の通知	3-5
Oracle Object Storage との間でのスナップショットのコピー	3-12
ユーザーの作成および事前定義済役割への割当て	3-14
ライセンスされたユーザー(役割に割り当てられたユーザー)の数のカウント	3-18
役割に割り当てられたユーザーの監査レポートの作成	3-20
役割割当ての作成と監査レポートの取消し	3-24
個人情報保護法に準拠するためのアクセス・ログとアクティビティ・レポートのマスク	3-27
アクティビティ・レポートのローカル・コンピュータへのダウンロード自動化	3-32
環境からのアクセス・ログのダウンロード	3-36
環境のクローニングの自動化	3-39
プライマリ環境での日次メンテナンス完了後の、プライマリ環境からスタンバイ環境への毎日のクローニング	3-43
環境からの不要なファイルの削除	3-52

環境からのファイルの検索およびダウンロード	3-54
監査用の古い Cloud EPM 環境の再作成	3-55
データベース・アクセスの監査およびコンプライアンスの自動化	3-66
ユーザーおよび事前定義済役割割当てのレプリケート	3-77
あるアイデンティティ・ドメインから別のアイデンティティ・ドメインへのユーザーのレプリケート	3-78
ある環境から別の環境への事前定義済役割割当てのレプリケート	3-84
四半期の Cloud EPM アップグレード頻度の作成	3-91
Windows スクリプトと手順	3-92
UNIX/Linux スクリプトと手順	3-95
Groovy スクリプト	3-99
6 週間のテスト・サイクルを使用した四半期の Cloud EPM アップグレード頻度の作成	3-102
Planning、連結、Tax Reporting および Enterprise Profitability and Cost Management 用のサンプル・シナリオ	3-116
集約ストレージ・キューブからの大量のセル・エクスポートの自動化	3-117
アプリケーションへのメタデータのインポート	3-126
データのインポート、計算スクリプトの実行、ブロック・ストレージ・データベースから集約ストレージ・データベースへのデータのコピー	3-128
メタデータおよびデータのエクスポートおよびダウンロード	3-131
アプリケーション・データのエクスポートおよびダウンロード	3-133
アプリケーション監査レコードのアーカイブの自動化	3-135
Windows スクリプト	3-137
Linux スクリプト	3-137
環境へのデータ・ファイルのアップロードおよびデータ・ロード・ルールの実行	3-138
日次データ統合の自動化	3-141
Account Reconciliation のサンプル・シナリオ	3-143
期間への事前フォーマット済残高のロード	3-144
バックアップ・スナップショットのアップロードとインポート	3-146
照合した古いトランザクションのアーカイブおよびアーカイブしたトランザクションのページ	3-148
Profitability and Cost Management のサンプル・シナリオ	3-154
アプリケーションへのメタデータのインポート	3-154
データのインポートとプログラム・ルールの実行	3-156
Oracle Enterprise Data Management Cloud のサンプル・シナリオ	3-159
Oracle Enterprise Data Management Cloud ディメンションおよびマッピングの Cloud EPM アプリケーションとの同期	3-159
Cloud EPM ディメンションの Oracle Enterprise Data Management Cloud アプリケーションとの同期	3-161
スクリプトの実行の自動化	3-163
EPM 自動化アクティビティのモニタリング	3-163

4 EPM 自動化をインストールしないコマンドの実行

サーバー側のコマンド実行をサポートする環境	4-1
情報ソース	4-2
サポートされているコマンド	4-2
サーバー側の Groovy を使用して EPM 自動化を実行するために使用されるメソッド	4-2
サーバー側の Groovy スクリプトを使用した環境のクローニング	4-3
サーバー側の Groovy スクリプトを使用したアクティビティ・レポートの電子メール送信	4-5

5 Cloud EPM 環境のレプリケート

日次レプリケーションの設定	5-1
オンデマンド・レプリケーションの設定	5-2
セカンダリ環境の構成	5-3

A simulateConcurrentUsage コマンドの実行の準備

requirement.csv ファイルの作成	A-2
入力ファイルの作成	A-3
フォーム入力ファイルを開く	A-4
フォーム入力ファイルの保存	A-4
ビジネス・ルール入力ファイルの実行	A-5
ビジネス・ルールセット入力ファイルの実行	A-5
データ・ルール入力ファイルの実行	A-6
アド・ホック・グリッド入力ファイル	A-6
レポート入力ファイルの実行	A-7
ブック入力ファイルの実行	A-7
UserVarMemberMapping.csv ファイルの作成	A-8
options.xml ファイルの作成	A-8
users.csv ファイルの作成	A-8
入力 ZIP ファイルの作成および環境へのアップロード	A-9
同時使用のシミュレート・レポートのサンプル	A-9

B replay コマンドの実行準備

replay コマンドについて	B-1
前提条件	B-1
HAR ファイルの作成	B-2
リプレイ・ファイルの作成	B-5
トレース・ファイルの生成	B-6

C 特殊文字の処理

D 各 Cloud EPM サービスに固有のコマンド

Account Reconciliation のコマンド	D-2
Financial Consolidation and Close のコマンド	D-3
Narrative Reporting のコマンド	D-4
Oracle Enterprise Data Management Cloud のコマンド	D-5
Planning、Planning モジュール、フリーフォーム、Strategic Workforce Planning および Sales Planning のコマンド	D-6
Profitability and Cost Management のコマンド	D-7
Enterprise Profitability and Cost Management のコマンド	D-8
Tax Reporting のコマンド	D-9

ドキュメントのアクセシビリティについて

オラクルのアクセシビリティについての詳細情報は、Oracle Accessibility Program の Web サイト(<http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>)を参照してください。

Oracle サポートへのアクセス

サポートをご契約のお客様には、My Oracle Support を通して電子支援サービスを提供しています。詳細情報は <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> か、聴覚に障害のあるお客様は <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> を参照してください。

ドキュメントのフィードバック

このドキュメントに対するフィードバックを送るには、Oracle Help Center トピックのページの下部にあるフィードバック・ボタンをクリックします。epmdoc_yw@oracle.com に電子メールを送信することもできます。

1

EPM 自動化について

EPM 自動化を使用すると、Oracle Fusion Cloud Enterprise Performance Management 環境内のタスクをリモートで実行できます。

Cloud EPM サービス管理者は、次のような多数の繰り返し可能なタスクを自動化できます:

- メタデータ、データ、アーティファクト・スナップショットとアプリケーション・スナップショット、テンプレートおよびデータ管理マッピングのインポートとエクスポート
- 環境へのファイルのアップロード、ファイルのリスト、サービスからのファイルの削除
- スナップショット、レポートおよびメタデータ・ファイルとデータ・ファイルのサービスからのダウンロード
- データに対するビジネス・ルールの実行、およびアプリケーションのリフレッシュ
- あるデータベースから別のデータベースへのデータのコピー; 通常はブロック・ストレージ・データベースから集約ストレージ・データベース、またはブロック・ストレージ・データベースから別のブロック・ストレージ・データベース
- データ管理バッチ・ルールの実行
- データ管理レポート、プロビジョニング・レポートおよびユーザー監査レポートの生成
- 事前マップ済残高データ、通貨レート、事前マップ済トランザクション、残高データおよびプロファイルの **Account Reconciliation** へのインポート
- 照合プロセスを開始する期間へのプロファイルのコピー
- **Profitability and Cost Management** アプリケーションの計算キューブのデプロイ
- **Enterprise Profitability and Cost Management** および **Profitability and Cost Management** アプリケーションでの視点のクリア、コピーおよび削除
- 環境に対する **Oracle Smart View for Office** または **REST API** の負荷がリプレイされ、高負荷の状況でのパフォーマンス・テストが可能になります
- ファイルから **Financial Consolidation and Close** への補足データのインポート

様々なタスクを実行できるスクリプトを作成し、その実行をスケジューラを使用することで自動化できます。たとえば、環境から日次メンテナンス・バックアップをダウンロードするスクリプトを作成し、アーティファクトおよびデータのローカル・バックアップを作成できます。



[チュートリアル: EPM 自動化を使用した Planning コマンドの実行方法](#)

EPM 自動化のインストール

EPM 自動化をインストールしてコマンドを実行します。一部のコマンドは、EPM 自動化をインストールせずに、Groovy スクリプトを使用して Oracle Fusion Cloud Enterprise Performance Management で直接実行することもできます。

Windows、Linux/UNIX および macOS X 用の EPM 自動化のインストーラは Cloud EPM 環境から入手できます。

Windows バージョン 10 以降では Windows 管理者のみに EPM 自動化のインストールが許可されているため、Windows 管理者のみがインストールおよびアップグレードできます。EPM 自動化は、インストールしたユーザーまたは別の Windows 管理者がアップグレードできます。

この項の内容:

- [能力およびポートの要件](#)
- [サポートされているプラットフォーム](#)
- [Java Runtime Environment および EPM 自動化](#)
- [OpenJDK の使用](#)
- [Windows での手順](#)
- [Linux/UNIX/macOS X での手順](#)
- [EPM 自動化コマンドのサーバー側での実行](#)

能力およびポートの要件

EPM 自動化は軽量のクライアントであるため、大きなクライアント・フットプリントは必要ありません。すべての処理は Oracle Fusion Cloud Enterprise Performance Management で実行されます。

EPM 自動化は、セキュアな HTTP 接続を介して外部ホストにアクセスできる Oracle Integration Cloud マシン、標準クライアント・マシンおよび仮想マシンにインストールできます。

EPM 自動化は、標準 TLS ポート(ポート 443)を使用して Cloud EPM に接続します。EPM 自動化用に追加の送信ポートを開く必要はありません。

この時点で、EPM 自動化は相互 TLS (mTLS)認証をサポートしません。

サポートされているプラットフォーム

EPM 自動化は、セキュアな HTTP 接続を介して外部ホストにアクセスできる Oracle Integration Cloud (OIC)マシンおよび仮想マシンにインストールできます。

 **Note:**

- EPM 自動化は、オペレーティング・システムのベンダーによって現在サポートされている 64 ビット・オペレーティング・システムのみで使用される可能性があります。
- EPM 自動化は SOCKS プロキシでは機能しません。HTTP または HTTPS プロキシでのみ機能します。
- EPM 自動化では、プロキシ・サーバーに接続するために、基本、ダイジェスト、Kerberos、ネゴシエートおよび NTLM 認証メカニズムをサポートしています。
- EPM 自動化は、API ゲートウェイ(Google APIGEE、IBM Data Power、他のリバース・プロキシ・サーバーなど)を介して Oracle Fusion Cloud Enterprise Performance Management に接続できます。
これが機能するには/epmcloud などのコンテキストなしで、Cloud EPM 環境の URL としてターゲットを設定して、ゲートウェイまたはリバース・プロキシを構成します。例: https://epm-idDomain.epm.dataCenterRegion.oraclecloud.com。次に、login コマンドで、Cloud EPM URL のかわりにリバース・プロキシ URL を使用します。構成情報については、ゲートウェイまたはプロキシ・サーバーのドキュメントを参照してください。

プロキシ設定の構成中は、Cloud EPM からの応答コードを EPM 自動化に必ず渡してください。このとき、EPM 自動化が 200、206、400、404、500、501 などの応答コードを正しく処理できるように、応答コードは一切変更しないでください。たとえば、IBM Datapower の場合、proxy HTTP Response を ON に設定します。さらに、API ゲートウェイは HTTP メソッド(GET、POST、PUT、PATCH、および DELETE)を許可する必要があります。

Linux および UNIX コンピュータでは、EPM 自動化は次の環境変数を探して、HTTP または HTTPS プロキシ設定を決定します。

- proxyHost
- proxyPort

HTTP プロキシ設定の例:

```
export proxyHost=host.example.com
export proxyPort=8000
```

HTTPS プロキシ設定の例:

```
export proxyHost=host.example.com
export proxyPort=8080
```

 **Note:**

EPM 自動化は、OAuth 2.0 承認プロトコルを使用して、コマンドを実行するため、特にコマンドの実行を自動化するために Cloud EPM 環境にアクセスできます (OAuth 用に構成されている場合)。

基本認証を使用する環境では、EPM 自動化は企業の SSO (アイデンティティ・プロバイダ) 資格情報では機能しません。ユーザーは、企業資格証明を使用してサインインできないため、EPM 自動化にアクセスするためのユーザー・アカウントは、アイデンティティ・ドメインで管理する必要があります。サブスクリプションが SSO 用に構成されている場合は、EPM 自動化ユーザーがアイデンティティ・ドメイン資格情報でサインインできるようにする必要があります。Oracle Cloud Identity Management の管理のアイデンティティ・ドメイン資格証明によるサインインの有効化を参照してください。

ダウンロードの手順: 管理者用スタート・ガイドのクライアントのダウンロードおよびインストール。

Java Runtime Environment および EPM 自動化

Windows への EPM 自動化のインストールでは、必要な Java Runtime Environment (JRE) がインストールされます。しかし、Linux、Unix および macOS X インストーラに JRE は含まれていません。EPM 自動化を使用するには、JRE インストール(バージョン 8 からバージョン 11) にアクセスする必要があります。

お客様には Oracle Java Standard Edition (SE) を EPM 自動化とともに使用する権利があり、Java SE サブスクリプションを別途購入する必要はありません。EPM 自動化での Oracle JDK ライセンスの詳細は、Oracle サポート・ドキュメント 1557737.1: 『Support Entitlement for Java SE When Used As Part of Another Oracle Product』を参照してください。

OpenJDK の使用

Linux、Unix および macOS X プラットフォームでは、JRE のかわりに、OpenJDK バージョン 14 以降を使用できます。

Oracle による無償の、GPL でライセンスされた、本番対応の JDK である OpenJDK は、<https://openjdk.java.net> からダウンロードできます。OpenJDK をインストールする手順も、この Web サイトにあります。

EPM 自動化セッションを開始する前に、JAVA_HOME 環境変数が OpenJDK インストールを指すように設定します。

ホーム・ディレクトリにインストールされた OpenJDK バージョン 14 を使用する macOS X の例(Bash シェルを前提)。

```
cd ~/
export JAVA_HOME=$(/usr/jdk-14.jdk/Contents/Home)
```

ホーム・ディレクトリにインストールされた OpenJDK バージョン 14 を使用する **Linux の例** (Bash シェルを前提)。

```
cd ~/
export JAVA_HOME=/openjdk/jdk-14.0.2
```

Windows での手順

デフォルトでは、EPM 自動化は C:/Oracle/EPM Automate にインストールされます。

EPM 自動化をインストールするには:

1. EPM 自動化をインストールする Windows コンピュータから環境にアクセスします。
2. ホーム・ページで、自分のユーザー名をクリックして「**設定およびアクション**」にアクセスします。
3. 「**ダウンロード**」をクリックします。
4. ダウンロード・ページで、「EPM 自動化」セクションの「**Windows 用にダウンロード**」をクリックします。
5. インストーラをコンピュータに保存します。
6. インストーラ(EPM Automate.exe)を右クリックして「**管理者として実行**」を選択します。
7. 「**ユーザー アカウント制御**」で、「**はい**」をクリックします。
8. 画面上の指示に従ってインストールを完了します。

Linux/UNIX/macOS X での手順

EPM 自動化には、サポートされている JRE (バージョン 8 から 11)のデプロイメントへのアクセスが必要です。環境変数 JAVA_HOME は、JRE インストールを指し示すよう設定する必要があります。

EPM 自動化をインストールするには:

1. 環境にアクセスします。
2. ホーム・ページで、自分のユーザー名をクリックして「**設定およびアクション**」にアクセスします。
3. 「**ダウンロード**」をクリックします。
4. 「**ダウンロード**」ページで、「EPM 自動化」セクションの **Linux/macOS X 用にダウンロード**をクリックします。
5. 読取り/書込み/実行の権限のあるディレクトリにインストーラ(EPMAutomate.tar)を保存します。
6. 次のステップを完了します:
 - a. インストーラのコンテンツを抽出します
 - b. 必要な環境変数を設定します
 - c. インストーラのコンテンツを抽出したディレクトリの epmautomate/bin にディレクトリを変更します
 - d. epmautomate.sh を実行します:

次のスクリプトでは、必要に応じて実際のディレクトリ・パスを使用してください。たとえば、EPMAutomate.tar が HOME/oracle/epmautomate/bin に抽出された場合は、それを EPMAutomate_extract_directory の値として使用します。

ホーム・ディレクトリからインストールおよび実行する **macOS X の例**(Bash シェルを前提)。

```
cd ~/
tar xf directory_containing_downloaded_installer/EPMAutomate.tar
export JAVA_HOME=$(/usr/libexec/java_home)
export PATH $HOME/epmautomate/bin:$PATH
cd EPMAutomate_extract_directory/epmautomate/bin
./epmautomate.sh
```

ホーム・ディレクトリからインストールおよび実行する **Linux の例** (Bash シェルを前提)。JDK バージョン 1.8.0_191 を前提としています。

```
cd ~/
tar xf directory_containing_downloaded_installer/EPMAutomate.tar
export JAVA_HOME=/opt/jdk1.8.0_191
export PATH ~/Downloads/epmautomate/bin:$PATH
cd EPMAutomate_extract_directory/epmautomate/bin
./epmautomate.sh
```

EPM 自動化のコマンドのサーバー側での実行

EPM 自動化の一部のコマンドは、Groovy を使用して Oracle Fusion Cloud Enterprise Performance Management で直接実行できます。Groovy スクリプトを使用してコマンドを実行するために EPM 自動化をインストールする必要はありません。

サーバー側でのコマンドの実行は、クライアント・コンピュータで Groovy スクリプトを実行して EPM 自動化のコマンドを実行するのとは同じではないことに注意してください。

詳細は、[EPM 自動化をインストールしないコマンドの実行](#)を参照してください。

EPM 自動化の更新

EPM 自動化を使用すると、最新バージョンにサイレント・アップグレードできます。

ログインしてセッションを開始すると、EPM 自動化は現在インストールされているバージョンを識別します。インストールされているのが利用可能な最新バージョンでない場合、新しいバージョンが利用できるというメッセージが表示されます。

`upgrade` コマンドを実行して、EPM 自動化をサイレント・アップグレードします。

EPM 自動化のアンインストール

Note:

これらの手順は、クライアント・マシンから EPM 自動化を削除する場合にのみ使用します。現在の EPM 自動化バージョンを更新する場合は、[EPM 自動化の更新](#)の指示を使用します。

Windows

EPM 自動化をアンインストールするには:

1. 「設定」から「アプリ」を開き、「インストールされているアプリ」を開きます
2. インストールされているアプリケーションのリストで EPM 自動化を見つけます。
3. オーバーフロー・メニューから「アンインストール」を選択します。
4. 「アンインストール」をクリックします。
5. インストール・ディレクトリに、残っているファイルがあれば削除します(通常、C:\Oracle\Epm Automate)

Linux/UNIX/macOS X

EPM 自動化のディレクトリおよび EPMAutomate.tar ファイルを削除します。

EPM 自動化の暗号化レベルの理解

Oracle Fusion Cloud Enterprise Performance Management では、SHA-2/SHA-256 暗号的ハッシュ・アルゴリズムによる Transport Layer Security (TLS)を使用して、EPM 自動化との通信を保護します。

OCI (Gen2)環境での OAuth 2.0 承認プロトコルの使用

EPM 自動化は、OAuth 2.0 承認プロトコルを使用して、コマンドを実行するため、特にコマンドの実行を自動化するために OCI (GEN 2) Oracle Fusion Cloud Enterprise Performance Management 環境にアクセスできます。

OAuth 2.0 アクセスを有効にするには、アイデンティティ・ドメイン管理者が、アプリケーションを Oracle Cloud Identity Services のパブリック・クライアントとして登録する必要があります。OAuth はアプリケーションに適用されます。サブスクリプション全体ではありません。

OCI (GEN 2)環境用に OAuth 2.0 を設定する手順の詳細は、*REST API* ガイドの OAuth 2 での認証 - OCI (Gen 2)環境のみを参照してください。

 **Note:**

基本認証は、環境で OAuth が有効になっている場合でも機能します。既存の暗号化されたパスワード・ファイルは、将来使用する予定がある場合は上書きしないでください。

リフレッシュ・トークンとクライアント ID を含む暗号化されたパスワード・ファイルの作成

EPM 自動化に OAuth 2.0 を使用して環境にアクセスするサービス管理者は、暗号化されたパスワード・ファイルを作成するためにこれらの詳細が必要です。これは、環境にサインインするために使用されます:

- リフレッシュ・トークン
リフレッシュ・トークンの取得方法の詳細は、*REST API* ガイドの OAuth 2 での認証 - OCI (Gen 2)環境のみを参照してください。
- クライアント ID
クライアント ID は、アイデンティティ・ドメイン管理者がアプリケーションを OAuth 用に構成するときに生成されます。これは、アプリケーションの「構成」タブの「**全般情報**」の下に表示されます。

OAuth 認証用の暗号化されたパスワード・ファイルを作成するには:

1. EPM 自動化セッションを開始します。
2. 次のようなコマンドを実行します。

```
epmautomate encrypt REFRESH_TOKEN ENCRYPTION_KEY PASSWORD_FILE
```

ClientID=CLIENT_ID。ここで、REFRESH_TOKEN はセキュリティ・ストアからの復号化されたリフレッシュ・トークン、ENCRYPTION_KEY はパスワードを暗号化するための秘密キー、PASSWORD_FILE は暗号化されたリフレッシュ・トークンを格納するファイルの名前と場所です。パスワード・ファイルには、.epw 拡張子を使用する必要があります。

手順の詳細は、[encrypt](#) を参照してください。
3. 新しく生成されたパスワード・ファイルを使用して、OAuth を使用してサインインします。スクリプトを自動実行する場合は、新しく生成されたパスワード・ファイルを指すようにスクリプトを更新してください。

2

コマンド・リファレンス

- [EPM 自動化コマンドの実行について](#)
- [EPM 自動化の実行](#)
- [コマンドの概要](#)
- [EPM 自動化コマンド](#)
- [終了コード](#)

一部の EPM 自動化コマンドはすべてのビジネス・プロセスに適用されますが、一部はビジネス・プロセスのグループに適用されます。特に指定しないかぎり、特定のビジネス・プロセス (Planning など) に適用可能なコマンドは、別のビジネス・プロセス (Financial Consolidation and Close など) では機能しません。コマンドをサポートしていないビジネス・プロセスに対してコマンドを実行しようとすると、エラーが発生します。各ビジネス・プロセスに適用可能なコマンドのリストは、[各 Cloud EPM サービスに固有のコマンド](#)を参照してください。

EPM 自動化コマンドの実行について

すべての Oracle Fusion Cloud Enterprise Performance Management サービスは、環境のリモート管理に EPM 自動化コマンドを使用します。

- [前提条件](#)
- [デフォルトのファイルの場所](#)
- [Transport Layer Security プロトコル 1.2 の有効化](#)
- [EPM 自動化コマンドの使用](#)
- [パラメータに対する複数の値の指定](#)
- [日次メンテナンス中の動作](#)

前提条件

この項では、EPM 自動化を使用するための前提条件(環境での Oracle Fusion Cloud Enterprise Performance Management 資格証明の使用やデフォルトのファイルの場所など)を示します。

全般

すべての Cloud EPM ユーザーは、自分のアイデンティティ・ドメイン資格証明を使用し、EPM 自動化を使用して環境に接続できます。ユーザーに割り当てられている事前定義済役割とアプリケーション役割によって、ユーザーが実行できるコマンドが決まります。

- また、アイデンティティ・ドメイン内のユーザーを追加または削除するコマンドを実行するにはアイデンティティ・ドメイン管理者役割が必要です。
- コマンドを実行する必要があるファイルは環境内に存在する必要があります。[uploadFile](#) コマンドを使用して、ファイルをアップロードします。

各サービスで使用されるデフォルトのファイルの場所については、[デフォルトのファイルの場所](#)を参照してください。

- コマンドでのファイル拡張子の使用:
 - ファイル拡張子を含めたファイル名全体(たとえば `data.csv`)を指定して、ファイル操作を実行するコマンドを実行します。ファイル操作コマンドの例としては、`deletefile`、`listfiles`、`uploadfile` があります。
 - 移行操作を実行するコマンドを実行するときは、ファイル拡張子を使用しません。移行操作では、スナップショットの名前を指定する必要があります。
- スペース文字を含むパラメータ値(コメント、場所の名前、フォルダ・パスなど)は、引用符で囲む必要があります。

Planning

- ジョブ
次の項で説明するコマンドの多くではジョブが必要です。ジョブはデータのインポートまたはエクスポートなどの、すぐに開始するか、後の時刻にスケジュールできるアクションです。たとえばデータのインポートまたはエクスポートや、データベースのリフレッシュなどです。

ジョブ・コンソールを使用して、次の操作を実行するための適切なジョブを作成する必要があります。**Planning** でのジョブの作成方法の詳細は、*Planning の管理*のジョブの管理を参照してください。

- アプリケーションへのデータのインポート
 - アプリケーションからのデータのエクスポート
 - アプリケーションへのメタデータのインポート
 - アプリケーションからのメタデータのエクスポート
 - あるブロック・ストレージ・データベースから集約ストレージ・データベースへ、またはブロック・ストレージ・データベースから別のブロック・ストレージ・データベースへのデータのコピー
- ビジネス・ルール
実行するビジネス・ルールはアプリケーションに存在する必要があります。

Calculation Manager を使用してビジネス・ルールを作成すると、アプリケーションにデプロイされます。*Calculation Manager* での設計を参照してください。

データ管理

- データ・ルール
データ・ロード・ルールでは、データ管理によってファイルからデータをロードする方法を定義します。**EPM 自動化**を使用してデータをロードするには、データ・ロード・ルールを事前に定義しておく必要があります。
- バッチ
データ管理で定義されているバッチを使用してデータをロードできます。バッチを使用すると、ユーザーはバッチで多種多様なロード・ルールを組み合わせ、逐次モードまたは並列モードで実行できます。

デフォルトのファイルの場所

デフォルトのアップロード場所

デフォルトでは、Oracle Fusion Cloud Enterprise Performance Management にアップロードされたすべてのファイルは、移行からアクセス可能なデフォルトの場所に格納されます。

移行で処理されるファイル(たとえば、サービスにインポートするスナップショット)をデフォルトの場所にアップロードする必要があります。

受信ボックスと送信ボックス

受信ボックスと送信ボックスの場所は、Cloud EPM ビジネス・プロセス間で異なる場合があります。受信ボックスを使用して、インポートするか、または Profitability and Cost Management 以外のビジネス・プロセスを使用して処理するファイルをアップロードします。データ管理では、受信ボックスまたはその中のディレクトリ内のファイルを処理できます。

通常、Cloud EPM は、ビジネス・プロセスから生成したファイル(データまたはメタデータ・エクスポート・ファイル)を送信ボックスに格納します。

- EPM 自動化がファイルをアップロードする受信ボックス、およびダウンロードのためにファイルを保存する送信ボックスには、次に示すアプリケーションがアクセスできます。ファイルをこの場所にアップロードする必要があるのは、これらのアプリケーション固有のプロセスを使用してファイルを処理する予定がある場合です。ファイルを送信ボックスにアップロードすることもできます。
 - Planning
 - Planning モジュール
 - Account Reconciliation
 - Financial Consolidation and Close
 - Tax Reporting
 - Narrative Reporting
 - Enterprise Profitability and Cost Management

受信ボックスまたは送信ボックス・エクスプローラを使用して、デフォルトの場所に格納されたファイルを参照できます。EPM 自動化を使用して作成したアプリケーション・スナップショットは受信ボックス/送信ボックス・エクスプローラに表示されません。Migration の「スナップショット」タブから表示できます。

- Profitability and Cost Management プロセスを使用して処理される予定のファイルは、profitinbox にアップロードする必要があります。ファイルを profitoutbox にアップロードすることもできます。Profitability and Cost Management プロセスからエクスポートされるファイルは、profitinbox に格納されます。これらのファイルを参照するにはファイル・エクスプローラを使用します。
- データ管理を使用して処理される予定のファイルは、受信ボックスまたはその中のフォルダにあることが必要です。デフォルトでは、データ管理を使用してエクスポートされるファイルは送信ボックスに格納され、データ管理レポート出力はデータ管理の outbox/report フォルダに格納されます。これらのファイルを参照するにはデータ管理のファイル・ブラウザを使用します。
- Oracle Fusion Cloud Enterprise Data Management では、アップロード、コピーまたはダウンロードされたインポート・ファイルおよびエクスポート・ファイルにデフォルトの場所

を使用します。デフォルトの場所のファイルは、ListFiles コマンドを使用して表示できます。

ログ・ファイル

EPM 自動化コマンドの実行のたびにデバッグ・ファイルが生成され、コマンドが成功すると自動的に削除されます。コマンドの実行中にエラーが発生すると、EPM 自動化の実行元ディレクトリに失敗したコマンドのデバッグ・ファイルが保持されます。デフォルトで、これは Oracle/epm_automate/bin ディレクトリ (Windows) または home/user/epmautomate/bin (Linux/UNIX) です。

EPM 自動化のデバッグ・ファイルでは、次の命名規則が使用されます。

commandname_date_timestamp.log。たとえば、2020 年 11 月 23 日の 09:28:02 に listfiles コマンドの実行に失敗した場合、デバッグ・ファイル名は listfiles_23_11_2020_09_28_02.log になります。

失敗したコマンドのデバッグ・ファイルの作成を抑制することはできません。ただし、次の Windows の例に示すように、コマンドの末尾に -d とともに、デバッグ・ファイル名、エラーおよび出力ストリーム(-d >> c:\logs\LOG_FILE_NAME.log 2>&1)を追加することで、デバッグ情報とコマンド出力を別のディレクトリのファイルに書き込むことができます。

```
epmautomate listfiles -d >> c:\logs\listfiles.log 2>&1
```

Transport Layer Security プロトコル 1.2 の有効化

EPM 自動化は、Transport Layer Security (TLS) プロトコル 1.2 以上をサポートするオペレーティング・システムにインストールする必要があります。

認証およびデータ暗号化に対する最高レベルのセキュリティを確保するために、EPM 自動化では TLS 1.2 のみをサポートします。EPM 自動化が実行されるコンピュータで TLS 1.2 が有効化されていない場合は、EPMAT-7: Unable to connect.Unsupported Protocol: HTTPS というエラーが表示されます。このエラーを解決するには、IT 管理者とともに作業して TLS 1.2 を有効化してください。

TLS 1.2 を有効化する手順はオペレーティング・システムによって異なります。次の情報ソースを使用してください。サポートされているその他のオペレーティング・システムについて同様の Web リソースを使用できます。

- Windows コンピュータに対する TLS 1.2 の有効化の詳細は、[Windows の WinHTTP でデフォルト・セキュア・プロトコルとして TLS 1.1 および TLS 1.2 を有効にするための更新](#)を参照してください。
- Red Hat Enterprise Linux の OpenSSL での TLS 1.2 の有効化の詳細は、[TLS 構成のハードニング](#)を参照してください。

EPM 自動化コマンドの使用

コマンド・パラメータの順序

コマンドの必須パラメータはすべて、コマンド使用方法で指定された順序で渡す必要があります。必須パラメータとその値はオプション・パラメータ(任意の順序で渡すことができる)より前にあります。オプションのパラメータは位置指定ではありません。

たとえば、login コマンドの次の使用方法を考えてみます。

```
epmautomate login USERNAME PASSWORD CLOUD_EPM_BASE_URL
[ProxyServerUserName=PROXY_USERNAME] [ProxyServerPassword=PROXY_PASSWORD]
[ProxyServerDomain=PROXY_DOMAIN]
```

このコマンドには、USERNAME、PASSWORD および CLOUD_EPM_BASE_URL の 3 つの必須パラメータがあり、使用方法で指定された順序で出現する必要があります。この順序が保持されていない場合は、コマンドからエラーが返されます。オプション・パラメータ (ProxyServerUserName、ProxyServerPassword および ProxyServerDomain) とその値は任意の順序で指定できます。

EPM 自動化コマンドの大文字と小文字は区別されるか。

EPM 自動化コマンドの大文字と小文字は区別されません。コマンド名をどのように入力するかは、コマンドの実行に影響を与えません。たとえば、addUsers コマンドを addusers、ADDUSERS または AdDuSeRs と入力できます。

EPM 自動化コマンド・パラメータの大文字と小文字は区別されるか。

EPM 自動化コマンド・パラメータの大文字と小文字は区別されません。コマンド・パラメータ名をどのように入力するかは、コマンドの実行に影響を与えません。たとえば、FileName パラメータは、コマンドの実行に影響を与えることなく、filename、fileName または filEnAmE と入力できます。

パラメータに対する複数の値の指定

一部の EPM 自動化のコマンドは、カンマで区切られた複数のパラメータ値を受け入れます。たとえば、Planning アプリケーションのビジネス・ルール、ルールセット、およびテンプレート内のタイプ・メンバーのランタイム・プロンプトなどです。

EPM 自動化コマンド内の Entities という名前のランタイム・プロンプトのメンバー・タイプに複数のメンバーを設定するには、runbusinessrule コメントの実行に関する次の例に示される「,」(カンマ)を使用します。

```
epmautomate runbusinessrule clearDistData TargetYear=FY19
TargetMonth=Feb Entities=District1,District2
```

スペースやカンマなどの特殊文字を含むメンバー名は、次の例に示すように、引用符で囲み、\ (バックスラッシュ)を使用してエスケープする必要があります。

```
epmautomate runbusinessrule clearDistData TargetYear=FY19
TargetMonth=Feb Entities="\ "District 1\","entity_name, with comma\ ""
```

日次メンテナンス中の動作

環境の日次メンテナンスの進行中は、EPM 自動化コマンドを実行しないでください。

日次メンテナンス中はユーザー・アクティビティが許可されません。日次メンテナンスの進行中に直接またはスクリプトを使用して EPM 自動化コマンドを実行しようとすると、次のエラーが表示されます。

```
EPMAT-11:Internal server error.日次メンテナンスにより、Oracle EPM Cloud
Service 環境は現在使用できません。
```

EPM 自動化の実行

EPM 自動化を使用してサインインするには、Oracle Fusion Cloud Enterprise Performance Management 資格証明を使用します。SSO 資格証明を使用してサインインすることはできません。

すべての Cloud EPM ユーザーは、自分のアイデンティティ・ドメイン資格証明を使用し、EPM 自動化を使用して環境に接続できます。ユーザーに割り当てられている事前定義済役割とアプリケーション役割によって、ユーザーが実行できるコマンドが決まります。

また、コマンドの中には、サービス管理者の役割のみで実行できるものと、アイデンティティ・ドメイン管理者の役割も実行に必要なものがあります。

デバッグ・ログ・ファイルの生成

EPM 自動化の実行中に発生した問題をトラブルシューティングするために、Oracle サポートは、セッションのデバッグ・ログ・ファイルの提供をお願いしています。-d オプションにより、デバッグ・メッセージが生成されますが、これは>ディレクティブを使用してファイルにリダイレクトできます。デバッグ・ファイルは、1つのコマンド、バッチ実行ファイルまたは複数のコマンドを含むスクリプトに対して作成できます。

使用方法: `epmautomate command [command_parameters] -d > log_file 2>&1`

Windows の例: `epmautomate downloadfile "Artifact Snapshot" -d > C:\logs\download_log.txt 2>&1`

Linux の例: `epmautomate.sh downloadfile "Artifact Snapshot" -d > ./logs/download_log 2>&1`

Windows

EPM 自動化を実行する前に、実行する予定のコンピュータから環境にアクセスできることを確認します。

EPM 自動化では、ユーザー情報を含む .prefs ファイルとログ・ファイルが現在のディレクトリに作成されます。Windows コンピュータ上では、.prefs ファイルの内容は、そのファイルを作成したユーザーおよび Windows 管理者にのみ表示されます。Linux、UNIX および macOSX 環境では、.prefs ファイルは、所有者のみにこのファイルへの読取りおよび書込みを許可する権限 600 を使用して生成されます。

EPM 自動化の実行元である Windows ディレクトリの書込み権限がない場合は、Windows 環境に FileNotFoundException: .prefs (アクセスが拒否されました) エラーが表示されます。このエラーを解決するには、現在のユーザーの Windows アカウントに、EPM 自動化の実行元のディレクトリに対する読取り/書込みアクセスがあることを確認してください。さらに、このユーザーには、ファイルのアクセス元(たとえば、uploadFile コマンドの実行時)または書込み元(たとえば、downloadFile コマンドの実行時)のその他のディレクトリに対する適切なアクセス権も必要です。

ノート:

名前に&が含まれるフォルダ(C:\Oracle\A&B など)から EPM 自動化を実行することはできません。

EPM 自動化を Windows クライアントで実行するには:

1. 「スタート」、「すべてのプログラム」、「EPM 自動化」、「EPM 自動化の起動」の順にクリックします。EPM 自動化コマンドのプロンプトが表示されます。
2. **オプション:** 操作を実行するディレクトリにナビゲートします。
3. **オプション:** パスワード暗号化ファイルを生成します。パスワード暗号化ファイルを使用し、暗号化されたパスワードを渡してセッションを開始します。

```
epmautomate encrypt P@ssword1 myKey C:/mySecuredir/password.epw
```

4. サービス管理者としてセッションを開始します。次のようなコマンドを使用します。
 - 暗号化されていないパスワードの使用:

```
epmautomate login serviceAdmin P@ssword1  
https://test-cloudpln.pbcs_us1.oraclecloud.com
```

- 暗号化されたパスワードの使用:

```
epmautomate login serviceAdmin C:\mySecuredir\password.epw  
https://test-cloudpln.pbcs_us1.oraclecloud.com
```

5. 完了するタスクを実行するためのコマンドを入力します。
コマンドの実行ステータスの詳細は、[終了コード](#)を参照してください。
6. 環境からサインアウトします。次のコマンドを使用します。

```
epmautomate logout
```

Linux

ノート:

`JAVA_HOME` が `.profile` ファイルの `PATH` 変数で、またはシェルの環境変数として設定されていることを確認してください。サポートされている **JRE** (バージョン 8 から 11)が必要です。

EPM 自動化を Linux クライアントで実行するには:

1. 端末ウィンドウを開いて、EPM 自動化をインストールしたディレクトリにナビゲートします。
2. **オプション:** パスワード暗号化ファイルを生成します。パスワード暗号化ファイルを使用し、暗号化されていないパスワードではなく、暗号化されたパスワードを渡してセッションを開始します。

```
epmautomate encrypt P@ssword1 myKey ../misc/encrypt/password.epw
```

3. サービス管理者としてセッションを開始します。次のようなコマンドを使用します。

- 暗号化されていないパスワードの使用:

```
./bin/epmautomate.sh login serviceAdmin P@ssword1
https://test-cloudpln.pbcs_us1.oraclecloud.com
```

- 暗号化されたパスワードの使用:

```
./bin/epmautomate.sh login serviceAdmin ../misc/encrypt/password.epw
https://test-cloudpln.pbcs_us1.oraclecloud.com
```

4. 完了するタスクを実行するためのコマンドを入力します。

コマンドの実行ステータスの詳細は、[終了コード](#)を参照してください。

5. 環境からサインアウトします。次のコマンドを使用します。

```
./bin/epmautomate.sh logout
```

EPM 自動化の複数インスタンスの実行

同一のディレクトリから 1 つの環境に対して EPM 自動化の複数のインスタンスを実行できます。同様に、同じまたは異なるディレクトリから様々な環境に対して複数のインスタンスを実行できます。

たとえば、<https://cloudpln.pbcs.us1.oraclecloud.com> および <https://testcloudpln.pbcs.us1.oraclecloud.com> の **Planning** アプリケーション・キューブを同時にリフレッシュする必要がある場合があります。このシナリオには、2 つのオプションがあります。

- 同じディレクトリから EPM 自動化の 2 つのインスタンスを実行して、様々な環境内のアプリケーション・キューブをリフレッシュします
- 複数の異なるディレクトリから EPM 自動化を実行してこれらの環境に接続してから、アプリケーション・キューブをリフレッシュします

どちらのシナリオでも、各インスタンスは別々に作動します。1 つのインスタンスからログアウトしても、その他のインスタンスからはログアウトしません。その他のインスタンスからサインアウトした場合でも、EPM 自動化を使用して開始されたアクティビティは、その環境で完了するまで実行を継続します。

この項には、2 つの EPM 自動化セッションを作成してタスクを実行するために使用できる **Windows** および **Unix/Linux** のサンプル・スクリプト (`caller` および `multisession`) が記載されています。複数の同時セッションを実行するには、次の接続情報を `caller` スクリプト (`multisession` スクリプトをコールして `login`、`uploadfile`、`listfiles` および `logout` コマンドを実行する) に追加する必要があります。これらのコマンド以外のタスクを実行するために `multisession` スクリプトを変更できます。これら両方のスクリプトが同じディレクトリに格納されていることを確認してください。

- EPM 自動化では、環境変数 `EPM_SID` を使用して複数のセッションを区別します。この変数は、`caller` スクリプトでセッションごとに一意の値に設定する必要があります。サンプル・スクリプトでは、次のように一意の値が設定されています:
 - `caller.BAT` では、`EPM_SID` に `!RANDOM!` が設定され、これによって一意のシステム生成番号が割り当てられます。この番号は、各セッションのログ・ファイルの生成にも使用されます。各セッションのログ・ファイルを追跡する必要がある場合は、`!RANDOM!` のかわりに一意の番号を指定できます。

- caller.sh では、EPM_SID には一意のプロセス ID が設定されます。各セッションのログ・ファイルを追跡する必要がある場合に、一意の EPM_SID を指定するには、渡された値を使用するように multisession スクリプトの export EPM_SID=\$\$文を変更した後、各セッションの caller スクリプトでこのパラメータに一意の値を渡します。たとえば、caller.sh では EPM_SID の値を次のように指定します:

```
$SCRIPT_DIR/multisession.sh EPM_SID "USERNAME" "PASSWORD" "URL" "/home/  
user/Snapshot1.zip" &  
$SCRIPT_DIR/multisession.sh EPM_SID "USERNAME" "PASSWORD" "URL" "/home/  
user/Snapshot2.zip" &
```

- USERNAME: サービス管理者のログイン ID
- PASSWORD: サービス管理者のパスワード
- URL: 環境の接続 URL

サンプルの Windows スクリプト

caller.BAT

```
@echo off  
setlocal EnableExtensions EnableDelayedExpansion  
  
REM syntax: start /B multisession.bat EPM_SID "USERNAME" "PASSWORD" "URL"  
"SNAPSHOTPATH"  
start /B multisession.bat !RANDOM! "USERNAME" "PASSWORD" "URL"  
"C:\Snapshot1.zip"  
start /B multisession.bat !RANDOM! "USERNAME" "PASSWORD" "URL"  
"C:\Snapshot2.zip"  
  
endlocal
```

multisession.BAT

```
@echo off  
  
set EPM_SID=%1  
set USERNAME=%2  
set PASSWORD=%3  
set URL=%4  
set SNAPSHOTNAME=%5  
  
echo User: %USERNAME% > %EPM_SID%.log  
echo Cloud Instance: %URL% >> %EPM_SID%.log  
  
call epmautomate login %USERNAME% %PASSWORD% %URL% >> %EPM_SID%.log  
call epmautomate uploadfile %SNAPSHOTNAME% >> %EPM_SID%.log  
call epmautomate listfiles >> %EPM_SID%.log  
call epmautomate logout
```

サンプルの Bourne Shell スクリプト

caller.sh

```
#!/bin/sh

set +x
SCRIPT_DIR=`dirname "${0}"`

# syntax: /home/user/multisession.sh "USERNAME" "PASSWORD" "URL"
# "SNAPSHOTPATH" &
$SCRIPT_DIR/multisession.sh "USERNAME" "PASSWORD" "URL" "/home/user/
Snapshot1.zip" &
$SCRIPT_DIR/multisession.sh "USERNAME" "PASSWORD" "URL" "/home/user/
Snapshot2.zip" &
```

multisession.sh

```
#!/bin/sh

set +x

EPM_AUTOMATE_HOME=/home/user/epmautomate

export JAVA_HOME=/home/user/jre
export EPM_SID=$$

USERNAME=$1
PASSWORD=$2
URL=$3
SNAPSHOTNAME=$4

echo User: $USERNAME > $EPM_SID.log
echo Cloud Instance: $URL >> $EPM_SID.log

$EPM_AUTOMATE_HOME/bin/epmautomate.sh login $USERNAME $PASSWORD $URL
>> $EPM_SID.log
$EPM_AUTOMATE_HOME/bin/epmautomate.sh uploadfile $SNAPSHOTNAME >> $EPM_SID.log
$EPM_AUTOMATE_HOME/bin/epmautomate.sh listfiles >> $EPM_SID.log
$EPM_AUTOMATE_HOME/bin/epmautomate.sh logout
```

コマンドの概要

すべての EPM 自動化コマンドをアルファベット順にリストします。

表 2-1 すべての EPM Automate コマンド

コマンド名	PLN、 SWP、 SP、FF	FCC	TR	PCM	EPCM	AR	EDM	NR
addUsers	✓	✓	✓	✓	✓	✓	✓	✓
addUsersToGroup	✓	✓	✓	✓	✓	✓	✓	✓

表 2-1 (続き) すべての EPM Automate コマンド

コマンド名	PLN、 SWP、 SP、FF	FCC	TR	PCM	EPCM	AR	EDM	NR
addUsersToTeam		✓	✓			✓		
addUserToGroups	✓	✓	✓	✓	✓	✓	✓	✓
applicationAdminMode	✓	✓	✓		✓			
applyDataGrants				✓				
archiveTmTransactions						✓		
assignRole	✓	✓	✓	✓	✓	✓	✓	✓
autoPredict* 脚注を参照	✓							
calculateModel					✓			
clearCube	✓				✓			
clearDataByPointOfView					✓			
clearDataByProfile		✓	✓					
clearPOV				✓				
cloneEnvironment	✓	✓	✓	✓	✓	✓	✓	✓
compactCube	✓				✓			
copyDataByPointOfView					✓			
copyDataByProfile		✓	✓					
copyFileFromInstance	✓	✓	✓	✓	✓	✓	✓	✓
copyFromObjectStorage	✓	✓	✓	✓	✓	✓	✓	✓
copyFromSFTP	✓	✓	✓	✓	✓	✓	✓	✓
copyOwnershipDataToNextYear		✓	✓					
copyPOV				✓				
copySnapshotFromInstance	✓	✓	✓	✓	✓	✓	✓	
copyToObjectStorage	✓	✓	✓	✓	✓	✓	✓	✓
copyToSFTP	✓	✓	✓	✓	✓	✓	✓	✓
createGroups	✓	✓	✓	✓	✓	✓	✓	✓
createNRSnapshot								✓
createReconciliations						✓		
deleteFile	✓	✓	✓	✓	✓	✓	✓	✓
deleteGroups	✓	✓	✓	✓	✓	✓	✓	✓
deletePointOfView					✓			
deletePOV				✓				
deployCube				✓				
deployEJTemplates		✓						
deployFormTemplates		✓	✓					
deployTaskManagerTemplate		✓			✓			
dismissIPMInsights**	✓							
downloadFile	✓	✓	✓	✓	✓	✓	✓	✓
enableApp				✓				
enableQueryTracking	✓				✓			
encrypt	✓	✓	✓	✓	✓	✓	✓	✓

表 2-1 (続き) すべての EPM Automate コマンド

コマンド名	PLN、 SWP、 SP、FF	FCC	TR	PCM	EPCM	AR	EDM	NR
essbaseBlockAnalysisReport	✓	✓	✓					
executeAggregationProcess	✓				✓			
executeBurstDefinition								✓
executeReportBurstingDefinition	✓	✓	✓		✓			
exportAccessControl						✓		
exportAppAudit	✓	✓	✓		✓			
exportAppSecurity	✓	✓	✓		✓			
exportARApplicationProperties						✓		
exportBackgroundImage						✓		
exportCellLevelSecurity	✓		✓		✓			
exportData	✓	✓	✓		✓			
exportConsolidationJournals		✓						
exportDataManagement	✓	✓	✓	✓	✓			
exportDimension							✓	
exportDimensionMapping							✓	
exportEJournals		✓						
exportEssbaseData	✓	✓	✓		✓			
exportJobConsole	✓	✓	✓		✓			
exportLibraryArtifact								✓
exportLibraryDocument	✓	✓	✓		✓			
exportLogImage						✓		
exportMapping	✓	✓	✓	✓	✓	✓		
exportMetadata	✓	✓	✓		✓			
exportOwnershipData		✓	✓					
exportQueryResults				✓				
exportSnapshot	✓	✓	✓	✓	✓	✓	✓	
exportTemplate				✓				
exportTaskManagerAccessControl		✓	✓		✓			
exportValidIntersections	✓	✓	✓		✓			
extractDimension							✓	
extractPackage							✓	
feedback	✓	✓	✓	✓	✓	✓	✓	✓
getApplicationAdminMode	✓	✓	✓		✓	✓		
getDailyMaintenanceStartTime	✓	✓	✓	✓	✓	✓	✓	✓
getEssbaseQryGovExecTime	✓	✓	✓	✓	✓			
getIdleSessionTimeout	✓	✓	✓	✓	✓	✓	✓	✓
getIPAllowlist	✓	✓	✓	✓	✓	✓	✓	✓
getRestrictedDataAccess	✓	✓	✓	✓	✓	✓	✓	✓
getSubstVar	✓	✓	✓		✓			
getVirusScanOnFileUploads	✓	✓	✓	✓	✓	✓	✓	✓

表 2-1 (続き) すべての EPM Automate コマンド

コマンド名	PLN、 SWP、 SP、FF	FCC	TR	PCM	EPCM	AR	EDM	NR
groupAssignmentAuditReport	✓	✓	✓	✓	✓	✓	✓	✓
help	✓	✓	✓	✓	✓	✓	✓	✓
importAppAudit	✓				✓			
importAppSecurity	✓	✓	✓		✓			
importARApplicationProperties						✓		
importBackgroundImage						✓		
importBalances						✓		
importCellLevelSecurity	✓		✓		✓			
importConsolidationJournals		✓						
importData	✓	✓	✓	✓	✓			
importDataManagement	✓	✓	✓	✓	✓			
importDimension							✓	
importJobConsole	✓	✓	✓		✓			
importLibraryArtifact								✓
importLogImage						✓		
importMapping	✓	✓	✓	✓	✓	✓		
importMetadata	✓	✓	✓		✓			
importOwnershipData		✓	✓					
importPreMappedBalances						✓		
importPreMappedTransactions						✓		
importProfiles						✓		
importRates						✓		
importRCAAttributeValues						✓		
importReconciliationAttributes						✓		
importSnapshot	✓	✓	✓	✓	✓	✓	✓	
importSupplementalCollectionData		✓	✓					
importSupplementalData		✓	✓					
importTemplate				✓				
importTMAAttributeValues						✓		
importValidIntersections	✓	✓	✓		✓			
invalidLoginReport	✓	✓	✓	✓	✓	✓	✓	✓
listBackups	✓	✓	✓	✓	✓	✓	✓	✓
listFiles	✓	✓	✓	✓	✓	✓	✓	✓
loadData				✓				
loadDimData				✓				
loadDimensionViewpoint							✓	
loadViewpoint							✓	
login	✓	✓	✓	✓	✓	✓	✓	✓
logout	✓	✓	✓	✓	✓	✓	✓	✓
maskData	✓	✓	✓		✓			

表 2-1 (続き) すべての EPM Automate コマンド

コマンド名	PLN、 SWP、 SP、FF	FCC	TR	PCM	EPCM	AR	EDM	NR
mergeDataSlices	✓				✓			
mergeSlices				✓				
optimizeASOCube				✓				
programDocumentationReport				✓				
provisionReport	✓	✓	✓	✓	✓	✓	✓	✓
purgeArchivedTmTransactions						✓		
purgeTmTransactions						✓		
recomputeOwnershipData		✓	✓					
recreate	✓	✓	✓	✓	✓	✓	✓	✓
refreshCube	✓	✓	✓		✓			
removeUserFromGroups	✓	✓	✓	✓	✓	✓	✓	✓
removeUsers	✓	✓	✓	✓	✓	✓	✓	✓
removeUsersFromGroup	✓	✓	✓	✓	✓	✓	✓	✓
removeUsersFromTeam		✓	✓			✓		
renameSnapshot	✓	✓	✓	✓	✓	✓	✓	
replay	✓	✓	✓	✓	✓	✓	✓	✓
resetService	✓	✓	✓	✓	✓	✓	✓	✓
restoreBackup	✓	✓	✓	✓	✓	✓	✓	✓
restructureCube	✓	✓	✓					
roleAssignmentAuditReport	✓	✓	✓	✓	✓	✓	✓	✓
roleAssignmentReport	✓	✓	✓	✓	✓	✓	✓	✓
runAutomatch						✓		
runBatch	✓	✓	✓	✓	✓	✓		
runBusinessRule	✓	✓	✓		✓			
runCalc				✓				
runComplianceReport						✓		
runDailyMaintenance	✓	✓	✓	✓	✓	✓	✓	✓
runDataRule	✓	✓	✓	✓	✓	✓		
runDMReport	✓	✓	✓	✓	✓	✓		
runIntegration	✓	✓	✓	✓	✓	✓		
runIntercompanyMatchingReport		✓						
runMatchingReport						✓		
runPipeline	✓	✓	✓		✓			
runPlanTypeMap	✓							
runRuleSet	✓	✓	✓		✓			
runSupplementalDataReport		✓	✓					
runTaskManagerReport		✓	✓		✓			
sendMail	✓	✓	✓	✓	✓	✓	✓	✓
setApplicationAdminMode	✓	✓	✓		✓	✓		
setDailyMaintenanceStartTime	✓	✓	✓	✓	✓	✓	✓	✓

表 2-1 (続き) すべての EPM Automate コマンド

コマンド名	PLN、 SWP、 SP、FF	FCC	TR	PCM	EPCM	AR	EDM	NR
setDemoDates		✓	✓		✓	✓		
setEJJournalStatus		✓						
setEncryptionKey	✓	✓	✓	✓	✓	✓	✓	✓
setEssbaseQryGovExecTime	✓	✓	✓	✓	✓			
setIdleSessionTimeout	✓	✓	✓	✓	✓	✓	✓	✓
setIPAllowlist	✓	✓	✓	✓	✓	✓	✓	✓
setManualDataAccess	✓	✓	✓	✓	✓	✓	✓	✓
setPeriodStatus						✓		
setRestrictedDataAccess	✓	✓	✓	✓	✓	✓	✓	✓
setSubstVars	✓	✓	✓		✓			
setVirusScanOnFileUploads	✓	✓	✓	✓	✓	✓	✓	✓
simulateConcurrentUsage	✓	✓	✓					
skipUpdate	✓	✓	✓	✓	✓	✓	✓	✓
snapshotCompareReport	✓	✓	✓		✓			
sortMember	✓				✓			
unassignRole	✓	✓	✓	✓	✓	✓	✓	✓
updateGuidedLearningSettings	✓	✓	✓		✓			
updateUsers	✓	✓	✓	✓	✓	✓	✓	✓
upgrade	✓	✓	✓	✓	✓	✓	✓	✓
uploadFile	✓	✓	✓	✓	✓	✓	✓	✓
userAuditReport	✓	✓	✓	✓	✓	✓	✓	✓
userGroupReport	✓	✓	✓	✓	✓	✓	✓	✓
validateConsolidationMetadata		✓						
validateModel					✓			

- * このコマンドは、ハイブリッド Oracle Essbase キューブがアプリケーションで有効になっている場合にのみサポートされます。Strategic Workforce Planning および Sales Planning は、ハイブリッド Essbase をサポートしていません。このコマンドは、フリーフォームではサポートされていません。
- **このコマンドは、フリーフォームではサポートされていません。

省略形

- PLN: Planning (Planning モジュールを含む)
- FF: フリーフォーム
- SWP: Strategic Workforce Planning
- SP: Sales Planning
- FCC: Financial Consolidation and Close
- TR: Tax Reporting
- PCM: Profitability and Cost Management

- EPCM: Enterprise Profitability and Cost Management
- AR: Account Reconciliation
- EDM: Oracle Fusion Cloud Enterprise Data Management
- NR: Narrative Reporting

EPM 自動化コマンド

この項では、各 EPM 自動化コマンドについて詳述します。各コマンドの情報には、そのコマンドを使用できるサービス、コマンドの使用法および例が含まれます。

addUsers

環境にアップロードされた ANSI または UTF-8 エンコーディングのカンマ区切り値(CSV)ファイルを使用して、アイデンティティ・ドメインに一連のユーザーを作成します。また、新しいユーザーにユーザー名と一時パスワードを通知します。

サービス管理者は、[uploadFile](#) コマンドを使用して、ファイルを環境にアップロードします。CSV ファイルのすべての列は必須です。このコマンドは、定義の各列の値を検証し、欠落している値または無効な値を特定するエラー・メッセージを表示します。CSV ファイルの形式は次のとおりです。

```
First Name,Last Name,Email,User Login  
Jane,Doe,jane.doe@example.com,jdoe  
John,Doe,john.doe@example.com,john.doe@example.com
```

CSV ファイル形式の詳細は、『*Oracle Cloud スタート・ガイド*』のユーザー・アカウントのバッチのインポートに関する項を参照してください。

インポート・ファイルで指定されるユーザー・ログインの値では大文字と小文字が区別されません。たとえば、値 John.doe@example.com は、John.Doe@example.com など、大文字と小文字のすべてのバリエーションと同じであるものとして処理されます。

CSV ファイル内のユーザー定義がアイデンティティ・ドメインに存在するユーザー・アカウントと一致する場合、既存のユーザー・アカウントは変更されません。このコマンドは、ファイルにアカウント情報が含まれている新規ユーザーのアカウントのみを作成します。ユーザー・アカウントはアイデンティティ・ドメインがサポートするすべての環境に共通するため、新規ユーザーはアイデンティティ・ドメインを共有するすべての環境で使用できます。

終了すると、失敗した各エントリに関する情報がコンソールに出力されます。この情報を確認して、CSV ファイルの一部のエントリでコマンドの実行が失敗した理由を理解してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Strategic Workforce Planning および Sales Planning。

必要な役割

アイデンティティ・ドメイン管理者と任意の事前定義済役割

使用方法

epmautomate addUsers *FILE_NAME* [userPassword=*PASSWORD*] [resetPassword=true|false]。ここで:

- *FILE_NAME* は、ユーザー情報を含む **CSV** ファイルの名前ですマルチバイト文字を含む入力ファイルでは、**UTF-8** 文字エンコードを使用する必要があります。
- userPassword は、オプションで、アイデンティティ・ドメインに作成されるすべての新規ユーザーに対するデフォルト・パスワードを示します。指定した場合、このパスワードは、アイデンティティ・ドメインの最小のパスワード要件を満たす必要があります。パラメータが指定されていない場合、一意の一時パスワードが各ユーザーに割り当てられます。指定した場合、userPassword パラメータの値は、**CSV** ファイルで指定されたすべてのユーザーのパスワードとして使用されます。すべてのユーザーへの同じパスワードの割当てが望ましいのは、単にテスト目的でユーザーを作成している場合です。実際の **Oracle Fusion Cloud Enterprise Performance Management** ユーザーを作成していて、各ユーザーに特定のパスワードを割り当てる場合は、userPassword オプション・パラメータの値を指定せずに、このコマンドを使用します。
- resetPassword はオプションで、新規ユーザーが最初のログイン時にパスワードを変更する必要があるかどうかを指定します。デフォルトは true です。このパラメータが false に設定されている場合以外は、新規ユーザーは最初のサインイン時にパスワードの変更が強制されます。
resetPassword が true に設定されている場合、このコマンドは、新しい各ユーザーに、アカウントの詳細(ユーザー名とパスワード)を記載した電子メールを送信します。
resetPassword が false に設定されている場合、電子メールは送信されません。
resetPassword を **false** に設定する場合は、userPassword を指定する必要があります。そうしないと、各ユーザーに一意の一時パスワードが割り当てられますが、電子メールが送信されないため、パスワードがユーザーに認識されず、ユーザーはログインできません。

例


- 同じパスワードを持つテスト・ユーザーをアイデンティティ・ドメインに追加し、パスワードの変更を要求しない:
epmautomate addUsers user_file.CSV userPassword=Example@Pwd12
resetPassword=false
- 一時パスワードを使用してアイデンティティ・ドメインにユーザーを追加し、パスワードの変更を要求します。
epmautomate addUsers user_file.CSV

addUsersToGroup

環境にアップロードされた **ANSI** または **UTF-8** エンコーディングの **CSV** ファイルを使用して、アクセス制御の既存のグループにユーザーのバッチを追加します。

サービス管理者は、**uploadFile** コマンドを使用して、ファイルを環境にアップロードします。ユーザー・ログイン値では大文字と小文字が区別されません。ファイル形式は次のとおりです。

```
User Login
jdoe
john.doe@example.com
```

 ノート:

ユーザーは、次の両方の条件が満たされている場合にのみグループに追加されます。

- ファイルに含まれるユーザー・ログイン値が、環境にサービスを提供するアイデンティティ・ドメインに存在する。ユーザー・ログイン値では大文字と小文字が区別されません。
- ユーザーが、アイデンティティ・ドメイン内の事前定義済役割に割り当てられている

完了すると、失敗した各エントリに関する情報がコンソールに出力されます。この情報を確認して、CSV ファイルの一部のエントリでコマンドの実行が失敗した理由を理解してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Enterprise Profitability and Cost Management、Narrative Reporting、Strategic Workforce Planning および Sales Planning。

必要な役割

- サービス管理者
- 任意の事前定義済役割およびアクセス制御 - 管理アプリケーション役割

使用方法

`epmautomate addUsersToGroup FILE_NAME GROUP_NAME`。ここで:

- `FILE_NAME` は、アクセス制御のグループに割り当てるユーザーのログイン名を含む CSV ファイルの名前です
- `GROUP_NAME` は、アクセス制御に存在するグループの名前ですこの値では大文字と小文字が区別されません。

例


```
epmautomate addUsersToGroup user_file.CSV example_group
```

addUsersToTeam

CSV ファイルにリストされている Oracle Fusion Cloud Enterprise Performance Management ユーザーを既存のチームに追加します。

CSV ファイルに含まれているユーザーがすでにチームのメンバーである場合、このコマンドはそのユーザーを無視します。このファイル内の値では大文字と小文字が区別されません。サービス管理者は、[uploadFile](#) コマンドを使用してこのファイルを環境にロードします。CSV ファイルの形式は次のとおりです。

```
User Login, primary_user  
jdoe, yes  
jane.doe@example.com, no
```

 ノート:

プライマリ・ユーザーはチームに割り当てられたタスクを実行するようにデフォルトで指定されます。

適用対象

Financial Consolidation and Close、Tax Reporting および Account Reconciliation。

必要な役割

- サービス管理者
- 任意の事前定義済役割およびチーム - 管理アプリケーション役割
- 任意の事前定義済役割およびユーザー - 管理アプリケーション役割

使用方法

`epmautomate addUsersToTeam FILE TEAM_NAME`。ここで:

- `FILE` は、チームに追加するユーザーのログイン ID がリストされた UTF-8 形式の CSV ファイルを識別します。このコマンドを実行する前に、[uploadFile](#) コマンドを使用して、ファイルを環境にアップロードします。
- `TEAM_NAME` は、アクセス制御で定義されたチーム名を識別します。この値では大文字と小文字が区別されません。

例

```
epmautomate addUsersToTeam example_users.csv example_team
```

addUserToGroups

ユーザーを ANSI または UTF-8 エンコーディングの CSV ファイルで識別されたアクセス制御グループのメンバーとして追加します。

サービス管理者は、[uploadFile](#) コマンドを使用して、ファイルを環境にアップロードします。ファイル形式は次のとおりです。

```
Group Name  
Group1  
Group2
```

グループ名値では大文字と小文字が区別されません。

終了すると、失敗した各エントリに関する情報がコンソールに出力されます。この情報を確認して、CSV ファイルの一部のエントリでコマンドの実行が失敗した理由を理解してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Strategic Workforce Planning および Sales Planning。

必要な役割

- サービス管理者
- 任意の事前定義済役割およびアクセス制御 - 管理アプリケーション役割

使用方法

`epmautomate addUserToGroups FILE_NAME User_Login`。ここで:

- `FILE_NAME` は、ユーザーに割り当てるアクセス制御グループ名を含む CSV ファイルの名前です
- `User_Login` は、アクセス制御グループに割り当てる Oracle Fusion Cloud Enterprise Performance Management ユーザーのログイン ID です。ユーザーのログイン ID (大文字と小文字は区別されません)は、環境にサービスを提供するアイデンティティ・ドメインに存在し、かつ事前定義済役割に割り当てられている必要があります。

例

```
epmautomate addUserToGroups groups.CSV jdoe@example.com
```

applicationAdminMode

アプリケーションを管理モードにし、アプリケーションへのアクセス権がサービス管理者のみに限定されるようにします。

このコマンドは、サービス管理者が管理操作を実行しているときにユーザーがアプリケーションを操作できないようにするときに役立ちます。アプリケーションは、モードを戻してすべてのユーザーがアクセスできるようにするまでは管理モードのままです。

ノート:

このコマンドは非推奨になりましたが、EPM 自動化から削除されていません。かわりに、[setApplicationAdminMode](#) コマンドを使用することをお勧めします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

`epmautomate applicationAdminMode VALUE`。ここで、`VALUE` は、アプリケーションを管理モードにするかどうかを指定します。使用可能な値は次のとおりです。

- `true`: アプリケーションを管理モードにします
- `false`: アプリケーションを通常モードに戻して、すべてのユーザーがアクセスできるようにします

例

- アプリケーションを管理モードにする場合: `epmautomate applicationAdminMode true`
- アプリケーションを通常モードに戻す場合: `epmautomate applicationAdminMode false`

applyDataGrants

Oracle Essbase データ・スライスへのアクセス権を制御するデータ権限をリフレッシュして、Profitability and Cost Management アプリケーションに定義されたデータ権限と一致するようにします。

Profitability and Cost Management アプリケーションで作成したユーザーおよびグループ・レベルのデータ権限は、Essbase で自動的に同期化されます。アプリケーション内のデータ権限と Essbase 内のフィルタの不一致があると思われる場合は、このコマンドを使用して Essbase データへのアクセス権を同期します。

この操作を完了するまでにかかる時間は、アプリケーションのサイズによって変わります。次のメンテナンス・ウィンドウでアプリケーションがバックアップされる前に、データ権限のリフレッシュ操作が完了するようにしてください。この操作が行われている間はアプリケーションを使用できないため、ユーザーがアプリケーションを使用しない時間帯にこの操作をスケジュール設定するようにお勧めします。

適用対象

Profitability and Cost Management

必要な役割

「サービス管理者」、「パワー・ユーザー」

使用方法

`epmautomate applyDataGrants APPLICATION_NAME`。ここで、`APPLICATION_NAME` は、データ権限を再作成する Profitability and Cost Management アプリケーションの名前です。

例

```
epmautomate applyDataGrants BksML12
```

archiveTmTransactions

サポートおよび調整の詳細を含めて、指定の経過期間以上経過した照合済トランザクションをアーカイブします。照合済トランザクションは ZIP ファイルに記録されます。

このコマンドを使用して、組織のトランザクション保持ポリシーに基づいて古い照合済トランザクションをアーカイブしてパージすることにより、Account Reconciliation アプリケーションのサイズを最適に維持します。

適用対象

Account Reconciliation

必要な役割

「サービス管理者」、「パワー・ユーザー」、「ユーザー」、「参照者」

「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

epmautomate archiveTmTransactions *matchType* *age* [*filterOperator*=*VALUE*]
[*filterValue*=*VALUE*] [*logFilename*=*FILE_NAME*] [*filename*=*FILE_NAME*]. ここで:

- *matchType* は、照合済トランザクションをアーカイブする照合タイプの識別子(TextID)です。
- *age* は、トランザクションを照合してからの日数を識別します。この値以上経過した照合済トランザクションがアーカイブされます。
- *filterOperator* はオプションで、アーカイブ対象の照合済トランザクションが格納されている勘定科目を識別するための次のいずれかのフィルタ条件です。この値を *filterValue* と組み合わせて、照合済トランザクションをアーカイブする勘定科目を識別します:
 - equals
 - not_equals
 - starts_with
 - ends_with
 - contains
 - not_contains
- *filterValue* はオプションで、アーカイブするトランザクションを識別するためのフィルタ値です。*filterOperator* が equals または not_equals の場合は、スペース区切りのリストを使用して複数の値を指定できます(例: *filterValue*=101-120 *filterValue*=102-202)。複数の値が指定されている場合、フィルタ演算子とフィルタ値の組合せに一致する勘定科目のトランザクションがアーカイブ対象として選択されます。

Note:

filterOperator および *filterValue* が指定されていない場合は、指定された *matchType* のすべての勘定科目から *age* 以上経過した照合済トランザクションがすべてアーカイブされます。

- *logFilename* はオプションで、コマンド・アクティビティに関する情報を記録するログ・ファイルの名前です。ファイル名が指定されていない場合は、`Archive_Transactions_matchType_JOBID.log` というログ・ファイルが自動的に生成されます。
- *filename* はオプションで、アーカイブされたトランザクションを格納する.ZIP ファイルの名前です。指定されていない場合は、コマンドにより、デフォルトで `Archived_Transactions_matchType_JOBID.zip` が作成されます。[downloadFile](#) コマンドを使用して、このファイルをローカル・コンピュータにダウンロードします。

 **Note:**

このコマンドは、指定するパラメータを使用して、TM トランザクションのアーカイブ・ジョブを実行します。 [purgeArchivedTmTransactions](#) コマンドとの使用を容易にするために、ジョブ ID がコマンド出力に返されます。ジョブはジョブ・コンソールでモニターできます。

例

- フィルタは使用しませんが、カスタム・ログと.ZIP ファイル名を使用して、古い照合済トランザクションをアーカイブします:

```
epmautomate archiveTmTransactions cashrecon 180 logFile=tmlogs.log  
filename=trans.zip
```
- フィルタを使用して古い照合済トランザクションをアーカイブします:
 - ```
epmautomate archiveTmTransactions cashrecon 180 filterOperator>equals
filterValue=101-120 FilterValue=102-202
```
  - ```
epmautomate archiveTmTransactions cashrecon 180 filterOperator=contains  
filterValue=11
```


assignRole

ログイン ID が ANSI または UTF-8 でエンコードされた CSV ファイルに含まれているユーザー(このコマンドを実行するユーザーを含む)に、役割を割り当てます。このコマンドを使用して、事前定義済役割またはアプリケーション役割にユーザーを割り当てます。

このコマンドを使用する前に、サービス管理者は、[uploadFile](#) コマンドを使用して、入力ファイルを環境にアップロードします。ファイル形式は次のとおりです。

```
User Login  
jane.doe@example.com  
jdoe
```

『Oracle Cloud スタート・ガイド』の1つの役割の多数のユーザーへの割当てに関する項を参照してください。

 **ノート:**

- ファイルに含まれるユーザー・ログイン値では大文字と小文字が区別されません。
- 空白文字を含む役割名は二重引用符で囲みます。

完了すると、失敗した各エントリに関する情報がコンソールに出力されます。この情報を確認して、CSV ファイルの一部のエントリでコマンドの実行が失敗した理由を理解してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability

and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Strategic Workforce Planning および Sales Planning。

必要な役割

事前定義済役割を割り当てる場合:

- 事前定義済役割を割り当てる場合:
 - サービス管理者
 - アイデンティティ・ドメイン管理者と任意の事前定義済役割
- アプリケーション役割を割り当てる場合:
 - サービス管理者
 - 任意の事前定義済役割およびアクセス制御 - 管理アプリケーション役割

使用方法

epmautomate assignRole *FILE_NAME* *ROLE*。ここで:

- *FILE_NAME* は、ユーザー・ログイン ID を含む CSV ファイルの名前です CSV 拡張子は小文字で指定してください。
- *ROLE* は、次のいずれかです。この値では大文字と小文字が区別されません:
 - ユーザーを事前定義済のアイデンティティ・ドメイン役割に割り当てる場合、*ROLE* ではサービスに適用可能な事前定義済役割を識別する必要があります。 *管理者用スタート・ガイド*の事前定義済役割の理解を参照してください。
これらの役割の説明は、 *アクセス制御の管理*のアプリケーション・レベルでの役割割当ての管理を参照してください
 - ユーザーをアプリケーション役割に割り当てる場合、*ROLE* では、現在の環境のアプリケーションに属する役割を指定する必要があります。アプリケーション役割は、「アクセス制御」の「**役割**」タブにリストされます。各ビジネス・プロセスのアプリケーション役割の説明については、 *Oracle Enterprise Performance Management Cloud アクセス制御の管理*の次のトピックを参照してください:
 - * Account Reconciliation
 - * Enterprise Profitability and Cost Management
 - * Planning、フリーフォーム、Financial Consolidation and Close および Tax Reporting
 - * Profitability and Cost Management
 - * Oracle Enterprise Data Management
 - * Narrative Reporting

例

- ユーザーの事前定義済のアイデンティティ・ドメイン役割への割当て:
epmautomate assignRole admin_role_file.csv "Service Administrator"
- ユーザーのアプリケーション役割への割当て:
epmautomate assignRole example_file.csv "Task List Access Manager"

autoPredict

Planning または Planning モジュールの既存の自動プレディクト定義に基づいて、将来のパフォーマンスのプレディクションを生成します。

このコマンドは、アプリケーションで指定された自動プレディクト定義で定義された各メンバーの履歴データを使用するジョブを開始します。自動プレディクト機能を使用するアプリケーションおよびプレディクション設定の詳細は、*Planning の管理*の自動プレディクトを使用して自動的に実行するプレディクションの設定を参照してください。

適用対象

Planning、Planning モジュール(ハイブリッド Oracle Essbase キューブがアプリケーションで有効になっている場合)。

必要な役割

サービス管理者

使用方法

```
epmautomate autoPredict PREDICTION_DEFINITION [forceRun=true|false]
[paginatedDim=DIMENSION_NAME]。ここで:
```

- `PREDICTION_DEFINITION` は、アプリケーションで使用可能な自動プレディクション定義の名前です。
- `forceRun` はオプションで、最初の実行後に基礎となる定義が変更されていない場合に、プレディクションを実行するかどうかを指定します。デフォルトは `false` です。ジョブ定義に変更がない場合でも自動プレディクト・ジョブを実行するには、このパラメータの値を `true` に設定します。プレディクションを一度(最初の実行時のみに)実行するには、デフォルト(`false`)を使用します。
- `paginatedDim` はオプションで、別々のスレッドでプレディクションを並列で実行することにより、自動プレディクト・ジョブを高速化するために使用されるディメンションを指定します。これらの並列スレッドを効率的にするために、データを各プレディクション・スレッドに均一に分散するディメンションを指定します。

例

```
epmautomate autoPredict ASOtoBSO forceRun=true paginatedDim=Entity
```

calculateModel

Enterprise Profitability and Cost Management アプリケーションで計算プロセスを実行します。

適用対象

Enterprise Profitability and Cost Management

必要な役割

サービス管理者

使用方法

```
epmautomate calculateModel POV_NAME MODEL_NAME EXECUTION_TYPE
[povDelimiter=DELIMITER] [optimizeForReporting=true|false]
[captureDebugScripts=true|false] [comment=COMMENT] [PARAMETER=VALUE]。ここで:
```

- *POV_NAME* は、計算するデータ **POV** の名前です。複数の **POV** を計算するには、区切り文字としてカンマで区切って **POV** 名をリストします。他の区切り文字を使用して **POV** 名を区切らないでください。メンバー名に空白が含まれる場合は、**POV** 名のリストを二重引用符で囲みます。
- *MODEL_NAME* は、計算するモデルの名前です。モデル名に空白が含まれる場合は、モデル名を二重引用符で囲みます。
- *EXECUTION_TYPE* は、ルールの実行タイプを識別する次のいずれかです。
 - *ALL_RULES*: すべてのルールを使用して **POV** を計算します。
この値を指定する場合は、*rulesetSeqNumStart*、*rulesetSeqNumEnd*、*ruleName*などのルールのサブセットまたは1つのルールに関連するランタイム・パラメータを含めないでください。
 - *RULESET_SUBSET*: ルールセットのサブセットを使用して **POV** を計算します。
この値を使用する場合は、ランタイム・パラメータとして *rulesetSeqNumStart* および *rulesetSeqNumEnd* の値を指定する必要があります。
 - *SINGLE_RULE*: 特定のルールを実行して **POV** を計算します。
この値を使用する場合は、ランタイム・パラメータとして *ruleName* のみを指定する必要があります。
 - *RUN_FROM_RULE*: 特定のルールから開始して **POV** に対する計算を実行します。
この値を使用する場合は、ランタイム・パラメータとして *ruleName* のみを指定する必要があります。
 - *STOP_AFTER_RULE*: 特定のルールで計算が終了したら **POV** の計算を停止します。
この値を使用する場合は、ランタイム・パラメータとして *ruleName* のみを指定する必要があります。
- *povDelimiter* はオプションで、**POV** 値で使用される区切り文字です。デフォルトの区切り文字は:: (二重コロン)です。区切り文字は二重引用符で囲む必要があります。次の区切り文字のみがサポートされています:
 - `_` (アンダースコア)
 - `#` (ハッシュ)
 - `&` (アンパサンド)
 - `~` (チルダ)
 - `%` (パーセンテージ)
 - `;` (セミコロン)
 - `:` (コロン)
 - `-` (ダッシュ)
- *optimizeForReporting=true|false* はオプションで、計算をレポート用に最適化して実行するか、または最適化しないで実行するかを指定します。デフォルトは *false* です。1つのルールまたは連続した **POV** を実行するときなど、集計作成ステップをスキップして処理時間を節約するには、この値を *false* に設定します。複数の同時実行計算ジョブを実

行するときは、すべてのジョブで `optimizeForReporting=true` を設定します。最後に実行するジョブのみで集計が行われるため、冗長な処理は回避され、ジョブの実行速度が低下することを防ぎます。

- `captureDebugScripts=true|false` はオプションで、受信ボックスにデバッグ・スクリプトを生成するかどうかを指定します。計算の問題をトラブルシューティングするためにこれらのスクリプトが必要になる場合があります。デフォルトは `false` です。
- `comment="COMMENT"` はオプションで、二重引用符で囲まれたプロセスに関するコメントを指定します。
- `PARAMETER=VALUE` はオプションで、計算を実行するためのランタイム・パラメータとその値を指定します。パラメータと値のペアをプロセスの必要に応じて指定します。有効なパラメータと値は次のとおりです。
 - `rulesetSeqNumStart`: ルールセット内で最初に行うルールシーケンス番号。`EXECUTION_TYPE=RULESET_SUBSET` が使用されている場合にのみ有効です。
 - `rulesetSeqNumEnd`: ルールセット内で最後に実行するルールシーケンス番号を指定します。`EXECUTION_TYPE=RULESET_SUBSET` が使用されている場合にのみ有効です。
 - `ruleName`: 実行するルール名。空白が含まれる場合は、値を二重引用符で囲みます。`EXECUTION_TYPE` の値が `SINGLE_RULE`、`RUN_FROM_RULE` または `STOP_AFTER_RULE` に設定されている場合にのみ有効です。
 - `clearCalculatedData=true|false`: 既存の計算をクリアするかどうかを指定します。デフォルトは `false` です。
 - `executeCalculations=true|false`: 計算を実行するかどうかを指定します。デフォルトは `false` です。

 **Note:**

パラメータ値(`true` および `false`)はすべて小文字の必要があります。

例

- すべてのルールを実行して、1つの **POV** を計算します:

```
epmautomate calculateModel FY22::Jan::Actual::Working "10 Actuals Allocation Process" ALL_RULES clearCalculatedData=true executeCalculations=true optimizeForReporting=true comment="Running all rules to calculate a POV"
```
- すべてのルールを実行し、カスタム区切り文字も使用して、複数の **POV** を計算します:

```
epmautomate calculateModel "FY22_Jan_Actual_Working,FY22_Feb_Actual_Working,FY22_Mar_Actual_Working" "10 Actuals Allocation Process" ALL_RULES clearCalculatedData=true executeCalculations=true optimizeForReporting=true captureDebugScripts=true comment="Runing calculation for multiple POVs" povDelimiter="_"
```
- ルール・セットの範囲を実行して **POV** を計算します:

```
epmautomate calculateModel FY22::Jan::Actual::Working "10 Actuals Allocation Process" RULESET_SUBSET rulesetSeqNumStart=10 rulesetSeqNumEnd=20 clearCalculatedData=true executeCalculations=true comment="Running a subset of rule sets"
```
- 特定のルールを実行して、**POV** を計算します:

```
epmautomate calculateModel FY22::Jan::Actual::Working "10 Actuals Allocation Process" SINGLE_RULE ruleName="Rent and Utilities Reassignment"
```

```
clearCalculatedData=true executeCalculations=true comment="Running a specific rule"
```

clearCube

タイプ `clear cube` のジョブで指定した設定を使用して、入力およびレポート・キューブから特定のデータを削除します。

このコマンドは、アプリケーションのリレーショナル表内のアプリケーション定義を削除しません。*Planning の管理*のキューブのクリアを参照してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

`epmautomate clearCube JOB_NAME`。ここで、`JOB_NAME` は、アプリケーションで定義されたジョブの名前です。

例

```
epmautomate clearCube ClearPlan1
```

clearDataByPointOfView

Enterprise Profitability and Cost Management キューブの特定の POV のデータをクリアします。

適用対象

Enterprise Profitability and Cost Management

必要な役割

サービス管理者

使用方法

`epmAutomate clearDataByPointOfView POV_NAME [cubeName=CUBE_NAME] [PARAMETER=VALUE]`。ここで:

- `POV_NAME` は、アプリケーションの POV の名前です。
- `cubeName` はオプションで、クリアされるデータのキューブの名前です。デフォルトは `PCM_CLC` です。
- `PARAMETER=VALUE` は、オプションのランタイム・パラメータとその値を示します。パラメータと値のペアをプロセスの必要に応じて指定します。有効なパラメータと値は次のとおりです。
 - `povDelimiter` は POV 値で使用される区切り文字です。デフォルトは `::` (二重コロン) です。この値は、二重引用符で囲まれている必要があります。例: `povDelimiter="_"`

デフォルト以外では、次の区切り文字のみがサポートされています: _ (アンダースコア)、# (ハッシュ)、& (アンパサンド)、~ (チルダ)、% (パーセンテージ)、;(セミコロン)、:(コロン)、-(ダッシュ)。

- `clearInput=true|false` は、入力データをクリアするかどうかを指定します。デフォルトは `false` です。
- `clearAllocatedValues=true|false` は、割当て値をクリアするかどうかを指定します。デフォルトは `false` です。
- `clearAdjustmentValues=true|false` は、調整値をクリアするかどうかを指定します。デフォルトは `false` です。

 **Note:**

- * パラメータ値(`true` または `false`)はすべて小文字の必要があります。
- * `clearInput`、`clearAllocatedValues` または `clearAdjustmentValues` の少なくとも 1 つのパラメータを `true` に設定する必要があります。

例

- デフォルトの **POV** 区切り文字を使用して、デフォルトの **PCM_CLC** キューブの **POV** からデータをクリアします:
`epmAutomate clearDataByPointOfView FY21::Jan::Actual::Working clearInput=true clearAllocatedValues=true clearAdjustmentValues=true`
- カスタム **POV** 区切り文字を使用して、特定のキューブの **POV** から入力データと割当て済の値をクリアします:
`epmAutomate clearDataByPointOfView FY21_Jan_Actual_Working cubeName=PCM_REP povDelimiter="_" clearInput=true clearAllocatedValues=true`
- カスタム **POV** 区切り文字を使用して、特定のキューブの **POV** から入力データをクリアします:
`epmAutomate clearDataByPointOfView FY21:Jan:Actual:Working cubeName=PCM_REP povDelimiter=":" clearInput=true`

clearDataByProfile

Financial Consolidation and Close および Tax Reporting で定義されたデータのクリア・プロファイルで指定されたアイテム(リージョンなど)からデータをクリアします。

適用対象

Financial Consolidation and Close、Tax Reporting

必要な役割

サービス管理者

使用方法

`epmAutomate clearDataByProfile PROFILE_NAME`。ここで、`PROFILE_NAME` は、データのクリア・プロファイルの名前です。

例

```
epmautomate clearDataByProfile clearDataProfile_01
```

clearPOV

モデル・アーティファクトとデータを、**Profitability and Cost Management** アプリケーションにおける視点(POV)の組合せから、または POV 内のデータ・リージョンからクリアします。

適用対象

Profitability and Cost Management

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

```
epmautomate clearPOV APPLICATION_NAME POV_NAME [QUERY_NAME] PARAMETER=VALUE  
stringDelimiter="DELIMITER"。ここで:
```

- *APPLICATION_NAME* は、**Profitability and Cost Management** アプリケーションの名前です
- *POV_NAME* はアプリケーションの **POV** ですこの値は必須です。
- *QUERY_NAME* はオプションで、**Profitability and Cost Management** に定義されている問合せの名前を正確に指定します。これを指定すると、POV 内のデータ・リージョンをクリアするためにこの問合せが使用されます。

ノート:

問合せの名前を指定する場合は、すべてのランタイム・パラメータ(下記参照)の値を **false** に設定する必要があります。

- *PARAMETER=VALUE* には、**POV** をクリアするためのランタイム・パラメータとその値を指定します。パラメータと値のペアをプロセスの必要に応じて指定します。有効なパラメータ(1つ以上が必須)およびその値は次のとおりです:
 - *isManageRule=true|false* は、ルールをクリアするかどうかを指定します
 - *isInputData=true|false* は、入力データをクリアするかどうかを指定します
 - *isAllocatedValues=true|false* は、割当て値をクリアするかどうかを指定します
 - *isAdjustmentValues=true|false* は、調整値をクリアするかどうかを指定します

 ノート:

パラメータ値(true または false)はすべて小文字の必要があります。

POV のデータ・リージョンをクリアするには(QUERY_NAME が指定されている場合)、ランタイム・パラメータ(isManageRule、isInputData、isAllocatedValues、isAdjustmentValues)の値を false に設定する必要があります。

- stringDelimiter="DELIMITER"には、POV 値で使用される区切り文字を指定します。区切り文字は二重引用符で囲む必要があります。デフォルト値は_ (アンダースコア)です

例

- POV からすべてのモデル・アーティファクトとデータをクリア: `epmautomate clearPOV BksML12 2012_Jan_Actual isManageRule=true isInputData=true isAllocatedValues=true isAdjustmentValues=true stringDelimiter="_"`
- POV 内のデータ・リージョンをクリア: `epmautomate clearPOV BksML12 2012_Jan_Actual queryName=BksML12_2012_Jan_clear_query isManageRule=false isInputData=false isAllocatedValues=false isAdjustmentValues=false stringDelimiter="_"`

cloneEnvironment

現在の環境、およびオプションでアイデンティティ・ドメイン・アーティファクト(ユーザーおよび事前定義済役割の割当て)、データ管理レコード、監査レコード、ジョブ・コンソール・レコード、アプリケーション・プロパティ、受信ボックスと送信ボックスのコンテンツおよび保存されたスナップショットをクローニングします。このコマンドは、移行に環境のクローニング機能を使用する方法にかわるものです。

クローニングは、ソース環境およびターゲット環境のスケジュール済の日次メンテナンスの後に開始します。クローニングの進行中にソース環境の日次メンテナンスが開始された場合、クローニング・プロセスは終了します。日次メンテナンスの開始時にクローニングが進行中の場合でも、ターゲット環境のクローニング・プロセスは影響を受けません。このシナリオでは、日次メンテナンスは、クローニングが完了してから実行されます。

環境のクローニングに時間がかかる場合は、ソース環境の日次メンテナンスの開始時間を再スケジュールして、クローニング・プロセスが終了しないようにします。日次メンテナンスの開始時間をリセットする方法については、次の情報ソースを参照してください。

- [setDailyMaintenanceStartTime](#)
- *管理者用スタート・ガイド*の日次メンテナンスの管理。
- *REST API ガイド*の日次メンテナンス・ウィンドウの時間の表示および設定。

 **Note:**

- **データ管理:** ステージング表に非常に多数のレコードが含まれている場合は、データ管理レコードのクローニングに長時間かかることがあります。同様に、受信ボックスと送信ボックスのコンテンツや保存されたスナップショットのクローニングでは、特に大量のデータが含まれている場合は、かなりの時間を要する可能性があります。
- **レガシー環境:** 次のシナリオで説明するように、クローニングにより現在の Oracle Essbase バージョンを維持します:
 - シナリオ 1: ハイブリッド・キューブをサポートしない Essbase バージョンを使用するソース・レガシー環境を、ハイブリッド・キューブをサポートする Essbase バージョンを使用するターゲット・レガシー環境にクローニングします。このシナリオでは、ターゲット環境の Essbase は、ソース環境のバージョンと一致するようにダウングレードされます。
 - シナリオ 2: ハイブリッド・キューブをサポートする Essbase バージョンを使用するソース・レガシー環境を、ハイブリッド・キューブをサポートしない Essbase バージョンを使用するターゲット・レガシー環境にクローニングします。このシナリオでは、ターゲット環境の Essbase は、ソース環境のバージョンと一致するようにアップグレードされます。
 - シナリオ 3: ハイブリッド・キューブをサポートしない Essbase バージョンを使用するソース・レガシー環境を、ハイブリッド・キューブをサポートする Essbase バージョンをデフォルトで使用するターゲット EPM Standard Cloud Service 環境または EPM Enterprise Cloud Service 環境にクローニングします。このシナリオでは、ターゲット環境の Essbase は、ソース環境のバージョンと一致するようにダウングレードされません。
- **Planning:** カスタム期間メンバーに置き換えられ、名前が変更されたシード済期間メンバーが Planning ビジネス・プロセスに含まれると、クローニングが失敗する可能性があります。たとえば、シード済の *YearTotal* 期間メンバーの名前を *unused_YearTotal* に変更し、元のシード済メンバー名(この例では *YearTotal*)を使用して別タイプの期間メンバーを追加したとします。この場合、環境のクローニングは失敗する可能性があります。

これらのトピックの詳細は、*移行の管理*の Cloud EPM 環境のクローニングを参照してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

ユーザー役割および事前定義済役割をクローニングするには、アイデンティティ・ドメイン管理者役割が必要です。

使用方法

```
epmAutomate cloneEnvironment TARGET_USERNAME TARGET_PASSWORD TARGET_URL
[SnapshotName=NAME] [UsersAndPreDefinedRoles=true|false] [DataManagement=true|
false] [appAudit=true|false] [jobConsole=true|false]
[storedSnapshotsAndFiles=true|false] [DailyMaintenanceStartTime=true|false]
[ApplicationProperties=true|false]。ここで:
```

Note:

- dataManagement パラメータは、Oracle Enterprise Data Management Cloud および Narrative Reporting 環境には適用されません。
データ管理レコードは、ソース環境とターゲット環境の両方が同じ月次更新の場合、またはターゲット環境がソース環境よりも 1 つ新しい更新である場合のみクローニングします。たとえば、22.01 のデータ管理レコードは、別の 22.01 環境または 22.02 環境のみにクローニングできます。
- jobConsole パラメータは、Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning にのみ適用されます。
- appAudit パラメータは、Planning、Planning モジュール、フリーフォーム、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning にのみ適用されます。
Financial Consolidation and Close および Tax Reporting の監査情報は、デフォルトでスナップショットに含まれます。
- dataManagement、jobConsole または appAudit パラメータを環境に適用できない場合、指定された値は無視されます。
- ApplicationProperties パラメータは Account Reconciliation にのみ適用されます。

- TARGET_USERNAME は、ターゲット環境のサービス管理者の ID です。ターゲットのアイデンティティ・ドメイン・ユーザー名(SSO ユーザー名ではなく)を使用する必要があります。ターゲット環境でユーザーと役割の割当てをクローニングする予定がある場合、このユーザーにはアイデンティティ・ドメイン管理者の役割も必要です。
- TARGET_PASSWORD は、TARGET_USERNAME で識別されるユーザーの暗号化されたパスワード・ファイルの場所です。
- TARGET_URL は、クローニング先の環境となる環境の CLOUD-EPM_BASE_URL です。
- SnapshotName はオプションで、クローニングに使用されるスナップショットの名前です。このスナップショットはソース環境に存在する必要があります。デフォルトは Artifact Snapshot で、最後のメンテナンス・スナップショットを使用して環境をクローニングします。
- UsersAndPreDefinedRoles はオプションで、ユーザーとその事前定義済役割の割当てをクローニングするかどうかを識別します(アクセス制御グループは常にクローニングされます)。デフォルトは false です。

このオプションが機能するためには、`TARGET_USER_NAME` で識別されるユーザーに、ターゲット環境のアイデンティティ・ドメイン管理者の役割が必要です。

このチェック・ボックスを選択した後に、アイデンティティ・ドメイン管理者ではないユーザーが環境をクローニングすると、ユーザーとその事前定義済役割のインポートは失敗します。次のエラーが移行ステータス・レポートに記録されます: 外部ディレクトリ・アーティファクト<artifact_name>のインポートに失敗しました。ユーザー<user_name>には、この操作を実行する権限はありません。この操作を実行するには、ユーザーにアイデンティティ・ドメイン管理者役割が必要です。

- ユーザーをインポートせず、ソース・スナップショット内のユーザーがターゲット環境の事前定義済役割に割り当てられていない場合は、エラー(EPmie-00070: 割り当てられた役割のインポート中にユーザーが見つかりませんでした)が表示されます。
 - アイデンティティ・ドメイン管理者の役割の割り当てはクローニングされません。アイデンティティ・ドメイン管理者の役割のみが割り当てられているユーザーは、ターゲット環境にクローニングされません。
アイデンティティ・ドメイン管理者の役割とソース環境の事前定義済役割の組合せに割り当てられているユーザーはクローニングされますが、ターゲット環境の個々の事前定義済役割にのみ割り当てられます。これらのユーザーは、ターゲット環境のアイデンティティ・ドメイン管理者の役割を持ちません。
 - ユーザーの事前定義済役割に対する変更は、ソース・スナップショットで割り当てられている役割に基づいて更新されます。ただし、ソース・スナップショットの割り当てと一致させるためにターゲットの役割の割り当てが削除されることはありません。たとえば、jdoe がターゲット環境ではパワー・ユーザーという事前定義済役割に割り当てられているが、ソース・スナップショットではユーザー役割のみを持っているとします。この状況では、このコマンドによって、ターゲット環境で jdoe がユーザー役割に割り当てられますが、パワー・ユーザーの役割の割り当ては削除されません。
 - このコマンドでは、ターゲット環境に存在するユーザーは、ソース・スナップショットに存在しなくても削除されません。たとえば、jdoe にはターゲット環境にアカウントがありますが、ソース・スナップショットにはこのアカウントが存在しないとします。この状況では、ターゲット環境の jdoe のアカウントは削除されません。
 - このコマンドでは、ターゲット環境に存在しないユーザーは追加されます。ターゲット環境の現在のユーザー・プロパティは、ソース・スナップショットと異なる場合でも更新されません。たとえば、ソース・スナップショットの jdoe の姓のスペルがターゲット環境で異なる場合、ターゲット環境では変更は加えられません。ターゲット環境の新しいユーザーにはランダムなパスワードが割り当てられます。新しいユーザーは、パスワードの変更を求めるアカウントのアクティブ化の電子メールを受信します。
 - このコマンドでは、ターゲット環境の既存のユーザーのパスワードがソース・スナップショットと異なる場合でも変更されません。
- `dataManagement=true|false` はオプションで、ソース環境のデータ管理レコードをターゲット環境にクローニングします。デフォルトは `true` で、データ管理レコードをクローニングします。データ管理レコードをクローニングしない場合は、この値を `false` に設定します。
 - `appAudit=true|false` はオプションで、ソース環境の監査レコードをターゲット環境にクローニングします。デフォルトは `true` で、アプリケーション監査データをクローニングします。アプリケーション監査データをターゲット環境にクローニングしない場合は、この値を `false` に設定します。
 - `jobConsole=true|false` はオプションで、ソース環境のジョブ・コンソール・レコードをターゲット環境にクローニングします。デフォルトは `true` です。ジョブ・コンソール・レコードをクローニングしない場合は、この値を `false` に設定します。

- `storedSnapshotsAndFiles` はオプションで、受信ボックスと送信ボックスのコンテンツおよび保存されたスナップショットをクローニングするかどうかを識別します。デフォルトは `false` です。

 **Note:**

受信ボックスおよび送信ボックスの最上位フォルダのみがクローニングされ、サブフォルダはクローニングされません。サブフォルダの内容を保持する必要がある場合は、サブフォルダの内容をローカル・コンピュータにバックアップしてから、ターゲット環境にアップロードします。

- `DailyMaintenanceStartTime` はオプションで、クローニングされたターゲット環境のメンテナンス開始時間をソース環境のメンテナンス開始時間にリセットします。デフォルトは `true` です。ターゲット環境の現在のメンテナンス開始時間を維持するには、この値を `false` に設定します。
- `ApplicationProperties` はオプションで、**Account Reconciliation** アプリケーション設定 (レッドウッド・エクスペリエンス、テーマ、ビジネス・プロセス名、ロゴ・イメージおよび背景イメージ)をクローニングします。デフォルトは `true` です。ターゲット環境で現在のアプリケーション設定を維持するには、この値を `false` に設定します。

例

- 環境、ユーザーおよび事前定義済役割の割当て、監査データ、ジョブ・コンソール・レコードおよびデータ管理レコードをクローニングします。また、ターゲット環境のメンテナンス開始時間もソース環境のメンテナンス開始時間に変更します:

```
epmAutomate cloneEnvironment serviceAdmin Password.epw https://test-cloudpln.pbcs.us1.oraclecloud.com UsersAndPreDefinedRoles=true
```
- 受信ボックスと送信ボックスのコンテンツ、保存されたスナップショットを対象に含めて、ユーザーおよび事前定義済役割の割当て、データ管理レコード、監査データおよびジョブ・コンソール・レコードは対象から除外し、ターゲット環境のメンテナンス開始時間を変更せずに環境をクローニングします:

```
epmAutomate cloneEnvironment serviceAdmin Password.epw https://test-cloudpln.pbcs.us1.oraclecloud.com DataManagement=false appAudit=false jobConsole=false storedSnapshotsAndFiles=true DailyMaintenanceStartTime=false
```
- カスタム・スナップショットを使用して、環境全体(ユーザーおよび事前定義済役割の割当て、監査データ、ジョブ・コンソール・レコード、受信ボックスと送信ボックスのコンテンツ、保存されたスナップショットおよびデータ管理レコード)をクローニングします。また、ターゲット環境のメンテナンス開始時間もソース環境のメンテナンス開始時間に変更します:

```
epmAutomate cloneEnvironment serviceAdmin Password.epw https://test-cloudpln.pbcs.us1.oraclecloud.com UsersAndPreDefinedRoles=true storedSnapshotsAndFiles=true SnapshotName=SampleSnapshot
```
- **Account Reconciliation のみ:** 環境、ユーザーおよび事前定義済役割の割当て、およびデータ管理レコードをクローニングしますが、ターゲット環境のアプリケーション設定は維持します:

```
epmAutomate cloneEnvironment serviceAdmin Password.epw https://test-cloudarcs.arcs.epm.us.oraclecloud.com UsersAndPreDefinedRoles=true ApplicationProperties=false
```

compactCube

キューブの圧縮タイプのジョブで定義された設定を使用して、集約ストレージ(ASO)キューブのアウトライン・ファイルを圧縮します。アウトラインの圧縮は、キューブ・アウトライン・ファイルを最適なサイズに保持するのに役立ちます。圧縮してもキューブ内のデータはクリアされず、ビジネス・プロセスから Oracle Essbase に変更がプッシュされることはありません。

適用対象

Planning、Planning モジュール、フリーフォーム、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning

必要な役割

サービス管理者

使用方法

`epmautomate compactCube ASO_CUBE_NAME`。ここで、`ASO_CUBE_NAME` は、アウトラインを圧縮する ASO キューブの名前です。

例

```
epmautomate compactCube Vis1Aso
```

copyDataByPointOfView

データをキューブのソース POV から同じまたは別の Enterprise Profitability and Cost Management キューブの宛先 POV にコピーします。

適用対象

Enterprise Profitability and Cost Management

必要な役割

サービス管理者

使用方法

`epmAutomate copyDataByPointOfView SOURCE_POV_NAME TARGET_POV_NAME copyType=ALL_DATA|INPUT SOURCE_CUBE_NAME TARGET_CUBE_NAME [PARAMETER=VALUE]`。ここで:

- `SOURCE_POV_NAME` は、データのコピー元のソース POV の名前です。
- `TARGET_POV_NAME` は、ソースのデータのコピー先の有効なターゲット POV の名前です。
- `copyType` は、ソース POV からコピーするデータを識別します。有効な値は次のとおりです。
 - `ALL_DATA`: すべての入力データと計算データを宛先 POV にコピーします。
 - `INPUT`: ドライバ・データを含むすべての入力データを宛先 POV にコピーします。
- `SOURCE_CUBE_NAME` は、ソース POV を含むキューブの名前です。
- `TARGET_CUBE_NAME` は、ターゲット POV を含むキューブの名前です。

- `PARAMETER=VALUE` は、オプションのランタイム・パラメータとその値を示します。パラメータと値のペアをプロセスの必要に応じて指定します。有効なパラメータと値は次のとおりです。
 - `povDelimiter` はオプションで、**POV** 値で使用される区切り文字です。デフォルトは:: (二重コロン)です。この値は、二重引用符で囲まれている必要があります。例:
`povDelimiter="_"`
 デフォルト以外では、次の区切り文字のみがサポートされています: `_` (アンダースコア)、`#` (ハッシュ)、`&` (アンパサンド)、`~` (チルダ)、`%` (パーセンテージ)、`;` (セミコロン)、`:` (コロン)、`-` (ダッシュ)。
 - `createDestPOV=true|false` は、ターゲット **POV** が存在しない場合に作成するかどうかを指定しますデフォルトは `false` です。宛先 **POV** が存在しない場合は、このパラメータ値を `true` に設定する必要があります。

例

- すべてのデータを同じキューブの別の **POV** にコピーします:
`epmAutomate copyDataByPointOfView FY21_Jan_Actual_Working
FY22_Jan_Actual_Working ALL_DATA PCM_CLC PCM_CLC povDelimiter="_"
createDestPOV=true`
- すべてのデータを別のキューブの別の **POV** にコピーします:
`epmAutomate copyDataByPointOfView FY21_Jan_Actual_Working
FY22_Jan_Actual_Working ALL_DATA PCM_CLC PCM_REP povDelimiter="_"
createDestPOV=true`
- 入力データを同じキューブの別の **POV** にコピーします:
`epmAutomate copyDataByPointOfView FY21_Jan_Actual_Working
FY22_Jan_Actual_Working INPUT PCM_CLC PCM_CLC povDelimiter="_"
createDestPOV=true`
- 入力データを別のキューブの別の **POV** にコピーします:
`epmAutomate copyDataByPointOfView FY21_Jan_Actual_Working
FY22_Jan_Actual_Working INPUT PCM_CLC PCM_REP povDelimiter="_"
createDestPOV=true`

copyDataByProfile

データのコピー・プロファイルで識別されたアイテム(リージョンなど)のデータをコピーします。

適用対象

Financial Consolidation and Close、Tax Reporting

必要な役割

サービス管理者

使用方法

`epmAutomate copyDataByProfile PROFILE_NAME`。ここで、`PROFILE_NAME` は、**Financial Consolidation and Close** および **Tax Reporting** で定義されたデータのコピー・プロファイルの名前です。

例

```
epmautomate copyDataByProfile copyDataProfile_01
```

copyFileFromInstance

ソース環境にあるファイルを、このコマンドを実行している環境にコピーします。

このコマンドを実行する前に、EPM 自動化を使用して、ファイルのコピー先の環境にサインインします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Strategic Workforce Planning および Sales Planning。

必要な役割

- サービス管理者
- パワー・ユーザーおよび移行 - 管理アプリケーション役割(Oracle Enterprise Data Management Cloud および Profitability and Cost Management)

使用方法

```
epmautomate copyFileFromInstance SOURCE_FILE_NAME USERNAME PASSWORD_FILE URL  
TARGET_FILE_NAME。ここで:
```

- *SOURCE_FILE_NAME* は、ソース環境からコピーするファイルの名前(拡張子含む)です。
- *USERNAME* は、ソース環境のサービス管理者のユーザー名です。
- *PASSWORD_FILE* は、ソース環境のサービス管理者の暗号化されたパスワードを格納するファイルの名前と場所です。
- *URL* は、ソース環境の URL です。
- *TARGET_FILE_NAME* は、このコマンドを実行する環境におけるファイルの一意的な名前(拡張子含む)です。

例

```
epmautomate copyFileFromInstance "my data file.zip" serviceAdmin  
C:\mySecuredir\password.epw https://test-cloud-pln.pbcs.us1.oraclecloud.com "my  
target data file.zip"
```

copyFromObjectStorage

Oracle Object Storage バケットから現在の環境にファイルまたはバックアップ・スナップショットをコピーします。

バックアップ・スナップショットをコピーする場合、このコマンドは Object Storage バケットからバックアップ・スナップショットをコピーし、Oracle Fusion Cloud Enterprise Performance Management でその内容を抽出します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

`epmautomate copyFromObjectStorage USERNAME PASSWORD URL TARGET_FILE_NAME`。ここで:

- `USERNAME` は、**Oracle Object Storage Cloud** の必要なアクセス権を持つユーザーの ID です。フェデレーション・アイデンティティ・プロバイダで作成されたユーザーの場合は、ユーザーの完全修飾名を指定します(たとえば、`exampleIdP/jdoe` や `exampleIdP/john.doe@example.com`。ここで、`exampleIdP` はフェデレーション・アイデンティティ・プロバイダの名前)。その他のユーザーの場合は、ユーザー ID を指定します。
- `PASSWORD` は、ユーザーに関連付けられている **Swift** パスワードまたは認証トークンです。このパスワードは、オブジェクト・ストレージ・コンソールへのサインインに使用するパスワードとは異なります。認証トークンは、**Oracle** で生成されるトークンであり、たとえば、**Swift** クライアントでの認証など、サード・パーティの API での認証に使用します。このトークンを作成する手順は、**Oracle Cloud Infrastructure** ドキュメントの [認証トークンを作成するには](#) を参照してください。
- `URL` は、バケット名とコピーするオブジェクトの名前を含む、**Oracle Object Storage Cloud** バケットの URL です。

URL の形式:

```
https://swiftobjectstorage.region_identifier.oraclecloud.com/v1/namespace/  
bucket_name/object_name
```

この URL のコンポーネント:

- `region_identifier` は、**Oracle Cloud Infrastructure** ホスティング・リージョンです。
- `namespace` は、すべてのバケットとオブジェクトの最上位のコンテナです。各 **Oracle Cloud Infrastructure** テナントには、アカウントの作成時に、システムによって生成された一意の **Object Storage** 名前空間名が割り当てられます。テナントの名前空間名 (`axaxnpcrorw5` など)は、すべてのリージョンで有効です。
- `bucket_name` は、データとファイルを保存する論理コンテナの名前です。バケットはコンパートメントの下に整理され、維持されます。システムで生成されたバケット名(たとえば、`bucket-20210301-1359`)は、現在の年、月、日、および時刻を反映します。
- `object_name` は、**Oracle Object Storage Cloud** からコピーするスナップショットまたはファイルの名前です。この値は、**Object Storage Cloud** のオブジェクトのフル・ネームと正確に一致する必要があります。オブジェクト名に含まれていないかぎり、`.zip` などの拡張子は使用しないでください。

詳細は、**Oracle Cloud Infrastructure** のドキュメントのこれらのトピックを参照してください。

- [リージョンと可用性ドメイン](#)
- [オブジェクト・ストレージ名前空間の理解](#)

– バケットの管理

- `TARGET_FILE_NAME` は、Cloud EPM 環境内のファイルまたはスナップショットの一意の名前です。スナップショットをコピーするときは、このファイル名を `importSnapshot` コマンドで使用できるように、ZIP 拡張子を指定しないでください。
100 MB を超えるファイルは、そのセグメントを識別するマニフェスト・ファイルとともに Oracle Object Storage の論理ディレクトリ内に格納されます。論理ディレクトリの名前を `TARGET_FILE_NAME` として指定します。

例

これらの例では、`URL_OF_THE_ORACLE_OBJECT_STORAGE_BUCKET` を、次の形式の作業 URL に置換します：`https://swiftobjectstorage.region_identifier.oraclecloud.com/v1/namespace/bucket_name/`。

- `backup_Snapshot_12_05_20.zip` という名前のスナップショットを Oracle Object Storage バケットから Cloud EPM にコピーし、名前を変更します：

```
epmautomate copyFromObjectStorage oracleidentitycloudservice/jDoe example_pwd URL_OF_THE_ORACLE_OBJECT_STORAGE_BUCKET/backup_Snapshot_12_05_20.zip snapshot_from_osc
```
- `backup_Snapshot_12_05_20` という名前のスナップショットを Oracle Object Storage バケットから Cloud EPM にコピーし、名前を変更します：

```
epmautomate copyFromObjectStorage oracleidentitycloudservice/jDoe example_pwd URL_OF_THE_ORACLE_OBJECT_STORAGE_BUCKET/backup_Snapshot_12_05_20 snapshot_from_osc
```
- 名前を変更せずに、`backup_Snapshot_12_05_20` という名前のスナップショットを Oracle Object Storage バケットから Cloud EPM にコピーします：

```
epmautomate copyFromObjectStorage oracleidentitycloudservice/jDoe example_pwd URL_OF_THE_ORACLE_OBJECT_STORAGE_BUCKET/backup_snapshot_12_05_20 backup_snapshot_12_05_20
```
- ファイルを Oracle Object Storage バケットから Cloud EPM にコピーします：

```
epmautomate copyFromObjectStorage oracleidentitycloudservice/jDoe example_pwd URL_OF_THE_ORACLE_OBJECT_STORAGE_BUCKET/example_file.txt copied_from_osc.txt
```

copyFromSFTP

ファイルを、SFTP サーバーから OCI (Gen 2)環境に直接コピーします。このコマンドは、データ・ファイルをセキュアな SFTP サーバーから Oracle Fusion Cloud Enterprise Performance Management 環境に直接コピーするときに役立ちます。このコマンドを使用すると、ファイルを SFTP サーバーから直接コピーでき、最初に SFTP クライアントを使用してファイルを別のサーバーにダウンロードし、次に EPM 自動化または REST API を使用して Cloud EPM にアップロードする必要がありません。

IP 許可リストが SFTP サーバーに対して有効になっている場合は、Cloud EPM 環境をホストする OCI リージョンのアウトバウンド IP アドレスを SFTP サーバーの IP 許可リストに追加する必要があります。OCI リージョンのアウトバウンド IP アドレスについては、*オペレーション・ガイド*の Cloud EPM のデータ・センターおよびリージョンのアウトバウンド IP アドレスを参照してください。

 **Note:**

このコマンドは、ポート 22 で実行されている SFTP サーバーのみをサポートします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

```
epmautomate copyFromSftp SFTP_SERVER_URL FILE_NAME username=USERNAME  
password=PASSWORD。ここで:
```

- *SFTP_SERVER_URL* は SFTP サーバーの URL で、サーバーからコピーするファイルの場所と名前が含まれます。
- *FILE_NAME* は、Cloud EPM 環境内のファイルのファイル名です。たとえば、デフォルトのアップロード場所をコピーしない場合などは、パスを指定します。
- *USERNAME* は、SFTP サーバーの必要なアクセス権を持つユーザーの ID です。
- *PASSWORD* は SFTP ユーザーのパスワードです。

例

```
epmautomate copyFromSftp sftp://mySFTP_Server/someDirectory/someFile.csv  
myFile.csv username=jdoe password=Sftpwelcome
```

copyOwnershipDataToNextYear

出資比率データを、年の最後の期間から翌年の最初の期間にコピーします。

出資比率の初期デフォルト設定と上書き設定は、同一年度内の期間には自動的に繰り越されませんが、後続の年度の期間には繰り越されません。最新の出資比率設定を年の最後の期間から次の年の最初の期間に繰り越すには、POV で年の最後の期間から次の年の最初の期間に出資比率設定をコピーする必要があります。

適用対象

Financial Consolidation and Close および Tax Reporting

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー

使用方法

epmautomate copyOwnershipDataToNextYear Scenario Year。ここで:

- Scenario は、出資比率データのコピー元のシナリオ名です。
- Year は、次の年の最初の期間への出資比率データのコピー元の年です。

例

```
epmautomate copyOwnershipDataToNextYear FCCS_total_Actual FY18
```

copyPOV

モデル・アーティファクトと Oracle Essbase キューブ・データをソース POV から宛先 POV にコピーします。

適用対象

Profitability and Cost Management

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

```
epmautomate copyPOV APPLICATION_NAME SOURCE_POV_NAME TARGET_POV_NAME  
PARAMETER=VALUE stringDelimiter="DELIMITER" [isInputData=true|false  
isAllInputData=true|false]。ここで、
```

- APPLICATION_NAME は、ソース POV を含む Profitability and Cost Management アプリケーションの名前です。
- SOURCE_POV_NAME は、指定したアプリケーションのソース POV の名前です
- TARGET_POV_NAME は、Draft ステータスの有効なターゲット POV の名前です
- PARAMETER=VALUE には、POV をコピーするためのランタイム・パラメータとその値を指定します。パラメータと値のペアをプロセスの必要に応じて指定します。有効なパラメータと値は次のとおりです。
 - isManageRule=true|false は、ルールをコピーするかどうかを指定します。
 - isInputData=true | isAllData=true | isAllInputData=true は省略可能であり、データのコピー方法を指定します。これらのパラメータの場合、デフォルト値は false です。次のいずれか 1 つのみを true として指定します。
 - * isInputData=true を指定して、入力データを宛先 POV にコピーします。
 - * isAllData=true を指定して、すべての入力データと計算データを宛先 POV にコピーします。
 - * AllInputData=true を使用して、ドライバ・データを含むすべての入力データを宛先 POV にコピーします。
 - modelViewName=NAME は、ソース POV からターゲット POV にコピーするデータ・スライスの名前を指定します。

- createDestPOV=true|false は、ターゲット POV が存在しない場合に作成するかどうかを指定します
- nonEmptyTupleEnabled=true|false は、コマンドがデータを持つ交差のみを考慮するように、空でないタプル(NET)を有効にするかどうかを指定します。デフォルトは true です。これにより、まれに、Essbase データのコピーに対してコマンドが適切に実行されない場合があります。そのような場合は、nonEmptyTupleEnabled=false を使用してデフォルトをオーバーライドし、パフォーマンスを向上させます。

 ノート:

パラメータ値(true または false)はすべて小文字の必要があります。

- stringDelimiter="DELIMITER"には、POV 値で使用される区切り文字を指定します。区切り文字は二重引用符で囲む必要があります。

例

- `epmautomate copyPOV BksML12 2012_Jan_Actual 2012_Feb_Actual isManageRule=true isInputData=true modelViewName="Balancing - 5 Customer Costs" createDestPOV=true stringDelimiter="_"`
- `epmautomate copyPOV BksML12 2012_Jan_Actual 2012_Feb_Actual isManageRule=true isAllInputData=true createDestPOV=true stringDelimiter="_"`
- `epmautomate copyPOV BksML12 2012_Jan_Actual 2012_Feb_Actual isManageRule=true isAllData=true createDestPOV=true stringDelimiter="_"`

copySnapshotFromInstance

ソース環境にある最新スナップショットを、このコマンドを実行する環境(ターゲット)にコピーします。

このコマンドは、現在のスナップショットを別の環境からコピーして、テスト環境から本番環境のように環境を移行する最初のステップとして主に使用されます。[importSnapshot](#) コマンドを使用して、移行プロセスを完了します。

このコマンドを実行する前に、EPM 自動化セッションを開始し、ターゲット環境にサインインします。

このコマンドがソース環境のスナップショットの生成中、たとえば日次メンテナンス中に現在のスナップショットをコピーするために実行された場合は、「ファイルが見つかりません」エラーが表示されます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Strategic Workforce Planning および Sales Planning。

必要な役割

- サービス管理者

- パワー・ユーザーおよび移行 - 管理アプリケーション役割(Oracle Enterprise Data Management Cloud および Profitability and Cost Management)

使用方法

epmautomate copySnapshotFromInstance *SNAPSHOT_NAME USERNAME PASSWORD_FILE URL*。ここで:

- *SNAPSHOT_NAME* は、ソース環境の既存のスナップショットの名前です。
- *USERNAME* は、ソース環境のサービス管理者のユーザー名です。
- *PASSWORD_FILE* は、ソース環境のサービス管理者の暗号化されたパスワードを格納するファイルの名前と場所です。
- *URL* は、ソース環境の URL です。

例

```
epmautomate copySnapshotFromInstance "Artifact Snapshot" serviceAdmin
C:\mySecuredir\password.epw https://test-cloud-pln.pbcs.us1.oraclecloud.com
```

copyToObjectStorage

現在の環境から **Oracle Object Storage Cloud** バケットにファイルまたはスナップショットをコピーします。

スナップショットをコピーする場合、このコマンドは、その内容を **Oracle Object Storage** にコピーする前に圧縮します。

ファイルの高速コピーを円滑にするため、このコマンドで大きなファイル(100 MB を超える)を 10 MB のセグメント(*FILE_NAME/FILE_NAME_object_store_bytes_seg_0* から *FILE_NAME/FILE_NAME_object_store_bytes_seg_n* という名前)に分割し、マニフェスト・ファイル(*FILE_NAME/FILE_NAME.manifest* という名前)を作成します。ファイル・セグメントは、マニフェスト・ファイルとともに **Oracle Object Storage** に格納されます。オブジェクト・ストレージ・コンソールでは、ファイルは、ファイル・セグメントおよびマニフェスト・ファイルを含む論理ディレクトリとして表示されます。

100 MB 未満のファイルはセグメント化されず、元のファイル名で格納されます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

epmautomate copyToObjectStorage *SOURCE_FILE_NAME USERNAME PASSWORD URL*。ここで:

- *SOURCE_FILE_NAME* は、**Oracle Fusion Cloud Enterprise Performance Management** のファイルまたはスナップショットの名前です。スナップショットをコピーする場合、ZIP 拡張子を指定しないでください。

- `USERNAME` は、**Oracle Object Storage Cloud** への書込みに必要なアクセス権を持つユーザーの ID です。
フェデレーション・アイデンティティ・プロバイダで作成されたユーザーの場合は、ユーザーの完全修飾名を指定します(たとえば、`exampleIdP/jdoe` や `exampleIdP/john.doe@example.com`。ここで、`exampleIdP` はフェデレーション・アイデンティティ・プロバイダの名前)。その他のユーザーの場合は、ユーザー ID を指定します。

- `PASSWORD` は、ユーザーに関連付けられている **Swift** パスワードまたは認証トークンです。このパスワードは、オブジェクト・ストレージ・コンソールへのサインインに使用するパスワードとは異なります。認証トークンは、**Oracle** で生成されるトークンであり、たとえば、**Swift** クライアントでの認証など、サード・パーティの API での認証に使用します。このトークンを作成する手順は、**Oracle Cloud Infrastructure** ドキュメントの [認証トークンを作成するには](#) を参照してください。

- `URL` は、オプションのオブジェクト名が追加された **Oracle Object Storage Cloud** バケットの URL です。

オブジェクト名のない URL 形式:

```
https://swiftobjectstorage.region_identifier.oraclecloud.com/v1/namespace/  
bucket_name
```

オブジェクト名がある URL 形式:

```
https://swiftobjectstorage.region_identifier.oraclecloud.com/v1/namespace/  
bucket_name/object_name
```

この URL のコンポーネント:

- `region_identifier` は、**Oracle Cloud Infrastructure** ホスティング・リージョンです。
- `namespace` は、すべてのバケットとオブジェクトの最上位のコンテナです。各 **Oracle Cloud Infrastructure** テナントには、アカウントの作成時に、システムによって生成された一意の **Object Storage** 名前空間名が割り当てられます。テナンシの名前空間名 (`axaxnpcrorw5` など)は、すべてのリージョンで有効です。
- `bucket_name` は、データとファイルを保存する論理コンテナの名前です。バケットはコンパートメントの下に整理され、維持されます。システムで生成されたバケット名(たとえば、`bucket-20210301-1359`)は、現在の年、月、日、および時刻を反映します。
- `object_name` は、オプションであり、**Oracle Object Storage Cloud** 上のファイルに使用する名前です。オブジェクト名を指定しない場合、ファイルは元の名前でコピーされます。

詳細は、**Oracle Cloud Infrastructure** のドキュメントのこれらのトピックを参照してください。

- [リージョンと可用性ドメイン](#)
- [オブジェクト・ストレージ名前空間の理解](#)
- [バケットの管理](#)

例

これらの例では、`URL_OF_THE_ORACLE_OBJECT_STORAGE_BUCKET` を、次の形式の作業 URL に置換します:`https://swiftobjectstorage.region_identifier.oraclecloud.com/v1/namespace/bucket_name/`。

- スナップショットを **Object Storage** バケットにコピーし、名前を変更します。

```
epmautomate copyToObjectStorage "Artifact Snapshot"
oracleidentitycloudservice/jDoe example_pwd
URL_OF_THE_ORACLE_OBJECT_STORAGE_BUCKET/Snapshot_04_30_21
```

- ファイルを **Object Storage** バケットにコピーします。
epmautomate copyToObjectStorage example_file.txt oracleidentitycloudservice/jDoe example_pwd URL_OF_THE_ORACLE_OBJECT_STORAGE_BUCKET
- ファイルを **Object Storage** バケットにコピーし、名前を変更します。
epmautomate copyToObjectStorage example_file.txt eoracleidentitycloudservice/jDoe example_pwd URL_OF_THE_ORACLE_OBJECT_STORAGE_BUCKET/epm_text_file.txt

copyToSFTP

ファイルを **OCI (Gen 2)**環境の場所からセキュアな **FTP** サーバーの場所にコピーします。このコマンドを使用すると、ファイルを **Oracle Fusion Cloud Enterprise Performance Management** 環境から **SFTP** サーバーに直接コピーでき、最初に **EPM** 自動化または **REST API** を使用してファイルを別のサーバーにダウンロードし、次に **SFTP** クライアントを使用して **SFTP** サーバーにアップロードする必要がありません。

IP 許可リストが **SFTP** サーバーに対して有効になっている場合は、**Cloud EPM** 環境をホストする **OCI** リージョンのアウトバウンド **IP** アドレスを **SFTP** サーバーの **IP** 許可リストに追加する必要があります。**OCI** リージョンのアウトバウンド **IP** アドレスについては、*オペレーション・ガイド*の **Cloud EPM** のデータ・センターおよびリージョンのアウトバウンド **IP** アドレスを参照してください。

Note:

このコマンドは、ポート 22 で実行されている **SFTP** サーバーのみをサポートします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

```
epmautomate copyToSftp SFTP_SERVER_URL EPM_FILE_NAME username=USERNAME
password=PASSWORD。ここで:
```

- **SFTP_SERVER_URL** は、**SFTP** サーバーの **URL** で、ファイルをコピーするディレクトリとファイル名を含みます。
- **EPM_FILE_NAME** は、**Cloud EPM** 環境からコピーしようとするファイルの名前と場所(デフォルト送信ボックスからではない場合)です。スナップショットをコピーしている場合は、**.ZIP** 拡張子を指定しないでください
- **USERNAME** は、**SFTP** サーバーの必要なアクセス権を持つユーザーの **ID** です。
- **PASSWORD** は **SFTP** ユーザーのパスワードです。

例

```
epmautomate copyToSftp sftp://mySFTP_Server/someDirectory/some_file.csv  
myFile.csv username=jDoe password=SftpWelcome
```

createGroups

環境にアップロードされた ANSI または UTF-8 エンコーディングの CSV ファイルを使用して、アクセス制御にグループを追加します。

サービス管理者は、[uploadFile](#) コマンドを使用して、ファイルを環境にアップロードします。ファイル形式は次のとおりです。

```
Group Name,Description  
Example_grp1,My test group  
Example_grp2,My other test group
```

グループ名では大文字と小文字は区別されません。終了すると、失敗した各エントリに関する情報がコンソールに出力されます。この情報を確認して、CSV ファイルの一部のエントリでコマンドの実行が失敗した理由を理解してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- 任意の事前定義済役割およびアクセス制御 - 管理アプリケーション役割

使用方法

epmautomate createGroups *FILE_NAME*。ここで、*FILE_NAME* は、グループ名と説明が含まれる CSV ファイルの名前です。

例

```
epmautomate createGroups group_file.CSV
```

createNRSnapshot

Narrative Reporting 環境の EPRCS_Backup.tar.gz という名前のオンデマンド・スナップショットを作成します。

EPRCS_Backup.tar.gz とエラー・ファイルを、[downloadFile](#) コマンドを使用してローカル・コンピュータにダウンロードするか、[copyFileFromInstance](#) コマンドを使用して別の環境にコピーできます。

EPRCS_Backup.tar.gz のアプリケーション・データは、最後の日次メンテナンス時点のデータです。最新のデータをバックアップする必要がある場合は、データ・エクスポート Narrative Reporting 機能を使用します。

適用対象

Narrative Reporting

必要な役割

サービス管理者

使用方法

`epmautomate createNRSnapshot [errorFile=Error_File.txt]`。ここで、`errorFile` はオプションで、コマンドでエラーが発生した場合に、エラーを記録するための一意のテキスト・ファイルの名前を識別します。

例

```
epmautomate createNRSnapshot errorFile=EPRCS_backup_Error.txt
```

createReconciliations

プロファイルを指定の期間にコピーします。

適用対象

Account Reconciliation。

必要な役割

- サービス管理者
- 任意の事前定義済役割(参照者の役割には、照合 - 策定者のアプリケーション役割が必要)

使用方法

`epmautomate createreconciliations PERIOD SAVED_FILTER`。ここで:

- `PERIOD` は、期間の名前です
- `SAVED_FILTER` は、保存済パブリック・フィルタの名前です。保存済フィルタを指定しないと、すべての適用可能なプロファイルがコピーされます

例

- 対象期間のすべてのプロファイルをコピーします:`epmautomate createReconciliations "January 2015"`
- 特定のフィルタのプロファイルをコピーします:`epmautomate createReconciliations "January 2015" "Corporate Recs"`

deleteFile

デフォルトのアップロード場所、受信ボックスまたは送信ボックス、データ管理のフォルダ、または `profitinbox/profitoutbox` からファイルまたはスナップショットを削除します。

デフォルトのアップロード場所以外からファイルを削除するには、ファイルの場所を指定する必要があります。

このコマンドが生成中またはアーカイブ中のスナップショットを削除するために実行された場合は、次のいずれかのエラーが表示されます:

- ファイルが見つかりません: スナップショットが生成中の場合
- アーカイブ・プロセスが進行中です。名前変更または削除できません: スナップショットがアーカイブ中の場合

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザーおよび移行 - 管理アプリケーション役割(Oracle Enterprise Data Management Cloud および Profitability and Cost Management)

使用方法

```
epmautomate deleteFile FILE_NAME
```

ノート:

該当する場合は拡張子を含むファイル名を指定する必要があります(たとえば、data.csv、data.zip など)。スナップショットはファイル拡張子(.ZIP)を指定しなくても削除できます。ただし、この使用方法は非推奨です。ファイルがデフォルトの場所がない場合は、ファイルの場所を指定してください。詳細は、[デフォルトのファイルの場所](#)を参照してください。サポートされている場所には、inbox、profitinbox、outbox、profitoutbox、to_be_imported および inbox/directory_name が含まれます。

例

- デフォルトのアップロード場所からファイルを削除します:
epmautomate deleteFile data.csv
- 受信ボックスからファイルを削除します:
epmautomate deleteFile inbox/data.csv
- 送信ボックスから削除します:
epmautomate deleteFile outbox/data.csv
- 「移行」を使用して作成したスナップショットを削除します。
 - epmautomate deleteFile "Backup 18-06-12.zip" または
 - epmautomate deleteFile "Backup 18-06-12" (非推奨)
- profitinbox から削除します(Profitability and Cost Management):
epmautomate deleteFile profitinbox/data.csv
- profitoutbox から削除します(Profitability and Cost Management):
epmautomate deleteFile profitoutbox/data.csv

- データ管理アップロード・フォルダから削除します:
`epmautomate deleteFile inbox/dm_data/data.csv`
- データ管理フォルダから削除します:
`epmautomate deleteFile outbox/dm_data/data.csv`

deleteGroups

環境にアップロードされた ANSI または UTF-8 エンコーディングの CSV ファイルで使用可能な情報に基づいて、アクセス制御からグループを削除します。

サービス管理者は、[uploadFile](#) コマンドを使用して、ファイルを環境にアップロードします。ファイル形式は次のとおりです。

```
Group Name  
Example_grp1  
Example_grp2
```

ファイル内のグループ名では大文字と小文字が区別されません。完了すると、失敗した各エントリに関する情報がコンソールに出力されます。この情報を確認して、CSV ファイルの一部のエントリでコマンドの実行が失敗した理由を理解してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- 任意の事前定義済役割およびアクセス制御 - 管理アプリケーション役割

使用方法

`epmautomate deleteGroups FILE_NAME`。ここで、`FILE_NAME` は、アクセス制御から削除するグループの名前を含む CSV ファイルの名前です。

例

```
epmautomate deleteGroups group_file.CSV
```

deletePointOfView

アーティファクトと Oracle Essbase キューブ・データを Enterprise Profitability and Cost Management アプリケーションの POV から削除します。

適用対象

Enterprise Profitability and Cost Management

必要な役割

サービス管理者

使用方法

`epmautomate deletePointOfView POV_NAME [povDelimiter="DELIMITER"]`。ここで:

- `POV_NAME` は、削除する **POV** の名前を識別します。
- `povDelimiter` は **POV** 値で使用される区切り文字です。デフォルトは `::` (二重コロン) です。この値は、二重引用符で囲まれている必要があります。例: `povDelimiter="_"`
デフォルト以外では、次の区切り文字のみがサポートされています: `_` (アンダースコア)、`#` (ハッシュ)、`&` (アンパサンド)、`-` (チルダ)、`%` (パーセンテージ)、`;` (セミコロン)、`:` (コロン)、`-` (ダッシュ)。

例

- カスタム **POV** 区切り文字を使用する **POV** の削除
`epmAutomate deletePointOfView FY21_Jan_Actual_Working povDelimiter="_"`
- デフォルトの **POV** 区切り文字を使用する **POV** の削除
`epmAutomate deletePointOfView FY21::Jan::Actual::Working`

deletePOV

モデル・アーティファクトと Oracle Essbase キューブ・データを Profitability and Cost Management の **POV** から削除します。

適用対象

Profitability and Cost Management

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

`epmautomate deletePOV APPLICATION_NAME POV_NAME stringDelimiter="DELIMITER"`。ここで:

- `APPLICATION_NAME` は、削除する **POV** を含む Profitability and Cost Management アプリケーションの名前です。
- `POV_NAME` は、削除する **POV** の名前ですこの値は必須です。
- `stringDelimiter="DELIMITER"`には、**POV** 値で使用される区切り文字を指定します。区切り文字は二重引用符で囲む必要があります。

例

`epmautomate deletePOV BksML12 2012_Jan_Actual stringDelimiter="_"`

deployCube

Profitability and Cost Management アプリケーションの計算キューブをデプロイまたは再デプロイします。

適用対象

Profitability and Cost Management

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

epmautomate deployCube APPLICATION_NAME PARAMETER=VALUE comment="comment"。ここで:

- APPLICATION_NAME は、Profitability and Cost Management アプリケーションの名前です
- PARAMETER=VALUE には、キューブをデプロイするためのランタイム・パラメータとその値を指定します。パラメータと値のペアをプロセスの必要に応じて指定します。有効なパラメータと値は次のとおりです。

ノート:

パラメータ値(true または false)はすべて小文字の必要があります。

- isKeepData=true|false

既存のデータ(ある場合)を保存するかどうかを指定します

- isReplaceCube=true|false は、既存のキューブを置換するかどうかを指定します

ノート:

isKeepData および isReplaceCube の両方の値を、true に設定することはできません。

- isRunNow=true|false は、プロセスをすぐに実行するかどうかを指定します
- comment には、二重引用符でコメントを囲んで指定します(オプション)

例

```
epmautomate deployCube BksML12 isKeepData=true isReplaceCube=false isRunNow=true  
comment="Test cube deployment"
```

deployEJTemplates

完成したエンタープライズ仕訳テンプレートを **Financial Consolidation and Close** のオープン期間にデプロイします。エンタープライズ仕訳テンプレートをデプロイすると、選択した期間のテンプレートに関連付けられた定型仕訳が作成されます。また、デプロイされたテンプレートを使用して、アドホック仕訳を作成することもできます。

このコマンドは、月初に新しいエンタープライズ仕訳テンプレートをデプロイする際に **Financial Consolidation and Close** 画面を使用する方法にかわるものです。

適用対象

Financial Consolidation and Close

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

```
epmautomate deployEJTemplates YEAR PERIOD [Template=TEMPLATE_NAME]
[ResetJournals=true|false]。ここで:
```

- Year は仕訳の年です。
- Period は仕訳の期間です。この値は、年が指定されている場合にのみ指定できます。
- Template=TEMPLATE_NAME は、デプロイする仕訳の名前を識別します。複数の仕訳をデプロイするには、それぞれ一意のテンプレート名を Template=TEMPLATE_NAME 形式で指定します(例: Template="Loan Details" Template="Housing Details" Template="Repayment Details")。このパラメータ値が指定されていない場合、指定した年と期間の組合せのすべてのテンプレートがデプロイされます。
- ResetJournals はオプションで、テンプレートを再デプロイした後、すべての仕訳を最初のステージにリセットする必要があるかどうかを示します。デフォルトは false です。**Financial Consolidation and Close** により、テンプレートに対する変更に基づいてこの値が内部的に検証され、指定した値が必要に応じて上書きされる可能性があります。

例

```
epmautomate deployEJTemplates 2021 May Template="Loan Details" Template="Housing
Details" ResetJournals=true
```

deployFormTemplates

完成したフォーム・テンプレートを新しいデータ収集期間にデプロイして、補足データ・フォームを作成し、データ収集プロセスの一貫性と反復性を確保します。

適用対象

Financial Consolidation and Close、Tax Reporting

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

`epmautomate deployFormTemplates COLLECTION_INTERVAL [DIMENSION] [Template] [resetWorkFlows=true|false]`。ここで:

- `COLLECTION_INTERVAL` は、テンプレートをデプロイする収集間隔の名前です。
- `DIMENSION` はオプションで、`DIMENSION=MEMBER_NAME` の形式で、データ収集プロセスの頻度ディメンションを指定します。収集間隔で定義した数と同じ数のディメンションを指定します。(Year、Period など、最大で 4 つ。例: "Year=2020" "Period=July" "Product=Oracle EPM" "Consolidation=entity Input")。このパラメータ値を指定しない場合、デフォルト値は使用されません。
- `Template` はオプションで、`Template=TEMPLATE_NAME` の形式で、デプロイするフォーム・テンプレートの一意の名前を指定します。この形式で、必要に応じて任意の数の一意の名前を指定できます。例: `Template="Loan Details Template" Template="Housing Details Template" Template="Repayment Details Template"`。このプロパティ値を指定しない場合、指定した間隔のすべてのテンプレートがデプロイされます。
- `resetWorkFlows` はオプションで、再デプロイ後にすべてのフォームが最初のステージにリセットされるかどうかを指定します。デフォルトは `false` です。

例

```
epmautomate deployFormTemplates "Journal Collection Interval" "Year=2020"
"Period=July" "Product=Oracle EPM" "Consolidation=entity Input" Template="Loan
Details Template" Template="Housing Details Template" resetWorkFlows=true
```

deployTaskManagerTemplate

タスク・マネージャ・テンプレートからタスク・スケジュールにタスクをデプロイし、反復的なビジネス・プロセスを一貫して実行できるようにします。

適用対象

Enterprise Profitability and Cost Management、Financial Consolidation and Close および Tax Reporting

必要な役割

サービス管理者

使用方法

`epmAutomate deployTaskManagerTemplate TEMPLATE_NAME SCHEDULE_NAME YEAR PERIOD DAY_ZERO_DATE [dateFormat=DATE_FORMAT] [orgUnit=ORGANIZATION UNIT]`。ここで:

- `TEMPLATE_NAME` は、デプロイするタスク・マネージャ・テンプレートの名前です。
- `SCHEDULE_NAME` は、テンプレートから作成するスケジュール名です。

- `YEAR` は、テンプレートをデプロイする年ディメンション・メンバーです。
- `PERIOD` は、テンプレートをデプロイする期間ディメンション・メンバーです。
- `DAY_ZERO_DATE` は、スケジュールの作成に使用する有効な形式のゼロ日の日付です。
- `dateFormat` はオプションで、ゼロ日の日付の日付形式です。デフォルトの形式は `YYYY-MM-DD` です。
- `orgUnit` はオプションで、組織単位の名前です。値が指定されていない場合、スケジュールは標準の日付マッピングを使用して作成されます。休日ルールは使用されません。

例

- ゼロの日付のデフォルトの日付形式(`YYYY-MM-DD`)を使用して、`Ind` という組織単位のタスク・マネージャ・テンプレートをデプロイします:

```
epmautomate deployTaskManagerTemplate "Vision Monthly Close" "Qtr 2 Close"
2021 July 2021-07-10 orgUnit=Ind
```
- ゼロの日付の日付形式として `dd/mm/yyyy` を使用して、`Ind` という組織単位のタスク・マネージャ・テンプレートをデプロイします:

```
epmautomate deployTaskManagerTemplate "Vision Monthly Close" "Qtr 2 Close"
2021 July 02/07/2021 dateFormat=dd/MM/yyyy orgUnit=Ind
```

dismissIPMInsights

新しい IPM インサイト・ジョブを実行する前に、インテリジェント・パフォーマンス管理(IPM) インサイト・データの破棄を自動化します。データを閉じると、アクションを実行する予定のないすべてのオープン・インサイトが閉じられます。このコマンドは、IPM インサイト・ダッシュボードを使用してデータを手動で閉じるかわりの方法です。

適用対象

Planning、Planning モジュール、Strategic Workforce Planning、Sales Planning。

必要な役割

サービス管理者

使用方法

`epmautomate dismissIPMInsights [comment="comment"]`。ここで、`comment` は省略可能であり、オープン・インサイトを却下する理由です。

例

```
epmautomate dismissIPMInsights comment="dismissing unusable insights"
```

downloadFile

環境からローカル・コンピュータにファイルをダウンロードします。

このコマンドを使用して、ローカル・ストレージに対するデータ、メタデータおよびバックアップ・スナップショットをダウンロードします。ファイルは EPM 自動化を実行するフォルダにダウンロードされます。

このコマンドが環境のスナップショットの生成中、たとえば日次メンテナンス中に現在のスナップショットをダウンロードするために実行された場合は、「ファイルが見つかりません」エラーが表示されます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザーおよび移行 - 管理アプリケーション役割(Oracle Enterprise Data Management Cloud および Profitability and Cost Management)

使用方法

```
epmautomate downloadFile "[FILE_PATH]/FILE_NAME"
```

例

- メンテナンス・スナップショットのダウンロード: `epmautomate downloadFile "Artifact Snapshot"`
- カスタム・スナップショットのダウンロード: `epmautomate downloadFile "mySnapshot.zip"`
- Narrative Reporting のメンテナンス・スナップショットのダウンロード: `epmautomate downloadFile "EPRCS_Backup.tar.gz"`
- デフォルトのダウンロード場所からのファイルのダウンロード: `epmautomate downloadFile data.csv`
- データ管理のフォルダからのダウンロード: `epmautomate downloadfile outbox/dm_data/data.csv`
- profitoutbox からのダウンロード: `epmautomate downloadFile profitOutbox/data.csv`

enableApp

アプリケーションを有効にします。

適用対象

Profitability and Cost Management

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

`epmautomate enableapp APPLICATION_NAME`。APPLICATION_NAME は、有効にする Profitability and Cost Management アプリケーションの名前です。

例

```
epmautomate enableApp BksML12
```

enableQueryTracking

ASO キューブの問合せトラッキングを有効にして、ユーザー・データ取得パターン(問合せ)のキャプチャを開始します。

キャプチャされたデータ取得パターンを使用して、ASO キューブ集約を最適化します。これは、[executeAggregationProcess](#) コマンドを使用して開始されます。

適用対象

Planning、Planning モジュール、フリーフォーム、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

`epmautomate enableQueryTracking ASO_CUBE_NAME`。ここで、`ASO_CUBE_NAME` は、問合せトラッキングをアクティブにする ASO キューブの名前です。

例

```
epmautomate enableQueryTracking VISION_ASO
```

encrypt

Advanced Encryption Standard (AES/CBC/PKCS5Padding (256))を使用して、Oracle Fusion Cloud Enterprise Performance Management パスワード(または OCI (Gen 2)環境にアクセスするための OAuth2.0 リフレッシュ・トークンとクライアント ID)、およびオプションで、Oracle Fusion Cloud EPM 環境へのサインインに使用されるインターネット・プロキシ・サーバーのパスワードを暗号化し、パスワード・ファイルに保存します。

シークレットを暗号化すると、サービス管理者は、EPM 自動化スクリプトを記述する開発者がスクリプトを実行できるように、暗号化されたパスワード・ファイルを開発者と共有できます。これにより、スクリプトを実行するためにサービス管理者パスワードを共有したり、汎用の共有 Cloud EPM アカウントを作成する必要がなくなります。

パスワードの暗号化はワンタイム・プロセスです。

ノート:

特殊文字を含むパスワードの暗号化の詳細は、[特殊文字の処理](#)を参照してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

使用方法

`epmautomate encrypt PASSWORD|REFRESH_TOKEN KEY PASSWORD_FILE [ClientID=CLIENT_ID] [ProxyServerPassword=PROXY_PASSWORD]`。ここで:

- `PASSWORD|REFRESH_TOKEN PASSWORD` は、暗号化するパスワードまたは OAuth リフレッシュ・トークンです。会社の資格証明を EPM 自動化で使用することはできません。
- `KEY` は、パスワードの暗号化に使用する秘密キーです。
- `PASSWORD_FILE` は、暗号化されたパスワードまたはリフレッシュ・トークンを格納するファイルの名前と場所です。パスワード・ファイルは、.epw 拡張子を使用する必要があります。
- `ClientID` は省略可能であり、OAuth 2.0 のセットアップ中に作成されるクライアント識別子です。この値は、OAuth 2.0 リフレッシュ・トークンを暗号化するときに指定する必要があります。パスワードの暗号化中にこの値を指定しないでください。
- `ProxyServerPassword` は、HTTP プロキシ・サーバーでユーザーを認証するパスワードです。必要なのは、ネットワークに対してプロキシ・サーバーでの認証が有効になっている場合のみです。

例

- **Cloud EPM パスワードのみを暗号化:** `epmautomate encrypt P@ssword1 myKey C:\mySecuredir\password.epw`
- **Cloud EPM とインターネット・プロキシ・サーバーのパスワードを暗号化:** `epmautomate encrypt E@example1 myKey C:\mySecuredir\password.epw ProxyServerPassword=Proxy_Pwd1`
- **リフレッシュ・トークンとクライアント ID を暗号化:** `epmautomate encrypt AAyyilyBAWD4...FVxkxefd8kjoJr6HJPA= myEncryption42Key C:\mySecuredir\oauthfile1.epw ClientID=6fdf2e72fd343430ABR22394C`

essbaseBlockAnalysisReport

Essbase ブロック分析レポートを作成します。これは、Oracle Essbase データを分析して、アプリケーションのブロック・ストレージ・オプション(BSO)キューブ(通常、計算に使用)の調整をサポートするのに役立ちます。

Essbase ブロック分析レポートは、たとえば、Essbase BSO キューブに含まれる繰り返しの数など、データのパターンから生じるパフォーマンスの問題を解決するのに役立ちます。このレポートは、次の 3 つの領域に関する情報を提供します:

1. **ゼロのみを含むブロックのパーセンテージ。**これは、ゼロのみを含むブロックを、エクスポート・ファイルに含まれるすべてのブロックに対するパーセンテージとして示します
2. **数値セルの繰り返し値上位 10 件(数値セルのパーセンテージ)。**この表は、繰り返し値の上位 10 件を、エクスポート・ファイル内のすべての値に対するパーセンテージとして示します。

3. **繰返し値を持つ密メンバーの組合せ上位 100 件**。この表は、キューブ内で繰返し値を持つ密の組合せ上位 100 件を示します。「セルの値」列には、各メンバーの値が、階層内に表示される順序で個別の列として表示されます。たとえば、期間が列にある場合、1 月、2 月などには別の列が設けられます。それ以外の密ディメンションは行に表示されます。これにより、繰返し値の場所が識別しやすくなります。

このレポートを実行する前に、[exportEssbaseData](#) コマンドを使用して、ブロック分析レポートを作成するキューブのデータを zip ファイルにエクスポートします。必要に応じて、**level0** またはすべてのデータをエクスポートできます。このコマンドを実行して、この zip ファイルに対してブロック分析レポートを作成します。レポートは送信ボックスに作成されます。[downloadFile](#) コマンドを使用してローカル・コンピュータにダウンロードするか、[sendMail](#) コマンドを使用して電子メールで送信します。

適用対象

Financial Consolidation and Close、Planning、Planning モジュール、フリーフォーム、Strategic Workforce Planning、Sales Planning および Tax Reporting。

必要な役割

サービス管理者

使用方法

`epmautomate essbaseBlockAnalysisReport EXPORT_DATA_FILE.zip REPORT_FILE`。ここで

- `EXPORT_DATA_FILE.zip` は、[exportEssbaseData](#) コマンドを使用して BSO キューブから以前にエクスポートされた Essbase データを含む zip ファイルの名前です。
- `REPORT_FILE` は、HTML 形式のブロック分析レポート・ファイルの名前です。

例

```
epmautomate essbaseBlockAnalysisReport Plan1_cube_data.zip block_report.html
```

executeAggregationProcess

ASO キューブのパフォーマンスを高めるために、必要に応じて問合せトラッキング統計を使用して集約プロセスを開始します。これは、ASO キューブを最適化するための重要なステップです。

このコマンドを実行する前に、次のことを実行します:

- [enableQueryTracking](#) コマンドを使用してデータ取得統計をキャプチャし、ASO 集約を最適化します。
- 集約ビューの作成に使用されるユーザー・データ取得パターン(問合せ)をビジネス・プロセスがキャプチャするのに十分な時間を確保します。

適用対象

Planning、Planning モジュール、フリーフォーム、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

`epmautomate executeAggregationProcess ASO_CUBE_NAME [useQueryData=true|false] [includeAlternateRollups=disable|enable] [growthSizeRatio=VALUE]`。ここで:

- `useQueryData` は、最も適切な集約ビューのセットを選択するために、問合せトラッキングを使用して収集された、記録されている問合せデータを使用します。デフォルトは `false` です。
- `includeAlternateRollups` は、セカンダリ階層(デフォルトのレベルの使用方法)をビュー選択プロセスに含めます。デフォルトは `disable` です。
- `growthSizeRatio` はオプションで、サーバーが選択するビューを集約するための最大のキューブ拡張比率です。最大拡張が指定の比率に達すると、キューブ拡張は停止します。デフォルト設定では、拡張比率に制限なくキューブを拡張できます。

Note:

デフォルトの集約ビューを作成するには、オプションのパラメータを指定せずに、このコマンドを実行します。

例

- `enableQueryTracking` コマンドを使用してキャプチャされた問合せデータに基づいて集約ビューを作成します:

```
epmautomate executeAggregationProcess VISION_ASO useQueryData=true
includeAlternateRollups=enable
```
- デフォルトの集約ビューを作成します:

```
epmautomate executeAggregationProcess Vis1ASO
```

executeBurstDefinition

1つのデータ・ソースの単一ディメンションの複数のメンバーについてレポートまたはブックを実行するために必要なアーティファクト、POV およびその他の設定を指定するバースト定義を実行します。

適用対象

Narrative Reporting

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」を持つユーザーには、ACL を介して追加のセキュリティが割り当てられる必要があります

使用方法

`epmAutomate executeBurstDefinition ARTIFACT_NAME`。ここで、`ARTIFACT_NAME` は、バースト定義のパスと名前です。

例

```
epmAutomate executeBurstDefinition "library/Reports/Example BurstDef1"
```

executeReportBurstingDefinition

バースト定義を使用して、単一ディメンションの複数メンバーに対する単一のレポートまたはブックのバーストを実行し、メンバーごとに PDF 出力または静的な(Oracle Smart View for Office でリフレッシュできない) Excel 出力を発行します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmAutomate executeReportBurstingDefinition BURST_DEFINITION_NAME`
[`jobName=JOB_NAME`]。ここで:

- `BURST_DEFINITION_NAME` は、バースト定義のパスと名前です。
- `JOB_NAME` はオプションで、バースト定義の実行で使用するジョブの名前です。デフォルトは「バースト定義の実行」です。

例

```
epmAutomate executeReportBurstingDefinition /Library/MonthlySalesBurstDef
```

exportAccessControl

環境内で事前定義済役割を持つユーザーに関する情報が含まれ、各ユーザーの属性(名前や電子メールなど)と、アクセスに関する情報(グループ、チーム、組織への割り当てなど)が一覧表示されるユーザー詳細レポートを CSV または XLS ファイルにエクスポートします。

サンプル・レポート:

	A	B	C	D	E	F	G	H	I	J	K	L
	Name	User Login	Status	Teams	Email	Role	Workflow Roles	Preparer	Reviewer	Organizations	Power User Filter	Last Login
1	John Doe	john.doe@example.com	Available	AP Preparers	john.doe@example.com	Administrator		Yes	Yes			
2	Jane Doe	jane.doe@example.com	Available	AR Preparers	jane.doe@example.com	Power User		No	No			
3				AR Reviewers								
4	ats_power_user4	ats_power_user4	Available		example1@example.com	User		No	No			
5	ats_user1	ats_user1	Available		example2@example.com	User		No	No			
6	ats_view_user1	ats_view_user1	Available		view.user@example.com	Viewer		No	No			

[downloadFile](#) コマンドを使用してこのレポートをダウンロードできます。

適用対象

Account Reconciliation

必要な役割

サービス管理者

使用方法

```
epmAutomate exportAccessControl REPORT_NAME [reportFormat=XLS|CSV]。ここで:
```

- `REPORT_NAME` は、レポートを含むエクスポート・ファイルの名前です。
- `reportFormat` は省略可能であり、ファイル形式です。有効な値は `XLS` と `CSV` (デフォルト)です。

例

```
epmAutomate exportAccessControl aclreport.xls reportFormat=XLS
```

exportAppAudit

ローカル・コンピュータにダウンロードおよびアーカイブできる ZIP ファイルに、データ監査レコードをエクスポートします。最大 365 日間の監査情報を環境で使用できます。このレポートは、日数または日付範囲に対して生成できます。

出力 CSV ファイルの最初の文字は、バイト・オーダー・マーク(BOM)文字 `\ufeff` で、その後二重引用符で囲まれた暗号化されたアプリケーション識別子が続きます。CSV ファイル・ヘッダーは、アプリケーション識別子の後に続きます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

```
epmautomate exportAppAudit EXPORT_FILE_NAME [userNames=USER_NAMES]  
[nDays=Number_of_Days] [startDate=START_DATE endDate=END_DATE]  
[excludeApplicationId=true|false]。ここで:
```

- `EXPORT_FILE_NAME` は、エクスポートされた監査データを格納する ZIP ファイルの名前です。 [downloadFile](#) コマンドを使用して、ファイルを環境からダウンロードします。
- `userNames` はオプションで、カンマで区切られたユーザー・ログイン名のリストです。指定した場合は、これらのユーザーによって作成された監査データのみがエクスポートされます。すべてのユーザーの監査データをエクスポートする場合は、この値を指定しないでください。
- `nDays` はオプションで、エクスポートする監査レコードの日数を指定します。デフォルトは 7 日です。使用可能な値: `all` (過去 365 日間の使用可能な監査データをエクスポートす

る場合)、1、2、7、30、60 および 180。日付範囲に対してレポートを生成している場合は、この値を指定しないでください。

- startDate および endDate は、オプションで、監査レコードをエクスポートする開始日と終了日を識別します。これらの値は YYYY-MM-DD 形式で指定する必要があります。
 - 開始日は終了日より前の日付にする必要があります。レポートを生成するには、開始日と終了日の両方を指定する必要があります。
 - これらの日付は、将来日付にしないでください。
 - これらの日付は、オプション値 nDays が指定されている場合は無視されます
- excludeApplicationId はオプションで、アプリケーション識別子をエクスポート・ファイルに書き込むかどうかを識別します。デフォルトは false です。

ノート:

アプリケーション識別子が格納されていないエクスポート・ファイルからのデータは、Oracle Fusion Cloud Enterprise Performance Management 環境にインポートできません。

例

- 指定したユーザーの監査データを過去 30 日間分、アプリケーション識別子ありでエクスポートします:

```
epmautomate exportAppAudit auditData userNames=johnDoe,jane.doe@example.com ndays=30
```
- 指定したユーザーの監査データを過去 30 日間分、アプリケーション識別子なしでエクスポートします:

```
epmautomate exportAppAudit auditData userNames=johnDoe,jane.doe@example.com ndays=30 excludeApplicationId=true
```
- すべてのユーザーの 2024 年 8 月の監査データを、アプリケーション識別子ありでエクスポートします:

```
epmautomate exportAppAudit auditData startDate=2024-08-01 endDate=2024-08-31
```

exportAppSecurity

ローカル・ストレージに対してダウンロードできる CSV ファイルに、アーティファクト・レベルのアクセス権の割当(ACL)をエクスポートします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

`epmautomate exportAppSecurity EXPORT_FILE_NAME.CSV`。ここで、`EXPORT_FILE_NAME` は、エクスポートされたセキュリティ・データを格納するファイルの名前です。このファイルは送信ボックスに作成され、そこからコンピュータにダウンロードできます。

例

```
epmautomate exportAppSecurity app_security.CSV
```

exportARApplicationProperties

Account Reconciliation アプリケーション設定(レッドウッド・エクスペリエンス、テーマ、電子メール通知、およびビジネス・プロセス名に関連)、背景イメージ、およびロゴ・イメージを **JSON** ファイルにエクスポートして、それらを同じ環境または別の環境にインポートできるようにします。

このコマンドは、アプリケーションを本番環境からテスト環境にインポートするときに役立ちます。アプリケーションの設定が本番環境とテスト環境で異なる場合は、本番環境からアプリケーションをインポートする前にテスト環境からそれらをエクスポートしてからそれらの設定をテスト環境にインポートし、元の設定を維持できます。

適用対象

Account Reconciliation

必要な役割

サービス管理者

使用方法

```
epmautomate exportARApplicationProperties FILE_NAME  
[Properties=PROPERTIES_TO_EXPORT]
```

- `FILE_NAME` は、エクスポートされたプロパティ値を格納する **JSON** ファイルの名前です。
[downloadFile](#) コマンドを使用してエクスポート・ファイルをダウンロードできます。
[uploadFile](#) コマンドを使用してターゲット環境にアップロードしてから、
[importARApplicationProperties](#) コマンドを実行してこれらの設定をターゲット環境に復元します。
- `Properties` は省略可能であり、エクスポートするプロパティのカンマ区切りのリストです。次のプロパティの一部またはすべてをエクスポートできます。このプロパティを省略すると、次のすべてのプロパティがエクスポートされます。
 - `Theme`: 環境で使用されている表示テーマをエクスポートします。
 - `EmailNotification`: 環境で定義された電子メール通知設定をエクスポートします。
 - `DisplayBusinessProcessName`: 環境内のページにビジネス・プロセス名を表示するかどうかをエクスポートします。
 - `RedwoodExperience`: 環境のレッドウッド・エクスペリエンス設定をエクスポートします。
 - `BackgroundImage`: 環境で使用されている背景イメージをエクスポートします。
 - `LogoImage`: 環境で使用されているロゴ・イメージをエクスポートします。

例

環境から電子メール通知とレッドウッド・エクスペリエンス設定、ロゴ・イメージのみをエクスポートします。

```
epmautomate exportARApplicationProperties myProp.JSON  
Properties=EmailNotification,RedwoodExperience,LogoImage
```

exportBackgroundImage

Account Reconciliation 環境で使用されている背景イメージを JPG ファイルにエクスポートして、別の環境にインポートできるようにします。

適用対象

Account Reconciliation

必要な役割

サービス管理者

使用方法

epmautomate exportBackgroundImage *IMAGE_NAME*.jpg。ここで、*IMAGE_NAME* は、背景イメージ・ファイルの名前です。

[downloadFile](#) コマンドを使用して、イメージ・ファイルをダウンロードできます。[uploadFile](#) コマンドを使用してターゲット環境にアップロードしてから、[importBackgroundImage](#) コマンドを実行してインポートします。

例

```
epmautomate exportBackgroundImage corpImage.jpg
```

exportCellLevelSecurity

セルレベルのセキュリティ設定をビジネス・プロセスから ZIP ファイルにエクスポートします。このファイルは、[downloadFile](#) コマンドを使用してローカル・コンピュータにダウンロードできます。

適用対象

Planning、Planning モジュール、フリーフォーム、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

```
epmautomate exportCellLevelSecurity FILE_NAME.ZIP [names=SECURITY_RECORD_NAMES]。  
ここで:
```

- *FILE_NAME* は、セルレベルのセキュリティ情報を含む Excel ファイルを保持するために作成される ZIP ファイルの名前です。

- names は、オプションであり、アプリケーションのセルレベルのセキュリティ定義のカンマ区切りのリストを識別します。このオプションが指定されていない場合、アプリケーションのすべてのセルレベルのセキュリティ定義がエクスポートされます。

例

- 特定のセルレベルのセキュリティ定義のエクスポート**
epmautomate exportCellLevelSecurity ExportCLSDRecordsFile.zip
names=CLSDAccountPeriod,CLSDEntityPeriod,CLSDProductPeriod
- すべてのセルレベルのセキュリティ定義のエクスポート**
epmautomate exportCellLevelSecurity ExportCLSDRecordsFile.zip

exportConsolidationJournals

Financial Consolidation and Close で定義したジョブを使用して、連結仕訳をエクスポートします。

適用対象

Financial Consolidation and Close

必要な役割

サービス管理者

使用方法

epmautomate exportConsolidationJournals jobName [fileName=FILE_NAME]。ここで

- jobName は、Financial Consolidation and Close で作成された仕訳のエクスポート・ジョブの名前です。
- fileName はオプションで、仕訳のエクスポート先の.JLF ファイルの名前です。
[downloadFile](#) コマンドを使用して、このファイルをローカル・コンピュータにダウンロードします。

例

```
epmautomate exportConsolidationJournals "JEXPORT1" fileName=Export_Test.jlf
```

exportData

export data タイプのジョブで指定されたデータのエクスポート設定(ファイル名を含む)を使用して、アプリケーション・データを ZIP ファイルにエクスポートします。

エクスポートされたデータ・ファイルはデフォルトのダウンロード場所に格納されます。そこからコンピュータにダウンロードできます。受信ボックスまたは送信ボックス・エクスプローラを使用して、エクスポートされたファイルの詳細を確認します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

`epmautomate exportData JOB_NAME [FILE_NAME]`。ここで、`JOB_NAME` はアプリケーションに定義されたジョブ名です。`FILE_NAME` は、データのエクスポート先の ZIP ファイルの名前です (オプション)。

例

```
epmautomate exportData dailydataexport dailyData.zip
```

exportDataManagement

データ管理レコードを環境から ZIP ファイルにエクスポートします。

このコマンドは、参照整合性を失うことなくデータをインポートできるように、ID 列を含めた設定およびステージング表データの完全なセットを ZIP ファイルにエクスポートします。

エクスポートされたファイル(たとえば、`dataFile.zip`)は、送信ボックスに保存されます。エクスポートされたファイルは、[downloadFile](#) コマンド(たとえば、`epmAutomate downloadFile outbox/dataFile.zip`)を使用してダウンロードできます。この ZIP ファイルを使用して、[importDataManagement](#) コマンドでデータをインポートできます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

`epmautomate exportDataManagement FILE_NAME.zip`。ここで、`FILE_NAME` は、データのエクスポート先となる ZIP ファイルの名前です。

例

```
epmautomate exportDataManagement dataFile.zip
```

exportDimension

Oracle Fusion Cloud Enterprise Data Management アプリケーションからステージング領域内のファイルに(またはオプションで、接続で定義されたターゲット環境に)ディメンションをエクスポートします。

適用対象

Oracle Enterprise Data Management Cloud

必要な役割

- サービス管理者
- ユーザー(データ・マネージャ 権限が必要)

使用方法

`epmautomate exportDimension APPLICATION DIMENSION FILE_NAME [connection=NAME]`。ここで:

- `APPLICATION` は、Oracle Enterprise Data Management Cloud アプリケーションの名前です
- `DIMENSION` は、アプリケーションのディメンションの名前です
- `FILE_NAME` は、エクスポートされたデータを格納するためのファイル(ファイルにエクスポートする場合は CSV、Oracle Financials Cloud にエクスポートする場合は ZIP)の名前です。`connection` パラメータ値が設定されていない場合、このファイルはステージング領域に作成されます。このファイルは、`downloadFile` コマンドを使用してローカル・コンピュータにダウンロードするか、`copyFileFromInstance` コマンドを使用して別の Oracle Fusion Cloud Enterprise Performance Management 環境にコピーできます。
- `connection=NAME` はオプションで、Oracle Enterprise Data Management Cloud で定義された接続名(インスタンスの場所)を指定します。指定すると、エクスポート・ファイルはターゲット環境(Cloud EPM の場合は受信ボックス、Oracle Financials Cloud の場合はデフォルトのアップロードの場所)にアップロードされます。

ノート:

接続の定義で指定された資格証明には、ターゲット環境に書き込むためのアクセス権が含まれている必要があります。

例

- Oracle Enterprise Data Management Cloud のステージング領域へのエクスポート:
`epmautomate exportDimension USOperations Entity EntityData.CSV`
- エクスポートおよび Oracle Financials Cloud へのアップロード:
`epmautomate exportDimension USOperations Entity EntityData.zip Connection=ora_fusion_gl`
- エクスポートおよびターゲット Cloud EPM の受信ボックスへのアップロード:
`epmautomate exportDimension USOperations Entity EntityData.CSV Connection=EPM_cloud_pln`

exportDimensionMapping

マッピング・ルール・ファイルを作成するために、場所に対する特定の Oracle Fusion Cloud Enterprise Data Management ディメンションのマッピング・ルールをエクスポートし、オプションで、エクスポートされたファイルを別の Oracle Fusion Cloud Enterprise Performance Management 環境の Data Management の受信ボックスにアップロードします。

適用対象

Oracle Enterprise Data Management Cloud

必要な役割

- サービス管理者
- ユーザー(データ・マネージャ 権限付き)

使用方法

`epmautomate exportDimensionMapping APPLICATION DIMENSION LOCATION FILE_NAME [connection=NAME]`。ここで:

- `APPLICATION` は、**Oracle Enterprise Data Management Cloud** アプリケーションの名前です
- `DIMENSION` は、アプリケーションのディメンションの名前です
- `LOCATION` は、マッピング・ルールがエクスポートされる特定の場所です。
- `FILE_NAME` は、エクスポートしたマッピングを格納する **CSV** ファイルの名前です。
`connection` パラメータを設定していない場合、このファイルはステージング領域に作成されます。このファイルは、[downloadFile](#) コマンドを使用してローカル・コンピュータにダウンロードするか、[copyFileFromInstance](#) コマンドを使用して別の **Cloud EPM** 環境にコピーできます。
- `connection=NAME` はオプションで、**Oracle Enterprise Data Management Cloud** で定義された接続名(インスタンスの場所)を指定します。指定すると、コマンドにより、エクスポートされたファイルがターゲット環境のデフォルトのアップロードの場所にアップロードされます。

ノート:

接続で指定された資格証明には、ターゲット環境に書き込むためのアクセス権が含まれている必要があります。

例

- ステージング領域へのエクスポート:
`epmautomate exportDimensionMapping USOperations Entity Loc1 Loc1Mappings.CSV`
- エクスポートおよびターゲット **Cloud EPM** 環境へのアップロード:
`epmautomate exportDimensionMapping USOperations Entity Loc1 Loc1Mappings.CSV Connection=EPM_cloud_pln`

exportEJournals

転記の準備が整ったエンタープライズ仕訳を **Financial Consolidation and Close** から **ZIP** ファイルにエクスポートします。次に、このファイルを使用して仕訳データを **ERP** システムにインポートできます。

仕訳をエクスポート・ファイルにエクスポートした後、このコマンドは、エクスポートされた各仕訳の転記ステータスを **Ready To Post** から **Post In Progress** に更新します。

適用対象

Financial Consolidation and Close

必要な役割

サービス管理者

使用方法

epmautomate exportEJournals *FILE_NAME*.zip [year=*YEAR*] [period=*PERIOD*]
[OPERATION=posting|validation]。ここで:

- *FILE_NAME* は、仕訳エクスポート CSV ファイルをアーカイブする ZIP ファイルを識別します。各仕訳に対して 1 つの CSV ファイル(名前の形式は *YEAR_PERIOD_JOURNALID_YYYYDDMMHHMSS.csv*)が生成され、それらを圧縮してこの ZIP ファイルが作成されます。
- *YEAR* はオプションで、仕訳データをエクスポートするデータ収集年です。指定されていない場合、すべての年のデータがエクスポートされます。
- *PERIOD* はオプションで、仕訳データをエクスポートするデータ収集期間です。データ収集年が指定されている場合にのみ設定できます。値が指定されていない場合、すべての期間のデータがエクスポートされます。

Note:

YEAR と *PERIOD* が指定されていない場合は、すべての年と期間について転記ステータスが Ready To Post であるすべての仕訳がエクスポートされます。

- Operation はオプションで、実行する操作を示します。使用可能な値は posting および validation です。デフォルトは posting です。
operation=posting では、仕訳転記の結果を格納した CSV ファイルを提供します。このファイルには、「年」、「期間」、「仕訳 ID」、「転記ステータス」(転記済/失敗)、「メッセージ」、「エラー・コード」および「エラー・メッセージ」の列が含まれます。「メッセージ」、「エラー・コード」および「エラー・メッセージ」列はオプションです。仕訳の転記ステータスは、ファイルで提供されたステータスに基づいて更新されます。転記ステータスが **転記進行中の仕訳のみ**が更新されます。

operation = validation では、仕訳検証の結果を格納した CSV ファイルを生成します。このファイルには、「年」、「期間」、「仕訳 ID」、「転記ステータス」(有効/失敗)、「メッセージ」、「エラー・コード」および「エラー・メッセージ」の列が含まれます。「メッセージ」、「エラー・コード」および「エラー・メッセージ」列はオプションです。仕訳の検証ステータスは、ファイルで提供されたステータスに基づいて更新されます。検証ステータスが **検証進行中の仕訳のみ**が更新されます。

例

- すべての年と期間の仕訳データをエクスポートします:
epmautomate exportEJournals Journal_Export.zip
- 特定の年の仕訳データをエクスポートします:
epmautomate exportEJournals Journal_Export.zip year=2020
- 特定の年と期間の組合せの仕訳データをエクスポートします:
epmautomate exportEJournals Journal_Export.zip year=2024 period=March
- 検証操作を実行して、特定の年と期間の組合せの仕訳データをエクスポートします:
epmautomate exportEJournals Journal_Export.zip year=2024 period=March
Operation=validation

exportEssbaseData

アプリケーション・キューブからアーカイブにデータをエクスポートします。レベル 0 のデータ (ASO キューブと BSO キューブ) のみ、またはキューブ (BSO キューブ) 内のすべてのデータをエクスポートできます。

エクスポートされたアーカイブを使用して Oracle Essbase データのパターンを分析し、たとえばパフォーマンスの向上に役立てます。

Note:

- アーカイブに含まれるデータ・ファイルには .txt 拡張子が付きます。
- スマートリストおよびセル・テキスト・オブジェクトは UUID としてエクスポートされます。

データ・エクスポートを実行すると、キューブは読み取り専用モードになり、エクスポート操作の進行中はすべての書き込みアクティビティが防止されます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

epmautomate exportEssbaseData CUBE_NAME FILE_NAME [level=0|All]。ここで:

- *CUBE_NAME* は、データのエクスポート元のキューブを指定します。
- *FILE_NAME* は、エクスポートされたデータを含む ZIP ファイルの名前です。このアーカイブをダウンロードするには、[downloadFile](#) コマンドを実行します。
- level (オプション) は、エクスポートするデータのレベルを指定します。デフォルトは 0 です。
 - **ASO キューブ:** レベル 0 のデータをエクスポートするには、0 を指定します。All オプションは使用できません。
 - **BSO キューブ:** レベル 0 のデータをエクスポートするには 0 を指定し、すべてのデータをエクスポートするには All を指定します。

例

- BSO キューブからすべてのデータをエクスポート:
epmautomate exportEssbaseData Report1 Report1_all_data.zip level=All
- キューブからレベル 0 のデータをエクスポート:
epmautomate exportEssbaseData Plan1 Plan1_lvl0_data.zip

exportJobConsole

ジョブ・コンソール・レコードを CSV ファイルにエクスポートし、エクスポート ZIP ファイルを作成します。

出力 CSV ファイルの最初の文字は、バイト・オーダー・マーク(BOM)文字\ufeffで、その後二重引用符で囲まれた暗号化されたアプリケーション識別子が続きます。CSV ファイル・ヘッダーは、アプリケーション識別子の後に続きます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

```
epmautomate exportJobConsole FILE_NAME.zip [nDays=NUMBER_OF_DAYS]
[jobtypes=JOB_TYPE] [jobStatusCodes=STATUS_CODE] [exportErrorDetails=true|false]
[excludeApplicationId=true|false]。ここで:
```

- `FILE_NAME` は、エクスポートされたジョブ・コンソール・レコードを保存する ZIP ファイルの名前です。 `downloadFile` コマンドを使用して、このファイルを環境からダウンロードします。
- `nDays` はオプションで、エクスポートするジョブ・コンソール・レコードの日数を指定します。使用可能な値: `all` (すべて小文字で指定。使用可能なすべてのジョブ・コンソール・レコードをエクスポートする場合)、1、2、7、30 および 60。デフォルトは 7 です。
- `jobTypes` はオプションで、コンソール・レコードがエクスポートされるジョブ・コードのカンマ区切りのリストです。デフォルトは `Rules` です。有効な値は次のとおりです。
 - `all` (すべて小文字で指定)
 - `RULES`
 - `RULESET`
 - `CLEAR_CELL_DETAILS`
 - `COPY_DATA`
 - `INVALID_INTERSECTION_RPT`
 - `COPY_VERSIONS`
 - `CONTENT_UPGRADE`
 - `PLAN_TYPE_MAP`
 - `IMPORT_DATA`
 - `EXPORT_DATA`
 - `EXPORT_METADATA`
 - `IMPORT_METADATA`

- CUBE_REFRESH
- CLEAR_CUBE
- ADMIN_MODE
- COMPACT_CUBE
- RESTRUCTURE_CUBE
- MERGE_DATA_SLICES
- OPTIMIZE_AGGREGATION
- SECURITY_IMPORT
- SECURITY_EXPORT
- AUDIT_EXPORT
- JOB_CONSOLE_EXPORT
- SORT_MEMBERS
- SMART_PUSH
- IMPORT_EXCHANGE_RATES
- `jobStatusCodes` はオプションで、レコードをエクスポートするジョブ・ステータス・コードのカンマ区切りのリストです。デフォルトは 2 (正常に完了) です。使用可能な値は次のとおりです:
 - all (すべて小文字で指定)あらゆるステータスのすべてのジョブを対象
 - 1 - 処理中
 - 2 - 正常に完了
 - 3 - エラーで失敗
 - 4 - 不明なステータスで完了
 - 5 - しきい値違反ステータスで完了
 - 6 - 取消保留中
 - 7 - 取消し済
 - 8 - エラーありで完了
 - 9 - 警告ありで完了
- `exportErrorDetails` はオプションで、この値が `true` に設定されている場合は、失敗したジョブまたはエラーが報告されたジョブの詳細をログ・ファイルにエクスポートします。このエラー・ログ・ファイルは、出力 ZIP ファイルに含まれます。デフォルトは `false` です。この値が `true` に設定されている場合は、次のステータスのジョブのステータス詳細がエクスポートされます。
 - エラーで失敗
 - 不明なステータスで完了
 - しきい値違反ステータスで完了
 - エラーありで完了
 - 警告ありで完了

- `excludeApplicationId` はオプションで、アプリケーション識別子をエクスポート・ファイルに書き込むかどうかを識別します。デフォルトは `false` です。

 **Note:**

アプリケーション識別子が格納されていないエクスポート・ファイルからのデータは、Oracle Fusion Cloud Enterprise Performance Management 環境にインポートできません。

例

- 使用可能なすべてのジョブ・コンソール・レコードをエクスポートします:
`epmautomate exportJobConsole jobs.zip nDays=all jobTypes=all jobStatusCodes=all`
- 使用可能なすべてのルール・ジョブ・コンソール・レコードをエクスポートします:
`epmautomate exportJobConsole jobs.zip nDays=all jobStatusCodes=all`
- アプリケーション識別子なしで、使用可能なすべてのルール・ジョブ・コンソール・レコードをエクスポートします:
`epmautomate exportJobConsole jobs.zip nDays=all jobStatusCodes=all excludeApplicationId=true`
- 過去 14 日間に正常に終了したルール・ジョブのレコードのみをエクスポートします:
`epmautomate exportJobConsole jobs.zip nDays=14`
- 過去 7 日間に実行され、エラーで失敗またはエラーありで完了した、メタデータのインポート・ジョブおよびキューブのクリア・ジョブのコンソール・レコードとエラーをエクスポートします:
`epmautomate exportJobConsole jobs.zip jobtypes=IMPORT_METADATA,CLEAR_CUBE jobStatusCodes=3,8 exportErrorDetails=true`

exportLibraryArtifact

Narrative Reporting ライブラリ・アーティファクトをエクスポートします。オプションで、レポート・アーティファクトの場合のみ、エクスポートを **Financial Consolidation and Close**、**Planning**、**Planning** モジュールまたは **Tax Reporting** にインポートできる LCM ファイルに変換できます。

エクスポートの完了時に、[downloadFile](#) コマンドを使用して、エクスポートおよびエラー・ファイルをローカル・コンピュータにダウンロードします。

適用対象

Narrative Reporting

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」を持つユーザーには、ACL を介して追加のセキュリティが付与される必要があります

使用方法

epmautomate exportLibraryArtifact *ARTIFACT_PATH* *EXPORT_FILE* [exportFormat=Native|File|LCM] [applicationName=*APP_NAME*] [errorFile=*ERROR_FILE.txt*]. ここで:

- *ARTIFACT_PATH* は、**Narrative Reporting** ライブラリのアーティファクトの場所です。
- *EXPORT_FILE* は、アーティファクトのエクスポート先のファイルの一意の名前です。
- exportFormat はオプションで、次のいずれかを使用します:
 - Native は、他の **Narrative Reporting** 環境で使用できる zip ファイルとしてアーティファクトをエクスポートします。これがデフォルト値です。
 - File は、**Narrative Reporting** 内で使用できる元のバイナリ形式(PDF、DOCX、Zip、JPEG など)でファイルをエクスポートします。このパラメータは、バイナリ・ファイルのエクスポートにのみ使用できます。レポート・アーティファクトと一緒に使用しないでください。
 - LCM は、レポートを移行で使用される形式に変換し、**Financial Consolidation and Close、Planning、Planning** モジュールまたは **Tax Reporting** 環境にインポートできる ZIP ファイルにエクスポートします。
- applicationName はオプションで、レポートのインポートを予定しているターゲット・アプリケーションの名前です。この値は、exportFormat パラメータの値として LCM を使用している場合にのみ必要です。
- errorFile はオプションで、エクスポート関連のエラーを格納するテキスト・ファイルの一意の名前です。

例

- レポートを別の **Narrative Reporting** 環境にインポートできるように、ネイティブ形式でエクスポートします:

```
epmautomate exportLibraryArtifact "Library/Samples/Sample Report 1"
exp_SampleReport1.doc errorFile=export_errors.txt
```
- スプレッドシートを元のバイナリ形式でエクスポートします:

```
epmautomate exportLibraryArtifact "Library/Spreadsheets/Sheet1.xlsx"
exp_Sheet1.xlsx exportFormat=File errorFile=export_errors.txt
```
- レポートをエクスポートし、**Financial Consolidation and Close、Planning、Planning** モジュールまたは **Tax Reporting** にインポートするためにフォーマットします:

```
epmautomate exportLibraryArtifact "Library/Samples/Sample Report 1"
exp_SampleReport1.zip exportFormat=LCM applicationName=Vision
errorFile=report_exp_errors.txt
```

exportLibraryDocument

レポート・ライブラリで使用可能なドキュメントをファイルにエクスポートします。

[downloadFile](#) コマンドを使用して、エクスポートされたファイルをローカル・コンピュータにダウンロードできます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

```
epmautomate exportLibraryDocument ARTIFACT_PATH [jobName=JOB_NAME]
[exportFile=FILE_NAME] [exportFormat=file|zip] [errorFile=FILE_NAME.log]
[overWrite=true|false]。ここで:
```

- `ARTIFACT_PATH` は、レポート・ライブラリ内のドキュメントの場所または **Universally Unique Identifier (UUID)** です。複数のドキュメントをエクスポートするには、UUID またはドキュメントの場所のカンマ区切りリストを指定します。
- `jobName` は、オプションで、ドキュメントのエクスポートに使用されるライブラリ・ドキュメント・エクスポート・ジョブの名前です。デフォルトのジョブ名は Copy Artifact From Library です。
- `EXPORT_FILE` は、オプションで、エクスポート・ファイルの一意の名前です。`exportFormat=ZIP` を使用している場合、生成される ZIP ファイルはこの名前を使用します。この値には、`.zip` 拡張子を使用してください。

`exportFormat=file` を使用している場合:

- 1つのドキュメントをエクスポートする場合: 指定したファイル拡張子がドキュメントの拡張子と一致する場合、エクスポート・ファイルはこの名前で作成されます。
- 多数のドキュメントをエクスポートする場合: コマンドはこの値を無視し、元の名前を使用して各ドキュメントをエクスポートします。
- `exportFormat` はオプションで、次のいずれかを使用します:
 - `file` では、ライブラリで使用できる元のバイナリ形式(PDF、DOCX、Zip、JPEG など)でドキュメントをエクスポートします。エクスポートされたファイルは、それぞれ元の名前を使用して送信ボックスに作成されます。これがデフォルト値です。
 - `zip` では、エクスポートされたすべてのドキュメントを元のバイナリ形式で格納した ZIP ファイルが作成されます。`exportFile` オプション・パラメータに値が指定されていない場合は、コマンドを実行しているユーザーのユーザー名およびタイムスタンプを使用して、`zip` ファイル名が自動的に生成されます。ファイル名の形式は、`username_exported_artifacts_yyyyMMddHHmm.zip` で、たとえば、`JDoe_exported_artifacts_202410201559.zip` のようになります。
- `errorFile` はオプションで、エクスポート関連のエラーを格納するファイルの一意の名前です。この値を指定していない場合、エラー・ファイルは作成されません。
- `overwrite` はオプションで、デフォルトのダウンロード場所に現在ある同じ名前のファイルを上書きするかどうかを制御します。デフォルトは `false` です。これは、送信ボックスに同じ名前のファイルが存在する場合、コマンドが失敗となることを意味します。

例

- 1つのドキュメントをエクスポートする場合:

```
epmautomate exportLibraryDocument Library/folder1/WeeklySales.html
jobName="Copy Weekly Sales" exportFile=WeeklySales.zip
errorFile=WeeklySalesError.log overWrite=true exportFormat=zip
```
- 複数のドキュメントをエクスポートする場合

```
epmautomate exportLibraryDocument Library/folder1/WeeklySales.html,Library/  
folder2/WeeklySalesReport.pdf jobName="Copy Weekly Sales"  
exportFile=WeeklySales.zip errorFile=WeeklySalesError.log overWrite=true  
exportFormat=zip
```

exportLogoImage

Account Reconciliation ビジネス・プロセスで使用される企業ロゴを **JPG** ファイルにエクスポートして、別の環境にインポートできるようにします。

適用対象

Account Reconciliation

必要な役割

サービス管理者

使用方法

epmautomate exportLogoImage *IMAGE_NAME*.jpg。ここで、*IMAGE_NAME* は、ロゴ・イメージ・ファイルの名前です。
エクスポートされたロゴ・ファイルは、[downloadFile](#) コマンドを使用してダウンロードできます。[uploadFile](#) コマンドを使用してターゲット環境にアップロードしてから、[importLogoImage](#) コマンドを実行します。

例

```
epmautomate exportLogoImage corpLogo.jpg
```

exportMapping

特定のディメンションまたは場所のマッピング・ルールをエクスポートして、マッピング・ルール・ファイルを作成します。マッピングをエクスポートする、受信ボックス内のファイルの名前と場所(たとえば、inbox/exportedAccountMap.txt または inbox/france sales/exportedAccountMap.txt)を指定する必要があります。

[downloadFile](#) コマンドを使用して、エクスポートされたマッピング・ファイルをローカル・コンピュータにダウンロードします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

```
epmautomate exportMapping DIMENSION_NAME|ALL FILE_NAME LOCATION。ここで:
```

- `DIMENSION_NAME|ALL` は、マッピングのエクスポート元のソース・ディメンションです。マッピングのエクスポート元のディメンション名を指定するか、ある場所のすべてのディメンションからマッピングをエクスポートする場合は `ALL` を指定します。
- `FILE_NAME` は、送信ボックス内のマッピング・ファイルと場所の一意名です。
- `LOCATION` は、マッピング・ルールがエクスポートされるデータ管理の場所です。

例

- `epmautomate exportMapping Account inbox/exportedAccountMap.txt "France Sales"`
- `epmautomate exportMapping ALL "inbox/france sales/exportedAccountMap.txt" "France Sales"`

exportMetadata

`export metadata` タイプのジョブで指定された設定を使用して、メタデータをファイルにエクスポートします。エクスポートされたデータを含むファイルはデフォルトのダウンロード場所に格納されます。そこからローカル・コンピュータにダウンロードできます。

オプションで、エクスポートされたデータのファイル名を指定できます。これはデフォルト・ファイル名(メタデータのエクスポートに使用されるジョブ名)より優先されます。メタデータは ZIP ファイルとしてのみエクスポートされます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

`epmautomate exportMetadata JOB_NAME [FILE_NAME]`。ここで、`JOB_NAME` は、アプリケーションに定義されたジョブ名です。`FILE_NAME` は、メタデータのエクスポート先の ZIP ファイルの名前です。

[downloadFile](#) コマンドを使用して、このファイルをローカル・サーバーにダウンロードします。

例

```
epmautomate exportMetadata dailyAccountexport Accountexport.ZIP
```

exportOwnershipData

出資比率データを、エンティティからカンマ区切りの CSV ファイルにエクスポートします。

Financial Consolidation and Close によって移入されたデフォルトの出資比率データは、エクスポート・ファイルに含まれません。エクスポート・ファイルには、デフォルト設定を上書きするためにユーザーが入力したデータのみが含まれます。

適用対象

Financial Consolidation and Close および Tax Reporting

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー

使用方法

`epmautomate exportOwnershipData Entity Scenario Year Period FILE_NAME`。ここで:

- `Entity` は、データのエクスポート元のエンティティ名です。
- `Scenario` は、データのエクスポート元のシナリオです。
- `Year` は、データのエクスポート元の年です。
- `Period` は、データのエクスポート元の年の期間です。
- `FILE_NAME` は、データのエクスポート先の **CSV** ファイルの名前です。 [downloadFile](#) コマンドを使用して、このファイルをローカル・サーバーにダウンロードします。

例

```
epmautomate exportOwnershipData FCCS_TotalActual FY18 Dec exportfile.csv
```

exportQueryResults

アプリケーションに定義された問合せを実行し、結果をテキスト・ファイルにエクスポートします。

問合せの結果ファイルは `profitoutbox` に格納され、 [downloadFile](#) コマンドを使用するか、 **Profitability and Cost Management** のファイル・エクスプローラを使用することでダウンロードできます。

適用対象

Profitability and Cost Management

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

使用方法

```
epmautomate exportQueryResults APPLICATION_NAME fileName=FILE_NAME  
[fileOutputOptions=ZIP_ONLY|ZIP_AND_TEXT|TEXT_ONLY] [queryName=QUERY_NAME]  
[exportOnlyLevel0Flg=true|false] [roundingPrecision=2] [dataFormat=NATIVE|  
COLUMNAR] [memberFilters=JSON_FILTER] [includeHeader=true|false]  
[delimiter="DELIMITER"] [keepDuplicateMemberFormat=true|false]。ここで:
```

- `APPLICATION_NAME` は、問合せを実行する **Profitability and Cost Management** アプリケーションの名前です。

- `fileName` は、問合せ結果が格納されるファイルの名前です。このパラメータ値は、`queryName` パラメータ値が指定されていない場合に必要です。`queryName` パラメータ値が指定されている場合はオプションです。この場合、問合せ名が問合せ結果ファイルの名前として使用されます。
指定するデータ・フォーマットによって、出力ファイルの形式が決まります。`dataFormat=NATIVE` (デフォルト)を使用すると、エクスポート・プロセスでテキスト・ファイルが作成されます。`dataFormat=COLUMNAR`を使用すると、エクスポート・プロセスで連番付きの複数のテキスト・ファイルが作成され、Zip ファイルに圧縮されます。
- `fileOutputOptions` はオプションで、問合せ結果ファイルの出力形式を識別します。デフォルトは `ZIP_ONLY` で、`fileName` パラメータの値が指定されているかどうかに応じて `fileName.ZIP` または `queryName.ZIP` が作成されます。その他のオプションは、出力ファイルをテキスト・ファイルとして作成する `TEXT_ONLY` と、テキスト・ファイルと zip ファイルの両方を生成する `ZIP_AND_TEXT` です。
- `queryName` は、アプリケーションに定義された問合せを指定するオプション・パラメータです。空白が含まれる問合せ名は、二重引用符で囲む必要があります。アプリケーションに属するすべての Oracle Essbase データをエクスポートする場合は、問合せ名を指定しないでください。

次の条件がある場合、このコマンドによって空のデータ・ファイルが作成されることがあります。

- 問合せの形式が正しくないためにデータを取得できない
- 問合せによって生成されるデータが多すぎる。このシナリオでは、問合せの対象範囲を絞り込んで取得データを減らすことを検討するか、小さな問合せに分割してください
Profitability and Cost Management の管理の Oracle Profitability and Cost Management Cloud 問合せの管理に関する項を参照してください。
- `exportOnlyLevel0Flg` はオプションで、問合せでレベル 0 データのみを取得するかどうかを指定します。このパラメータ値はすべて小文字で指定します。
問合せ名を省略してすべてのアプリケーション・データをエクスポートしている場合、このパラメータは無視されます。
- `roundingPrecision` はオプションで、問合せ結果をエクスポートするとき使用する小数点以下桁数(丸め処理精度)を指定します。`queryName` が指定されている場合にのみ適用されます。デフォルトは 2 です。
- `dataFormat` はオプションで、出力形式を識別します。有効な値は次のとおりです。
 - `NATIVE` は、問合せ結果を Essbase ネイティブ形式のデータとして保持します。これがデフォルト値です。
 - `COLUMNAR` は、Essbase ネイティブ形式のデータを変換し、他のアプリケーションに簡単に解釈してインポートできるように列に並べます。
このオプションは、すべての Essbase データをエクスポートし、`queryName` パラメータ値を無視します。`memberFilters` パラメータ値を設定することで、データをフィルタできます。

ノート:

コマンドでは、`dataFormat` が `COLUMNAR` に指定されている場合にのみ、次のオプションのパラメータが考慮されます。

- memberFilters はオプションで、JSON 形式の文字列を受け入れて、ディメンションとレベル 0 のメンバーでフィルタします。例: {"Dim1\":[\"Mem1\"],\"Dim2\":[\"Mem21\", \"Mem22\"]}"
- includeHeader はオプションで、列ヘッダーとしてディメンション名を追加します。列ヘッダーを除外するには、この値を false に設定します。デフォルトは true です。
- delimiter はオプションで、問合せ結果ファイルのディメンション・メンバーの区切りに使用する区切り文字を識別します。区切り文字は二重引用符で囲む必要があります。デフォルトはスペース(" ")です。
- keepDuplicateMemberFormat はオプションで、メンバー・フォーマットを Essbase の重複メンバー・フォーマット(例: [Account]@[Account1])で印刷するかどうかを指定します。メンバー名のみを印刷するには、この値を false に設定します。デフォルトは true です。

例

- すべてのアプリケーション・データのエクスポート:
epmautomate exportQueryResults BksML12 fileName="BksML12_MyQuery1.txt"
fileOutputOptions=TEXT_ONLY
- 特定の問合せ結果のエクスポート:
epmautomate exportQueryResults BksML12 queryName="My Product Query"
roundingPrecision=3
- NATIVE データ・フォーマットでのレベル 0 データのエクスポート:
epmautomate exportQueryResults BksML30 fileName="BksML30_ExportLevel0-Data"
fileOutputOptions=ZIP_AND_TEXT exportOnlyLevel0Flg=true
- 単一のディメンションおよび単一のメンバー・フィルタによる COLUMNAR データ・フォーマットでのレベル 0 データのエクスポート:
epmautomate exportQueryResults BksML30 fileName="BksML30_Level0-Data"
dataFormat="COLUMNAR" memberFilters="{\"Period\":[\"December\"]}"
includeHeader="true" delimiter="," roundingPrecision="3"
- 単一のディメンションおよび複数のメンバー・フィルタによる COLUMNAR データ・フォーマットでのレベル 0 データのエクスポート:
epmautomate exportQueryResults BksML30 fileName="BksML30_Level0-Data"
dataFormat="COLUMNAR" memberFilters="{\"Period\":[\"November\", \"December\"]}"
includeHeader="true" delimiter="," roundingPrecision="3"
- 複数のディメンションおよび複数のメンバー・フィルタによる COLUMNAR データ・フォーマットでのレベル 0 データのエクスポート:
epmautomate exportQueryResults BksML30 fileName="BksML30_Level0-Data"
dataFormat="COLUMNAR" memberFilters="{\"Year\":[\"2016\"],\"Period\":[\"November\", \"December\"]}" includeHeader="true" delimiter=","
roundingPrecision="3"

exportSnapshot

以前実行したエクスポート操作を繰り返して Migration コンテンツのスナップショットを作成します。

「移行」,を使用して、必要なアーティファクトを選択してスナップショット(たとえば January16FullApp)にエクスポートします。このコマンドのスナップショット名を使用して、後でエクスポート操作を繰り返すと、元のエクスポート操作時に選択されたアーティファクトのみがエクスポートされます。移行の管理のアーティファクトおよびアプリケーションのエクスポートを参照してください。

- Planning、Planning モジュールおよびフリーフォーム・アプリケーション・スナップショットに含まれていないものは次のとおりです:
 - 監査データ
 - ジョブ・コンソール・データ

監査データおよびジョブ・コンソール・データをターゲット環境にコピーする場合は、[cloneEnvironment](#) コマンドまたは環境のクローニング機能を使用します。

- スナップショットには、データ管理のステージング表データは含まれません。このデータをインポートするには、[exportDataManagement](#) コマンドと [importDataManagement](#) コマンドまたはデータ管理システム・メンテナンス・スクリプト・インタフェースを使用します。[cloneEnvironment](#) コマンドまたは環境のクローニング機能を使用して、データ管理ステージング表データを含む環境の同一コピーを作成できます。

[downloadFile](#) コマンドを使用して、エクスポートされたスナップショットをデフォルトの場所からダウンロードできます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザーおよび移行 - 管理アプリケーション役割(Oracle Enterprise Data Management Cloud および Profitability and Cost Management)

使用方法

`epmautomate exportSnapshot SNAPSHOT_NAME`。ここで、`SNAPSHOT_NAME` は Migration の既存するスナップショットの名前です。このスナップショットは新しいスナップショットによって置き換えられます。

例

```
epmautomate exportSnapshot January16FullApp
```

exportTemplate

アプリケーションをテンプレートとして .ZIP ファイルにエクスポートします。エクスポートされたファイルは `profitoutbox` に格納されます。

[downloadFile](#) コマンドを使用して、エクスポートされたファイルをローカル・コンピュータにダウンロードできます。

適用対象

Profitability and Cost Management

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

`epmautomate exportTemplate APPLICATION_NAME File_Name`。ここで:

- `APPLICATION_NAME` は、テンプレートとしてエクスポートする **Profitability and Cost Management** アプリケーションの名前です
- `File_Name` はテンプレート・ファイルの名前です

例

```
epmautomate exportTemplate BksML12 template1
```

exportTaskManagerAccessControl

タスク・マネージャ、補足データおよびエンタープライズ仕訳のユーザー割当てのユーザー詳細レポートをエクスポートします。レポートには、環境内で事前定義済役割を持つユーザーに関する情報が含まれ、各ユーザーの属性(名前や電子メールなど)、ユーザーのステータス、チーム、事前定義済役割、ワークフローの役割、組織、グループ、最終ログインのタイムスタンプが **Excel** または **CSV** ファイルに一覧表示されます。

タスク・マネージャのアクセス制御レポートのサンプル:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Name	User Login	Status	Teams	Email	Role	Workflow Task Assignee	Task Approver	Organizati	Last Login	Form Preparer	Form Approver	Form Integrator	Groups	
2	John Doe	john.doe@example.com	Available	Admins	john.doe@example.com	Administrator									Service Administrator
3	Jane Doe	jane.doe@example.com	Available		jane.doe@example.com	Power User									Power User
4	user Ats	ats_user	Available		example1@example.com	User									User
5	user2 Ats	ats_user2	Available		example2@example.com	User									User
6	View User	viewUser	Available		view.user@exaple.com	Viewer									Viewer

適用対象

Enterprise Profitability and Cost Management、Financial Consolidation and Close および Tax Reporting

必要な役割

サービス管理者

使用方法

`epmAutomate exportTaskManagerAccessControl REPORT_NAME`。ここで、`REPORT_NAME` は、レポートが格納されているエクスポート・ファイルの名前(有効な **CSV** または **XLS** の拡張子を含む)です。

このレポートは、**CSV** または **XSL** 形式で生成できます。[downloadFile](#) コマンドを使用してそれをダウンロードできます。

例

- `epmAutomate exportTaskManagerAccessControl aclreport.csv`
- `epmAutomate exportTaskManagerAccessControl aclreport.xls`

exportValidIntersections

有効な交差グループをビジネス・プロセスから ZIP ファイルにエクスポートします。このファイルは、[downloadFile](#) コマンドを使用して、ローカル・コンピュータにダウンロードできます。有効な交差は、定義した有効な交差ルールと呼ばれるルールに基づいてフィルタリングされるセルの相互作用です。これらのルールによって、ユーザーがデータの入力や実行時プロンプトを選択するときに、特定のセル交差がフィルタされます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmautomate exportValidIntersections FILE_NAME.zip [names=INTERSECTION_NAMES]`。ここで:

- `FILE_NAME` は、エクスポート ZIP ファイルの名前です。コマンドで識別されたすべての有効な交差は、Microsoft Excel ファイルにエクスポートされ、圧縮されてこのファイルが作成されます。
- `names` は、オプションであり、エクスポートする有効な交差のカンマ区切りのリストを識別します。このパラメータ値を指定していない場合は、アプリケーション内のすべての有効な交差がエクスポートされます。

例

- **特定の有効な交差のエクスポート**
`epmautomate exportValidIntersections VI_export_File.zip names=VIAccountPeriod,VIEntityPeriod,VIProductPeriod`
- **すべての有効な交差のエクスポート**
`epmautomate exportValidIntersections VI_export_File.zip`

extractDimension

Oracle Fusion Cloud Enterprise Data Management デイメンションをファイルまたはグローバル接続に抽出します。

適用対象

Oracle Enterprise Data Management Cloud

必要な役割

- サービス管理者
- ユーザー(データ・マネージャ 権限付き)

使用方法

`epmautomate extractDimension APPLICATION DIMENSION EXTRACT_PROFILE FILE_NAME [connection=NAME]`。ここで:

- `APPLICATION` は、Oracle Enterprise Data Management Cloud アプリケーションの名前です。
- `DIMENSION` は、抽出されるディメンションの名前です。
- `EXTRACT_PROFILE` は、アプリケーションで定義された抽出プロファイルの名前です。このプロファイルは、ディメンションを抽出するために使用されます。
- `FILE_NAME` は、抽出されたデータを格納するためのファイル(ファイルにエクスポートする場合は CSV、Oracle Financials Cloud にエクスポートする場合は ZIP)の名前です。接続パラメータ値が設定されていない場合、このファイルはステージング領域に作成されます。このファイルは、[downloadFile](#) コマンドを使用してローカル・コンピュータにダウンロードするか、[copyFileFromInstance](#) コマンドを使用して別の Oracle Enterprise Data Management Cloud 環境にコピーできます。
- `connection=NAME` はオプションで、Oracle Enterprise Data Management Cloud で定義されたグローバル接続名(インスタンスの場所)をファイルの場所として指定します。指定した場合、抽出ファイルはターゲット環境(Oracle Fusion Cloud Enterprise Performance Management の受信ボックスと Oracle ERP の指定したドキュメント・アカウント)にアップロードされます。

Note:

グローバル接続で指定された資格証明には、ターゲット環境に書き込むためのアクセス権が含まれている必要があります。

例

- Oracle Enterprise Data Management Cloud ステージング領域に抽出: `epmautomate extractDimension USOperations Entity EntityExtProfile EntityData.CSV`
- 抽出して Oracle ERP にアップロード: `epmautomate extractDimension USOperations Entity EntityExtProfile EntityData.zip Connection=ora_fusion_gl`
- 抽出してターゲット Cloud EPM 受信ボックスにアップロード: `epmautomate extractDimension USOperations Entity EntityExtProfile EntityData.CSV Connection=EPM_cloud_pln`

extractPackage

単一の操作を使用して、アプリケーションの複数の抽出で構成される抽出パッケージを実行します。パッケージ内の各抽出の結果は 1 つの ZIP ファイルに追加されます。これは、ステージング領域またはグローバル接続に書き込まれる場合があります。

適用対象

Oracle Fusion Cloud Enterprise Data Management

必要な役割

- サービス管理者

- ユーザー(データ・マネージャ権限付き)

使用方法

`epmautomate extractPackage APPLICATION_NAME EXTRACT_PACKAGE_NAME FILE_NAME.zip [connection=CONNECTION_NAME]`。ここで:

- `APPLICATION_NAME` は、Oracle Enterprise Data Management Cloud アプリケーションの名前です。
- `EXTRACT_PACKAGE_NAME` は、Oracle Enterprise Data Management Cloud アプリケーションに定義されているパッケージの名前です。
- `FILE_NAME` は抽出 ZIP ファイルの名前で、このファイルには抽出の結果が含まれます。このファイルは、ステージング領域に作成されるか、接続名が指定されている場合はグローバル接続に書き込まれます。パッケージには 1 つ以上の抽出プロファイルが含まれる場合があり、それぞれに個別名が付けられています。抽出パッケージ ZIP ファイルには、パッケージ内の抽出の出力ファイルが含まれます。ZIP ファイル内のファイル数は、パッケージの設定によって決まりません。
- `CONNECTION_NAME` は、オプションで、Oracle Enterprise Data Management Cloud に定義されたグローバル接続名(インスタンスの場所)です。これは、抽出の結果が格納された ZIP ファイルをアップロードする場所を識別します。接続名を指定した場合、抽出データを格納する ZIP ファイルはターゲット環境(Oracle Fusion Cloud Enterprise Performance Management 環境の受信ボックスまたは Oracle ERP の指定したドキュメント・アカウント)にアップロードされます。値が指定されていない場合、エクスポート・ファイルは、Oracle Enterprise Data Management Cloud 環境のデフォルトの送信ボックス(ステージング領域)内に作成されます。

例

- COARedesignMaps パッケージを Oracle Enterprise Data Management Cloud ステージング領域内のファイルに抽出します:

```
epmautomate extractPackage FinancialsCloud COARedesignMaps COARedesign.zip
```

- COARedesignMaps パッケージをファイルに抽出し、接続を使用して Oracle Fusion Cloud EPM 受信ボックスにアップロードします。

```
epmautomate extractPackage CorporateAccount COARedesignMaps  
COARedesign.zip Connection=EPM_cloud_pln
```

feedback

オラクル社および環境のサービス管理者にフィードバックを送信し、過去 24 時間に作成されたすべての EPM 自動化ログ・ファイルを現在のディレクトリから自動的にアップロードします。

必要に応じて、現在の問題が発生した理由を診断するために Oracle サポートが使用できる追加ファイル(たとえば、Fiddler トレース・ファイル)をアップロードします。

このコマンドは、サービスの「フィードバックの提供」機能によく似ており、ユーザー・インタフェースが反応しない場合や EPM 自動化の実行中に問題が発生した場合にオラクル社にフィードバックを提供するのに特に便利です。

「フィードバックの提供」機能の詳細は、*管理者用スタート・ガイド*でフィードバックの提供ユーティリティを使用してオラクル社の診断情報収集に協力するを参照してください。

このコマンドにより、フィードバックではサービス・リクエストは作成されないことを通知する次のようなメッセージが返されます。問題を解決するためにオラクル社の支援が必要な場合は、サービス・リクエストを提出する必要があります。このコマンドでは **UDR** 参照番号が表示されます。この番号は、提出するサービス・リクエストに含める必要があります。

```
Thank you for providing feedback. If you need Oracle's assistance with this issue please log into My Oracle Support and log a service request.
Make a note of the feedback reference below as you will be asked to provide this information during the SR submission process.
Reference is UDR_502367689_example@example.com_2022_10_17_06_29_41
feedback completed successfully
c:\Oracle\EPM Automate\bin>
```

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

使用方法

epmautomate feedback "*Comment*" [Screenshot="*FILE_PATH*"] [File="*FILE_PATH*"]。ここで:

- *Comment* は、このフィードバックを送信することになった問題を説明するテキストです。コメントは引用符で囲む必要があります。
- *Screenshot* はオプションで、このフィードバックを送信することになった問題を示すグラフィック・ファイルの名前を識別します。このパラメータと値を必要に応じて繰り返すことによって、複数のスクリーンショットを送信できます。
- *File* はオプションで、現在の問題を解決するために **Oracle** サポートが使用できるファイルの名前を識別します。このパラメータを使用して、**Fiddler** トレースまたはその他のファイルを **Oracle** に送信します。このパラメータと値を必要に応じて繰り返すことによって、複数のファイルを送信できます。

例

- **Windows:** epmautomate Feedback "runplantypemap CampaignToReporting ClearData=True did not clear data from aggregate storage" Screenshot=C:/feedback/issue.jpg File=exampleScript.ps1 file=trace.har
- **Linux:** epmautomate Feedback "runplantypemap CampaignToReporting ClearData=True did not clear data from aggregate storage" Screenshot=/scratch/screens/issue.jpg File=/home/feedback/trace.har

getApplicationAdminMode

アプリケーションが管理モードであり、アクセスがサービス管理者のみに制限されているかどうかをチェックします。

このコマンドは、アプリケーションが管理モードの場合は true を返し、それ以外の場合は false を返します。これは、自動化スクリプトを実行する前にアプリケーションのステータスをチェックするのに便利です。たとえば、[refreshCube](#) コマンドはアプリケーションが管理モードになっている必要があります。次のように自動化スクリプトでこのコマンドを使用して、アプリケーションが管理モードになっているかどうかをチェックできます。

```
adminMode = `epmautomate.sh getApplicationAdminMode`  
if ["$adminMode" == "true"]  
    epmautomate.sh refreshCube
```

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Account Reconciliation、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

```
epmautomate getApplicationAdminMode
```

例

```
epmautomate getApplicationAdminMode
```

getDailyMaintenanceStartTime

環境の日次メンテナンスの開始がスケジュールされている時間の協定世界時(UTC)、またはオプションでタイム・ゾーンをコンソールに表示します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者、ユーザー(Oracle Enterprise Data Management Cloud のみ)、任意の事前定義済役割および移行 - 管理アプリケーション役割

使用方法

```
epmautomate getDailyMaintenanceStartTime [timezone=true|false]。ここで、  
timezone=true はオプションで、設定時に指定したタイム・ゾーン(たとえば、America/
```

Los_Angeles)で日次メンテナンス開始時間を表示するかどうかを識別します。デフォルトは false です。

例

- 設定時に指定したタイム・ゾーンでメンテナンス時間を表示します:
epmautomate getDailyMaintenanceStartTime timezone=true
- UTC でメンテナンス時間を表示します:
epmautomate getDailyMaintenanceStartTime

getEssbaseQryGovExecTime

Oracle Essbase 問合せが終了する前に問合せで情報を取得し配信するために使用できる現在の最大時間(秒単位)を表示します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Profitability and Cost Management、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

epmautomate getEssbaseQryGovExecTime

サンプル・コマンド出力:

```
c:\Oracle\EPM Automate\bin>epmautomate getEssbaseQryGovExecTime
300

c:\Oracle\EPM Automate\bin>epmautomate setEssbaseQryGovExecTime 600
setEssbaseQryGovExecTime completed successfully

c:\Oracle\EPM Automate\bin>epmautomate getEssbaseQryGovExecTime
600
```

例

epmautomate getEssbaseQryGovExecTime

getIdleSessionTimeout

Oracle Fusion Cloud Enterprise Performance Management 環境のセッションのタイムアウト(分単位)を表示します。この期間にセッションがアイドルになると、ユーザーはログイン・ページにリダイレクトされます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

```
epmAutomate getIdleSessionTimeout
```

サンプル・コマンド出力:

```
c:\Oracle\EPM Automate\bin>epmAutomate getIdleSessionTimeout  
75
```

getIPAllowlist

OCI (Gen 2)環境で、現在の許可リストに含まれている IP アドレスおよび Classless Inter-Domain Routing (CIDR)を表示します。

このコマンドは、特定の IP アドレスまたは CIDR が現在 OCI (Gen 2)環境へのアクセスを許可されているかどうかをチェックするのに役立ちます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

```
epmAutomate getIPAllowlist
```

 **Note:**

既存のすべての IP アドレスと CIDR をファイルに書き込むには、出力をテキスト・ファイルにリダイレクトし、そのファイルを編集して(いくつかの、またはすべての IP アドレスと CIDR を削除して)環境にアップロードし、[setIPAllowlist](#) コマンドを使用してファイル内のエントリを許可リストから削除します。コマンドの実行例:

```
epmAutomate getIPAllowlist > myRemoveList.txt
epmAutomate uploadFile myRemoveList.txt
epmAutomate setIPAllowlist remove myRemoveList.txt
```

例

現在の許可リストに含まれている IP アドレスおよび CIDR を表示します:

```
epmAutomate getIPAllowlist
```

getRestrictedDataAccess

フィードバックの提供ユーティリティを使用している間にサービス管理者がアプリケーション・スナップショットを Oracle に送信することに同意しないように環境が構成されているかどうかを表示します。

このコマンドは、サービス管理者がアプリケーション・スナップショットを送信することに同意しないように環境が構成されている場合は true を報告し、それ以外の場合は false を報告します

適用対象

Account Reconciliation、Oracle Fusion Cloud Enterprise Data Management、Enterprise Profitability and Cost Management、Financial Consolidation and Close、フリーフォーム、Planning、Planning モジュール、Profitability and Cost Management、Narrative Reporting、Sales Planning、Strategic Workforce Planning および Tax Reporting。

必要な役割

サービス管理者

使用方法

現在のデータ・アクセス制限のステータスを変更するには、[setRestrictedDataAccess](#) を使用します。

例

```
epmAutomate getRestrictedDataAccess
```

getSubstVar

代替変数の値を取得し、画面に表示します。

表示形式は、`CUBE_NAME.SUBSTVAR=value` (例: `Plan2.CurYear=2016`)です。アプリケーション・レベルの代替変数の値は、`ALL.CurYear=2016` のように `ALL.SUBSTVAR=value` 形式で表示されます

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

「サービス管理者」、「パワー・ユーザー」(ルール起動アクセス許可付き)

使用方法

`epmautomate getSubstVar CUBE_NAME|ALL [name=VARIABLE_NAME]`。ここで:

- `CUBE_NAME` は、代替変数を取得するキューブ(**Plan1**、**Plan2** など)です。アプリケーション・レベルで代替変数を取得するには、`ALL` を使用します。
- `name=VARIABLE_NAME`(オプション)には、値を取得する代替変数を指定します。変数名を指定しないと、すべての代替変数の値が取得されます。

例

- アプリケーション・レベルとキューブ・レベルですべての代替変数の値を取得: `epmautomate getSubstVar ALL`
- アプリケーション・レベルで特定の 1 つの代替変数の値を取得: `epmautomate getSubstVar ALL name=CurYear`
- キューブ・レベルですべての代替変数の値を取得: `epmautomate getSubstVar Plan2`
- キューブ・レベルで特定の 1 つの代替変数の値を取得: `epmautomate getSubstVar Plan2 name=CurYear`

getVirusScanOnFileUploads

OCI (Gen 2)環境が、アップロードされるすべてのファイルをスキャンしてウィルス・フリーであることを確認するかどうかを確認します。

このコマンドは、ファイルが環境にアップロードされる前に、ウィルス・スキャンが強制されるかどうかを確認します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmautomate getVirusScanOnFileUploads`

このコマンドは、アップロードされるファイルを環境でウイルス・スキャンする場合は true を出力し、そうでない場合は false を出力します。

groupAssignmentAuditReport

指定した日付範囲にアクセス制御グループに対して割り当てられた、または割当てを解除(削除)されたユーザーとグループをリストするレポートを作成します。このレポートは最大 120 日間について生成できます。

このレポートは CSV ファイルとして生成され、セキュリティ監査操作のサポートに使用できます。生成された CSV ファイルの各行には、追加または削除されたユーザーやグループ、ユーザーやグループが追加または削除されたグループ、アクションを実行したサービス管理者、アクションが完了した日次が含まれます。

	A	B	C	D	E
1	User/Group Name	Group	Action	Administrator	Date and Time
2	testGroup1	exampleGroup1	Added	john.smith@example.com	May 10, 2022 23:13:20 UTC
3	johndoe@example.com	exampleGroup1	Added	joe.doe@example.com	May 11, 2022 23:14:20 UTC
4	JaneDoe@example.com	testGroup1	Added	joe.doe@example.com	May 11, 2022 23:14:50 UTC
5	jaDoe@example.com	testGroup1	Added	john.smith@example.com	May 12, 2022 23:25:20 UTC
6	JaneDoe@example.com	testGroup1	Removed	john.smith@example.com	May 13, 2022 18:13:20 UTC
7	jaDoe@example.com	testGroup1	Removed	john.smith@example.com	May 13, 2022 18:22:20 UTC
8	testGroup1	exampleGroup1	Removed	joe.doe@example.com	May 13, 2022 23:13:20 UTC

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Strategic Workforce Planning および Sales Planning。

必要な役割

- サービス管理者
- 任意の事前定義済役割およびアクセス制御 - 管理アプリケーション役割
- 任意の事前定義済役割およびアクセス制御 - 表示アプリケーション役割

使用方法

epmAutomate groupAssignmentAuditReport FROM_DATE TO_DATE REPORT_NAME。ここで、

- FROM_DATE は、レポートが生成される期間の開始日(YYYY-MM-DD 形式)です。
- TO_DATE は、レポートが生成される期間の終了日(YYYY-MM-DD 形式)です。
- REPORT_NAME は、レポートの CSV ファイルの名前です。downloadFile コマンドを使用して、生成されたレポートをダウンロードできます。

例

```
epmAutomate groupAssignmentAuditReport 2022-03-01 2022-05-01
GroupAssignmentReport.CSV
```

help

EPM 自動化のすべてのコマンドのまたは特定のコマンドのヘルプを表示します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

使用方法

`epmautomate [COMMAND_NAME] help`。ここで、`COMMAND_NAME` はオプションで、コンソールにヘルプを表示するコマンドの名前です

例

- すべてのコマンドのヘルプにアクセスできる **Web** ページを表示します:
`epmautomate help`
- **encrypt** コマンドのコマンド・ヘルプをコンソールに表示します:
`epmautomate encrypt help`

importAppAudit

環境から監査データをエクスポートすることで作成した ZIP ファイルから、データ監査レコードをインポートします。

インポート・ファイルは [exportAppAudit](#) コマンド (`epmautomate exportAppAudit auditData ndays=All`) を使用して作成します。このコマンドを使用して、障害回復のための移行またはクローニング中に、ある環境から別の環境に監査レコードをクローニングします。

適用対象

Planning、Planning モジュール、フリーフォーム、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

`epmautomate importAppAudit FILE_NAME [logfile=LOG_FILE_NAME]`。ここで:

- FILE_NAME は、アプリケーションにインポートするデータ監査レコードが格納されている ZIP ファイルの名前です。このコマンドを実行する前に、[uploadFile](#) コマンドを使用して、このファイルを環境にアップロードします。
- logFileName はオプションで、インポート中に発生したエラーが記録されるエラー・ログ・ファイルを識別します。この値が指定されていない場合、コマンドは、次の規則を使用して名前が指定されたエラー・ファイルを生成します: username_date_timestamp。このファイルは、[downloadFile](#) コマンドを使用してダウンロードできます。

例

```
epmautomate importAppaudit Audit_data.zip logFileName=auditImportLog
```

importAppSecurity

アプリケーションのユーザーまたはグループのアクセス権限を受信ボックスで使用可能な CSV ファイルからロードします。

アクセス権限をインポートすることにより、インポートされたメンバー、データ・フォーム、データ・フォーム・フォルダ、タスク・リスト、計算マネージャのビジネス・ルールおよびビジネス・ルール・フォルダの既存の割当てが上書きされます。その他の既存アクセス権限は影響を受けません。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

epmautomate importAppSecurity ACL_FILE_NAME ERROR_FILE [clearall=true|false]。ここで:

- ACL_FILE_NAME は、アプリケーションにインポートするアクセス権限を含む CSV ファイルの名前です。このコマンドを実行する前に、[uploadFile](#) コマンドを使用して、このファイルを受信ボックスにアップロードします。サンプルの入力ファイルの内容を次の図に示します。

	A	B	C	D	E	F	G	H
1	Object Name	Name	Parent	Is User	Object Type	Access Type	Access Mode	Remove
2	AllAB	Interactive User		N	SL_DIMENSION	READWRITE	@IDESCENDANTS	N
3	FormsB	Interactive User	/	N	SL_FORMFOLDER	READWRITE	@IDESCENDANTS	N
4	Statistics	Interactive User		N	SL_DIMENSION	READWRITE	@IDESCENDANTS	N
5	TD	Interactive User		N	SL_DIMENSION	READWRITE	@IDESCENDANTS	N
6	No Entity	Interactive User		N	SL_DIMENSION	READWRITE	MEMBER	N

列ヘッダーおよび指定可能な値の説明は、*REST API* ガイドのセキュリティのインポートを参照してください。

- ERROR_FILE は、この操作中に検出されたエラーを記録するために EPM 自動化によって作成される CSV ファイルの名前です。このファイルをローカル・コンピュータにダウンロードして分析し、レポートされたエラーを修正できます。サンプルのエラー・ファイルの内容を次の図に示します。このファイルの列が入力ファイルのヘッダー列に対応します。

	A	B	C	D	E	F	G	H
1	AllAB	Interactive User		N	SL_DIMENSION	READWRITE	@IDESCENDANTS	N
2	FormsB	Interactive User	/	N	SL_FORMFOLDER	READWRITE	@IDESCENDANTS	N

- `clearall` はオプションで、新しいアクセス権限をファイルからロードする前に既存の権限を削除するかどうかを指定します。デフォルトは `false` です。

例

```
epmautomate importAppSecurity Acl_file.CSV Acl_import_error.CSV clearall=true
```

importARApplicationProperties

エクスポート JSON ファイルで利用可能なアプリケーション設定(レッドウッド・エクスペリエンス、テーマ、電子メール通知、およびビジネス・プロセス名)、ロゴ、および背景イメージを Account Reconciliation 環境にインポートします。

適用対象

Account Reconciliation

必要な役割

サービス管理者

使用方法

`epmautomate importARApplicationProperties FILE_NAME`。ここで、`FILE_NAME` は、環境からエクスポートされた JSON ファイルの名前です。

[exportARApplicationProperties](#) コマンドを使用して別の環境からエクスポートされたこのファイルは、アプリケーション設定を復元する環境で使用可能である必要があります。

例

```
epmautomate importARApplicationProperties myProp.JSON
```

importBackgroundImage

背景イメージをエクスポート・ファイルから Account Reconciliation 環境にインポートします。

適用対象

Account Reconciliation

必要な役割

サービス管理者

使用方法

`epmautomate importBackgroundImage FILE_NAME.jpg`。ここで、`FILE_NAME` は、別の環境からエクスポートされた背景イメージ・ファイルの名前です。

例

```
epmautomate importBackgroundImage image_file.jpg
```

importBalances

データ管理を使用して、データ・ロード定義から残高データをインポートします。

適用対象

Account Reconciliation。

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」を持つユーザーには、ACL を介して追加のセキュリティが付与される必要があります

使用方法

epmautomate importBalances *DL_DEFINITION PERIOD*。ここで:

- *DL_DEFINITION* は、Account Reconciliation の既存のデータ・ロード定義です。
- *PERIOD* は、期間の名前です

例

```
epmautomate importBalances DailyLoad "January 2020"
```

importCellLevelSecurity

セルレベルのセキュリティ・レコードを含む 1 つの Excel ファイルを含む ZIP ファイルから、セルレベルのセキュリティ設定をビジネス・プロセスにインポートします。このコマンドを実行する前に、[uploadFile](#) コマンドを使用してインポート・ファイルを環境にアップロードします。

インポート ZIP ファイルには、セルレベルのセキュリティを正常にインポートするために、2 つのワークシート(ルールとサブ・ルール)を含む 1 つの Excel ファイルが含まれている必要があります。ルール・シートには、セルレベルのセキュリティ定義、含まれるディメンション、未指定の有効なディメンションや追加のディメンションが必要などのプロパティを含める必要があります。サブ・ルール・シートには、メンバーの選択と除外が含まれている必要があります。インポート・ファイル形式のテンプレートを取得する最良の方法は、アプリケーションからセルレベルのセキュリティをエクスポートすることです。次の図にサンプル形式を示します。

	A	B	C	D	E	F	G	H	I	J	K
	Name	Position	Description	Enabled	Valid Cubes	Anchor Dim Name	Anchor Dimension Apply to Selected	Dim1	Dim1 Required	Dim2	Dim2 Required
1											
2	Sample-CLS	1		true	All	Product	true	Account	false		
3	Sample-CLS-Dup	2		true	All	Product	true	Account	false		
4											
5											

	A	B	C	D	E	F	G	H	I	J	K	L
	Name	Users	User Groups	Restriction	Anchor Members	Anchor Exclusion	Dim1 Members	Dim1 Exclusion	Dim2 Members	Dim2 Exclusion	Dim3 Members	Dim3 Exclusion
1												
2	Sample-CLS		Service Administrator	Deny Write P_TP			Statistics					
3	Sample-CLS-Dup		Service Administrator	Deny Write P_TP			Statistics					
4												

適用対象

Planning、Planning モジュール、フリーフォーム、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmautomate importCellLevelSecurity FILE_NAME.ZIP [ErrorFile=FILE_NAME.txt]`。ここで:

- `FILE_NAME` は、セルレベルのセキュリティ情報を含む Excel ファイルを含む ZIP ファイルの名前です。
- `ErrorFile` は、オプションで、エラー・レコードが書き込まれるテキスト・ファイルの名前を識別します。このパラメータ値を指定していない場合は、エラー・ファイルが自動的に生成されます。この名前はジョブ・コンソールで確認できます。
[downloadFile](#) コマンドを使用して、エラー・ファイルをローカル・コンピュータにダウンロードします。

例

```
epmautomate importCellLevelSecurity ImportCLSDRecordsFile.zip
ErrorFile=ImportCLSDRecords_errors.txt
```

importConsolidationJournals

.JLF ファイルから Financial Consolidation and Close に連結仕訳をインポートします。

- `exportConsolidationJournals` コマンドを使用して、このコマンドの入力として使用される `.JLF` ファイルを作成します。
- このコマンドを実行する前に、`uploadFile` コマンドを使用して、入力ファイルを環境にロードします。

適用対象

Financial Consolidation and Close

必要な役割

サービス管理者

使用方法

```
epmautomate importConsolidationJournals jobName [fileName=FILE_NAME]
[errorFileName=ERROR_FILE_NAME]。ここで
```

- `jobName` は、**Financial Consolidation and Close** で作成された仕訳のインポート・ジョブの名前です。
- `fileName` はオプションで、仕訳のインポート元の `.JLF` ファイルの名前です。
- `errorFileName` はオプションで、インポート・プロセス中に生成されたメッセージが記録されるログ・ファイルの名前です。

例

```
epmautomate importConsolidationJournals "JIMPORT1" fileName="TestImport1.jlf"
errorFileName="TestImport1_error.log"
```

importData

`import data` タイプのジョブで指定されたデータのインポート設定を使用して、ファイルからデータをアプリケーションにインポートします。

`uploadFile` コマンドを使用して、アプリケーション・データを含むファイルをデフォルトのアップロード場所にアップロードします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

```
epmautomate importData JOB_NAME [FILE_NAME] errorFile=ERROR_FILE.zip。ここで:
```

- `JOB_NAME` は、アプリケーションに定義されたジョブ名です。
- `FILE_NAME` はオプションで、データのインポート元の `ZIP`、`CSV` または `TXT (Essbase 形式データ・ファイル)` ファイルの名前を指定します。ファイル名を指定する場合、ジョブのインポート・ファイル名は無視されます。

ジョブが Essbase 形式のデータをインポートするように定義されている場合、ZIP ファイルには Essbase 形式の TXT ファイルが含まれている必要があります。他のインポート・ジョブでは、ZIP ファイルにはファイル名でインポート順序を示す 1 つ以上の CSV ファイルが含まれる場合があります(例: data1-3.csv、data2-3.csv および data3-3.csv)。

- `errorFile` はオプションで、却下されたレコード(ある場合)をインポート操作中に記録する ZIP ファイルの名前を指定します。送信ボックスに同じ名前の ZIP ファイルがある場合は上書きされます。このファイルは、[downloadFile](#) コマンドを使用してダウンロードできます。

例

```
epmautomate importData dailydataload dailydata.zip errorFile=dataImport_error.zip
```

importDataManagement

データ管理レコードを ZIP ファイルから環境にインポートします。

このコマンドは、[exportDataManagement](#) コマンドを使用して作成された ZIP ファイルから設定表およびステージング表にデータをインポートします。サービス管理者は、たとえば、`epmAutomate uploadFile "C:/datafile/datafile.zip" inbox` など、[uploadFile](#) コマンドを使用して、インポート ZIP ファイルをデータ管理の受信ボックスまたはその中のフォルダにアップロードします。

Note:

このコマンドでインポートできるのは、同じ月次更新で実行されている別の環境からエクスポートされたデータ管理レコードのみです。たとえば、**21.11 Oracle Fusion Cloud Enterprise Performance Management** 環境からエクスポートされたレコードは、別の **21.11** 環境にのみインポートできます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Account Reconciliation、Tax Reporting、Profitability and Cost Management、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

`epmautomate importDataManagement FILE_NAME.zip`、ここで、`FILE_NAME` は、インポートするデータ管理データが含まれている ZIP ファイルの名前です。

例

- データ管理の受信ボックスからインポートします:
`epmautomate importDataManagement inbox/dataFile.zip`
- 受信ボックス内のフォルダからインポートします:
`epmautomate importDataManagement inbox/dm_data/dataFile.zip`

importDimension

ディメンションをファイルから Oracle Fusion Cloud Enterprise Data Management アプリケーションにインポートします。

このコマンドは、Oracle Enterprise Data Management Cloud またはステージング領域で定義された接続から、入力ファイルをインポートできます。

ファイルを Oracle Enterprise Data Management Cloud ステージング領域からインポートする場合、サービス管理者は、[uploadFile](#) コマンドを使用して、このファイルをターゲットの Oracle Enterprise Data Management Cloud 環境にアップロードします。また、サービス管理者は、[copyFileFromInstance](#) コマンドを使用して、別の Oracle Fusion Cloud Enterprise Performance Management 環境からファイルをコピーすることもできます。

適用対象

Oracle Enterprise Data Management Cloud

必要な役割

- サービス管理者
- ユーザー(データ・マネージャ 権限付き)

使用方法

```
epmautomate importDimension APPLICATION DIMENSION IMPORT_TYPE FILE_NAME  
[connection=NAME]。ここで:
```

- APPLICATION は、Oracle Enterprise Data Management Cloud アプリケーションの名前です
- DIMENSION は、インポートするアプリケーションのディメンションの名前です
- IMPORT_TYPE は、インポートの実行方法を示します。有効なインポート・タイプは次のとおりです:
 - ResetDimension は、既存のディメンション・データをすべて削除して、新規のデータをインポートします
 - ReplaceNodes は、ノードを追加または更新して、インポート中に既存の階層を置換します
 - Merge は、インポート要求を使用して、ノードおよび階層への増分変更を処理します
- FILE_NAME は、インポートするディメンション・データが含まれるファイル(CSV または ZIP)の名前です。ファイル名は、_ (アンダースコア文字)を前に付けたディメンション名で終わる必要があります(例: import_Entity.csv)。複数のインポート・ファイルが含まれる ZIP ファイルからインポートする場合、このコマンドは、ZIP ファイル内のファイル名に依存して、適切なインポート・ファイルを識別します。
connection の値を指定する場合、ZIP ファイル(例: importdata_Entity.zip)からディメンションをインポートする必要があります。
- connection=NAME はオプションで、Oracle Enterprise Data Management Cloud で定義された接続名(インスタンスの場所)をインポート・ファイルの場所として指定します。指定されていない場合、インポート・プロセスはローカル・ステージング領域でインポート・ファイルを探します。

例

- ステージング領域にアップロードされたファイルからのインポート: `epmautomate importDimension USOperations Entity ReplaceNodes data_Entity.CSV`
- 別の Cloud EPM 環境の送信ボックスからのインポート: `epmautomate importDimension USOperations Entity ReplaceNodes data_Entity.ZIP Connection=Cloud-EPM_pln`

importJobConsole

環境からエクスポートされたジョブ・コンソール・レコードが格納されている ZIP ファイルを使用して、ジョブ・コンソール・レコードをクローニングします。

このコマンドを使用したジョブ・コンソール・レコードのインポートは、`recreate` コマンドの実行後に実行する必要がある 1 回限りのタスクです。すでにこのコマンドを使用してジョブ・コンソール・レコードをインポートしている場合、コマンドの後続の呼出しは、環境を再作成するまで失敗します。

`exportJobConsole` コマンド(`epmAutomate exportJobConsole FILE_NAME.zip nDays=All jobTypes=All jobStatusCode=All`)を使用して、このコマンドの入力として使用される ZIP ファイルを作成します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

`epmautomate importJobConsole FILE_NAME.zip [logFileName=jobConsoleLog]`。ここで:

- `FILE_NAME` は、インポートするジョブ・コンソール・レコードが格納されている ZIP ファイルの名前です。 `uploadFile` コマンドを使用して、このファイルを環境にアップロードします。
- `logFileName` はオプションで、`jobConsoleLog` をインポート中に発生したエラーが記録されるログ・ファイルとして識別します。この値が指定されていない場合、コマンドは、次の規則を使用して名前が指定されたエラー・ファイルを生成します:
`usernameimportLog_date_timestamp.zip`。このファイルは、 `downloadFile` コマンドを使用してダウンロードできます。

例

```
epmautomate importJobConsole jobConsole.zip jobConsoleLog
```

importLibraryArtifact

アーカイブまたはファイルから Narrative Reporting ライブラリにライブラリ・アーティファクトをインポートします。

このコマンドを実行する前に、サービス管理者は、アップロードの `uploadFile` コマンドを使用して、ソース・アーカイブまたはファイルを環境にロードします。

適用対象

Narrative Reporting

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」を持つユーザーには、ACL を介して追加のセキュリティが付与される必要があります

使用方法

```
epmautomate importLibraryArtifact SOURCE_FILE [errorFile=ERROR_FILE.txt]
[importFormat=Native|File] [importFolder=FOLDER_PATH] [ importPermission=true|
false] [overwrite=true|false]。ここで:
```

- `SOURCE_FILE` は、ライブラリにインポートするアーティファクトが含まれているアーカイブの名前です。このファイルは受信ボックスで使用可能であることが必要です。
- `errorFile` はオプションで、インポート関連のエラーを格納するテキスト・ファイルの一意の名前です。
- `importFormat` はオプションで、次のいずれかを使用します:
 - `Native` は、`exportFormat=Native` オプションを指定した `exportLibraryArtifact` コマンドを使用して作成された zip ファイルからアーティファクトをインポートします。これがデフォルト値です。
 - `File` は、バイナリ・ファイルをインポートします。

ノート:

`importSnapshot` コマンドを使用して、ライブラリ・アーティファクト zip ファイル(`exportFormat=LCM` オプションを指定した `exportLibraryArtifact` コマンドを使用して作成)を **Financial Consolidation and Close**、**Planning**、**Planning** モジュールまたは **Tax Reporting** 環境にインポートします。

- `importFolder` はオプションで、インポートされたアーティファクトを格納するライブラリの場所です。この場所が `Library` (デフォルトのインポートの場所)と異なる場合は、このパスを指定します。
- `importPermission` は、アーティファクトに設定されたアクセス権限をインポートするかどうかを指定します。デフォルトは `False` です。
- `overwrite` は、指定したライブラリの場所で同じ名前のアーティファクト(ある場合)を上書きするかどうかを指定します。デフォルトは `False` です。これは、インポートの場所に同じ名前のアーティファクトが存在する場合、アーティファクトがインポートされないことを意味します。

インポートの完了時に、[downloadFile](#) コマンドを使用して、エラー・ファイルをローカル・コンピュータにダウンロードします。

例

- バイナリ形式のファイルをインポートします:

```
epmautomate importLibraryArtifact newReports.doc  
errorFile=report_imp_errors.txt importFormat=File importFolder="Library/My  
Reports" importPermission=true overwrite=true
```
- エクスポートされた zip ファイルからアーティファクトをインポートします:

```
epmautomate importLibraryArtifact newReports.zip  
errorFile=report_imp_errors.txt importFormat=Native importFolder="Library/My  
Reports" importPermission=true overwrite=true
```
- エクスポートされた zip ファイルから **Financial Consolidation and Close**、**Planning**、**Planning** モジュールまたは **Tax Reporting** 環境にレポートをインポートします:

```
epmautomate importSnapshot newReports.zip
```

importLogoImage

Account Reconciliation 環境で使用される企業ロゴをエクスポート・ファイルから別の環境にインポートします。

適用対象

Account Reconciliation

必要な役割

サービス管理者

使用方法

`epmautomate importLogoImage IMAGE_NAME.jpg`。ここで、`IMAGE_NAME` は、ロゴ・イメージ・ファイルの名前です。

[downloadFile](#) コマンドを使用してエクスポートしたイメージをダウンロードできます。

[uploadFile](#) コマンドを使用してターゲット環境にアップロードしてから、[importLogoImage](#) コマンドを実行してインポートします。

例

```
epmautomate importLogoImage corpLogo.jpg
```

importMapping

事前に環境にアップロードされていたマッピング・インポート・ファイルからマッピングをインポートします。

サービス管理者は、[uploadFile](#) コマンドを使用して、データ管理の受信ボックスまたはその中のフォルダにファイルをアップロードします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

`epmautomate importMapping DIMENSION_NAME|ALL FILE_NAME IMPORT_MODE VALIDATION_MODE LOCATION`。ここで:

- `DIMENSION_NAME|ALL` は、マッピングの受取り側です。マッピングのインポート先のディメンション名を指定するか、ファイルに含まれるすべてのマッピングを適切なディメンションにインポートする場合は `ALL` を指定します。
- `FILE_NAME` は、データ管理受信ボックスまたはその中のディレクトリにあるマッピング・インポート・ファイルの名前と場所です。ファイル名(標準データ管理形式の `TXT` ファイル)とパス(たとえば、`inbox/AccountMap.txt` または `inbox/pbcs_maps/AccountMap.txt`)を指定します。
- `IMPORT_MODE` は、`REPLACE` (既存のマッピング・ルールをクリアしてからマッピングをインポート)または `MERGE` (既存のマッピング・ルールに新しいルールを追加)です。
- `VALIDATION_MODE` は、`TRUE` (アプリケーションに対してターゲット・メンバーを検証)または `FALSE` (検証を実行せずにマッピング・ファイルをロード)です。
- `LOCATION` は、マッピング・ルールがロードされるデータ管理の場所です。

例

- `epmautomate importMapping Account inbox/AccountMap.txt MERGE FALSE "France Sales"`
- `epmautomate importMapping ALL "inbox/France Sales/AllMaps.txt" MERGE FALSE "France Sales"` (マッピング・インポート・ファイルのマッピングを、**France Sales** の場所にあるすべてのマップ済ディメンションにロードします)

importMetadata

`import metadata` タイプのジョブで指定されたインポート設定を使用して、メタデータをアプリケーションにインポートします。オプションで、メタデータのインポート元の `ZIP` ファイルの名前を指定できます。

[uploadFile](#) コマンドを使用して、メタデータを含むファイルをデフォルトのアップロード場所にアップロードします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

epmautomate importMetadata JOB_NAME [FILE_NAME] errorFile=ERROR_FILE.zip。ここで:

- JOB_NAME は、アプリケーションに定義されたジョブ名です。
- FILE_NAME はオプションで、メタデータのインポート元の ZIP ファイルの名前を指定します。指定した場合、この ZIP ファイルの内容はジョブで定義されたファイル名よりも優先されます。ZIP ファイルには 1 つ以上の CSV ファイルが含まれている場合があります。ディメンションのメタデータが含まれるファイル名は、ジョブで定義されたインポート・ファイル名と一致するか、末尾が `_DIMENSIONNAME.csv` である必要があります。たとえば、`metadata_Entity.csv`、`metadata_HSP_Smart Lists.csv`、`metadata_Exchange Rates.csv` などです。
- errorFile はオプションで、却下されたレコード(ある場合)をインポート操作中に記録する ZIP ファイルの名前を指定します。送信ボックスに同じ名前の ZIP ファイルがある場合は上書きされます。このファイルは、[downloadFile](#) コマンドを使用してダウンロードできます。

ノート:

- old_name または unique_name プロパティが変更されたロード・ファイルを使用してメタデータのインポート・ジョブを実行し、メンバーの名前を変更することはできません。メンバーの名前変更は無視されます。
- このコマンドを使用してメタデータをインポートしている間、属性ディメンションは削除できません。
- ジョブでメタデータのインポートが設定されているディメンションのメタデータのみがインポートされます。他のディメンションのメタデータは、ZIP ファイルに含まれる場合、無視されます。

ZIP ファイルについて次の条件がどちらも true の場合、インポート状況が曖昧になります。

- ジョブで定義されているファイル名と同じ名前のメタデータ・ファイルが ZIP に含まれている
- `_DIMENSIONNAME.CSV` または `_DIMENSIONNAME.TXT` で終わる名前のメタデータ・ファイルが ZIP に含まれている。ここで `DIMENSIONNAME` はメタデータのインポート先であるディメンションの名前。

ZIP ファイルには、ジョブで参照しているものと同じ名前のメタデータ・ファイルか、`_DIMENSIONNAME.CSV` (または `_DIMENSIONNAME.TXT`) で終わる名前のファイルのどちらかを含め、両方は含めないことをお勧めします。たとえば、`Employees_A-Z.CSV` というメタデータ・ファイルを参照するジョブを `Employees` ディメンションにロードする場合、ZIP ファイルには `Employees_A-Z.CSV` か `New_Employees.CSV` のどちらかを含め、両方は含めないようにします。ZIP に `Employees_A-Z.CSV` と `New_Employees.CSV` の両方が含まれている場合は、ZIP でのファイルの順序に応じて EPM 自動化がどちらかのファイルをインポート対象として選択することがあります。`Employees_A-Z.CSV` ファイルは、ジョブで参照されているファイル名と同じ名前なので、インポート対象として一致する可能性があります。一方、`New_Employees.CSV` も、`_DIMENSIONNAME.CSV` のパターンと同じ名前なので、一致する可能性があります。

例

```
epmautomate importMetadata importAccount importAccount.zip  
errorFile=metadataImport_error.zip
```

importOwnershipData

出資比率データを環境で使用可能な **CSV** ファイルから期間にインポートします。

このコマンドを実行する前に、サービス管理者は、**uploadFile** コマンドを使用して、インポート・ソース **CSV** ファイルを環境にロードします。この **CSV** ファイルのヘッダーは次のとおりです:

```
Scenario、Year、Period、Entity、Parent、POwn、Control、Method
```

POwn、Control および Method 値はオプションです。

インポートされた出資比率データがすべての既存データとマージされ、無効な出資比率エントリが作成される場合があります。エンティティが階層の複数のブランチに存在する場合、インポートされた出資比率データによって、エンティティの結合出資比率%が **100%**を超えることがあります。出資比率%が **100%**を超えないように手動で修正する必要があります。

適用対象

Financial Consolidation and Close および Tax Reporting

必要な役割

- サービス管理者
- パワー・ユーザー
- エンティティへの書込みアクセス権を持つユーザー。

使用方法

```
epmautomate importOwnershipData Scenario Year Period FILE_NAME。ここで:
```

- Scenario は、出資比率データのインポート先のシナリオです。
- Year は、データのインポート先の年です。
- Period は、出資比率データのインポート先の年の期間です。
- FILE_NAME は、インポート対象のデータが含まれる **CSV** ファイルの名前です

例

```
epmautomate importOwnershipData FCCS_TotalActual FY19 Jan importfile.csv
```

importPreMappedBalances

ファイルから事前マップ済残高データを **Account Reconciliation** リポジトリにインポートします。

適用対象

Account Reconciliation

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

`epmautomate importPreMappedBalances PERIOD FILE_NAME BALANCE_TYPE CURRENCY_BUCKET`。ここで:

- `PERIOD` は、期間の名前です
- `FILE_NAME` は、インポート対象のデータが含まれる **CSV** ファイルの名前です
- `BALANCE_TYPE` は、SRC または SUB です
- `CURRENCY_BUCKET` は、Entered、Functional または Reporting です

例

```
epmautomate importPreMappedBalances "January 2015" dailydata.csv SRC Reporting
```

importPreMappedTransactions

CSV ファイルから事前マップ済トランザクションを **Account Reconciliation** リポジトリにインポートします。

適用対象

Account Reconciliation

必要な役割

「サービス管理者」、「パワー・ユーザー」、「ユーザー」、「参照者」
「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

`epmautomate importPreMappedTransactions PERIOD TRANSACTION_TYPE FILE_NAME DATE_FORMAT`。ここで:

- `PERIOD` は、期間の名前です
- `TRANSACTION_TYPE` は、次のいずれかです。
 - BEX: 残高の説明をロードする場合
 - SRC: ソース・システムの調整をロードする場合
 - SUB: サブシステムの調整をロードする場合
 - VEX: 差異分析の説明をロードする場合

- `FILE_NAME` は、インポート対象のデータが含まれる **CSV** ファイルの名前です
- `DATE_FORMAT` は、日付書式のテキスト文字列(MMM d, yyyy など)です

例

```
epmautomate importPreMappedTransactions "January 2015" "BEX" transactions.csv  
"MMM d, yyyy"
```

importProfiles

CSV ファイルから新しいプロファイル定義を **Account Reconciliation** リポジトリにインポートします。

適用対象

Account Reconciliation

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

epmautomate importProfiles `FILE_NAME PROFILE_TYPE METHOD DATE_FORMAT`。ここで:

- `FILE_NAME` は、インポート対象のデータが含まれる **CSV** ファイルの名前です
- `PROFILE_TYPE` は、profiles または children です
- `METHOD` は、Replace または Update です
- `DATE_FORMAT` は、日付書式のテキスト文字列(MMM d, yyyy など)です

例

```
epmautomate importProfiles NewRecProfiles.csv Profiles Replace "MMM d, yyyy"
```

importRates

CSV ファイルから通貨レートを **Account Reconciliation** リポジトリにインポートします。

適用対象

Account Reconciliation

必要な役割

「サービス管理者」、「パワー・ユーザー」、「ユーザー」、「参照者」
「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

epmautomate importRates PERIOD RATE_TYPE REPLACE_MODE FILE_NAME。ここで:

- PERIOD は、期間の名前です
- RATE_TYPE は、事前定義済のレート・タイプです
- REPLACEMENT_MODE は、Replace または ReplaceAll です
- FILE_NAME は、インポート対象のレートが含まれる CSV ファイルの名前です

例

```
epmautomate importRates "January 2015" Actual ReplaceAll avgrates.csv
```

importRCAttributeValues

Account Reconciliation 照合コンプライアンス・リストまたはグループ属性に属性値をインポートします。

適用対象

Account Reconciliation

必要な役割

- サービス管理者
- パワー・ユーザー(ACL を介して提供される追加のセキュリティが必要)

使用方法

epmautomate importRCAttributeValues ATTRIBUTE_NAME FILE_NAME [METHOD=REPLACE|REPLACE ALL|UPDATE] [DATEFORMAT=DD/MM/YYYY|DD-MMM-YYYY|MMM d,yyyy|All]。ここで:

- ATTRIBUTE_NAME は、値のインポート先となるリストまたはグループ属性の名前です。
- FILE_NAME は、値のインポート元 CSV インポート・ファイルです。このコマンドを実行する前に、[uploadFile](#) コマンドを使用して、このファイルを環境にアップロードします。
- METHOD はオプションで、値のインポート方法を示します。有効な値:
 - Replace は、インポート・ファイルのすべての値を照合コンプライアンスの属性値として追加します。既存の属性値はインポート・ファイルの値で置換されます。値がない属性には、インポート・ファイルに存在する値が追加されます。インポート・ファイルにはなく、属性に存在する値は変更されません。特定のキー値のすべての属性データは、ファイルの内容で置換されるか、クリアされることに注意してください。新しい値は、ファイルに記載されている順序で下部に追加されます。
このタイプのインポートは、ソース・システムから最新の変更のみを移動する場合、たとえば、買収によって新しい店舗データを追加する場合など、指定された属性値(存在する場合)のみをインポート・ファイルの値で置換するのに最も役立ちます。これはデフォルトです。
 - Replace All は、インポートした値で既存の属性値を置換します。インポート・ファイルにはなく、属性に存在する値は削除されます。
このインポート・タイプは、たとえば、ソース・システムの店舗データと同期するために毎週更新を完了するなど、完全な更新でソース・システムからの値をミラーリングする場合に最も役立ちます。

- Update は、インポート・ファイルのすべての値を属性に置換または追加します。既存の属性値はインポート・ファイルの値で置換されます。インポート・ファイルにあり、属性に存在しない値は追加されます。インポート・ファイルにはなく、属性に存在する値は変更されません。特定のキー値の属性データのみがファイルの内容で置換され、ファイル内の使用可能でない属性のデータは操作されません。インポート・ファイルにキーが存在し、属性にキーがない場合は、エラーが発生します。このタイプのインポートは、たとえば、再編成後に店舗データの残りの部分に影響を与えずに店舗マネージャを更新する場合など、すべての属性値にわたっていくつかの属性を更新する場合に最も役立ちます。
- Dateformat はオプションで、解析する有効な日付形式を指定します(例: DD/MM/YYYY、DD-
MMM-YYYY (デフォルト)、MMM d,yyyy、All)。セミコロンを使用して区切った複数の日付形式の値を指定できます。

例

```
epmautomate importRCAttributeValues Stores StoreData.csv METHOD=Replace  
DATEFORMAT="All"
```

importReconciliationAttributes

サービス管理者が [uploadFile](#) コマンドを使用して **Account Reconciliation** 環境にアップロードしたファイルから、既存の照合に照合属性をインポートします。

適用対象

Account Reconciliation

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

```
epmautomate importReconciliationAttributes FILE.CSV Period [Rules=RULE_NAME]  
[Reopen=true|false] [Dateformat=DATE_FORMAT]。ここで:
```

- FILE は、照合にインポートする照合属性を含む CSV ファイルの名前です。
- Period は、照合が属する期間を識別します。
- Rules は、オプションであり、属性のインポート後に影響を受ける照合で実行されるルールを識別します。複数のルール名を区切るには、カンマを使用します。有効な値は次のとおりです。
 - None: 影響を受ける照合に対してルールを実行しません。これはデフォルト値です。他の値とは組み合わせられません。
 - ALL: 指定された期間の照合用に定義されたすべてのルールを実行します。この値は単独で使用する必要があります。他のルール名とは組み合わせられません。
 - SET_ATTR_VAL: 事前定義済ルールを実行して、属性値を設定します。

- CRT_ALT: 事前定義済ルールを実行してアラートを作成します。
- AUTO_APP: 事前定義済ルールを実行して、照合を自動的に承認します。
- AUTO_SUB: 事前定義済ルールを実行して、照合を自動的に送信します。
- EMAIL_ON_SAVE: 事前定義済ルールを実行して、照合を更新した後に自動的に電子メールを送信します。
- Reopen は、オプションであり、インポート操作の完了時に、変更された照合を再開するかどうかを指定します。デフォルトは false です。
- Dateformat は、オプションであり、有効な日付形式を指定します(たとえば、MM-dd-yyyy、dd-MMMM-yy、MMM d, yyyy を解析します)。セミコロンを使用して区切った複数の日付形式の値を指定できます。

例

- **期間の属性値をインポートし、多くの日付形式で複数のルールを実行:**

```
epmAutomate importReconciliationAttributes Reconciliations.csv "July 2020"
Rules=SET_ATTR_VAL,CRT_ALT,AUTO_APP,AUTO_SUB" Reopen=true "Dateformat=MM-dd-yyyy;dd-MMM-yy;MMM d, yyyy"
```
- **ルールを実行せずに期間の属性値をインポート:**

```
epmAutomate importReconciliationAttributes Reconciliations.csv "July 2020"
```
- **一定期間の属性値をインポートし、該当するすべてのルールを実行し、影響を受ける照合を再開:**

```
epmAutomate importReconciliationAttributes Reconciliations.csv "July 2020"
Rules=ALL Reopen=true
```

importSnapshot

サービス環境にスナップショットの内容をインポートします。インポートするスナップショットは、デフォルトのアップロード場所にある必要があります。

サービス管理者は、[uploadFile](#) コマンドを使用してスナップショットをアップロードするか、[copySnapshotFromInstance](#) コマンドを使用してスナップショットを別のインスタンスからコピーします。

- **Planning、Planning モジュールおよびフリーフォーム・アプリケーション・スナップショットに含まれていないものは次のとおりです:**
 - 監査データ
 - ジョブ・コンソール・データ

監査データおよびジョブ・コンソール・データをターゲット環境にコピーする場合は、[cloneEnvironment](#) コマンドまたは環境のクローニング機能を使用します。カスタム期間メンバーに置き換えられ、名前が変更されたシード済期間メンバーが **Planning** ビジネス・プロセスに含まれると、スナップショットのインポートが失敗する可能性があります。たとえば、シード済の *YearTotal* 期間メンバーの名前を *unused_YearTotal* に変更し、元のシード済メンバー名(この例では *YearTotal*)を使用して別タイプの期間メンバーを追加したとします。この場合、スナップショットのインポートは失敗する可能性があります。
- スナップショットには、データ管理のステージング表データは含まれません。このデータをインポートするには、[exportDataManagement](#) コマンドと [importDataManagement](#) コマンドまたはデータ管理システム・メンテナンス・スクリプト・インタフェースを使用しま

す。 `cloneEnvironment` コマンドまたは環境のクローニング機能を使用して、データ管理ステージング表データを含む環境の同一コピーを作成できます。

このコマンドを使用して実行できるアクティビティは役割によって異なります。

- サービス管理者は、アプリケーション・アーティファクトのみを環境にインポートできません。
- アプリケーション・コンテンツをサービス環境にインポートし、アイデンティティ・ドメイン・アーティファクト(ユーザーとその事前定義済役割の割当て)を環境のアイデンティティ・ドメインにインポートするには、サービス管理者とアイデンティティ・ドメイン管理者の両方の役割が必要です。
インポート中のスナップショットで、アイデンティティ・ドメインに存在しないユーザーが参照されている場合、このコマンドによってアイデンティティ・ドメインにユーザーが作成され、コマンドで指定したデフォルト・パスワードが割り当てられます。または、コマンドでパスワードを指定しない場合は、各ユーザーに対する一時的な一意的パスワードが割り当てられます。デフォルトでは、最初のサインインの際にユーザーがパスワードをリセットする必要があります。

ノート:

- Account Reconciliation、Profitability and Cost Management および Oracle Fusion Cloud Enterprise Data Management 以外のビジネス・プロセスの場合: メタデータのロード中に、アウトラインで基本メンバーの前に共有メンバーがあるために前の試行でレコードが却下された場合、Oracle Fusion Cloud Enterprise Performance Management は、複数のロード・パスを経由することがあります。このような試行により、コマンドの処理時間が長くなる可能性があります。
- アクセス制御のグループのメンバーであるユーザーには、事前定義済の役割を割り当てる必要があります。定義済の役割が割り当てられていないユーザーをグループに割り当てようとしても、許可されません。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Enterprise Data Management Cloud、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザーおよび移行 - 管理アプリケーション役割(Oracle Enterprise Data Management Cloud および Profitability and Cost Management)

ユーザーと事前定義済役割割当てをインポートするには、アイデンティティ・ドメイン管理者役割が必要です。

使用方法

```
epmautomate importSnapshot SNAPSHOT_NAME [importUsers=true|false]
[userPassword=DEFAULT_PASSWORD] [resetPassword=true|false]。ここで:
```

- `SNAPSHOT_NAME` は、デフォルトのアップロード場所のスナップショット名です。

- `importUsers` は、オプションであり、スナップショットからユーザーとその事前定義済役割の割当てをインポートするかどうかを指定します。デフォルトは `false` です。ソース・スナップショットに新しいユーザーに関するデータが含まれている場合、または現在のユーザーに新しい役割が割り当てられている場合は、`importUsers=true` を使用して、ユーザーと事前定義済役割の割当てをアイデンティティ・ドメインにインポートします。ユーザー・ログイン値では大文字と小文字が区別されません。たとえば、ユーザー・ログイン値 `jane.doe@example.com` は、`Jane.Doe@Example.com` など、大文字と小文字のすべてのバリエーションと同じであるものとして処理されます。いずれかのバリエーションがアイデンティティ・ドメインに存在するユーザー・ログインに一致する場合、ユーザーはスナップショットからインポートされません。

ノート:

- アイデンティティ・ドメイン管理者ではないユーザーがインポート操作を実行すると、ユーザーとその事前定義済役割のインポートは失敗します。次のエラーが移行ステータス・レポートに記録されます: 外部ディレクトリ・アーティファクト `ARTIFACT_NAME` のインポートに失敗しました。ユーザー `USER_NAME` には、この操作を実行する権限はありません。この操作を実行するには、ユーザーにアイデンティティ・ドメイン管理者役割が必要です。
 - ユーザーをインポートせず、ソース・スナップショット内のユーザーがターゲット環境の事前定義済役割に割り当てられていない場合は、エラー (EPMIE-00070: 割り当てられた役割のインポート中にユーザーが見つかりませんでした)が表示されます。
-
- ユーザーの事前定義済役割に対する変更は、ソース・スナップショットで割り当てられている役割に基づいて更新されます。ただし、ソース・スナップショットの割当てと一致させるためにターゲットの役割の割当てが削除されることはありません。たとえば、`jdoe` がターゲット環境ではパワー・ユーザーという事前定義済役割に割り当てられているが、ソース・スナップショットではユーザー役割のみを持っているとします。この状況では、このコマンドによって、ターゲット環境で `jdoe` がユーザー役割に割り当てられますが、パワー・ユーザーの役割の割当ては削除されません。
 - このコマンドでは、ターゲット環境に存在するユーザーは、ソース・スナップショットに存在しなくても削除されません。たとえば、`jdoe` にはターゲット環境にアカウントがありますが、ソース・スナップショットにはこのアカウントが存在しないとします。この状況では、ターゲット環境の `jdoe` のアカウントは削除されません。
 - このコマンドでは、ターゲット環境に存在しないユーザーは追加されます。ターゲット環境の現在のユーザー・プロパティは、ソース・スナップショットと異なる場合でも更新されません。たとえば、ソース・スナップショットの `jdoe` の姓のスペルがターゲット環境で異なる場合、ターゲット環境では変更は加えられません。
 - このコマンドでは、ターゲット環境の既存のユーザーのパスワードがソース・スナップショットと異なる場合でも変更されません。
- `userPassword` は、オプションであり、アイデンティティ・ドメインで作成された新しいユーザーに割り当てるデフォルトのパスワードを示します。指定するパスワードは、パスワードの最小要件を満たしている必要があります。このパラメータの値を指定しない場合、一意の一時パスワードが各ユーザーに割り当てられます。
 - `resetPassword` は、オプションであり、新しいユーザーが最初のログイン時にパスワードを変更する必要があるかどうかを示します。デフォルトは `true` であり、新しいユーザーは最初のサインイン時にパスワードを変更する必要があります。この値が `true` に設定され

ている場合、新しいユーザーは、パスワードの変更を求めるアカウントのアクティブ化の電子メールを受信します。

例

- アプリケーション・アーティファクトのみをインポート: `epmautomate importSnapshot April16FullApp`
- アプリケーションとアイデンティティ・ドメイン・アーティファクトをインポート(サービス管理者とアイデンティティ・ドメイン管理者の役割が必要):
 - 新しいユーザーごとに一意の一時パスワードを割当て、初めてサインインした後にパスワードをリセットするように強制します。
`epmautomate importSnapshot April16FullApp importUsers=true`
 - 特定のパスワードを割当て、ユーザーが選択した場合はパスワードを変更しないようにします。実稼働環境へのインポートには推奨されません。
`epmautomate importSnapshot April16FullApp importUsers=true
userPassword=P@ssw0rd1 resetPassword=false`

importSupplementalCollectionData

補足コレクション・データをファイルからアプリケーションにインポートします。

uploadFile コマンドを使用して、データを含むファイルをデフォルトのアップロード場所にアップロードします。インポート・ファイルの形式は次のとおりです:

```
#Workflow
Workflow_Dimension_1_Name,Workflow_Dimension_2_Name,Workflow_Dimension_n_Name
Workflow_Dimension_1_Member,Workflow_Dimension_2_Member,Workflow_Dimension_n_Member
#Collection
Collection_Attribute_1,Collection_Attribute_2,Collection_Attribute_n
Record1_Attr_Value_1,Record1_Attr_Value_2, Record1_Attr_Value_n
```

次に例を示します。

```
#Workflow
Entity
9100
#Collection
Custody Account Code,Trade Currency Code,Account Description,Base Currency Code,CIC Code,IFRS 13 Tier,SII Portfolio Type,WPM Detailed NAV ID,WPM Asset Description
1,,,,111,,,,6
```

適用対象

Financial Consolidation and Close,および Tax Reporting

必要な役割

サービス管理者

使用方法

ノート:

すべてのコマンド・パラメータを二重引用符で囲む必要があります。

```
epmautomate importSupplementalCollectionData "FILE_NAME" "COLLECTION_NAME" "YEAR"
"PERIOD" "[FREQUENCY_DIMENSION=MEMBER]"。ここで:
```

- FILE_NAME は、デフォルトのアップロード場所で使用可能な CSV ファイルで、適切にフォーマットされた補足データが含まれています。
- COLLECTION_NAME はコレクション名で、ここにファイルの補足データがインポートされます。
- YEAR は、コレクションに使用される年ディメンション・メンバーです。
- PERIOD は、コレクションに使用される期間ディメンションの名前です。
- FREQUENCY_DIMENSION はオプションで、コレクションに使用される頻度ディメンションの名前です。"FREQUENCY_DIMENSION1=MEMBER" "FREQUENCY_DIMENSION2=MEMBER"形式で、必要な数の頻度ディメンションを指定できます。

例

```
epmautomate importSupplementalCollectionData "datafile.csv" "Journal Data
Collection" "FY20" "Jan" "Account=PAYROLL" "JournalID=LNR 113"
```

importSupplementalData

補足データをファイルからアプリケーションにインポートします。

[uploadFile](#) コマンドを使用して、データを含むファイルをデフォルトのアップロード場所にアップロードします。

適用対象

Financial Consolidation and Close および Tax Reporting

必要な役割

サービス管理者

使用方法

ノート:

すべてのコマンド・パラメータを二重引用符で囲む必要があります。

```
epmautomate importSupplementalData "FILE_NAME" "DATA_SET_NAME" "YEAR"
"PERIOD_NAME" "SCENARIO_NAME"。ここで:
```

- FILE_NAME は、デフォルトのアップロード場所で使用可能な CSV ファイルで、適切にフォーマットされた補足データが含まれています。
- DATA_SET_NAME はデータセット名で、ここにファイルの補足データがインポートされます。
- YEAR は、データセットがデプロイされる年です。
- PERIOD_NAME は、データセットのデプロイ先の期間名です。
- SCENARIO_NAME は、データセットのデプロイ先のシナリオ名です。

例

```
epmautomate importSupplementalData "DatasetImport.csv" "EmployeeDataSet" "FY17"  
"Jan" "Actual"
```

importTemplate

profitinbox に存在するテンプレート・ファイルからインポートしてアプリケーション構造を作成します。

サービス管理者は、[uploadFile](#) コマンドを使用して、テンプレート・ファイルを profitinbox にアップロードします。

適用対象

Profitability and Cost Management

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

```
epmautomate importTemplate APPLICATION_NAME File_Name  
isApplicationOverwrite=true|false。ここで:
```

- APPLICATION_NAME は、テンプレートをインポートして作成する Profitability and Cost Management アプリケーションの名前です
- File_Name は、アプリケーション・テンプレートが含まれる .ZIP ファイルの名前です。このファイルは profitinbox に存在する必要があります。
- isApplicationOverwrite は、既存のアプリケーション(存在する場合)を上書きするかどうかを指定します。このパラメータ値はすべて小文字で指定します。

例

```
epmautomate importTemplate BksML12 template1.zip isApplicationOverwrite=true
```

importTMAttributeValues

Account Reconciliation トランザクション照合のグループ属性に値をインポートします。

サービス管理者は、このコマンドを実行する前に、[uploadFile](#) コマンドを使用して、トランザクション照合属性の値が格納されたインポート・ファイルを環境にアップロードします。

適用対象

Account Reconciliation

必要な役割

- サービス管理者
- パワー・ユーザー(ACL を介して提供される追加のセキュリティが必要)

使用方法

`epmautomate importTMAttributeValues ATTRIBUTE_NAME FILE_NAME [METHOD=REPLACE|REPLACE ALL|UPDATE] [DATEFORMAT=DD/MM/YYYY|DD-MMM-YYYY|MMM d,yyyy|All]`。ここで:

- `ATTRIBUTE_NAME` は、値のインポート先グループ属性の名前です。
- `FILE_NAME` は、トランザクション照合への値のインポート元 CSV インポート・ファイルです。
- `METHOD` はオプションで、値のインポート方法を示します。有効な値:
 - `Replace` は、インポート・ファイルのすべての値をトランザクション照合のグループ属性に追加します。既存の属性値はインポート・ファイルの値で置換されます。値がない属性には、インポート・ファイルに存在する値が追加されます。インポート・ファイルにはなく、属性に存在する値は変更されません。特定のキー値のすべての属性データは、ファイルの内容で置換されるか、クリアされることに注意してください。新しい値は、ファイルに記載されている順序で下部に追加されます。このタイプのインポートは、ソース・システムから最新の変更のみを移動する場合、たとえば、買収によって新しい店舗データを追加する場合など、指定された属性値(存在する場合)のみをインポート・ファイルの値で置換するのに最も役立ちます。これはデフォルトです。
 - `Replace All` は、インポートした値で既存の属性値を置換します。インポート・ファイルにはなく、属性に存在する値は削除されます。このインポート・タイプは、たとえば、ソース・システムの店舗データと同期するために毎週更新を完了するなど、完全な更新でソース・システムからの値をミラーリングする場合に最も役立ちます。
 - `Update` は、インポート・ファイルのすべての値を属性に置換または追加します。既存の属性値はインポート・ファイルの値で置換されます。インポート・ファイルにあり、属性に存在しない値は追加されます。インポート・ファイルにはなく、属性に存在する値は変更されません。特定のキー値の属性データのみがファイルの内容で置換され、ファイル内の使用可能でない属性のデータは操作されません。インポート・ファイルにキーが存在し、属性にキーがない場合は、エラーが発生します。このタイプのインポートは、たとえば、再編成後に店舗データの残りの部分に影響を与えずに店舗マネージャを更新する場合など、すべての属性値にわたっていくつかの属性を更新する場合に最も役立ちます。
- `Dateformat` はオプションで、解析する有効な日付形式を指定します(例: `DD/MM/YYYY`、`DD-MMM-YYYY` (デフォルト)、`MMM d,yyyy`、`All`)。セミコロンを使用して区切った複数の日付形式の値を指定できます。

例

```
epmautomate importTMAttributeValues TMGA TMGA.csv METHOD=Replace DATEFORMAT="All"
```

importTmPremappedTransactions

特定のデータ・ソースの場合、Account Reconciliation リポジトリのファイルから、事前マップ済トランザクション・データをトランザクション照合にインポートします。

サービス管理者は、[uploadFile](#) コマンドを使用して、トランザクション・ファイルをサービスにアップロードします。

このコマンドは、インポート・ステータスおよびインポート・ログ・ファイル名をコンソールに表示します。サービス管理者は [downloadFile](#) コマンドを使用して、ログ・ファイルをローカル・コンピュータにダウンロードします。

インポートのファイル・フォーマット要件およびデータのインポートの詳細は、*Account Reconciliation* を使用した勘定科目の照合のデータのインポートを参照してください。

ノート:

- 一度に 1 つの照合タイプのみトランザクションをインポートできます。ただし、様々な照合タイプへの並列インポートを実行できます。
- ジョブ画面の場合とは異なり、事前マップ済トランザクション・データは一度に 1 つのファイルからのみインポートできます。
- すべてのデータ・ソースの事前マップ済トランザクションをインポートした後で、runautomatch コマンドを実行します。

適用対象

Account Reconciliation

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

```
epmautomate importTmPremappedTransactions MATCH_TYPE DATA_SOURCE FILE_NAME  
[DATE_FORMAT]。ここで:
```

- MATCH_TYPE は、Account Reconciliation で定義された照合タイプです。
- DATA_SOURCE は、指定した照合タイプと関連付けられたデータ・ソースの識別子です。
- FILE_NAME は、インポートするトランザクションを含む CSV ファイルの名前です。サービスでこのファイルが使用可能な必要があります。
- DATE_FORMAT はオプション・パラメータで、トランザクション・インポート・ファイルに含まれる日付フィールドのフォーマットを示します。デフォルトは dd-MMM-YYYY です。サ

ポートされるその他の日付フォーマットは、MM/dd/yyyy、dd/MM/yyyy、MM-dd-yyyy、d-M-yyyy および MMM d.yyyy です。

例

```
epmautomate importTmPremappedTransactions "INTERCOMPANY" "AP" dailydata.csv d-M-YYYY
```

importValidIntersections

有効な交差定義を持つ 1 つの Excel ファイルを含む ZIP ファイルから、有効な交差グループをビジネス・プロセスにインポートします。このコマンドを実行する前に、[uploadFile](#) コマンドを使用してインポート・ファイルを環境にアップロードします。

インポートする ZIP ファイルには、有効な交差を正常にインポートするための 2 つのワークシート(ルールとサブ・ルール)を含む Excel ファイルが含まれている必要があります。最初のシートであるルールには、交差グループ、含まれるディメンション、および未指定の有効、追加のディメンションが必要などのプロパティを定義する必要があります。2 番目のシートであるサブ・ルールは、メンバーの選択と除外を提供する必要があります。詳細は、*Planning* の管理の次のトピックを参照してください。

- アンカーおよび非アンカー・ディメンション
- 有効な交差の例

インポート・ファイル形式のテンプレートを取得する最良の方法は、アプリケーションから有効な交差をエクスポートすることです。次の図にサンプル形式を示します。

	A	B	C	D	E	F	G	H	I	J
	Name	Position	Description	Enabled	Anchor Dim Name	Anchor Dimension	Dim1	Dim1 Required	Dim2	Dim2 Required
1						Apply				
2	Region - Product	1		true	Entity	true	Product	false		
3	Region-Product-VI-Copy	2		true	Entity	true	Product	false		
4										
5										
6										

	A	B	C	D	E	F	G
	Name	Anchor Members	Anchor Exclusion	Dim1 Members	Dim1 Exclusion	Dim2 Members	Dim2 Exclusion
1							
2	Region - Product	Children(403)	410,421	IDescendants(P_TP)			
3	Region - Product	410		IDescendants(P_TP)	P_260,P_270,P_280		
4	Region - Product	421		IDescendants(P_TP)	P_220,P_250		
5	Region-Product-VI-Copy	Children(403)	410,421	IDescendants(P_TP)			
6	Region-Product-VI-Copy	410		IDescendants(P_TP)	P_260,P_270,P_280		
7	Region-Product-VI-Copy	421		IDescendants(P_TP)	P_220,P_250		
8							

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

```
epmautomate importValidIntersections FILE_NAME.zip
[ErrorFile=ERROR_FILE_NAME.txt]。ここで:
```

- FILE_NAME は、有効な交差定義 Excel ファイルを含む ZIP ファイルの名前です。
- ErrorFile は、オプションで、エラー・レコードが書き込まれるテキスト・ファイルの名前を識別します。このパラメータ値を指定していない場合は、エラー・ファイルが自動的に生成されます。この名前はジョブ・コンソールで確認できます。

例

```
epmautomate importValidIntersections VI_Import_File.zip
ErrorFile=VI_Import_Log.txt
```

invalidLoginReport

OCI (Gen 2)環境で、無効なログイン・レポートを作成します。このレポートには、環境に指定された監査保持期間に対応する指定された期間において、環境への失敗したサインインの試行がリストされます。デフォルトの保持期間は 30 日です。Oracle Cloud Identity Console で「**監査保持期間(日)**」設定を変更して、最大 90 日まで拡張できます。90 日を超える期間の監査データを保持するには、このレポートおよび**役割割当監査レポート**を定期的にダウンロードしてアーカイブします。

無効なログイン・レポートには、次のような情報が含まれます:

- サインインを試行したユーザーのユーザー名
- ユーザーがサインインを試行したリモート IP アドレス
- サインイン試行のタイムスタンプ

このレポートには、相当する Identity Cloud Service への不成功となったログイン試行がすべて表示されます。これらがすべて 1 つの Oracle Fusion Cloud Enterprise Performance Management インスタンスに関連しているとはかぎりません。

	A	B	C
1	User Name	IP Address	Access Date and Time
2	john.doe@example.com	xxx.xx.xx.xx5	July 15, 2021 11:14:58 UTC
3	jane.doe@example.com	xxx.xx.xx.xx9	July 15, 2021 11:14:58 UTC
4	john.smith@example.com	xxx.xx.xx.xx3	July 15, 2021 11:14:57 UTC

[downloadFile](#) コマンドを使用して、レポートをローカル・コンピュータにダウンロードします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Strategic Workforce Planning および Sales Planning。

必要な役割

アイデンティティ・ドメイン管理者と任意の事前定義済役割

使用方法

`epmAutomate invalidLoginReport FROM_DATE TO_DATE FILE_NAME.CSV`。ここで:

- `FROM_DATE` は、レポートを生成する期間の開始日(YYYY-MM-DD 形式)を示します。この日付は、Oracle Cloud Identity Console で指定した監査データ保持期間内である必要があります。
- `TO_DATE` は、レポートを生成する期間の終了日(YYYY-MM-DD 形式)を示します。
- `FILE_NAME` は、レポートの CSV ファイルの名前です。

Note:

このレポートは、過去 90 日間についてのみ生成できます。

例

```
epmAutomate invalidLoginReport 2021-06-01 2021-06-30 invalidLoginReport.CSV
```

listBackups

環境の利用可能なバックアップ・スナップショットを一覧表示して、特定のバックアップが利用可能かどうかを判断し、アーカイブまたは現在の環境を自分で復元するために使用できるようにします。

特定のバックアップを復元する前に、このコマンドを使用して、必要なバックアップが Oracle Object Storage で使用可能かどうかを確認してください。バックアップが利用可能な場合は、[restoreBackup](#) コマンドを実行して、バックアップを復元(ご使用の環境にコピー)できます。バックアップをコピーした後、[importSnapshot](#) コマンドを使用してインポートできます。環境のセルフサービス復元により、処理時間を節約できます。

Narrative Reporting 以外のサービスの場合、このコマンドは、使用可能なバックアップ・スナップショット(日次メンテナンス・プロセスによって作成される)を一覧表示します。これには、命名規則 `YYYY-MM-DDTHH:MM:SS/Artifact_Snapshot.zip` が使用されます。たとえば、`2022-02-16T21:00:02/Artifact_Snapshot.zip` です。**Narrative Reporting** の場合、使用可能なスナップショットは命名規則 `YYYY-MM-DDTHH:MM:SS/EPRCS_Backup.tar.gz` を使用します。たとえば、`2022-02-16T21:00:02/EPRCS_Backup.tar.gz` です。どちらの場合も、タイムスタンプはスナップショットが作成された UTC 時刻を反映しています。次の図は、サンプルのコマンド出力を示しています。

```
c:\Oracle\EPM Automate\bin>epmautomate listbackups

2022-03-04T06:37:51/Artifact_Snapshot.zip
2022-03-08T06:32:01/Artifact_Snapshot.zip
2022-03-09T12:08:05/Artifact_Snapshot.zip
2022-03-10T06:37:48/Artifact_Snapshot.zip
2022-03-15T06:21:28/Artifact_Snapshot.zip
2022-03-16T06:20:52/Artifact_Snapshot.zip
2022-03-16T12:13:56/Artifact_Snapshot.zip

Total 7
```

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザーおよび移行 - 管理アプリケーション役割(Oracle Enterprise Data Management Cloud および Profitability and Cost Management)

使用方法

```
epmAutomate listBackups
```

例

```
epmAutomate listBackups
```

listFiles

デフォルトの場所、データ管理のフォルダ、および profitinbox/profitoutbox にあるファイル名をリストします(Profitability and Cost Management)。

このコマンドは、増分およびバックアップ・エクスポート・ファイル、移行のスナップショット、アクセス・ログ、アクティビティ・レポートもリストします。この図は、コマンド出力の切り捨てられたバージョンを示しています。

```
apr/2022-01-27 05_23_36/activityreport.json
apr/2022-01-28 05_24_07/2022-01-28 05_24_07.html
apr/2022-01-28 05_24_07/access_log.zip
apr/2022-01-28 05_24_07/activityreport.json
apr/2022-01-29 05_24_06/2022-01-29 05_24_06.html
apr/2022-01-29 05_24_06/access_log.zip
outbox/Vision_99.dat
roleassign.csv
RoleAssignment.csv
sanity_no_data_22-01-18.zip
U-1.csv
U2.csv
user1.csv
user12.csv
users12.csv
Uservariables-MemberFormula.zip
UsrGrpReport.CSV
Vision_DTsetup.zip
VisionADCForms2010.zip
```

このコマンドが環境のスナップショットの生成中、たとえば日次メンテナンス中に実行された場合、現在のスナップショットはリストされません。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザーおよび移行 - 管理アプリケーション役割(Oracle Enterprise Data Management Cloud および Profitability and Cost Management)

使用方法

```
epmautomate listFiles
```

例

```
epmautomate listFiles
```

loadData

profitinbox にあるファイルを使用して計算キューブにデータをロードします。

サービス管理者は、[uploadFile](#) コマンドを使用して、ファイルを profitinbox にロードします。

適用対象

Profitability and Cost Management

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

epmautomate loadData APPLICATION_NAME dataFileName=File_Name PARAMETER=VALUE。ここで:

- APPLICATION_NAME は、データをロードする Profitability and Cost Management アプリケーションの名前です
- dataFileName=File_Name には、profitinbox にあるデータ・ロード・ファイルを指定します。データ・ファイル名は二重引用符で括弧する必要があります。
- PARAMETER=VALUE には、データをロードするためのランタイム・パラメータとその値を指定します。パラメータと値のペアをプロセスの必要に応じて指定します。有効なパラメータと値は次のとおりです。
 - clearAllDataFlag=true|false は、アプリケーション・キューブの既存データをクリアするかどうかを指定します
 - dataLoadValue=OVERWRITE_EXISTING_VALUES|ADD_TO_EXISTING は、既存データの処理方法を指定します

例

```
epmautomate loadData BksML12 dataFileName="data1.txt"clearAllDataFlag=true
dataLoadValue="OVERWRITE_EXISTING_VALUES"
```

loadDimensionViewpoint

定義済ロードを使用して、バインドされていない、バインドされた、または部分的にバインドされた 1 つのビューポイント(ノードのサブセット)に、ロード・ファイルからデータをロードします。

ロード・ファイル(CSV、Excel (XLSX)ファイル、または 1 つの CSV または XLSX ファイルが格納された ZIP ファイル)は、ビューポイントをロードする環境で使用可能である必要があります。uploadFile または copyFileFromInstance コマンドを使用して、ロード・ファイルを環境にアップロードできます。

デフォルトのロードを使用してビューポイントをロードするには、loadViewpoint コマンドを使用します。

適用対象

Oracle Fusion Cloud Enterprise Data Management。

必要な役割

サービス管理者

使用方法

```
epmautomate loadDimensionViewpoint APPLICATION_NAME DIMENSION_NAME FILE_NAME
LOAD_NAME [loadOption=ReplaceNodes|Merge] [purpose="PURPOSE"]。ここで:
```

- APPLICATION_NAME は、Oracle Enterprise Data Management Cloud アプリケーションの名前です。
- DIMENSION_NAME は、ロードするディメンションの名前です。
- FILE_NAME は、ビューポイントのロード元のファイルの名前(CSV、XLSX または ZIP の拡張子付き)です。
- LOAD_NAME は、ビューポイントのロードに使用する定義済ロードの名前です。
- loadOption はオプションで、ビューポイントのロード方法を識別します。有効なロード・オプションは、次のとおりです:
 - ReplaceNodes は、ロード・ファイルからの関係以外のすべての関係(孤立関係または同じ階層セットを使用して他のビューポイントによって使用される関係を含む)を階層からクリアします。これがデフォルトのロード・タイプです。
 - Merge は、増分変更のみを処理して既存の関係を保持します。
- purpose はオプションで、ビューポイントをロードする理由を識別する、二重引用符で囲まれたテキスト文字列です。

例

- 既存の階層を置き換えてビューポイントをロード: `epmautomate loadDimensionViewPoint "Data Warehouse" Citizen Data_Warehouse_Citizen_20241108.csv DW-Citizen`
- 増分変更を指定のロードとマージしてビューポイントをロード: `epmautomate loadDimensionViewPoint "Data Warehouse" Citizen Data_Warehouse_Citizen_20241108.csv DW-Citizen loadOption=Merge purpose="Load Test"`

loadDimData

profitinbox にある 1 つ以上のファイルのディメンション・メタデータをアプリケーションにロードします。

サービス管理者は、[uploadFile](#) コマンドを使用して、メタデータ・ファイルを profitinbox にロードします。

適用対象

Profitability and Cost Management

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

`epmautomate loadDimData APPLICATION_NAME dataFileName=File_Name [stringDelimiter="DELIMITER"] [acceptableDecreasePercentage=PERCENTAGE]`。ここで:

- APPLICATION_NAME は、ディメンション・メタデータのロード先の Profitability and Cost Management アプリケーションの名前です

- dataFileName には、**profitinbox** にあるディメンション・メタデータ・ロード・ファイルを指定します。複数のファイルからメタデータをロードするには、区切り文字で区切ってファイル名をリストします
- stringDelimiter は、オプションで、メタデータ・ファイル名を区切るために使用される区切り文字を指定します。区切り文字は二重引用符で囲む必要があります。
- acceptableDecreasePercentage は、オプションで、操作で使用できるメンバー数のパーセンテージ(%記号なし)の差を指定します。受信ファイルの新しいメンバー数が既存のメンバー数より少ない場合、この値は、許容できるパーセンテージの減少を表します。メンバー数の偏差がこのパーセンテージを超えた場合、ディメンション・データのロードは失敗します。

例

```
epmautomate loadDimData BksML12 dataFileName="dimdata1.txt#dimdata1.txt"  
stringDelimiter="#" acceptableDecreasePercentage=5
```

loadViewpoint

ロード・ファイルから **Oracle Fusion Cloud Enterprise Data Management** アプリケーションにビューポイント(ノードのサブセット)をロードします。

ビューポイント・ロードを使用すると、未バインド、バインド済または一部バインド済のビューポイントにデータをロードできます。1つの **CSV** または **XLSX** ファイルを含むビューポイント・ロード・ファイル(**CSV**、**Excel (XLSX)**ファイルまたは **ZIP** ファイル)が、ビューポイントをロードする環境で使用可能である必要があります。[uploadFile](#) または [copyFileFromInstance](#) コマンドを使用して、ロード・ファイルを環境にアップロードできます。

適用対象

Oracle Enterprise Data Management Cloud

必要な役割

サービス管理者

使用方法

```
epmautomate loadViewpoint VIEW VIEWPOINT PURPOSE FILE_NAME  
[loadType=ReplaceNodes|Merge]
```

ここで:

- VIEW は、**Oracle Enterprise Data Management Cloud** ビューの名前です。
- VIEWPOINT は、ロードするビューポイントの名前です。
- PURPOSE は二重引用符に囲まれたテキスト文字列で、ビューポイントがロードされた理由を示します。
- FILE_NAME は、ビューポイントのロード元のファイルの名前(拡張子付き)です。
- loadType はオプションで、ビューポイントのロード方法を識別します。有効な値は Merge および ReplaceNodes です。
 - Merge を使用して、増分変更を処理して既存の関係を保持します。

- ReplaceNodes を使用して、ロード・ファイルからの関係以外のすべての関係(孤立関係または同じ階層セットを使用して他のビューポイントによって使用される関係を含む)を階層からクリアします。これがデフォルトのロード・タイプです。

例

- **増分変更のマージ:** `epmautomate loadViewpoint USOperations Entity "Daily Upstream Load" data_Entity.CSV loadType=Merge`
- **既存の階層の置換:** `epmautomate loadViewpoint USOperations Entity "Replace US Operations data" data_Entity.CSV`

login

環境へのセキュアな接続を確立します。このコマンドは、プレーン・テキストのパスワード、あるいはパスワードまたは OAuth 2.0 リフレッシュ・トークンが格納されている暗号化されたパスワード・ファイルを使用した環境へのサインインをサポートしています。OAuth 2.0 リフレッシュ・トークンを使用したログインは、OCI (Gen 2)環境でのみサポートされます。

サインインしてセッションを開始します。これはサインアウトするまでアクティブなままです。

ノート:

- このコマンドは、多要素認証(MFA)を使用した基本認証用に設定されたユーザーに対してサポートされていません。
- EPM 自動化は、組織の SSO 資格証明を使用したサインインをサポートしていません。
- EPM 自動化は SOCKS プロキシでは機能しません。HTTP または HTTPS プロキシでのみ機能します。
- このコマンドをバッチ・ファイルで使用してアクティビティを自動化するときは、暗号化されたパスワードまたは OAuth 2.0 リフレッシュ・トークンを使用し、バッチ・ファイルにクリア・テキスト・パスワードを記録しないようにしてください。
- Windows コンピュータでは、このコマンドは、接続の確立を妨げる可能性のある欠落しているプロキシ・サーバーの中間セキュリティ証明書を自動的に識別し、C:\Oracle\EPM Automate の下にインストールされている JRE に追加します。これにより、プロキシ・サーバーを使用したインターネットへのアクセス時のセキュリティ証明書に関連するログイン・エラーが防止されます。
Linux コンピュータでは、login コマンドは、プロキシ・サーバーから欠落しているセキュリティ証明書を識別し、それをダウンロードして、エラーを表示します。root アクセス権を持つユーザーは、ダウンロードした証明書を、JAVA_HOME 環境変数で識別される使用可能な JRE にインストールできます。次の情報ソースを参照してください。

- [Java Runtime Environment および EPM 自動化](#)
- [Keytool Java ドキュメント](#)

古いバージョンを使用している場合は、サインイン時に、EPM 自動化のアップグレードを促すメッセージが表示されます。[upgrade](#) コマンドを使用して、インストールをサイレント・アップグレードできます。

[addUsers](#)、[removeUsers](#)、[assignRole](#)、または [unassignRole](#) コマンドを実行する予定の場合、OAuth リフレッシュ・トークンを使用してログインしないでください。これらのコマンドでは、基本認証を使用する必要があります。他のすべてのコマンドは、OCI (GEN 2)環境の OAuth 2.0 で機能します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

使用方法

- **暗号化されていないパスワードの使用:** `epmautomate login USERNAME PASSWORD URL [IDENTITYDOMAIN] [ProxyServerUserName=PROXY_USERNAME ProxyServerPassword=PROXY_PASSWORD ProxyServerDomain=PROXY_DOMAIN] [KeystorePassword=PASSWORD]`
- **暗号化されたファイルの使用:** `epmautomate login USERNAME PASSWORD_FILE URL [IDENTITYDOMAIN] [ProxyServerUserName=PROXY_USERNAME] [ProxyServerPassword=PROXY_PASSWORD] [ProxyServerDomain=PROXY_DOMAIN] [KeystorePassword=KEYSTORE_PASSWORD]`

これらのコマンドでは:

- `USERNAME` は、ユーザーのユーザー名です。
- `PASSWORD` は、ユーザーのパスワードです。
- `PASSWORD_FILE` は、ユーザーの暗号化されたパスワードまたは OAuth 2.0 リフレッシュ・トークンを保存するファイルの名前と場所です。[encrypt](#) コマンドを参照してください。
- `URL` は、接続先の環境のベース URL です。Oracle Fusion Cloud Enterprise Performance Management の URL のかわりに、カスタム URL またはバニティ URL を使用できます。管理者用スタート・ガイドのバニティ URL の使用を参照してください

ノート:

API ゲートウェイまたはリーバース・プロキシを使用する場合は、Cloud EPM URL のかわりに、その URL と環境に定義されたコンテキストを使用します。

- `IDENTITYDOMAIN` はオプションで、環境のアイデンティティ・ドメインです。

この値は、Cloud EPM URL から自動的に導出されます。指定した値はすべて無視されま
す。

- ProxyServerUserName は、インターネットへのアクセスを制御する HTTP プロキシ・サー
バーとの安全なセッションを認証するためのユーザー名です。ドメイン名の接頭辞を付け
ずにユーザー名を指定します。必要なのは、ネットワークに対してプロキシ・サーバーで
の認証が有効になっている場合のみです。
- ProxyServerPassword は、プロキシ・サーバーでユーザーを認証するパスワードです。必
要なのは、ネットワークに対してプロキシ・サーバーでの認証が有効になっている場合の
みです。このパスワードは暗号化できます。encrypt コマンドを参照してください。このパ
スワードが暗号化されている場合は、PASSWORD_FILE から読み込まれます。
- ProxyServerDomain は、HTTP プロキシ・サーバー用に定義されたドメインの名前(サー
バー名またはプロキシ・サーバーのホスト名ではなく)です。必要なのは、ネットワークに対
してプロキシ・サーバーでの認証が有効になっており、プロキシ・サーバー・ドメインが
構成されている場合のみです。
- KeystorePassword は省略可能であり、プロキシ・サーバーのセキュリティ証明書をインポ
ートするために必要なキーストア・パスワードです。このパラメータは、Windows でのみ
使用し、プロキシ・サーバーを使用してインターネット・アクセスをチャネル化する環境
で次のエラーが発生した場合にのみ使用してください。

```
EPMAT-7: Unable to connect as few SSL certificates are missing in the keystore
```

```
EPMAT-7: Unable to connect as above-mentioned SSL certificates are missing in the keystore
```

ノート:

EPM 自動化は、コンピュータ上の HTTP または HTTPS プロキシ設定を検出して使
用します。

EPM 自動化では、プロキシ・サーバーに接続するための次の認証メカニズムをサポ
ートしています:

- 基本認証
- ダイジェスト認証
- Kerberos 認証
- ネゴシエート・プロキシ認証
- NTLM 認証

使用可能な認証方法とその構成は、使用するプロキシ・サーバーによって異なりま
す。

Linux コンピュータでは、プロキシ設定で、プロキシ・サーバーによる認証が必要と
されている場合は、このコマンドのパラメータとしてプロキシ・サーバーのドメイ
ン、ユーザー名およびパスワードを入力する必要があります。プロキシ・サーバーの
ドメイン名と資格証明については、ネットワーク管理者に連絡してください。

例

- 暗号化されていない Cloud EPM パスワードを使用し、プロキシ認証は使用しない:
epmautomate login serviceAdmin P@ssword1 https://test-cloud-
pln.pbcs.us1.oraclecloud.com

- 暗号化されたファイルを使用し、プロキシ認証は使用しない:
epmautomate login serviceAdmin C:\mySecuredir\password.epw https://test-cloud-pln.pbcs.us1.oraclecloud.com
- プロキシ・サーバーでの認証がサーバー・ドメイン付きで有効になっている場合に、暗号化されたファイルを使用:
epmautomate login serviceAdmin C:\mySecuredir\password.epw https://test-cloud-pln.pbcs.us1.oraclecloud.com ProxyServerUserName=john.doe@example.com ProxyServerPassword=example ProxyServerDomain=example
- プロキシ・サーバーでの認証がサーバー・ドメインなしで有効になっている場合に、暗号化されたファイルを使用:
epmautomate login serviceAdmin C:\mySecuredir\password.epw https://test-cloud-pln.pbcs.us1.oraclecloud.com ProxyServerUserName=john.doe@example.com ProxyServerPassword=example
- プロキシ・サーバーでの認証がサーバー・ドメイン付きで有効になっている場合は、暗号化された **Cloud EPM** およびプロキシ・サーバー・パスワードを使用:
epmautomate login serviceAdmin C:\mySecuredir\password.epw https://test-cloud-pln.pbcs.us1.oraclecloud.com ProxyServerUserName=john.doe@example.com ProxyServerDomain=example
- プロキシ・サーバーでの認証がサーバー・ドメインなしで有効になっている場合は、暗号化された **Cloud EPM** およびプロキシ・サーバー・パスワードを使用:
epmautomate login serviceAdmin C:\mySecuredir\password.epw https://test-cloud-pln.pbcs.us1.oraclecloud.com ProxyServerUserName=john.doe@example.com
- 暗号化されたファイルを **APIGEE API** ゲートウェイ付きで使用:
epmautomate login serviceAdmin C:\mySecuredir\password.epw https://exampleapigee.apigee.com/epm example_ID_DOM
- バニティ URL の使用:
epmautomate login serviceAdmin C:\mySecuredir\password.epw https://rebrand.ly/Automate

logout

環境との現在の接続を終了します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

使用方法

```
epmautomate logout
```

例

```
epmautomate logout
```

maskData

アプリケーション・データにマスクを適用して、データのプライバシーを確保します。このコマンドは、機密データをアプリケーション開発者に対して非表示にするために、テスト環境のみで使用してください。

警告: このコマンドは現在のアプリケーション・データをランダム化して無意味にするため、本番環境では使用しないでください。このコマンドの結果は元に戻せません。サービス環境のデータに誤ってマスクを適用した場合は、バックアップまたはメンテナンス・スナップショットからデータを復元する必要があります。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmautomate maskData [-f]`。ここで、`-f` は、ユーザーの確認なしでマスク適用プロセスを強制的に開始するオプションです。`-f` オプションを使用していない場合は、操作の確認を求められます。

例

```
epmautomate maskData [-f]
```

mergeDataSlices

集約ストレージ・キューブのすべての増分データ・スライスをメイン・データベース・スライスにマージし、オプションで値がゼロのセルを削除します。

適用対象

Planning、Planning モジュール、フリーフォーム、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmautomate mergeDataSlices CUBE_NAME [keepZeroCells=true|false]`。ここで:

- `CUBE_NAME` は、すべてのデータ・スライスがマージされる集約ストレージ・キューブを識別します。

- keepZeroCells はオプションで、値がゼロのセルを削除する(セルの値がゼロになるリージョンからデータを論理的にクリアする)かどうかを指定します。デフォルトは true です

例

```
epmautomate mergeDataSlices repl keepZeroCells=false
```

mergeSlices

増分データ・スライスを中心に、データベース・キューブにマージし、必要に応じて、値として 0 (ゼロ)を含む Oracle Essbase セルを削除し、キューブをコンパクトにします。

0 を含むセルを削除すると、キューブのパフォーマンスが最適化されます。

適用対象

Profitability and Cost Management

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

epmautomate mergeSlices applicationName [removeZeroCells=true|false]。ここで:

- applicationName は、Profitability and Cost Management アプリケーションの名前です。
- removeZeroCells は、必要に応じて、0 を含むセルを削除するかどうかを指定します。このパラメータのデフォルト値は、false です。

例

- 0 を含むセルを削除せずにスライスをマージ:
 - epmautomate mergeSlices BksML30
 - epmautomate mergeSlices BksML30 removeZeroCells=false
- スライスをマージして 0 を含むセルを削除: epmautomate mergeSlices BksML30 removeZeroCells=true

optimizeASOCube

ASO キューブからのデータ抽出用の集約ビューを選択するための問合せのパフォーマンスを最適化します。

データ・サイズが大きいためにデフォルトの集約ではデータ抽出またはレポートのニーズを満たすのに不十分とみなされる場合に、このコマンドを使用して、ASO キューブの問合せ最適化操作を実行できます。一般的な最適化プロセスは次のとおりです:

- デフォルトおよび問合せベースの集約を削除します。
- 問合せトラッキングを開始します。
- Profitability and Cost Management 問合せマネージャ、Oracle Smart View for Office またはデータ管理のサンプル問合せ、および Oracle Essbase をトレーニングするために最適化が望ましい問合せのタイプを表すその他の MDX 問合せを実行します。

- 最適化された問合せまたはデフォルトの問合せに基づいて集約を作成します。

適用対象

Profitability and Cost Management

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

`epmautomate optimizeASOCube APPLICATION_NAME OPTIMIZATION_TYPE`。ここで:

- `APPLICATION_NAME` は、**ASO** キューブが属する **Profitability and Cost Management** アプリケーションの名前です。
- `OPTIMIZATION_TYPE` は、キューブ最適化操作です。使用可能な値は次のとおりです。
 - `clearAggregations` は、デフォルトおよび問合せベースのビューを削除します。
 - `createAggregations` は、デフォルトの **Essbase** 集約ビューを作成します。このオプションを使用して、問合せベースの集約ではなくデフォルトの集約を実行します
 - `startQueryTracking` は、問合せトラッキングを開始します。
 - `stopQueryTracking` は、問合せトラッキングを停止します。このオプションを使用して、**Essbase** による最適化情報の収集を停止します。**Essbase** は、問合せトラッキングを停止するまで、または **Essbase** を停止するまで、最適化情報を収集し続けます。**Essbase** は、問合せトラッキングが停止するまで収集されたデータに基づいてビューを集約できます。
 - `createQBOAggregations` は、問合せトラッキングを有効にした後に実行する最適化された問合せに基づいて **Essbase** 集約ビューを作成します。

例

- デフォルトおよび問合せベースの集約ビューを削除します:
`epmautomate optimizeASOCube BksML12 clearAggregations`
- 問合せトラッキングを開始します
`epmautomate optimizeASOCube BksML12 startQueryTracking`
- 問合せトラッキングを開始した後に実行する最適化された問合せに基づいて **Essbase** 集約ビューを作成します:
`epmautomate optimizeASOCube BksML12 createQBOAggregations`

programDocumentationReport

Profitability and Cost Management アプリケーション・ロジックを含むプログラム・ドキュメンテーション・レポートを作成します。

[downloadFile](#) コマンドを使用して、レポートをローカル・コンピュータにダウンロードできます。

適用対象

Profitability and Cost Management

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

使用方法

```
epmautomate programDocumentationReport APPLICATION_NAME POV_NAME
[fileName=FILE_NAME] [fileType=PDF|WORD|EXCEL|HTML] [useAlias=true|false]
[skipFilters=true|false] stringDelimiter="DELIMITER"。ここで:
```

- `APPLICATION_NAME` は、プログラム・ドキュメンテーション・レポートを作成する **Profitability and Cost Management** アプリケーションの名前です。
- `POV_NAME` は、レポートを生成するアプリケーションのモデル **POV** の名前です。
- `fileName` はオプションで、レポート・ファイルの一意的名前(拡張子を含む)です。デフォルトのレポート・ファイル名は、`HPCMMLProgramDocumentationReport_APPLICATION_NAME_POV_NAME.pdf` です。
- `fileType` はオプションで、出力ファイル形式です。デフォルトは PDF です。
- `useAlias` はオプションで、メンバー名のかわりに別名を出力するかどうかを指定します。デフォルトは `false` です。
- `skipFilters` はオプションで、フィルタを使用するルールが多数ある大規模なモデルのレポート生成を高速化するために、フィルタを無視するかどうかを指定します。デフォルトは `false` です。
このパラメータを `true` に設定すると、各ルール・フィルタを解決してレポート内の推定ソース数、推定宛先数、推定ターゲット数を判断するプロセスがスキップされます。かわりに、対応するレベル 0 のメンバー数が使用されます。このパラメータ値を `false` に設定するか、パラメータ値を指定していない場合は、すべてのフィルタを解決してより正確な数が生成されます。
- `stringDelimiter` は **POV** 値で使用される区切り文字です。区切り文字は二重引用符で囲む必要があります。

例

- ルール・フィルタを解決した後にレポートを生成します:

```
epmautomate programDocumentationReport BksML30 2024_Feb_Actual fileName=Feb-Actual.xls fileType=Excel useAlias=true stringDelimiter="_"
```
- フィルタを無視してレポートを作成します:

```
epmautomate programDocumentationReport BksML30 2024_Feb_Actual fileName=Feb-Actual.xls fileType=Excel useAlias=true skipFilters=true stringDelimiter="_"
```

provisionReport

役割の割当レポート(.CSV ファイル)を生成し、デフォルトのダウンロード場所に格納します。

このレポートには、ユーザーに割り当てられた、事前定義済の役割(例: サービス名パワー・ユーザー)とアプリケーションの役割(例: **Planning** アプリケーションの役割である、一括割当て)がリストされます。[downloadFile](#) コマンドを使用して、レポートをダウンロードします。

レポートの2つのバージョン(簡略化またはクラシック)を生成できます。「アクセス制御」画面から入手できる役割の割当レポートと同じ簡略化レポートには、事前定義済役割に組み込まれるアプリケーション役割、またはユーザーに割り当てられているアプリケーション役割のコンポーネント役割はリストされません。クラシック・バージョンのレポートには、ユーザーが割り当てられる事前定義済役割に組み込まれるコンポーネント役割がリストされます。ユーザーに(直接またはグループを通じて)割り当てられたアプリケーション役割もリストします

このレポートを生成すると、アクセス制御で使用可能なユーザーおよび役割の情報がリフレッシュされます。

OCI (Gen 2)の場合のみ: Oracle Fusion Cloud Enterprise Performance Management では、非アクティブなユーザーは、非アクティブ化された時点で事前定義済役割を持っていたとしても、事前定義済役割が割り当てられていないユーザーと同じとみなされます。非アクティブなユーザーに関する情報はこのレポートには含まれません。

ノート:

このコマンドは次回のリリースで非推奨になります。このコマンドのかわりに、同等のレポートを生成する [roleAssignmentReport](#) コマンドを使用してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- 任意の事前定義済役割およびアクセス制御 - 管理アプリケーション役割
- 任意の事前定義済役割およびアクセス制御 - 表示アプリケーション役割

使用方法

```
epmautomate provisionReport REPORT_NAME [format=classic|simplified]
[userType=serviceUsers|IDAdmins]。ここで:
```

- `REPORT_NAME` はレポートの名前です。
- `format` はオプションで、レポートの書式設定方法を識別します。使用可能な値:
 - デフォルト・オプションである `simplified` は、「アクセス制御」画面から生成された役割の割当レポートと同じレポートを作成します。
 - `classic` は、ユーザーが割り当てられる事前定義済役割に組み込まれるコンポーネント役割をリストするレポートを作成します。ユーザーに(直接またはグループを通じて)割り当てられたアプリケーション役割もリストします
- `userType` はオプションで、レポートに含めるユーザーを識別します。このパラメータの値を指定しない場合は、デフォルト値の `serviceUsers` が使用されます。使用可能な値:
 - `serviceUsers` は、すべての機能ユーザーに関する情報を含むレポートを作成します(アプリケーションへのアクセス権を付与する事前定義済役割に割り当てられていない場合は、アイデンティティ・ドメイン管理者を含めないでください)

- IDAdmins は、アイデンティティ・ドメイン管理者役割に割り当てられたユーザーのみをリストするレポートを作成します。レポートは、クラシック形式と簡略化形式で同じです

例

- クラシック・レポートの作成: `epmautomate provisionReport myProvReport.CSV format=classic`
- 簡略化レポートの作成:
 - `epmautomate provisionReport myProvReport.CSV format=simplified`
 - `epmautomate provisionReport myProvReport.CSV userType=serviceUsers`
- アイデンティティ・ドメイン管理者のみをリストするレポートの作成:
 - `epmautomate provisionReport myProvReport.CSV userType=IDAdmins`
 - `epmautomate provisionReport myProvReport.CSV userType=IDAdmins format=classic`

purgeArchivedTmTransactions

アーカイブされた照合済トランザクションを **Account Reconciliation** アプリケーションからページします。

最適なアプリケーション・サイズを維持するために、[archiveTmTransactions](#) コマンドを定期的に使用して古い照合済トランザクションをアーカイブし、次にこのコマンドを実行して **Account Reconciliation** から削除します。

適用対象

Account Reconciliation

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

`epmautomate purgeArchivedTMTransactions JobID=JOB_ID`。ここで、JobID は、照合済トランザクションをアーカイブするために実行された **TM** トランザクションのアーカイブ・ジョブの識別子です。このジョブ ID は、[archiveTmTransactions](#) コマンドを実行すると **EPM** 自動化コンソールに表示されます。ジョブ・コンソールにも表示されます。

例

```
epmautomate purgeArchivedTMTransactions JobID=100000002655003
```

purgeTmTransactions

照合済トランザクションを Account Reconciliation から削除します。

適用対象

Account Reconciliation

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

```
epmautomate purgeTmTransactions matchType age [filterOperator=VALUE]  
[filterValue=VALUE] [logFilename=FILE_NAME]。ここで:
```

- matchType は、照合済トランザクションを削除する照合タイプの識別子(TextID)です。
- age は、トランザクションを照合してからの日数を識別します。この値以上経過した照合済トランザクションが削除されます。
- filterOperator はオプションで、削除対象の照合済トランザクションが格納されている勘定科目を識別するための次のいずれかのフィルタ条件です。この値を filterValue と組み合わせて、照合済トランザクションを削除する勘定科目を識別します:
 - equals
 - not_equals
 - starts_with
 - ends_with
 - contains
 - not_contains
- filterValue はオプションで、パージするトランザクションを識別するためのフィルタ値です。filterOperator が equals または not_equals の場合は、スペース区切りのリストを使用して複数の値を指定できます(例: filterValue=101-120 filterValue=102-202)。複数の値が指定されている場合、フィルタ演算子とフィルタ値の組合せに一致する勘定科目のトランザクションがパージ対象として選択されます。
- logFilename はオプションで、コマンド・アクティビティに関する情報を記録するログ・ファイルの名前です。ファイル名が指定されていない場合は、PurgeTransactions_JOB_ID という名前のログ・ファイルが自動的に生成されます。

 **Note:**

filterOperator および filterValue が指定されていない場合は、指定された matchType のすべての勘定科目から age 以上経過した照合済トランザクションがすべてページされます。

例

- 照合タイプ cashrecon の 180 日以上経過した照合済トランザクションをページします:
epmautomate purgeTMTransactions cashrecon 180 logFileName=tmlogs.log
- 勘定科目 101-120 または 102-202 の照合タイプ cashrecon で 180 日以上経過した照合済トランザクションをページします:
epmautomate purgeTMTransactions cashrecon 180 filterOperator>equals
filterValue=101-120 FilterValue=102-202
- 文字列 11 が含まれている勘定科目の照合タイプ cashrecon で 180 日以上経過した照合済トランザクションをページします:
epmautomate purgeTMTransactions cashrecon 180 filterOperator=contains
filterValue=11

recomputeOwnershipData

出資比率データを再計算します

Financial Consolidation and Close の出資比率データの再計算は、次の状況で必要です:

- 「出資比率の管理」勘定科目のオーバーライド・ルールの追加または削除後
- 連結メソッド範囲設定の変更後
- エンティティ構造が変更されたかどうかに関係なく、データベース・リフレッシュ後

Tax Reporting の出資比率データの再計算は、エンティティ構造が変更されていない場合でも、データ・リフレッシュの後、毎回必要になります。

適用対象

Financial Consolidation and Close および Tax Reporting

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー

使用方法

epmautomate recomputeOwnershipData Scenario Year Period。ここで:

- Scenario は、再計算するシナリオの名前です。
- Year は、再計算する年です。
- Period は、再計算する年の最初の期間です。
選択した期間およびすべての後続期間が再計算されます。

 ノート:

再計算が必要な POV は、出資比率データが再計算された後にのみ連結できます。

例

```
epmautomate recomputeOwnershipData FCCS_total_Actual FY19 Jan
```

recreate

再作成することにより、環境をクリーンな状態に復元します。

環境を再作成して、次のタスクを完了します:

- 完全なスナップショットをインポートする前に、環境をクリーン・アップします。
- 環境でデプロイできるビジネス・プロセスを変更します。

 注意:

- このコマンドでは、既存のアプリケーション、およびオプションでユーザー定義のすべてのアーティファクトが環境から削除されます。さらに、データベースが再作成されて、既存のすべてのデータが削除されます。サービスを再作成した後、移行または EPM 自動化を使用して、新しいビジネス・プロセスを作成するかインポートできます。
- このコマンドでは、移行履歴が削除されます。そのため、移行で使用できる移行ステータス・レポートには履歴情報が含まれません。
- このコマンドを使用する前に、環境の完全バックアップを実行してください。[runDailyMaintenance](#) コマンドを実行して、バックアップ・スナップショットを作成できます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

```
epmautomate recreate [-f] [removeAll=true|false] [TempServiceType=Service_type]。
ここで:
```

- -f は、ユーザーの確認なしで再作成プロセスを強制的に開始します。-f オプションを使用していない場合は、操作の確認を求められます。
- removeAll はオプションで、すべてのスナップショット、および受信ボックスのコンテンツ(アップロードされたファイル)および送信ボックスのコンテンツ(環境からエクスポート

されたファイル)を削除します。デフォルトは `false` で、スナップショット、および受信ボックスと送信ボックスのコンテンツは保持されます。

- `TempServiceType` は、オプションであり、環境を別のサービス環境に変換します。環境にデプロイできるビジネス・プロセスは、所有しているサブスクリプションのタイプによって管理されます。たとえば、**EPM Standard Cloud Service** サブスクリプションをお持ちの場合、**Account Reconciliation** から **Planning** へ環境を変換した後、フリー・フォーム・アプリケーションは作成できません。**EPM Enterprise Cloud Service** サブスクリプションをお持ちの場合、サービス・タイプを適切に変更した後、環境内に任意のビジネス・プロセスを作成できます。管理者用スタート・ガイドの新しい **Cloud EPM** サービスについてを参照してください

このパラメータの動作は、サブスクリプションによって異なります。

– **EPM Standard Cloud Service および EPM Enterprise Cloud Service 以外のサブスクリプション:**

`TempServiceType` オプションを使用して、**Planning**、**Enterprise Planning**、**Tax Reporting** または **Financial Consolidation and Close** 環境を **Account Reconciliation**、**Oracle Enterprise Data Management Cloud** または **Profitability and Cost Management** 環境に一時的に変換できます。たとえば、**Planning** 環境を購入した場合は、次のコマンドを実行して **Account Reconciliation** 環境に変換できます:

```
epmautomate recreate -f removeAll=true TempServiceType=ARCS
```

環境を **Account Reconciliation** に変換した後、これは、**Oracle Enterprise Data Management Cloud** 環境または **Profitability and Cost Management** 環境に、適切な `TempServiceType` 値を使用することで変換できます。たとえば、**Profitability and Cost Management** 環境に変換するには、次のコマンドを実行できます。

```
epmautomate recreate -f removeAll=true TempServiceType=PCMCS
```

環境を元のサービス・タイプに戻すには、次のコマンドを実行します。

```
epmautomate recreate -f
```

Profitability and Cost Management: 次のコマンドを実行して、**Profitability and Cost Management** 環境を **Planning**、**Enterprise Planning** または **Enterprise Profitability and Cost Management** 環境に変換できます:

```
epmautomate recreate -f removeAll=true TempServiceType=PBCS
```

環境を元の **Profitability and Cost Management** 環境に戻すには、次のコマンドを使用します:

```
epmautomate recreate -f TempServiceType=PCMCS
```

 ノート:

Profitability and Cost Management 環境を Account Reconciliation、Oracle Enterprise Data Management Cloud または Narrative Reporting 環境に変換することはできません。

– **EPM Standard Cloud Service および EPM Enterprise Cloud Service のサブスクリプション:**

TempServiceType オプションを使用して、Oracle Fusion Cloud Enterprise Performance Management 環境をサポートされている他の環境に変換できます。

EPM Enterprise Cloud Service サブスクリプションでは、共通の Cloud EPM プラットフォームが使用されます。最初に、サポートされている任意の Cloud EPM ビジネス・プロセスをデプロイできます。

デプロイされたビジネス・プロセスから別のビジネス・プロセスに切り替えるには、環境の新しいサービス・タイプを指定して、環境を再作成します。たとえば、Account Reconciliation ビジネス・プロセスの作成後、今度は Oracle Enterprise Data Management Cloud 環境を作成する場合、次のように再作成コマンドを実行します。

```
epmautomate recreate -f removeAll=true TempServiceType=EDMCS
```

ビジネス・プロセス(Account Reconciliation など)を Planning、Tax Reporting、Financial Consolidation and Close に変換するには、TempServiceType 値を指定しません。たとえば、Account Reconciliation ビジネス・プロセスの作成後、今度は Planning モジュール環境を作成する場合、次のように再作成コマンドを実行します。

```
epmautomate recreate -f removeAll=true
```

使用可能な TempServiceType 値:

- ARCS は、環境を Account Reconciliation 環境に変換します
- EDMCS は、環境を Oracle Enterprise Data Management Cloud 環境に変換します
- EPRCS は、環境を Narrative Reporting 環境に変換します
- PCMCS は、環境を Profitability and Cost Management 環境に変換します

例

- 現在の環境を再作成し、元のサービス・タイプに復元し(以前に TempServiceType パラメータを使用して再作成が発行された場合)、ユーザーが作成したスナップショットと受信ボックスおよび送信ボックスのコンテンツは削除しません:

```
epmautomate recreate -f
```

- 現在の環境を再作成し、元のサービス・タイプに復元し(以前に TempServiceType パラメータを使用して再作成が発行された場合)、スナップショットと受信ボックスおよび送信ボックスのコンテンツを削除します:

```
epmautomate recreate -f removeAll=true
```

- 現在の環境を Enterprise Data Management 環境として再作成し、受信ボックスと送信ボックスのコンテンツ、および既存のスナップショットを削除します:

```
epmautomate recreate -f removeAll=true TempServiceType=EDMCS
```

- 現在の EPM Enterprise Cloud Service Account Reconciliation 環境を、Financial Consolidation and Close 環境に再作成し、受信ボックスと送信ボックスのコンテンツおよび既存のスナップショットを削除します。

```
epmautomate recreate -f removeAll=true
```

refreshCube

アプリケーション・キューブをリフレッシュします。通常、アプリケーションにメタデータをインポートした後にキューブをリフレッシュします。

キューブのリフレッシュ操作を完了するまでにかかる時間は、アプリケーション構造に対して行った変更やその変更がキューブに及ぼす影響によって変わります。たとえば、疎ブロック・ストレージ・キューブ・メンバーを更新した後のリフレッシュにはあまり時間がかかりませんが、密ブロック・ストレージ・キューブ・メンバーすなわち集約ストレージ・キューブ・メンバーを更新した後のキューブ・リフレッシュには、かなりの時間がかかることがあります。次のメンテナンス・ウィンドウでアプリケーションがバックアップされる前に、キューブ・リフレッシュ操作を完了させる必要があります。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmautomate refreshCube [JOB_NAME]`。ここで、`JOB_NAME` (オプション)は、アプリケーションで定義されたデータベース・リフレッシュ・ジョブの名前です。操作のステータスは、コマンドが実行されたコンソールに表示されます。また、アプリケーションの「**ジョブ**」画面の**最近のアクティビティ**・ページからステータスを確認することもできます。

例

```
epmautomate refreshCube DaliyCubeRefresh
```

removeUserFromGroups

ANSI または UTF-8 エンコーディングの CSV ファイルで識別されたアクセス制御グループから、ユーザーのメンバーシップを削除します。

ファイル形式は次のとおりです。

```
Group Name  
Group1  
Group2
```

ノート:

これらのグループはアクセス制御に存在している必要があります。グループ名値では大文字と小文字が区別されません。

サービス管理者は、[uploadFile](#) コマンドを使用して、ファイルを環境にアップロードします。

終了すると、失敗した各エントリに関する情報がコンソールに出力されます。この情報を確認して、CSV ファイルの一部のエントリでコマンドの実行が失敗した理由を理解してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- 任意の事前定義済役割およびアクセス制御 - 管理アプリケーション役割

使用方法

`epmautomate removeUserFromGroups FILE_NAME User_Login`。ここで:

- `FILE_NAME` は、ユーザーのメンバーシップを削除するアクセス制御グループ名を含む CSV ファイルの名前です
- `User_Login` は、アクセス制御グループからメンバーシップを削除する Oracle Fusion Cloud Enterprise Performance Management ユーザーのログイン ID です。ユーザーのログイン ID は、環境にサービスを提供するアイデンティティ・ドメインに存在し、事前定義済役割に割り当てられている必要があります。この値では大文字と小文字が区別されません。

例

```
epmautomate removeUserFromGroups groups.CSV jdoe@example.com
```

removeUsers

アイデンティティ・ドメインから環境にアップロードされた ANSI または UTF-8 エンコーディングの CSV ファイルで識別されたアカウントを削除します。

ファイル形式は次のとおりです。

```
User Login  
jane.doe@example.com  
jdoe@example.com
```

サービス管理者は、[uploadFile](#) コマンドを使用して、ファイルを環境にアップロードします。ユーザー・ログイン値では大文字と小文字が区別されません。たとえば、`jane.doe@example.com` は、`Jane.Doe@Example.com` など、大文字と小文字のすべてのバリエーションと同じであるものとして処理されます。

 ノート:

- このコマンドを実行するユーザーのアカウントは CSV ファイルに含めないでください。
- ユーザー・アカウントはアイデンティティ・ドメイン管理者がサポートするすべてのサービス環境に共通するため、1つの環境でアカウントを削除すると、アイデンティティ・ドメイン管理者が同じすべての環境でそのアカウントが削除されます。

終了すると、失敗した各エントリに関する情報がコンソールに出力されます。この情報を確認して、CSV ファイルの一部のエントリでコマンドの実行が失敗した理由を理解してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

アイデンティティ・ドメイン管理者と任意の事前定義済役割

使用方法

`epmautomate removeUsers FILE_NAME`。ここで、`FILE_NAME` は、アイデンティティ・ドメインから削除するユーザーのログイン ID を含む CSV ファイルの名前です。

例

```
epmautomate removeUsers remove_users.CSV
```

removeUsersFromGroup

アクセス制御で管理されているグループから、ANSI または UTF-8 エンコーディングの CSV ファイルにリストされているユーザーを削除します。

ファイル形式は次のとおりです。

```
User Login  
jdoe  
john.doe@example.com
```

ユーザー・ログイン値では大文字と小文字が区別されません。たとえば、`jane.doe@example.com` は、`Jane.Doe@Example.com` など、大文字と小文字のすべてのバリエーションと同じであるものとして処理されます。サービス管理者は、[uploadFile](#) コマンドを使用して、ユーザー・ログインを含むファイルを環境にアップロードします。

終了すると、失敗した各エントリに関する情報がコンソールに出力されます。この情報を確認して、CSV ファイルの一部のエントリでコマンドの実行が失敗した理由を理解してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- 任意の事前定義済役割およびアクセス制御 - 管理アプリケーション役割

使用方法

`epmautomate removeUsersFromGroup FILE_NAME GROUP_NAME`。ここで:

- `FILE_NAME` は、アクセス制御で管理されているグループから削除するユーザーのログイン名を含む CSV ファイルの名前です
- `GROUP_NAME` は、ユーザーを削除するアクセス制御グループの名前ですこの値では大文字と小文字が区別されません。

ノート:

ユーザーは、次の両方の条件が満たされている場合にのみグループから削除されません。

- ファイルに含まれるユーザー・ログインが、環境にサービスを提供するアイデンティティ・ドメインに存在する
- ユーザーが、アイデンティティ・ドメイン内の事前定義済役割に割り当てられている

例

```
epmautomate removeUsersFromGroup user_file.CSV example_group
```

removeUsersFromTeam

CSV ファイルにリストされている Oracle Fusion Cloud Enterprise Performance Management ユーザーをチームから削除します。

CSV ファイルに含まれているユーザーがチームのメンバーでない場合、このコマンドはそのユーザーを無視します。このファイル内の値では大文字と小文字が区別されません。CSV ファイルの形式は次のとおりです。

```
User Login  
jdoe  
jane.doe@example.com
```

サービス管理者は、[uploadFile](#) を使用して、.CSV ファイルを環境にアップロードします。

適用対象

Financial Consolidation and Close、Tax Reporting および Account Reconciliation。

必要な役割

- サービス管理者
- 任意の事前定義済役割およびチーム - 管理アプリケーション役割
- 任意の事前定義済役割およびユーザー - 管理アプリケーション役割

「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

`epmautomate removeUsersFromTeam FILE.CSV TEAM_NAME`。ここで:

- `FILE` は、チームから削除するユーザーのログイン ID がリストされた UTF-8 形式の CSV ファイルを識別します。
- `TEAM_NAME` は、アクセス制御で定義されたチーム名を識別します。この値では大文字と小文字が区別されません。

例

```
epmautomate removeUsersFromTeam example_users.csv example_team
```

renameSnapshot

環境にアップロードまたは作成したスナップショットの名前を変更します。

このコマンドが生成中またはアーカイブ中のスナップショットの名前を変更するために実行された場合は、次のいずれかのエラーが表示されます:

- ファイルが見つかりません: スナップショットが生成中の場合
- アーカイブ・プロセスが進行中です。名前変更または削除できません: スナップショットがアーカイブ中の場合

環境内のメンテナンス・スナップショットの名前は変更しないでください。メンテナンス・スナップショットのバックアップを保持するには、Artifact Snapshot を環境からローカル・コンピュータにダウンロードした後、必要に応じて名前を変更する必要があります。管理者用スタート・ガイドのメンテナンス・スナップショットの概要を参照してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザーおよび移行 - 管理アプリケーション役割(Oracle Enterprise Data Management Cloud および Profitability and Cost Management)

使用方法

epmautomate renameSnapshot *SNAPSHOT_NAME* *NEW_SNAPSHOT_NAME*。ここで:

- *SNAPSHOT_NAME* は既存のスナップショットの名前です。この値に、スペース、\ (バックslash)、/ (スラッシュ)、* (アスタリスク)、? (疑問符)、" (引用符)、< (次より小さい)、> (次より大きい)などの特殊文字を含めることはできません。
- *NEW_SNAPSHOT_NAME* は、スナップショットに割り当てて一意の名前です。

例

```
epmautomate renameSnapshot "Example Snapshot" Example_Snapshot_18_09_25
```

replay

環境に対する Oracle Smart View for Office、REST API または EPM 自動化の負荷がリプレイされます。これにより、高負荷の状況でのパフォーマンス・テストが可能になり、指定された負荷の影響をサービスが受ける際のユーザー・エクスペリエンスを許容できるかどうかを確認できます。

サービスで実行する必要があるアクティビティを指定するリプレイ・ファイルを作成してください。リプレイ・ファイルを作成する方法の詳細は、[replay コマンドの実行準備](#)を参照してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

使用方法

```
epmautomate replay REPLAY_FILE_NAME.csv [duration=N] [trace=true] [lagTime=t] [encrypt=true|false]。ここで:
```

- *REPLAY_FILE_NAME* は、環境で実行されるアクティビティを格納する CSV ファイルです。
- Duration はオプションで、環境でアクティビティが実行される時間を分単位で示します。この値が設定されていると、HAR ファイルのアクティビティが一度実行されます。HAR ファイルのアクティビティが、このパラメータで指定した時間内に完了した場合、このコマンドはアクティビティが完了するまで HAR ファイルを返します。たとえば、実行に 3 分しかかからない HAR ファイルをリプレイする際に、duration=10 と設定したと仮定します。この場合、リプレイ・コマンドによって HAR ファイルのアクティビティは、4 回目が終わるまで 4 回(12 分間)実行されます。

- `trace=true` は、トレース・ファイルを XML 形式で作成するように指示するオプション設定です。
このオプション設定を指定すると、リプレイ CSV ファイルに含まれる HAR ファイルごとに 1 つのフォルダが作成され、関連するすべてのトレース・ファイルがその中に格納されます。HAR ファイル内のアクティビティごとに、**Smart View** レスポンスを含む 1 つのトレース・ファイルが生成されます。トレース・ファイルの名前は `trace-N.xml` のように付けられます。たとえば、`trace-1.xml` で、N は 1 から始まるカウンタです。

トレース・ファイルを格納するフォルダは、EPM 自動化が実行されるディレクトリに作成されます。環境の現在のシステム時刻と HAR ファイル名を `YYYY_MM_DD_HH_MM_SS_HAR_FILE_NAME` の形式で組み合わせて、フォルダの名前が付けられます。たとえば、HAR ファイル名が `forecast1.har` の場合、フォルダ名は `2016_06_08_10_21_42_forecast1` となることがあります。

- `[lagTime=t]` はオプションで、リプレイ・ファイルに含まれる各 HAR ファイルの実行の合間にコマンドが待機する秒数を指定します。デフォルトは 5 秒です。
5 秒未満の値を指定した場合はエラーが表示されます。負の数(-1 など)や分数(1/2 など)はパラメータ値として受け入れられません。小数値はサポートされています。

最初の HAR ファイルの実行を開始した後、このコマンドは、次の HAR ファイルの処理を開始するまで、このパラメータによって指定された秒数の間待機します。通常、ユーザー・アクティビティは同時に開始されないため、このパラメータを設定すると、環境への負荷のより現実的なシミュレーションを作成しやすくなります。

たとえば、アクティビティを実行するためにピーク時間中に環境にサインオンする 1000 人のユーザーによる負荷をシミュレートするとします。HAR ファイルを作成することにより、これらのセッションをシミュレートしてから、6 秒のラグ時間でこのコマンドを実行し、環境にかかった負荷をレプリケートできます。

- `encrypt=true|false` はオプションで、リプレイ・ファイルに含まれているすべてのパスワードを暗号化するかどうかを指定します。デフォルトは `true` です。パスワードの暗号化には、ランダム暗号化キーが使用されます。

このコマンドの実行に関連するステップの詳細は、[サンプル・リプレイ・セッション](#)を参照してください。

例

```
epmautomate replay forecast1.CSV duration=60 lagTime=5.6
```

resetService

環境を再起動します。オプションで、ブロック・ストレージ・オプション(BSO)キューブの Oracle Essbase インデックス・キャッシュがアプリケーションに対して最適化されるように、環境を再起動する前に自動調整できます。

デフォルトで、環境は日次メンテナンスの完了直後に再起動されます。たとえば、スナップショットを環境にインポートした後など、環境の自動調整は重要です。このコマンドは、深刻なパフォーマンス低下が観測された場合または環境が使用不可であることを示すエラー・メッセージが表示された場合にのみ使用します。環境の再起動によるアプリケーションのカスタマイズ(たとえば、ロケール変更、テーマや通貨に関連する設定など)への影響はありません。再起動には最大 15 分かかります。

このコマンドを使用する前に、環境でビジネス・ルールが実行されていないことを確認してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmautomate resetService "comment" [AutoTune=true|false] [-f]`。ここで:

- `Comment` は、環境のリセットを引き起こした問題の説明です。コメントは引用符で囲む必要があります。
- `AutoTune` はオプションで、アプリケーションの Essbase キャッシュの BSO キューブを最適化するために環境を自動調整するかどうかを示します。デフォルトは `false` です。
このパラメータは、Essbase BSO キューブ: Planning (Planning モジュールを含む)、Financial Consolidation and Close および Tax Reporting を使用する環境でのみ使用しません。
- `-f` はオプションで、追加のユーザー介入なしで環境の再起動を強制することを指定します。このオプションを使用していない場合は、操作の確認を求められます。このオプションは、このコマンドを使用するスクリプトをスケジュールする場合に便利です。

例

- `epmautomate resetService "Users experience slow connections; force restarting the environment" -f`
- `epmautomate resetService "Users experience unacceptably slow connections"`
- `epmautomate resetService "Optimizing the Essbase cache" AutoTune=true`

restoreBackup

使用可能なバックアップ・スナップショットをコピーして、環境へのインポートに使用できるようにします。

`listBackups` コマンドを使用して、復元するバックアップが使用可能かどうかを確認します。環境へのスナップショットのセルフサービス復元により、処理時間を節約できます。スナップショットを復元した後、`importSnapshot` コマンドを使用してスナップショットを環境にインポートします。

 **Note:**

Narrative Reporting のみ: 次のプロセスを使用して、復元されたスナップショットをインポートします:

1. **downloadFile** コマンドを使用して、復元されたスナップショットをローカル・コンピュータにダウンロードします。
2. ダウンロードしたスナップショットの名前を `EPRCS_Backup` に変更します。
3. **uploadFile** コマンドを使用して、名前を変更したスナップショット (`EPRCS_Backup`) を環境にロードします。コマンドには `to_be_imported` アップロード場所を指定してください。
4. Narrative Reporting で、「日次メンテナンス」カードを開きます。
 - a. **アップロードしたバックアップ・スナップショットの使用** を選択します。
 - b. **復元のスケジュール** をクリックし、次回の日次メンテナンスでスナップショットをインポートします。
 - c. 「はい」 をクリックします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザーおよび移行 - 管理アプリケーション役割(Oracle Enterprise Data Management Cloud および Profitability and Cost Management)

使用方法

`epmAutomate restoreBackup SNAPSHOT_NAME [targetName=TARGET_SNAPSHOT_NAME]`。ここで:

- `SNAPSHOT_NAME` は、**listBackups** コマンドでリストされる環境で使用可能なバックアップ・スナップショットの名前です。
- `targetName` はオプションで、ターゲット環境での拡張子なしのバックアップ・スナップショットの名前です。この値を指定しない場合、バックアップ・スナップショットは、`SNAPSHOT_NAME` を使用してターゲット環境に復元されますが、`/` (スラッシュ) は `_` (アンダースコア) に置き換えられます。たとえば、`SNAPSHOT_NAME` が `2022-05-14T00:08:56/Artifact_Snapshot.zip` の場合、`targetName` は `2022-05-14T00:08:56_Artifact_Snapshot.zip` になります。

例

- Narrative Reporting 以外のサービスの場合:
`epmAutomate restoreBackup 2022-05-14T00:08:56/Artifact_Snapshot.zip targetName=example_Artifact_Snapshot`
- Narrative Reporting のみの場合:

```
epmAutomate restoreBackup 2022-02-16T21:00:02/EPRCS_Backup  
targetName=Example_EPRCS_Backup
```

restructureCube

ブロック・ストレージ・キューブの再構築をすべて実行して、断片化を消去または削減します。再構築では空のブロックも削除され、変更内容はアプリケーションからキューブにプッシュされません。

ノート:

このコマンドを実行する前に、誰もこのアプリケーションを使用していないことを確認します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmautomate restructureCube CUBE_NAME`。ここで、`CUBE_NAME` は、アプリケーションに存在するとおりの正確なキューブの名前です

例

```
epmautomate restructureCube Plan1
```

roleAssignmentAuditReport

OCI (Gen 2)環境で、環境に指定された監査データ保持期間に対応する一定期間の事前定義済役割およびアプリケーション役割の割当てに対する変更をリストする監査レポートを作成します。デフォルトの保持期間は 30 日です。Oracle Cloud Identity Console で「**監査保持期間(日)**」設定を変更して、最大 90 日まで拡張できます。90 日を超える期間の監査データを保持するには、このレポートおよび**無効なログイン・レポート**を定期的にダウンロードしてアーカイブします。

役割割当監査レポートには、役割変更(「アクション」列)が行われたユーザー・ログイン名、IDCS グループ名または EPM グループ名がリストされます。また、割り当てられた、または割当てを解除された事前定義済役割またはアプリケーション役割、役割変更を実行したユーザー(「実行者」列)、およびアクションが完了したときの 24 時間フォーマットのタイムスタンプ(UTC)も含まれます。「タイプ」列は、「名前」列が表しているものを示します。「ユーザー」(「名前」列でユーザーのログイン名が識別される場合)、「IDCS グループ」(「名前」列に IDCS グループ名が表示される場合)または「EPM グループ」(「名前」列に EPM グループ名が表示される場合)の 3 つのいずれかの値になります。

	A	B	C	D	E	F
1	Name	Type	Role	Action	Performed By	Date and Time
2	IDCS-Group1	IDCS Group	Power User	Assign	serviceadmin@example.com	December 11, 2024 00:47:10 UTC
3	jane.doe@example.com	User	Power User	Assign	serviceadmin@example.com	December 11, 2024 00:47:27 UTC
4	epmIDCSGroup	User	Service Administrator	Assign	serviceadmin@example.com	December 11, 2024 00:48:47 UTC
5	john.doe@example.com	User	Power User	Assign	serviceadmin@example.com	December 11, 2024 00:59:06 UTC
6	john.doe@example.com	User	Ad Hoc - Create	Assign	serviceadmin@example.com	December 11, 2024 01:08:37 UTC
7	john.doe@example.com	User	Access Control - Manage	Unassign	serviceadmin@example.com	December 11, 2024 01:10:30 UTC
8	john.doe@example.com	User	Ad Hoc - Create	Unassign	serviceadmin@example.com	January 07, 2025 03:28:56 UTC

環境内で事前定義済役割に以前に割り当てられていた削除されたユーザーに関する情報は、「名前」列にユーザーの表示名(姓名)でリストされます。このような場合、役割の列には、ユーザーのアカウントが削除される前にユーザーに割り当てられていた事前定義済役割が表示されません。この変更は、削除されたユーザーに割り当てられていたアプリケーション役割には適用されません。このような割当ては、ユーザーのユーザー・ログイン名で表示されます。例については、次の図の赤いボックスの情報を参照してください。

	A	B	C	D	E	F
1	Name	Type	Role	Action	Performed By	Date and Time
2	Jane Doe	User	User	Assign	admin@example.com	January 08, 2025 05:42:47 UTC
3	jane.doe@example.com	User	Run Integrations	Assign	admin@example.com	January 08, 2025 05:44:18 UTC
4	john.smith@example.com	User	Power User	Assign	admin@example.com	January 08, 2025 05:44:18 UTC
5	john.smith@example.com	User	User	Unassign	admin@example.com	January 08, 2025 05:44:18 UTC
6	john.smith@example.com	User	Access Control - View	Assign	admin@example.com	January 08, 2025 05:44:18 UTC
7	epmIDCSGroup	IDCS Group	Viewer	Assign	admin@example.com	January 08, 2025 05:44:22 UTC
8	epmGroup	EPM Group	Ad Hoc - Create	Assign	admin@example.com	January 08, 2025 05:44:22 UTC

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Strategic Workforce Planning および Sales Planning。

必要な役割

- サービス管理者
- 任意の事前定義済役割およびアイデンティティ・ドメイン管理者役割
- 任意の事前定義済役割およびアクセス制御 - 管理アプリケーション役割
- 任意の事前定義済役割およびアクセス制御 - 表示アプリケーション役割

使用方法

`epmAutomate roleAssignmentAuditReport FROM_DATE TO_DATE FILE_NAME.CSV`。ここで:

- `FROM_DATE` は、レポートを生成する期間の開始日(YYYY-MM-DD 形式)を示します。この日付は、Oracle Cloud Identity Console で指定した監査保持期間内である必要があります。
- `TO_DATE` は、レポートを生成する期間の終了日(YYYY-MM-DD 形式)を示します。
- `FILE_NAME` は、レポートの CSV ファイルの名前です。 `downloadFile` コマンドを使用して、生成されたレポートをダウンロードできます。

例

```
epmAutomate roleAssignmentAuditReport 2024-12-11 2025-01-09 RoleAuditReport.CSV
```

roleAssignmentReport

役割の割り当てレポート(.CSV)を生成します。

このレポートには、デフォルトでは、事前定義済役割(サービス管理者など)およびユーザーに割り当てられているアプリケーション役割(Planning アプリケーション役割の承認所有権割当者、承認スーパーバイザ、承認管理者および承認プロセス・デザイナーなど)がリストされます。環境のアイデンティティ・ドメイン管理者をリストするために、必要に応じてこのレポートを生成することもできます。このレポートは、アクセス制御から生成される役割の割り当てレポートの CSV バージョンと一致します。

	A	B	C	D	E	F
1	User Login	First Name	Last Name	Email	Role	Granted through Group
2	Jdoe	John	Doe	jdoe@example.com	Planner	
3	jdoe	John	Doe	jdoe@example.com	Power User	
4	Jdoe	John	Doe	jdoe@example.com	Service Administrator	
5	jdoe	John	Doe	jdoe@example.com	Viewer	
6	Jdoe	John	Doe	jdoe@example.com	Mass Allocation	example->Power User
7	jdoe	John	Doe	jdoe@example.com	Run Integration	
8	jane.doe@example.com	Jane	Doe	jane.doe@example.com	Planner	
9	jane.doe@example.com	Jane	Doe	jane.doe@example.com	Power User	
10	jane.doe@example.com	Jane	Doe	jane.doe@example.com	Viewer	
11	jane.doe@example.com	Jane	Doe	jane.doe@example.com	Mass Allocation	example

このレポートを生成すると、アクセス制御で使用可能なユーザーおよび役割の情報がリフレッシュされます。

OCI (Gen 2)の場合のみ: Oracle Fusion Cloud Enterprise Performance Management では、非アクティブなユーザーは、非アクティブ化された時点で事前定義済役割を持っていたとしても、事前定義済役割が割り当てられていないユーザーと同じとみなされます。非アクティブなユーザーに関する情報はこのレポートには含まれません。

ノート:

このコマンドで、[provisionReport](#) コマンドを使用して作成されるレポートと同等のレポートが作成されます。

[downloadFile](#) コマンドを使用して、レポートをダウンロードできます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- 任意の事前定義済役割およびアクセス制御 - 管理アプリケーション役割
- 任意の事前定義済役割およびアクセス制御 - 表示アプリケーション役割

使用方法

`epmautomate roleAssignmentReport REPORT_NAME.CSV [userType=IDAdmins|serviceUsers]`。ここで:

- `REPORT_NAME` はレポートの名前です。
- `userType` はオプションで、レポートに情報を含めるユーザーのタイプを識別します。デフォルトは `serviceUsers` です。有効な値は次のとおりです。
 - `serviceUsers` は、すべての機能ユーザーに関する情報を含むレポートを作成します(アプリケーションへのアクセス権を付与する事前定義済役割に割り当てられていない場合は、アイデンティティ・ドメイン管理者を含めないでください)
 - `IDAdmins` は、アイデンティティ・ドメイン管理者役割に割り当てられたユーザーのみをリストするレポートを作成します。

例

- 機能ユーザーのみをリストするレポートの生成:
 - `epmautomate roleAssignmentReport myReport.CSV`
 - `epmautomate roleAssignmentReport myReport.CSV userType=serviceUsers`
- アイデンティティ・ドメイン管理者のみをリストするレポートの生成:
`epmautomate roleAssignmentReport myReport.CSV userType=IDAdmins`

runAutomatch

自動照合プロセスを実行し、サービス管理者が定義したルールを使用してトランザクションを照合します。

ノート:

`importTmPremappedTransactions` または `runDataRule` コマンドを使用してトランザクション照合にトランザクション・データをインポートした後、このコマンドを実行します。

Account Reconciliation の「**ジョブ履歴**」タブで、自動照合プロセスのステータスをモニターできます。

適用対象

Account Reconciliation

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

`epmautomate runAutomatch RECONCILIATION_TYPE`。ここで、`RECONCILIATION_TYPE` は、**Account Reconciliation** で定義された照合タイプです。

例

```
epmautomate runAutomatch INTERCOMPANY
```

runBatch

データ管理バッチを実行します。

データ管理でのバッチ実行モードが「シリアル」に設定されている場合は、バッチ内のジョブがすべて完了すると制御が戻ります。「並列」に設定されている場合は、バッチ内のジョブがすべて実行のために送信されると制御が戻ります。

ノート:

このコマンドを使用して、データ・ソースから **Oracle Fusion Cloud Enterprise Performance Management** に直接データ・ロード統合を実行することはできません。**EPM 統合エージェント**を使用して、直接データ・ロードを統合してください。詳細は、[データ統合の管理の EPM 統合エージェントを使用した直接データ・ロードの実行](#)を参照してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

`epmautomate runBatch BATCH_NAME`。ここで、`BATCH_NAME` はデータ管理で定義されるバッチの名前です。

例

```
epmautomate runBatch Accounting_batch
```

runBusinessRule

ビジネス・ルールを起動します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー(ルールへの起動アクセスが許可されている場合)

使用方法

`epmautomate runBusinessRule RULE_NAME [PARAMETER=VALUE]`。ここで:

- `RULE_NAME` は環境で定義されているとおりに正確なビジネス・ルールの名前です。
- `PARAMETER=VALUE` は、ビジネス・ルールの実行に必要なオプションのランタイム・パラメータとその値を示します。

ノート:

- このコマンドは、単一のビジネス・ルールのみを実行できます。ルールセットを実行するには、[runRuleSet](#) コマンドを使用します。
- ルールは、それがデプロイされたプラン・タイプに対して実行されます。
- ランタイム・パラメータに値を指定しない場合は、デフォルト値が使用されます。このコマンドは、ルールで定義されているものと完全一致しない実行時プロンプトを無視します。
- `PARAMETER=VALUE` のペアを使用して、ビジネス・ルールで必要とされるだけの実行時プロンプトを指定します。次の例では 2 つの実行時プロンプト (Period および Entity) とその値 (Q1 および USA) を使用します。パラメータに複数の値を入力する方法の詳細は、[パラメータに対する複数の値の指定](#)を参照してください。

例

```
epmautomate runBusinessRule RollupUSSales Period=Q1 Entity=USA
```

runCalc

アプリケーション内の計算を実行します。

このコマンドを使用すると、POV 間でルールをコピーせずに、別のデータ POV 内のデータに対してモデル POV 内のルールを使用して計算を実行できます。

適用対象

Profitability and Cost Management

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

epmautomate runCalc APPLICATION_NAME POV_NAME [DATA_POV_NAME] PARAMETER=VALUE
[comment="comment"] stringDelimiter="DELIMITER"。ここで:

- APPLICATION_NAME は、計算する POV を含む Profitability and Cost Management アプリケーションの名前です。
- POV_NAME は、計算するモデル POV の名前です。
- POV_NAME はオプションで、モデル POV のルールを使用して計算するデータ POV の名前です。
DATA_POV_NAME が指定されていない場合は、既定で POV_NAME が使用されます。
DATA_POV_NAME を指定した場合は、exeType=ALL_RULES のみを使用できます。
- PARAMETER=VALUE には、計算を実行するためのランタイム・パラメータとその値を指定します。パラメータと値のペアをプロセスの必要に応じて指定します。有効なパラメータと値は次のとおりです。
 - exeType=ALL_RULES|RULESET_SUBSET|SINGLE_RULE はルールの実行タイプを識別します。これは必須パラメータです。
exeType に設定された値に基づいて、次のパラメータを指定できます:
 - * exeType=ALL_RULES を指定した場合、subsetStart、subsetEnd、ruleSetName および ruleName など、ルールのサブセットまたは 1 つのルールに関連するパラメータを含めないでください。DATA_POV_NAME パラメータを設定する場合は、この exeType を使用する必要があります。
 - * exeType=SINGLE_RULE を指定した場合、ruleSetName および ruleName の値のみを指定してください。
 - * exeType=RULESET_SUBSET を指定する場合、subsetStart および subsetEnd の値を指定してください。
 - subsetStart は、ルール・セット内で最初に実行するルールのシーケンス番号を指定します
 - subsetEnd は、ルール・セット内で最後に実行するルールのシーケンス番号を指定します
 - ruleSetName には、実行する計算が含まれるルール・セットを指定します
 - ruleName は実行するルールの名前です(1 つのルールを実行する場合)
 - isClearCalculated=true|false は、既存の計算をクリアするかどうかを指定します
 - isExecuteCalculations=true|false は、計算を実行するかどうかを指定します
 - isRunNow=true|false では、プロセスをただちに実行する場合、この値を true に設定します

- optimizeReporting=true|false でこのオプション値を false に設定すると、レポート用の最適化なしで計算が実行されます。デフォルトは true です
ベスト・プラクティス:
 - * 1つのルールまたは連続したいくつかの POV を実行するときなど、処理時間の節約が必要な場合のみ、optimizeReporting=false を設定してください。
 - * 複数の同時実行計算ジョブを実行するときは、すべてのジョブで optimizeReporting=true を設定します。最後に実行するジョブのみで集計が行われるため、冗長な処理は回避され、ジョブの実行速度が低下することを防ぎます。

ノート:

パラメータ値(true または false)はすべて小文字の必要があります。

- comment には、二重引用符でコメントを囲んで指定します(オプション)
- stringDelimiter は POV 値で使用される区切り文字です。区切り文字は二重引用符で囲む必要があります。

例

```
epmautomate runCalc BksML12 2012_Jan_Actual Jan-2016 isClearCalculated=true
isExecuteCalculations=true isRunNow=true subsetStart=10 subsetEnd=20
ruleSetName="Utilities Expense Adjustment" ruleName="Occupancy Expense
Allocations" exeType="ALL_RULES" comment="Test calculation" stringDelimiter="_"
```

runComplianceReport

照合コンプライアンスで定義されたレポートを生成します。

Account Reconciliation の管理で次の情報ソースを参照してください:

- レポートの定義の詳細は、レポートの使用を参照してください。
- 事前定義済の照合コンプライアンス・レポートおよびそれらのレポートを生成するためのパラメータのリストは、照合コンプライアンスでの事前定義済レポートの生成を参照してください。

適用対象

Account Reconciliation

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

`epmautomate runComplianceReport FILE_NAME GROUP_NAME REPORT_NAME REPORT_FORMAT [Param=value]`。ここで:

- `FILE_NAME` は、生成されるレポートの一意的ファイル名です。この名前のレポートがサーバー上に存在する場合、そのレポートは上書きされます。`downloadFile` コマンドを使用して、このレポートをローカル・コンピュータにダウンロードします。
- `GROUP_NAME` は、レポートが関連付けられているグループの名前です。
- `REPORT_NAME` は、生成されるレポートの一意的名前です。
- `REPORT_FORMAT` は、次のいずれかのレポート形式です:
 - PDF
 - HTML (グラフおよびチャートではサポートされません)
 - XLSX (グラフではサポートされません)
 - CSV
 - CSV2

ノート:

`REPORT_FORMAT` CSV では、テンプレートに基づいたデータのフォーマットは許可されませんが、CSV2 では許可されます。CSV2 フォーマットのレポートの生成は、CSV 出力と比較して時間がかかります。

- `Param=value` はオプションで、レポートの生成に必要なパラメータを識別します。たとえば、「Balance By Account Type」レポートでは、「Period」(値「July 2017」)および「Currency Bucket」(値「Entered」)という2つのパラメータが使用されるとします。これらのパラメータは「Period=July 2017 "Currency Bucket=Entered"」のように指定する必要があります。

例

```
epmautomate runComplianceReport "Example_File Name""Reconciliation Manager"  
"Balance By Account Type" PDF "Period=July 2017" "Currency Bucket=Entered"
```

runDailyMaintenance

スケジュールされた日次メンテナンス・ウィンドウを待機せずに、日次サービス・メンテナンス・プロセスをすぐに開始します。

このコマンドでは、バックアップ・スナップショットを強制的に作成し、環境を更新できます。このコマンドを実行する前に、だれもこの環境を使用していないことを確認します。日次メンテナンス・スケジュールはこのコマンドによる影響を受けません。たとえば個別パッチの適用後など、次のメンテナンス・ウィンドウまで待たずに環境の変更を有効にする場合は、このコマンドを使用します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability

and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者、ユーザー(Oracle Enterprise Data Management Cloud のみ)

使用方法

`epmautomate runDailyMaintenance [skipNext=true|false] [-f]`。ここで:

- `skipNext` はオプションで、次回の日次メンテナンス・プロセスをスキップするかどうかを示します。デフォルトは **false** です。
- `-f` はオプションで、ユーザーの確認なしでメンテナンス・プロセスを強制的に開始するかどうかを示します。`-f` オプションを使用していない場合は、操作の確認を求められます。

例

- 次回のスケジュール済メンテナンスをスキップせずにサイクル外の日次メンテナンスを強制的に開始するには: `epmautomate runDailyMaintenance -f`
- サイクル外の日次メンテナンスを強制的に開始し、次回のスケジュール済メンテナンスをスキップするには: `epmautomate runDailyMaintenance -f`
- サイクル外の日次メンテナンスを開始し、次回のスケジュール済メンテナンスをスキップするには: `epmautomate runDailyMaintenance skipNext=true`

runDataRule

指定した開始期間および終了期間とインポート・オプションまたはエクスポート・オプションに基づいて、データ管理のデータ・ロード・ルールを実行します。

ノート:

このコマンドを使用して、データ・ソースから Oracle Fusion Cloud Enterprise Performance Management に直接データ・ロード統合を実行することはできません。EPM 統合エージェントを使用して、直接データ・ロードを統合してください。詳細は、[データ統合の管理の EPM 統合エージェントを使用した直接データ・ロードの実行](#)を参照してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

`epmautomate runDataRule RULE_NAME START_PERIOD END_PERIOD IMPORT_MODE EXPORT_MODE [FILE_NAME]`。ここで:

- `RULE_NAME` は、データ管理に定義されたデータ・ロード・ルールの名前です。ルール名に空白が含まれる場合は、引用符で囲む必要があります。
- `START_PERIOD` は、データがロードされる最初の期間です。この期間名は、データ管理の期間マッピングに定義されている必要があります。
- `END_PERIOD` は、複数期間データ・ロードの場合の、データがロードされる最後の期間です。単一期間ロードの場合は、開始期間と同じ期間を入力します。この期間名は、データ管理の期間マッピングに定義されている必要があります。
- `IMPORT_MODE` は、データをデータ管理にインポートする方法を決定します。

インポート・モードの設定では大文字と小文字が区別されます。使用可能な値は次のとおりです。

- APPEND: データ管理の既存の **POV** データに追加します
- REPLACE: **POV** データを削除してファイルからのデータで置換します
- RECALCULATE: データを再計算します
- NONE: データ管理のステージング表へのデータ・インポートをスキップします
- `EXPORT_MODE` は、アプリケーションにデータをエクスポートする方法を決定します。

エクスポート・モードの設定では大文字と小文字が区別されます。使用可能な値は次のとおりです。

- STORE_DATA: データ管理のステージング表のデータを既存のデータとマージします。データ管理ジョブをメタデータのロードに使用する場合は、常に、このエクスポート・オプションを使用します。
- ADD_DATA: データ管理のステージング表のデータをアプリケーションに追加します
- SUBTRACT_DATA: データ管理のステージング表のデータを既存のデータから削除します
- REPLACE_DATA: **POV** データをクリアしてデータ管理のステージング表のデータで置換します。データがクリアされる対象は、シナリオ、バージョン、年、期間およびエンティティです
- NONE: データ管理からアプリケーションへのデータ・エクスポートをスキップします

 ノート:

Financial Consolidation and Close では、次のエクスポート・モードのみサポートされます:

- MERGE: データ管理のステージング表のデータを既存のデータとマージします
- REPLACE: **DM** ステージング表からエントリを削除し、データ・ロードのエントリと置き換えます
- NONE: データ管理からアプリケーションへのデータ・エクスポートをスキップします

Oracle Fusion Cloud がターゲットの場合、次のエクスポート・モードのみサポートされます。

- MERGE: データ管理のステージング表のデータを既存のデータとマージします
- NONE: データ管理からアプリケーションへのデータ・エクスポートをスキップします

- *FILE_NAME* は、オプションのファイル名です。ファイル名を指定しないと、データ・ロード・ルールに指定されたファイル名に含まれるデータがインポートされます。このファイルは受信ボックス・フォルダまたはその中のフォルダにある必要があります。

Account Reconciliation 用の **Bank Administration Institute (BAI)** フォーマットのファイルをロードする場合は、このパラメータの値を指定しないでください。データ・ルール定義では、**BAI** ファイルをロードするためのファイル名を常に指定する必要があります。

 ノート:

データ・ルールにパスが指定されている場合、コマンドにはファイル・パスを指定せずに、ファイル名のみを指定してください。データ・ルールにパスが指定されていない場合、データ・ファイルへの完全なパスを指定してください。

例

- 複数期間インポート:
epmautomate runDataRule VisionActual Mar-15 Jun-15 REPLACE STORE_DATA inbox/Vision/GLActual.dat
- 単一期間インポート:
epmautomate runDataRule "Vision Actual" Mar-15 Mar-15 REPLACE STORE_DATA inbox/Vision/GLActual.dat

runDMReport

データ管理レポートを作成して、outbox/reports フォルダに格納します。

生成されたレポートの名前は、レポートを生成するデータ管理ジョブの ID とレポート形式に基づいて付けられます。たとえば、レポート・ジョブ ID が **2112** で、指定したレポート出力形式が **PDF** の場合、レポート名は 2112.pdf になります。レポート名は、レポートの生成後にコンソールに表示されます。データ管理の「プロセスの詳細」タブで、または [listFiles](#) コマンドを使用してもレポート名を識別できます。

`downloadFile` コマンドを使用して、レポートをローカル・コンピュータにダウンロードします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

`epmautomate runDMReport REPORT_NAME PARAMETER=Value "Report Output Format=[PDF|HTML|XLS|XLSX]"`。ここで:

- `REPORT_NAME` は、レポートの生成に使用されるデータ管理レポート・テンプレートの名前です。
- `PARAMETER=Value` には、レポートのパラメータとその値を指定します。必要なパラメータをいくつでも `PARAMETER=Value` 形式で指定します。必要なパラメータのリストは、生成しようとするレポートによって異なります。

ノート:

レポートのランタイム・パラメータは、レポートの設計時に定義されます。このコマンドを実行するには、これらのパラメータと値を生成して、「ワークフロー」タブからこれらを使用する必要があります。レポートのランタイム・パラメータを生成するには、データ管理の「ワークフロー」タブで「**レポートの実行**」をクリックし、「**レポート・グループ**」からグループを選択します。パラメータを生成するレポートを選択してから、「**レポート・スクリプトの作成**」をクリックします。必要な場合には、レポートのパラメータ値を指定し、出力形式を選択してから、「**OK**」をクリックします。「**レポート・スクリプトの生成**」に表示されるパラメータを使用して、ランタイム・パラメータと値を指定し、レポートを生成します。

- `Report Output Format` には、レポート出力形式を指定します。有効なオプションは、PDF、HTML、XLS、XLSX です。デフォルトのレポート形式は PDF です。

例

```
epmautomate runDMReport "TB Current Location By Target Acct (Cat,Per)"  
"Period=Jul 14" "Category=Forecast" "Location=FCSTtoVISCONSOL1" "Rule  
Name=FCSTtoVISCONSOL1" "Report Output Format=HTML"
```

runIntegration

データを Oracle Fusion Cloud Enterprise Performance Management ビジネス・プロセスにインポートするか、ビジネス・プロセスから外部システムにエクスポートするには、データ統合ジョブを実行します。

このコマンドにより、`runDataRule` コマンドが非推奨になります。`runDataRule` コマンドではなくこのコマンドの使用を開始することをお勧めします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー

使用方法

```
epmautomate runIntegration JOB_NAME importMode=Append|Replace|"Map and Validate"|"No Import"|Direct exportMode=Merge|Replace|Accumulate|Subtract|"No Export"|Check periodName={PERIOD_NAME} [inputFileName=FILE_NAME] [PARAMETERS]
```

- 標準モードの統合の場合は、importMode、exportMode および periodName の値を指定する必要があります
- クイック・モードの統合の場合は、exportMode の値を指定する必要があります
- パラメータ名とその値では大文字と小文字が区別されます

このコマンドのパラメータは次のとおりです:

- `JOB_NAME` は、データ統合に定義された統合ジョブ名です。
- `importMode` は、データをデータ統合にインポートする方法を決定します。使用可能なインポート・モードは次のとおりです:
 - `APPEND`: データ統合の既存の **POV** データに追加します。
 - `Replace`: **POV** データを削除してファイルからのデータで置換します。
 - `Map and Validate`: データのインポートをスキップし、更新されたマッピングおよびロジック勘定科目を使用してデータを再処理します。
 - `No Import`: データ統合のステージング表へのデータ・インポートをスキップします。
- `exportMode` は、データをターゲット・アプリケーションにロードする方法を決定します。クイック・モードの統合の場合は、`exportMode` パラメータの値として `Check` および `No Export` を使用できません。使用可能なエクスポート・モードの値は次のとおりです:
 - `Merge`: 既存のデータを更新し、新しいデータを追加します。
 - `Replace`: **POV** の既存のデータをクリアし、新しいデータを使用してロードします。標準モードの場合、データがクリアされる対象は、シナリオ、バージョン、年、期間およびエンティティ・ディメンションです。クイック・モードの場合、データがクリアされる対象は、年、期間およびエンティティ・ディメンションです。両方のモードでカスタム領域のクリアを定義できます。
 - `Accumulate`: データを既存のデータに追加します。Planning、Planning モジュール、Financial Consolidation and Close、Tax Reporting、Profitability and Cost Management および Enterprise Profitability and Cost Management に適用されます。
 - `Subtract`: データを既存の残高から削除します。Profitability and Cost Management および Enterprise Profitability and Cost Management に適用されます。クイック・モード統合の場合:

- * このパラメータの値に Check および No Export は使用できません。
 - * **Planning、Planning モジュール** および **Financial Consolidation and Close** の場合、有効な値は Replace、Merge および Accumulate のみです。
 - No Export: データ・エクスポートをスキップします。データをターゲット・アプリケーションにロードする前に確認のためにステージング表にロードするには、このモードを使用します。
 - Check: データの検証チェックのみを実行します。
- Oracle Fusion Cloud がターゲットの場合、次のエクスポート・モードのみサポートされます。
- * MERGE: データ統合のステージング表のデータを既存のデータとマージします
 - * NONE: データ統合からアプリケーションへのデータ・エクスポートをスキップします
- periodName は、データをインポートまたはエクスポートする 1 つ以上の期間または期間範囲の名前で、それぞれを中カッコで囲みます。使用可能な期間の命名規則は次のとおりです:
 - 単一期間ロードの場合は、{Jan-21} のように、期間名を中カッコで囲んで指定します
 - 複数期間ロードの場合は、{Jan-21}{Mar-21} (1 月 21 日から 3 月 21 日までのすべての期間のデータをロードする) のように、開始期間名と終了期間名を中カッコで囲みます
 - **Planning、Planning モジュール、Financial Consolidation and Close、フリーフォームおよび Tax Reporting の場合:** ビジネス・プロセス期間名と年を {Jan#FY21}{Mar#FY21} (1 月 21 日から 3 月 21 日までのすべての期間のデータをロードする) の形式で指定できます。
期間名は、中カッコで囲む必要があります。
 - * 単一期間—期間マッピングに定義されている単一期間のデータ管理期間名を参照します。
 - * 複数期間—複数期間ロードを参照します。パラメータは、{Month-Year}{Month-Year} の形式で指定します。たとえば、1 月 20 日から 3 月 20 日までの複数期間ロードの場合は、{Jan-20}{Mar-20} と指定します。
 - * **Planning 期間名—{Month#Year} 形式の Planning 期間名を参照します。** たとえば、{Jan#FY20}{Mar#FY20}。この規則を使用すると、データ統合期間名を指定する必要がありません。かわりに、年ディメンションとシナリオ・ディメンションのメンバー名を指定する必要があります。
このパラメータは、**Planning、Tax Reporting** および **Financial Consolidation and Close** ビジネス・プロセスでサポートされています。オンプレミス・データ・ソースから導出されたサービス・アプリケーションとクラウド・デプロイメントの両方で機能します。

この規則は、年および期間メンバー名をキャプチャして **Cloud EPM Groovy** スクリプトからトリガーされた場合に使用すると便利です。期間マッピングのターゲット値の年と月を持つアプリケーション期間マッピングまたはグローバル期間マッピングが存在する必要があります。
 - * 代替変数—前の **Planning 期間名** の拡張として、実際の年および月メンバー名かわりに、{Month#&CurYr}{&FcstMonth#&CurYr} 形式で代替変数を指定できるようにします。{Jan#&CurYr}{&FcstMonth#&CurYr} などです。
実際のメンバー名と代替変数の両方を組み合わせることがサポートされています。

この形式は、**Planning**、**Tax Reporting** および **Financial Consolidation and Close** ビジネス・プロセスでサポートされています。

期間マッピングのターゲット値で使用可能な年と月の値を持つアプリケーション期間マッピングまたはグローバル期間マッピングが、コマンドが実行される環境のデータ統合に存在する必要があります。この場合、年および月は実行中の代替変数の現在の値を参照します。

- * グローバル **POV**—グローバル **POV** 期間のデータ・ロードを実行します。
{GLOBAL_POV}形式を使用します。

 **Note:**

ここで説明されているパラメータ以外の期間名パラメータを使用した場合は、入力が無効です - HTTP 400 というエラー・メッセージが表示されません。

- {GLOBAL_POV}: システムのグローバル **POV** またはデータ統合のアプリケーション設定で定義されている期間のデータ・ロードを実行します。

 **Note:**

期間命名規則の{Month#Year}という形式は、**Planning**、**Planning** モジュール、**Financial Consolidation and Close** および **Tax Reporting** でサポートされています。この規則では、データ統合期間名のかわりに、年およびシナリオ・ディメンションのメンバー名を指定できます。このアプローチは、コマンドが年および期間メンバー名をキャプチャして **Groovy** スクリプトからトリガーされる場合に便利です。

代替変数の{Jan#&CurYr}{&FcstMonth#&CurYr}という命名規則は、前の期間命名規則の拡張です。代替変数は、このコマンドを **Planning**、**Planning** モジュール、**Financial Consolidation and Close** および **Tax Reporting** に対して実行する場合に、年および月メンバー名のかわりに指定できます。メンバー名と代替変数を組み合わせることもサポートされています。

前の期間名と代替変数の命名規則は、ターゲット値の年と月を持つアプリケーション期間マッピングまたはグローバル期間マッピングがデータ統合にすでに存在する場合にのみ機能します。

- `inputFileName` はファイルベースのデータ・ロード用で、受信ボックスで使用可能なデータのインポート元のファイル名を指定します。統合の定義でディレクトリ名を指定しない場合は、そのファイル名のみを渡します。統合の定義にディレクトリ名を含めない場合は、`inbox/DIR_NAME/FILE_NAME`形式を使用します。たとえば、`inbox/GLBALANCES.txt` または `inbox/EBSGL/GLBALANCES.txt` とします。ファイルが環境のデフォルトの場所にアップロードされた場合は、`#epminbox/FILE_NAME` 規則を使用して入力データ・ファイルを識別します。たとえば、`#epminbox/GLBALANCES.txt` とします。
このパラメータは、ネイティブ・ファイルベースのデータ・ロードにのみ適用されます。ファイルベースのデータ・ロードでこのパラメータ値を指定しない場合は、このコマンドによって、統合の定義に指定されているファイルからデータがインポートされます。ファイルベースではないデータ・ロードでこのパラメータ値を指定した場合は、無視されます。
- `PARAMETERS` はオプションで、`PARAMETER_NAME="PARAMETER"`形式のランタイム・パラメータを指定します。パラメータにはソース・フィルタとターゲット・オプションの両方が含まれます。

 **Note:**

現時点でディメンション(メタデータ)タイプのターゲット・アプリケーションに対して使用できる唯一のパラメータは"Refresh Database"=Yes|No です。

例

- 単一期間インポート:
`epmAutomate runIntegration VisionDataLoad importMode=Replace exportMode=Merge periodName="{Mar-15}"`
- 複数期間インポート:
`epmAutomate runIntegration VisionDataLoad importMode=Replace exportMode=Merge periodName="{Mar-15}{Jun-15}"`
- ファイルベースの増分データ統合:
`epmAutomate runIntegration IncrementalFileLoad importMode=Replace exportMode=Merge periodName="{Jan-20}{Mar-20}" inputFile=File1.txt`

runIntercompanyMatchingReport

会社間照合レポートを生成します。これは、連結プロセスの一環として、また、分析と監査の目的で、関連エンティティとパートナー間のトランザクションを正しく一致させるのに役立ちます。

適用対象

Financial Consolidation and Close

必要な役割

- サービス管理者
- パワー・ユーザー(ACL を介して追加のセキュリティが付与される場合)
- ユーザー(ACL を介して追加のセキュリティが付与される場合)

使用方法

`epmAutomate runIntercompanyMatchingReport JOB_NAME FILE_NAME [scenario=scenario] [years=years] [period=period_name] [ReportFormat=HTML]`。ここで:

- `JOB_NAME` は、アプリケーションに存在する会社間レポート・ジョブ定義の名前です。詳細は、*Financial Consolidation and Close* の操作で会社間照合レポートの管理を参照してください。このコマンドは、ジョブで使用可能な設定を使用して会社間照合レポートを生成します。生成されたレポートの場所として、送信ボックスを選択してください。

 **Note:**

このコマンドの実行中に、シナリオ、年、期間およびレポート形式のオプションの値を入力すると、ジョブに指定されたこれらの設定を上書きできます。

- `FILE_NAME` は、レポート・ファイルの名前です。このレポートは送信ボックスに生成されます。[downloadFile](#) コマンドを使用してローカル・コンピュータにダウンロードするか、[sendMail](#) コマンドを使用して電子メールで送信します。

- SCENARIO は、オプションで、アプリケーションに定義された、レポート生成対象となるシナリオの名前です。
- YEARS は、オプションで、レポート生成対象となる年です。
- PERIOD は、オプションで、レポート生成対象となる期間です。
- ReportFormat は、オプションで、レポートの形式です。許容値は HTML、PDF および XLSX です。

 **Note:**

- このレポートは、1つのシナリオ、年および期間の組合せに対して生成できます。複数のシナリオ、年および期間を指定しないでください。
- シナリオ、年、期間およびレポート形式の値を指定しないと、レポート・ジョブ定義に指定された対応する値(JOB_NAME で識別)が使用されます。指定されている場合、これらのオプション値はレポート・ジョブ定義の値セットを上書きします。

例

- ジョブ定義の値セットを上書きしてレポートを作成:

```
epmautomate runIntercompanyMatchingReport IC_Job_01 SampleICReport.html
scenario=Actual years=FY22 period=Jan reportFormat=HTML
```
- ジョブ定義の値セットを使用してレポートを作成:

```
epmautomate runIntercompanyMatchingReport IC_Job_01 SampleICReport
```

runMatchingReport

トランザクション照合で定義されたレポートを生成します。

事前定義済みのトランザクション照合レポートおよびそれらのレポートを生成するためのパラメータのリストは、**Account Reconciliation の管理**のトランザクション照合での事前定義済レポートの生成を参照してください。

適用対象

Account Reconciliation

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

epmautomate runMatchingReport *FILE_NAME* *GROUP_NAME* *REPORT_NAME* *REPORT_FORMAT* [Param=value]。ここで:

- *FILE_NAME* は、生成されるレポートの一意的ファイル名です。この名前のレポートがサーバー上に存在する場合、そのレポートは上書きされます。 [downloadFile](#) コマンドを使用して、このレポートをローカル・コンピュータにダウンロードします。
- *GROUP_NAME* は、レポートが関連付けられているグループの名前です。
- *REPORT_NAME* は、生成されるレポートの一意的名前です。
- *REPORT_FORMAT* は、次のいずれかのレポート形式です:
 - PDF
 - HTML (グラフおよびチャートではサポートされません)
 - XLSX (グラフではサポートされません)
 - CSV
 - CSV2

ノート:

REPORT_FORMAT CSV では、テンプレートに基づいたデータのフォーマットは許可されませんが、CSV2 では許可されます。CSV2 フォーマットのレポートの生成は、CSV 出力と比較して時間がかかります。

- Param=Value はオプションで、レポートの生成に必要なパラメータを識別します。たとえば、「Match Type Configuration」レポートで、「status」(値「approved」)というパラメータが使用される場合は、パラメータと値を status=Approved のように指定します。

例

```
epmautomate runMatchingReport Example_FileName "Transaction Matching" "Match Type Configuration" HTML "status=Approved"
```

runPipeline

パイプラインに定義された変数に基づいて、データ統合パイプライン(一連のジョブを単一のプロセスとして編成)を実行します。パイプラインの詳細は、*Oracle Enterprise Performance Management Cloud データ統合の管理*のパイプラインの使用を参照してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Enterprise Profitability and Cost Management、Tax Reporting、Sales Planning および Strategic Workforce Planning

必要な役割

サービス管理者または(場所のセキュリティを使用するか、システム設定の「**非管理ユーザーに対してパイプライン実行を使用可能にする**」設定を「はい」に設定することで)パイプライン定義を実行するプロキシ・ユーザーとして設定されたユーザー。

使用方法

epmautomate runPipeline PIPELINE_CODE [PARAMETER=VALUE]。ここで:

- PIPELINE_CODE は、データ統合でパイプラインを作成する際に定義されるコードです。詳細は、*データ統合の管理*のパイプライン・プロセスの説明を参照してください。
- PARAMETER=VALUE はオプションで、パイプラインを実行するためのパラメータとその値を識別します。パラメータと値のペアをプロセスの必要に応じて指定します。パラメータのリストは、データ統合のパイプライン変数の画面で定義された変数の数によって異なります。デフォルトのパイプライン・パラメータおよび使用可能な値は次のとおりです:
 - STARTPERIOD は、データがロードされる最初の期間です。この期間名は、データ統合の期間マッピングに定義されている必要があります。
 - ENDPERIOD は、データがロードされる最後の期間です。この期間名は、データ統合の期間マッピングに定義されている必要があります。
 - IMPORTMODE は、データをデータ統合にインポートする方法を決定します。使用可能な値は次のとおりです。
 - * Append: データ統合の既存の POV データに追加します。
 - * Replace: POV データを削除し、パイプライン定義またはパラメータに指定されたファイルのデータで置換します。
 - * Map and Validate: パイプライン定義またはパラメータに指定されたファイルからデータをインポートせずに、更新されたマッピングおよびロジック勘定科目でデータを再処理します。
 - * No Import: データ統合のステージング表へのデータ・インポートをスキップします。
 - EXPORTMODE は、データをデータ統合にエクスポートする方法を決定します。使用可能なエクスポート・モードは次のとおりです
 - * Merge: データ統合のステージング表のデータをビジネス・プロセスの既存のデータとマージします。
 - * Accumulate: データ統合のステージング表のデータをビジネス・プロセスに追加します。このエクスポート・モードは、**Planning**、**Planning** モジュール、**フリーフォーム**、**Sales Planning** および **Strategic Workforce Planning** にのみ適用されます。
 - * Replace: ビジネス・プロセスの POV データを削除し、データ統合のステージング表のデータで置換します。データがクリアされる対象は、シナリオ、バージョン、年、期間およびエンティティ・ディメンションです。
 - * No Export: データ統合からビジネス・プロセスへのデータのエクスポートをスキップします。
 - ATTACH_LOGS は、ログ・ファイルを zip に圧縮して、パイプライン実行に関連する通知に添付するかどうかを指定します。使用可能な値は、はいの場合は Y、いいえの場合は N です。
 - SEND_MAIL: パイプラインの実行が完了したときに電子メールを送信するかどうかを指定します。使用可能な値は、Always、No (デフォルト)、On Failure および On Success です。
 - SEND_TO は、電子メールを送信する電子メール ID のカンマ区切りリストです。

例

```
epmautomate runPipeline DAILYLOAD "STARTPERIOD=Jan-24" "ENDPERIOD=Jan-24"
"IMPORTMODE=Replace" "EXPORTMODE=Merge" "SEND_MAIL=Always"
"SEND_TO=John.Doe@example.com, Jane.Doe@example.com" "ATTACH_LOGS=Y"
```

runPlanTypeMap

plan type map タイプのジョブで指定された設定に基づいて、ブロック・ストレージ・データベースから集約ストレージ・データベースへ、またはブロック・ストレージから別のブロック・ストレージへデータをコピーします。

適用対象

Planning、Planning モジュール、フリーフォーム、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

epmautomate runPlanTypeMap *JOB_NAME* [clearData=true|false]。ここで:

- *JOB_NAME* はアプリケーションで定義された plan type map タイプのジョブの名前です。
- clearData は、データのコピー前にターゲット・データベースのデータを削除するかどうかを示すオプションの設定です。このパラメータ値が設定されない場合、デフォルト値 true が使用されます。

パラメータ値(true または false)はすべて小文字の必要があります。

例

```
epmautomate runPlanTypeMap CampaignToReporting clearData=false
```

runRuleSet

ビジネス・ルールセットを起動します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

「サービス管理者」、「パワー・ユーザー」(ルール起動アクセス許可が付与されている場合)

使用方法

epmautomate runRuleSet *RULESET_NAME* [PARAMETER=VALUE]。ここで:

- *RULESET_NAME* は、環境で定義されているとおりの正確なビジネス・ルールセットの名前です。

- `PARAMETER=VALUE` は、ルールセットの実行に必要なオプションのランタイム・パラメータとその値を示します。

ノート:

ルールセットは、それがデプロイされているプラン・タイプに対して実行されません。

`PARAMETER=VALUE` のペアを使用して、ルールセットで必要とされるだけの実行時プロンプトを指定します。次の例では2つの実行時プロンプト(Period および Entity)とその値(Q1 および USA)を使用します。

ランタイム・パラメータに値を指定しない場合は、デフォルト値が使用されます。このコマンドは、ルールセットで定義されているものと完全一致しない実行時プロンプトを無視します。

パラメータに複数の値を入力する方法の詳細は、[パラメータに対する複数の値の指定](#)を参照してください。

例

```
epmautomate runRuleSet RollupUSSales Period=Q1 Entity=USA
```

runSupplementalDataReport

補足データ・マネージャからのデータを表示するリレーショナル・レポートを生成します。

補足データ・レポートは、**Financial Consolidation and Close** および **Tax Reporting** で非連結レポートとしてグループ化されます。生成できるレポートおよびそれらを生成するパラメータのリストは、**REST API** の **Financial Consolidation and Close** および **Tax Reporting** のレポートの生成の事前定義済レポートおよびパラメータのリストの項を参照してください。

適用対象

Financial Consolidation and Close, および Tax Reporting

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

使用方法

```
epmautomate runSupplementalDataReport FILE_NAME GROUP_NAME REPORT_NAME  
REPORT_FORMAT [Param=value]。ここで:
```

- `FILE_NAME` はレポートの一意のファイル名です。
- `GROUP_NAME` は、レポートが関連付けられているグループの名前です。
- `REPORT_NAME` は、生成されるレポートの一意の名前です。
- `REPORT_FORMAT` は、次のいずれかのレポート形式です:

- PDF
- HTML (グラフおよびチャートではサポートされません)
- XLSX (グラフではサポートされません)
- CSV
- CSV2

`REPORT_FORMAT` CSV では、テンプレートに基づいたデータのフォーマットは許可されませんが、CSV2 では許可されます。CSV2 フォーマットのレポートの生成は、csv 出力と比較して時間がかかります。

- `Param=value` はオプションで、レポートの生成に必要なパラメータを識別します。たとえば、`schedule name` に `monthly` の値、`period` に `Jan` の値を使用する「リスクありタスク」レポートを生成するには、"`schedule name`"=`monthly` `period`=`Jan` を指定します。

例

```
epmautomate runSupplementalDataReport Example_File_name Group1 "At Risk Tasks"
html "schedule name"=monthly period=Jan
```

runTaskManagerReport

タスク・マネージャからのデータを表示するリレーショナル・レポートを生成します。

適用対象

Enterprise Profitability and Cost Management、Financial Consolidation and Close および Tax Reporting。


必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

使用方法

```
epmautomate runTaskManagerReport FILE_NAME GROUP_NAME REPORT_NAME REPORT_FORMAT
[Param=value]。ここで:
```

- `FILE_NAME` はレポートの一意のファイル名です。
- `GROUP_NAME` は、レポートが関連付けられているグループの名前です。
- `REPORT_NAME` は、生成されるレポートの一意の名前です。
- `REPORT_FORMAT` は、次のいずれかのレポート形式です:
 - PDF
 - HTML (グラフおよびチャートではサポートされません)
 - XLSX (グラフではサポートされません)
 - CSV
 - CSV2

 **ノート:**

`REPORT_FORMAT` CSV では、テンプレートに基づいたデータのフォーマットは許可されませんが、CSV2 では許可されます。CSV2 フォーマットのレポートの生成は、CSV 出力と比較して時間がかかります。

- `Param=value` はオプションで、レポートの生成に必要なパラメータを識別します。たとえば、`schedule name` に `monthly` の値、`period` に `Jan` の値を使用する「先行タスク」レポートを生成するには、"`schedule name=monthly period=Jan`" を指定します。

例

```
epmautomate runTaskManagerReport Example_File_name Group1 "Early Tasks" PDF
"schedule name=monthly period=Jan
```

sendMail

Oracle Fusion Cloud Enterprise Performance Management から、ファイルを添付するオプションを含む電子メールを送信します。

このコマンドをスクリプトに組み込み、様々な状態をユーザーに通知したり、レポートを送信できます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Narrative Reporting、Oracle Fusion Cloud Enterprise Data Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

```
epmautomate sendMail ToAddress Subject [Body="MessageBody"]
[Attachments=FILE1,FILE2]。ここで:
```

- `ToAddress` は、二重引用符で囲まれた受信者のセミコロン区切りの電子メール・アドレスを識別します。例: "`jd@example.com;jane.doe@example.com`"。
- `Subject` は、メールの件名を識別します。
- `Body="MessageBody"` は、オプションであり、電子メールのコンテンツです。指定されない場合、電子メールの本文はありません。

 **Note:**

有効な HTML タグを使用してメッセージ本文をフォーマットし、必要な電子メール・フォーマットを作成します。メッセージ本文全体(すべての HTML タグを含む)を 1 行として指定する必要があり、改行文字を含めることはできません。例を参照してください。

- Attachments はオプションで、電子メールに添付する Cloud EPM にあるファイルのカンマ区切りリストを識別します。例: outbox/errorFile.txt, inbox/users.csv。

 **Note:**

- * (アスタリスク)は、ファイル名の 1 文字に対するワイルドカードとして使用します。たとえば、outbox/user*.csv を指定して、パターンに適合する 5 文字のファイル名を持つ送信ボックス内のすべてのファイルを添付します。
- [listFiles](#) コマンドでリストされた、スナップショット以外の任意のファイルを電子メールの添付ファイルとして添付できます。添付のサイズは 10 MB を超えないようにする必要があります。

例

- **フォーマットされていない電子メール:** `epmautomate sendMail "jdoe@example.com;jane.doe@example.com" "Data Load Process Failed" Body="Data Load 1 Failed" Attachments=outbox/Errorfile.txt,outbox/Errofile2.txt`
- **フォーマットされている電子メール:** `epmautomate sendMail jdoe@example.com "Send Formatted Email" "Body=<!DOCTYPE html> <html> <head> <title>EpmAutomate Email Formatting</title> </head> <body> <h1>EpmAutomate Email Formatting</h1><p>Hi,</p><p>Test Allocation Rules, Volume, and SPT data were loaded into FY22_Feb_Actual_Version POV.</p><p>Check the attachment for details.</p></body></html>" Attachments=outbox/loadResults.txt`

setApplicationAdminMode

アプリケーションを管理モードにし、アプリケーションへのアクセス権がサービス管理者のみに限定されるようにします。

このコマンドは、サービス管理者が管理操作を実行しているときにユーザーがアプリケーションを操作できないようにするときに役立ちます。アプリケーションは、モードを戻してすべてのユーザーがアクセスできるようにするまでは管理モードのままです。

 **Note:**

このコマンドは、[applicationAdminMode](#) (現在は非推奨ですが、EPM 自動化から削除されていません)を置き換えるものです。

[getApplicationAdminMode](#) コマンドを使用して、環境の現在のステータスを確認します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Account Reconciliation、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

```
epmautomate setApplicationAdminMode true|false
```

このコマンドでは、アプリケーションを管理モードに設定するには `true` を指定し、通常モードに戻してすべてのユーザーがアクセスできるようにするには `false` を指定します

例

- アプリケーションを管理モードに設定:
`epmautomate setApplicationAdminMode true`
- アプリケーションを通常操作に戻す:
`epmautomate setApplicationAdminMode false`

setDailyMaintenanceStartTime

環境の日次メンテナンスを開始する時間(UTC または別のタイム・ゾーン)を設定します。環境のメンテナンスは、正時に開始する必要はありません。メンテナンスを開始する必要がある時間および分を設定できます。

このコマンドの使用が、バックアップを作成する Oracle の要件を干渉しないように、日次メンテナンス・プロセスが過去 36 時間実行されなかった場合、このコマンドは開始時間を変更しません。

ノート:

ブラウザを使用して現在環境にログインしているサービス管理者に新しい日次メンテナンスの開始時間が表示されるのは、サインアウトしてからサインインした後のみです。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者、ユーザー(Oracle Enterprise Data Management Cloud のみ)、任意の事前定義済役割および移行 - 管理アプリケーション役割

使用方法

```
epmautomate setDailyMaintenanceStartTime StartTime。ここで、StartTime は、メンテナ  
ンス・プロセスを開始する時間(24 時間制を使用した HH:MM 形式)およびオプションのタイ  
ム・ゾーンです。使用可能な開始時間の値の範囲は 00:00 から 23:59 です。開始時間を UTC で  
設定しない場合は、有効な標準タイム・ゾーンを指定します。たとえば、太平洋標準時午後 2  
時 35 分の場合は、"14:35 America/Los_Angeles"です。
```

例

- 日次メンテナンス開始を UTC 午後 2 時 20 分に設定します:

```
epmautomate setDailyMaintenanceStartTime 14:20
```

- 日次メンテナンス開始を太平洋標準時の午後 2 時 35 分に設定します:

```
epmautomate setDailyMaintenanceStartTime "14:35 America/Los Angeles"
```

setDemoDates

必要に応じて Oracle の内部デモ・データを更新します。

このコマンドは、Oracle の内部デモ・データを使用してセットアップされたインストールに対してのみ使用します。

Account Reconciliation のみ: このコマンドは、Demo Code 属性に値 `setdemodates` または `setdemodatesnostatuschange` が関連付けられている照合すべてについて日付をリセットします。このコマンドは、最大 12 期間(現在の期間と前(履歴)の 11 期間)の照合を処理します。Demo Code 属性で 3 つ以上の期間の照合にタグが付けられている場合、コマンドではこれらの期間が前の期間に属するものとして扱われます。この属性値を保持しない照合には作用しません。

- 値が `setdemodates` の場合、このコマンドは指定された日付およびランダム・ステータスに基づいて照合の日付をリセットします
- 値が `setdemodatesnostatuschange` の場合、このコマンドは指定された日付に基づいて照合の日付をリセットしますが、照合のステータスは変更しません

その他すべてのビジネス・プロセス: このコマンドはタスクの開始日と終了日、および他の関連する日付情報をリセットし、タスクがデモに適切になるようにします。タスクのスケジュールで設定された `SETDEMODATES` 属性の値、およびユーザーが指定した Demo Date の値に基づいて、新しいタスクの日付が計算されます。Demo Date 値を指定しない場合、コマンドは今日の日付を使用して新しいタスクの日付を計算します。

ノート:

`SETDEMODATES` 値を持たないスケジュール内のタスクには影響しません。

指定した Demo Date に基づいて、このコマンドはタスクに関連付けられたすべての日付を前方に移動します。これにはコア・ランタイムの日付(開始日、終了日など)および履歴、個々のワークフローの期限日および開始日(実際)などの補助的な日付が含まれます。タスクのステータスには影響しません。

適用対象

Account Reconciliation、Enterprise Profitability and Cost Management、Financial Consolidation and Close、フリーフォーム、Planning、Planning モジュール、Sales Planning、Strategic Workforce Planning および Tax Reporting

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

`epmautomate setDemoDates [DEMO_DATE]`。ここで、`DEMO_DATE` はオプションの日付(YYYY-MM-DD 形式)です。この値を指定しないと、照合は現在の日付にリセットされます。

例

```
epmautomate setDemoDates 2020-02-15
```

setEJJournalStatus

Financial Consolidation and Close で、ERP システムからのエンタープライズ仕訳の転記結果を設定します。このコマンドを使用して、ワークフロー・ステータスに関係なく、Post in Progress ステータスの仕訳の転記ステータスを更新します。

このコマンドは、ERP システムへのインポートのステータスを識別する **CSV** ファイルを使用します。`uploadFile` コマンドを使用して、インポート・ファイルを環境にアップロードします。CSV ファイルの形式は次のとおりです。

```
Year,Period,Journal ID,Posting Status,Message
2020,Dec,1000001021,Posted,"SUCCESS"
2020,Dec,1000001022,Failed,"Row Header No: 2,10000415 - Linked value 6 does
not exist Application-defined or object-defined error 65171"
2020,Dec,1000001022,Failed,"Row Header No: 7,10000415 - Z_ECS_MSG (001)Enter
a valid account number"
2020,Dec,1000001022,Failed,"Row Header No: 7,10000415 - Z_ECS_MSG (002) Enter
a valid cost center"
```

Message 列はオプションで、省略できます。

このコマンドは、**Financial Consolidation and Close** からのエンタープライズ仕訳データのエクスポートや、ERP システムへのインポートは行いません。

適用対象

Financial Consolidation and Close

必要な役割

サービス管理者

使用方法

`epmautomate setEJJournalStatus FILE_NAME.csv [OPERATION=posting|validation]`。ここで:

- `FILE_NAME` は、ERP システムへのインポートのステータスが含まれている **CSV** ファイルを識別します。
- Operation はオプションで、実行する操作を示します。使用可能な値は `posting` および `validation` です。デフォルトは `posting` です
`operation=posting` では、仕訳転記の結果を格納した **CSV** ファイルを提供します。このファイルには、「年」、「期間」、「仕訳 ID」、「転記ステータス」(転記済/失敗)、「メッセージ」、

「エラー・コード」および「エラー・メッセージ」の列が含まれます。「メッセージ」、「エラー・コード」および「エラー・メッセージ」列はオプションです。仕訳の転記ステータスは、ファイルで提供されたステータスに基づいて更新されます。転記ステータスが**転記進行中**の仕訳のみが更新されます。

operation = validation では、仕訳検証の結果を格納した CSV ファイルを生成します。このファイルには、「年」、「期間」、「仕訳 ID」、「転記ステータス」(有効/失敗)、「メッセージ」、「エラー・コード」および「エラー・メッセージ」の列が含まれます。「メッセージ」、「エラー・コード」および「エラー・メッセージ」列はオプションです。仕訳の検証ステータスは、ファイルで提供されたステータスに基づいて更新されます。検証ステータスが**検証進行中**の仕訳のみが更新されます。

例

- ファイルからエンタープライズ仕訳の転記結果を設定します:

```
epmautomate setEJJournalStatus JournalStatus.csv
```

- ファイルからエンタープライズ仕訳の転記結果を設定し、検証操作を実行します:

```
epmautomate esetEJJournalStatus JournalStatus.csv Operation=validation
```

setEncryptionKey

データベース・アクセス用のカスタム暗号化キーを設定します。

このコマンドの使用によって、**Bring Your Own Key (BYOK)**ソリューションが提供され、顧客は各自の標準キー管理ローテーションに **Oracle Fusion Cloud Enterprise Performance Management** を含めることができます。

カスタム暗号化キーは、環境の次の日次メンテナンス後に有効になります。すぐにアクティブにするには、[resetService](#) コマンドを実行します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

epmautomate setEncryptionKey key=key。ここで、key は、暗号化キーとして使用する任意の長さのカスタム文字列です。

例

- 暗号化キーの設定: epmautomate setEncryptionKey key=se!m+a2J
- 暗号化キーの削除: epmautomate setEncryptionKey key=

setEssbaseQryGovExecTime

Oracle Essbase 問合せが終了する前に問合せで情報を取得し配信するために使用できる最大時間(秒単位)を設定します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Profitability and Cost Management、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

`epmautomate setEssbaseQryGovExecTime TIME`。ここで、**TIME** は、**Essbase** 問合せが終了した後の秒数を識別します。この値は、**70000** を超えない自然数である必要があります。

Essbase 問合せが無期限に実行されないように、この値を **0 (ゼロ)** に設定しないことをお勧めします。

例

```
epmautomate setEssbaseQryGovExecTime 600
```

setIdleSessionTimeout

Oracle Fusion Cloud Enterprise Performance Management 環境のセッションのタイムアウト(分単位)を変更します。新規のセッションのタイムアウトは、環境の次の日次メンテナンス後にアクティブになります。このコマンドを使用して、デフォルトのセッションのタイムアウト(75分)を別の値に変更します。このコマンドを使用して指定した期間にセッションがアイドルになると、ユーザーはログイン・ページにリダイレクトされます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmautomate setIdleSessionTimeout MINUTES`。ここで、**MINUTES** は、新規のアイドル・セッションのタイムアウト(分単位、最小 **15** 分、最大 **150** 分)を識別します。

例

```
epmautomate setIdleSessionTimeout 30
```

setIPAllowlist

Oracle Fusion Cloud Enterprise Performance Management へのアクセスを許可する IP アドレスおよび Classless Inter-Domain Routing (CIDR)の許可リストを構成します。このコマンドは、IPv4 アドレスと CIDR を追加または削除します。

このコマンドによって、Cloud EPM 環境の許可リストを構成するためのセルフサービス方式が提供されます。

Note:

- Cloud EPM 環境の IP 許可リストを設定する場合、それらの特定の IP アドレスからの接続のみを許可します。このシナリオでは、別の Cloud EPM からのアクセスのリクエストは、リクエストしている環境が配置されているデータ・センターまたはリージョンのアウトバウンド IP アドレスを IP 許可リストに追加しないかぎり機能しません。IP 許可リストを設定する環境と他の環境が通信できることを確認するために追加する必要がある IP アドレスについては、オペレーション・ガイドの Cloud EPM のデータ・センターおよびリージョンのアウトバウンド IP アドレスを参照してください。
- 環境が配置されている Identity Cloud サービスの IDCS ネットワーク・ペリメータを使用している場合は、このネットワーク・ペリメータにも同じ IP アドレスを追加する必要があります。個別の環境用に許可リストを設定するかわりに、ネットワーク・ペリメータにのみ IP アドレスを追加することを選択できます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmAutomate setIPAllowlist add|remove FILE_NAME.txt`。ここで:

- `add`: テキスト・ファイルにリストされている IP アドレスと CIDR を許可リストに追加します。
- `remove`: テキスト・ファイルにリストされている IP アドレスと CIDR を許可リストから削除します。
- `FILE_NAME`: 許可リストに対して追加または削除する IP アドレスおよび CIDR をリストするテキスト・ファイルの名前です。ファイルの各エントリを改行文字で区切る必要があります。

ます。このファイルを環境にアップロードするには、`uploadFile` コマンドを使用します。ファイル内の各行は、次の形式の IPv4 アドレスまたは CIDR である必要があります:

```
xxx.xxx.xxx.xxx  
xxx.xxx.xxx.xxx/n
```

 **Note:**

- IPv4 IP アドレスのみがサポートされています。
- 個々の IP アドレスではなく、CIDR 形式を使用して、連続した範囲の IP アドレスを指定します。
- 任意の IP アドレスからのアクセスを許可する許可リストを無効にするには、`getIPAllowlist` コマンドを使用して、既存のすべての IP アドレスおよび CIDR をファイルに書き込みます。ファイルを環境にアップロードし、次の例に示すように `remove` オプションを指定してこのコマンドを実行します:

```
epmAutomate getIPAllowlist > myRemoveList.txt  
epmAutomate uploadFile myRemoveList.txt  
epmAutomate setIPAllowlist remove myRemoveList.txt
```

例

- 許可リストに IP アドレスおよび CIDR を追加する:

```
epmAutomate setIPAllowlist add myAddList.txt
```

- 許可リストから IP アドレスを削除する:

```
epmAutomate setIPAllowlist remove myRemoveList1.txt
```

setManualDataAccess

環境が応答しないため、その環境を調査して使用可能にするためのサービス・リクエストを顧客が提供していない緊急状況において、オラクル社によるその環境のリレーショナル・データベースと Oracle Essbase データベースへの手動アクセスを許可するかどうかを指定します。

緊急の状況では、オラクル社は内部プロセスを使用し、これによって、上級開発エグゼクティブは、プロセスに関する独立した検証の後、リレーショナル・データベースと Essbase データベースへの手動アクセスを許可します。このコマンドを使用して、ユーザーの明示的な承認なしにオラクル社がこれらのデータベースにアクセスしないようにします。さらに、サービス・リクエストがオープンであっても、緊急時にオラクル社がリレーショナル・データベースと Essbase データベースに手動でアクセスすることを禁止するオプションがあります。

このコマンドを使用して指定した設定はすぐに有効となります。デフォルトでは、オラクル社による手動データ・アクセスが許可されます。このコマンドを使用して、このアクセスを取り消します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmautomate setManualDataAccess Allow|Revoke [disableEmergencyAccess=true|false]`。
ここで、`disableEmergencyAccess` (オプション)は、リレーショナル・データベースと **Essbase** データベースへのすべての手動アクセスを禁止するかどうかを指定します。このプロパティ 値を `true` に設定すると、サービス・リクエストがオープンであっても、オラクル社がこれらのデータベースに手動でアクセスできなくなります。デフォルトは `false` です。

`disableEmergencyAccess` の値を `true` に指定すると、停止した環境のトラブルシューティングと修正のためにアクセスが必要ときに、オラクル社がリレーショナル・データベースと **Essbase** データベースにアクセスできません。環境が停止しているときに、このコマンドを発行して、オラクル社がこれらのデータベースに手動でアクセスできるように許可することはできません。

例

- 明示的な承認なしで緊急時にリレーショナル・データベースと **Essbase** データベースに手動でアクセスするために付与された権限を取り消します:
`epmautomate setManualDataAccess revoke`
- 緊急時におけるリレーショナル・データベースと **Essbase** データベースへの手動アクセスを許可します:
`epmautomate setManualDataAccess allow`
- サービス・リクエストがオープンであっても、リレーショナル・データベースと **Essbase** データベースへの手動アクセスを禁止します:
`epmautomate setManualDataAccess revoke disableEmergencyAccess=true`

setPeriodStatus

期間に対して特定のステータスを設定します。

適用対象

Account Reconciliation

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

「パワー・ユーザー」、「ユーザー」、「参照者」の各事前定義済役割には、追加のアプリケーション役割が必要な場合があります。

使用方法

`epmautomate setPeriodStatus PERIOD STATUS`。ここで:

- `PERIOD` は、期間の名前です
- `STATUS` は、OPEN、CLOSED または LOCKED です

例

```
epmautomate setPeriodStatus "January 2015" OPEN
```

setRestrictedDataAccess

フィードバックの提供ユーティリティを使用している間にサービス管理者がアプリケーション・スナップショットを Oracle に送信することに同意しないように、Oracle Fusion Cloud Enterprise Performance Management 環境を構成します。

ポリシーに従って、データへのアクセスを Oracle に許可しない場合は、このコマンドを使用して、Oracle へのフィードバック送信の間のデータ共有を管理します。環境に対して制限付きデータ・アクセスを設定すると、フィードバックの提供ユーティリティによって表示されるアプリケーション・スナップショットの送信オプションが無効化されます。

適用対象

Account Reconciliation、Oracle Fusion Cloud Enterprise Data Management、Enterprise Profitability and Cost Management、Financial Consolidation and Close、フリーフォーム、Planning、Planning モジュール、Profitability and Cost Management、Narrative Reporting、Sales Planning、Strategic Workforce Planning および Tax Reporting。

必要な役割

サービス管理者

使用方法

```
epmautomate setRestrictedDataAccess true|false
```

現在のデータ・アクセス制限のステータスを表示するには、[getRestrictedDataAccess](#) を使用します。

例

- サービス管理者がアプリケーション・スナップショットの送信に同意しないようにします:
`epmautomate setRestrictedDataAccess true`
- サービス管理者がアプリケーション・スナップショットの送信に同意することを許可します:
`epmautomate setRestrictedDataAccess false`

setSubstVars

アプリケーション・レベルまたはキューブ・レベルで代替変数を作成または更新します。

このコマンドを使用して代替変数に複数の値または関数を設定することはできません。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmautomate setSubstVars CUBE_NAME SUBSTVAR=VALUE [SUBSTVAR=VALUE]`。ここで:

- `CUBE_NAME` は、代替変数が作成または更新されるキューブ(Plan1、Plan2 など)です。アプリケーション・レベルで代替変数を設定または更新するには、キューブ名のかわりに ALL を使用します。
- `SUBSTVAR` は、値が設定または更新される代替変数の名前です。
- `VALUE` は、新しい代替変数の値です。

例

- アプリケーション・レベルでの代替変数の作成または更新: `epmautomate setSubstVars ALL CurYear=2015 CurPeriod=Jan`
- キューブ・レベルでの代替変数の作成または更新: `epmautomate setSubstVars Plan2 CurYear=2013 CurPeriod=Jan`

setVirusScanOnFileUploads

OCI (Gen 2)環境で、ファイルを Oracle Fusion Cloud Enterprise Performance Management にアップロードする前に、ファイルのウイルス・スキャンができます。

すべての OCI (Gen 2)環境は、アンチウイルス・プログラムを使用して保護されます。このコマンドにより、アップロード・ファイルに対するウイルス・スキャンを有効にでき、セキュリティを強化できます。ファイルをアップロードする前にスキャンすると、環境にウイルスがアップロードされなくなります。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmautomate setVirusScanOnFileUploads true|false`

デフォルトでは、ウイルス・スキャンは有効ではありません(false に設定)。この値が true に設定されると、Cloud EPM はすべてのアップロード・ファイルをスキャンします。ファイルがウイルスに感染している場合は、環境にアップロードされません。

例

- ウィルス・スキャンの有効化: `epmautomate setVirusScanOnFileUploads true`
- ウィルス・スキャンの無効化: `epmautomate setVirusScanOnFileUploads false`

simulateConcurrentUsage

ユーザーをシミュレートすることで、環境で様々な同時操作を実行します。

このコマンドを使用して、環境のパフォーマンスを検証し、特定の数のユーザーによって実行される特定の操作でサービスに負荷がかかっている状況でも許容できる応答時間であることを確認できます。たとえば、このコマンドを使用して、50人のユーザーが異なる POV を使用してフォームを同時に開いた場合のパフォーマンスを測定できます。これにより、環境のセルフサービス負荷テストが可能になります。

このコマンドは、指定されたユーザー数と反復に対して、指定された操作を実行することで、シミュレーションを実行します。複数の反復を実行し、特定の操作の最小時間、最大時間および平均時間を計算します。同時使用の負荷テストを実行するために、次の操作がサポートされています:

- フォームを開く
- フォームの保存
- ビジネス・ルールの実行
- ビジネス・ルールセットの実行
- データ・ルールの実行
- アド・ホック・グリッドを開く
- レポートの実行
- ブックの実行

Note:

このコマンドでは、**Financial Reporting** のレポートおよびブックをサポートしていません。レポート(以前の管理レポート)に属するブックおよびレポートのみがサポートされています。

Caution:

このコマンドは、指定された操作を現在の環境で実行しますが、操作によっては、環境のデータを更新する場合があります。このコマンドは、テスト環境で実行してください。このコマンドを本番環境で実行することはお薦めしません。

このコマンドは、環境の受信ボックスにすでにアップロードされている ZIP ファイルを入力として受け入れます。ZIP ファイルには、1つの `requirement.csv` ファイルと、`requirement.csv` に含まれているコース・ケースをサポートする入力ファイルが格納されています。オプションで、ZIP ファイルに含められるファイルとして、ユーザー変数メンバー・マッピングを提供するための `userVarMemberMapping.csv` ファイル、一部のコース・ケースに **Oracle Smart View**

for Office オプションを提供するための options.xml ファイル、新しいユーザーを作成するかわりに既存のユーザーが使用するユーザー名とパスワードを提供するための users.csv ファイルがあります。次に、このコマンドは、ユース・ケースをシミュレートし、1人以上の受信者に電子メールで送信できるレポートを作成します。

 **Note:**

このコマンドでは、フィードバックの提供の提供用資料は生成されません。ビジネス・プロセス画面の「フィードバックの提供」(*Oracle Enterprise Performance Management Cloud 管理者スタート・ガイド*のフィードバックの提供ユーティリティを使用してオラクル社の診断情報収集に協力するを参照)オプション、フィードバックの提供 REST API または `feedback` コマンドを使用して、フィードバックの提供用資料を生成して、シミュレーションの実行後に環境の詳細を取得します。

使用シナリオ 1: 50人のユーザーが同時にフォームを開いた場合のアプリケーション・パフォーマンスの受入テスト。

解決策:

1. Library/Global Assumption/フォルダに格納されている Exchange Rates という名前のフォームを開くことを想定して、次のようなエントリで requirement.csv を作成します:

```
# Type of Operation,Artifact Name,Number of Users,Input File,Additional Info
Open Form, Library/Global Assumption/Exchange Rates,50,open_form_input.csv,
```

2. [フォーム入力ファイルを開く](#) に指定されている形式を使用して、open_form_input.csv を作成します。このファイルには 1つのエントリがあり、50回使用されることとなります。異なる POV を含む同じフォームを開く場合は、使用する POV の数と同じ数のエントリが必要となります。
3. ユーザー変数メンバー・マッピングを設定する必要がある場合は、[UserVarMemberMapping.csv ファイルの作成](#) に指定された形式を使用して、userVarMemberMapping.csv を作成します。
4. Smart View オプションを使用する必要がある場合は、Smart View オプションを options.xml にエクスポートします。詳細は、[options.xml ファイルの作成](#) を参照してください。
5. 前のステップで作成したファイルを格納した ZIP ファイルを作成し、受信ボックスにアップロードします。
6. 前のステップの ZIP ファイルを入力ファイルとして使用して、simulateConcurrentUsage コマンドを実行します。

使用シナリオ 2: 会計年度末などの季節的な使用増加に関するパフォーマンスのシミュレート。仮定: 100人のユーザーが各ユーザー間のラグ時間 6秒でフォームを保存します。

解決策:

1. Library/Dashboards/フォルダに格納されている Accessories Revenue という名前のフォームを保存すると想定して、次のようなエントリで requirement.csv を作成します:

```
# Type of Operation,Artifact Name,Number of Users,Input File,Additional
Info
Save Form, Library/Dashboards/Accessories Revenue,100,save_form_input.csv,
```

2. [フォーム入力ファイルの保存](#)に指定されている形式を使用して、save_form_input.csv を作成します。
3. ユーザー変数メンバー・マッピングを設定する必要がある場合は、[UserVarMemberMapping.csv ファイルの作成](#)に指定された形式を使用して、userVarMemberMapping.csv を作成します。
4. Smart View オプションを使用する必要がある場合は、Smart View オプションを options.xml にエクスポートします。詳細は、[options.xml ファイルの作成](#)を参照してください。
5. 前のステップで作成したファイルを格納した ZIP ファイルを作成し、受信ボックスにアップロードします。
6. 前のステップの ZIP ファイルを入力ファイルとして使用し、iteration=1 および lagTime=6 のプロパティ値で simulateConcurrentUsage コマンドを実行します。

適用対象

Planning、Planning モジュール、FreeForm、Financial Consolidation and Close、Tax Reporting、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者。testModes 0、1 および 2 を使用するには、アイデンティティ・ドメイン管理者の役割も必要です。

使用方法

```
epmautomate simulateConcurrentUsage INPUT_FILE.zip [iterations=COUNT]
[notificationEmails="EMAIL_ADDRESS"] [testMode=0|1|2|3|4] [lagTime=LAG_TIME]。こ
こで:
```

- *INPUT_FILE.zip* は、コース・ケースを識別する ZIP ファイルの名前です。このコマンドを実行する前に、[uploadFile](#) コマンド(コマンド構文の例: epmautomate uploadFile "C:/uploads/*INPUT_FILE.zip*" inbox)を使用して、このファイルを受信ボックスにアップロードします。この ZIP ファイルには、次のファイルが格納されている必要があります:
 - requirement.csv という名前のコース・ケース CSV ファイル。この CSV ファイルの各行は、実行する操作のタイプ、アーティファクト名、同時ユーザー数、操作の詳細を指定した入力ファイル、および各コース・ケースに関連する追加情報を識別します。[requirement.csv ファイルの作成](#)を参照してください。
 - 各操作の詳細が格納されている入力ファイル。次のトピックを参照してください。
 - * [フォーム入力ファイルを開く](#)
 - * [フォーム入力ファイルの保存](#)
 - * [ビジネス・ルール入力ファイルの実行](#)
 - * [ビジネス・ルールセット入力ファイルの実行](#)

- * データ・ルール入力ファイルの実行
 - * アド・ホック・グリッド入力ファイル
 - * ブック入力ファイルの実行
 - * レポート入力ファイルの実行
- オプションで、userVarMemberMapping.csv を入力 ZIP ファイルに追加して、ユーザー変数メンバー・マッピングを提供します。[UserVarMemberMapping.csv ファイルの作成](#)を参照してください。
 - オプションで、options.xml を入力 ZIP ファイルに追加して、Smart View オプションを使用します。[options.xml ファイルの作成](#)を参照してください。
 - オプションで、users.csv を入力 ZIP ファイルに追加して、既存のユーザーのユーザー名とパスワードを提供します。[users.csv ファイルの作成](#)を参照してください。
- iterations は、応答時間を測定するために、requirement.csv で識別された各ユース・ケースの実行回数を示す正数です。指定されていない場合、操作は 1 回のみ実行されます。
 - notificationEmails はオプションで、このコマンドの結果を電子メールする送信先電子メール・アドレスを示します。複数の電子メール・アドレスを指定する場合は、セミコロンを使用してアドレスを区切ります。また、アドレスのリストは二重引用符で囲みます。指定されていない場合、結果はコマンドを開始したユーザーに電子メールで送信されます。このレポートの詳細は、[同時使用のシミュレート・レポートのサンプル](#)を参照してください。
 - [testMode]はオプションで、同時使用シミュレーション・モードを指定します。デフォルトは 0 です。使用可能な値は次のとおりです。
 - 0: デフォルトのシミュレーション・モードでは、シミュレート対象ユーザーを環境に追加し、サービス管理者の役割を割り当て、シミュレーションを実行してから、シミュレート対象ユーザーを削除します。このモードはテストを 1 回のみ実行する場合に便利です。
シミュレート対象ユーザーには次のプロパティがあります:
名: testuser1、testuser2 など
姓: testuser1、testuser2 など。
電子メール・アドレス: testuser1@discard.oracle.com、testuser2@discard.oracle.com など
ユーザー名: testuser1、testuser2 など
 - 1: シミュレート対象ユーザーを環境に追加し、サービス管理者の役割を割り当てます。シミュレーションの実行やシミュレート対象ユーザーの削除は実施しません。このモードを使用した後、モード 3 でコマンドを実行し、必要な回数のシミュレーションを実行します。最後に、モード 2 でコマンドを実行し、シミュレート対象ユーザーを削除します。
 - 2: シミュレート対象ユーザーを削除します。ユーザーの作成やシミュレーションの実行は実施しません。
 - 3: ユーザーの追加や削除を実施せずに、既存のシミュレート対象ユーザーを使用してシミュレーションを実行します。
 - 4: 入力 ZIP ファイルに格納された users.csv ファイルに定義されたユーザーを使用してコマンドを実行します。[users.csv ファイルの作成](#)を参照してください。このモードでは、シミュレーション用のユーザーは作成されません。かわりに、既存のユーザーが使用されます。

同時使用を 1 回のみ実行する場合は、testMode=0 を使用します。一連のテストを実行するには:

- 最初に、testMode=1 を使用してコマンドを実行し、シミュレート対象ユーザーを追加し、サービス管理者の役割を割り当てます。
 - 次に、testMode=3 を使用してコマンドを実行し、必要な回数のシミュレーションを実行します。
 - 最後に、testMode=2 を使用してコマンドを実行し、シミュレート対象ユーザーを削除します。
- [lagTime]はオプションで、requirement.csv の各コース・ケースの実行の合間に、コマンドが待機する秒数(5 秒以上)を指定します。デフォルトは 5 秒です。負数(-1 など)、分数(1/2 など)および小数値は使用しないでください。
1 人のユーザーによる requirement.csv のコース・ケースの実行が開始した後、コマンドは、このパラメータに指定した秒数が経過するまで実行を待機してから、次のユーザーによるコース・ケースの実行を開始します。通常、ユーザー・アクティビティは同時に開始されないため、このパラメータを設定すると、環境への負荷のより現実的なシミュレーションを作成しやすくなります。

例

```
epmautomate simulateConcurrentUsage test_simulation.zip iterations=5
notificationEmails="jane.doe@example.com;john.doe@example.com;example@example.com" lagTime=6
```


skipUpdate

環境への月次更新の適用をスキップ(最大で連続する 3 サイクル分)するようオラクルに要求するか、またはこのコマンドを使用して以前に作成された、更新のスキップ要求をすべて削除し、環境をメインのコード・ラインに更新できるようにします。

また、このコマンドを使用して、環境に対して現在指定されている、更新のスキップ要求をリストできます。環境の更新のスキップ・ステータスは、このコマンドを使用して環境に対する更新をスキップした後に生成されるアクティビティ・レポート(操作メトリック内)に含まれません。*管理者用スタート・ガイド*の操作メトリックを参照してください。

当月の週次および緊急パッチがあれば、引き続き環境に適用されます。アップグレードの遅延が要求された月の更新は行われません。

1 回限りのパッチが適用されている環境の更新をスキップすることはできません。また、環境に現在適用されている更新からの間隔が 3 か月を超える月次更新はスキップできません。たとえば、環境に現在 23.12 が適用されている場合、24.01、24.02 および 24.03 はスキップできませんが、24.04 はスキップできません。更新延期の仕組みの詳細は、*オペレーション・ガイド*の本番環境のアップグレード延期のリクエストを参照してください。

 ノート:

環境のうちの1つのみの更新を3か月間スキップした場合(たとえば、実稼働環境では更新をスキップし、テスト環境では更新をスキップしなかった場合)、それらの環境には3つのバージョンの差が生じます。このようなシナリオでは、これらの環境間でスナップショットを移行できない可能性があります。

たとえば、テスト環境と実稼働環境が現在バージョン 23.12 であり、実稼働環境でのみバージョン 24.01、24.02 および 24.03 の更新をスキップするとします。バージョン 24.03 が使用可能になると、テスト環境はバージョン 24.03 になりますが、実稼働環境は引き続きバージョン 23.12 のままです。この場合、テスト環境と本番環境の間の移行はサポートされません。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmautomate skipUpdate add|remove|list [version=UPDATE_NUMBER comment="COMMENT"]`。
ここで:

- `add` は、特定の月次更新について更新のスキップ要求を設定します。次のパラメータを指定する必要があります。
 - `version`: スキップする月次更新。今後の3回の月次更新のうち、1つ、2つまたは3つをスキップできます。たとえば、環境に 23.12 の月次更新が適用されている場合、24.01、24.02、24.03、またはこれらの複数の更新をスキップできます。3回の月次更新をスキップするには、コマンドを3回実行します。たとえば、`version=24.01`、次に `version=24.02`、さらに `version=24.03` を使用して、毎回スキップする特定の更新を1つ指定します。このシナリオの環境は、24.04 の月次サイクルでメインのコード・ラインに更新されます。
更新のスキップが要求された月次サイクルと現在の月次サイクルにギャップがある場合は、必要に応じて環境が更新され、その後、指定した月次サイクルの更新がスキップされます。たとえば、23.12 の月次更新が適用されている環境で、バージョン 24.02 および 24.03 の更新をスキップするよう指定したとします。この場合、環境は 24.01 に更新され、24.02 および 24.03 の更新がスキップされます。環境は、24.04 でメインのコード・ラインに更新されます。
 - `comment`: 更新のスキップが必要な理由を示すテキスト。コメントは二重引用符で囲む必要があります。
- `remove` は、環境に指定された更新のスキップ要求をすべて削除して、次の日次メンテナンス中にメインのコード・ラインに更新できるようにします。環境に複数の更新スキップ要求がある場合、このコマンドはそれらすべてを削除します。

- list は、次の図に示すように、環境に現在設定されている更新のスキップ要求(更新のスキップを要求したユーザーのログイン ID、コメント、更新をスキップするバージョン、および要求が行われた日付)を表示します。

```
c:\Oracle\EPM Automate\bin>epmautomate skipupdate add version=24.01 comment="Some Comment"
skipupdate completed successfully

c:\Oracle\EPM Automate\bin>epmautomate skipupdate add version=24.02 comment="Some Comment"
skipupdate completed successfully

c:\Oracle\EPM Automate\bin>epmautomate skipupdate add version=24.03 comment="Some Comment"
skipupdate completed successfully

c:\Oracle\EPM Automate\bin>epmautomate skipupdate list
skipupdate completed successfully

1] User: example@example.com | Version: 24.03 | Comments: Some Comment | Timestamp: 2023-11-15T19:17:09Z
2] User: example@example.com | Version: 24.02 | Comments: Some Comment | Timestamp: 2023-11-15T19:17:01Z
3] User: example@example.com | Version: 24.01 | Comments: Some Comment | Timestamp: 2023-11-15T19:16:51Z

c:\Oracle\EPM Automate\bin>epmautomate skipupdate remove
skipupdate completed successfully
```

例

- 更新のスキップの要求: `epmautomate skipUpdate add version=24.01 comment="We are in the process of closing the quarter"`
- 更新のスキップの詳細の表示: `epmautomate skipUpdate list`
- 更新のスキップ要求をすべて削除: `epmautomate skipUpdate remove`

snapshotCompareReport

2つのスナップショットを比較し、スナップショットに含まれる計算ルールとルールセットおよびデータ・フォームの差異を識別するスナップショット比較レポートを作成します。

レポートには、ルールおよびフォームの定義の差異が表示されます。また、一方のスナップショットに、もう一方のスナップショットに存在するルール、ルールセットまたはフォームが含まれているかどうかを識別します。ルールセットの差異は、ルールセットに含まれるルールに基づいて識別されます。生成されるレポートには、キューブまたはディメンションのプロパティの差異はレポートされないことに注意してください。

- 環境の最近のパフォーマンスの低下。以前のスナップショットと現在のスナップショットを比較して、パフォーマンスの低下を引き起こした可能性のある差異を確認できます。
- 機能の動作またはパフォーマンスが同一であると予想される2つの環境間で、動作またはパフォーマンスに差異があります。この場合、2つの環境のスナップショットを比較して、これらの差異を理解できます。
- いくつかのルールまたはフォームが環境から消失した疑いがあります。このレポートを使用して、以前存在していたアーティファクトと現在のアーティファクトを比較します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Strategic Workforce Planning および Sales Planning。

必要な役割

サービス管理者

使用方法

`epmAutomate snapshotCompareReport SOURCE_SNAPSHOT TARGET_SNAPSHOT [reportName=REPORT_NAME.html]`。ここで:

- `SOURCE_SNAPSHOT` は、比較対象のスナップショットの名前です。レポートには、このスナップショットのルールおよびフォームの差異に関するデータが含まれます。
- `TARGET_SNAPSHOT` は、比較するスナップショットの名前です。

Note:

- スナップショット名は、.ZIP 拡張子の有無にかかわらず指定できます。
- 環境に両方のスナップショットが存在している必要があります。
[uploadFile](#)、[copyFromObjectStorage](#) または [copySnapshotFromInstance](#) コマンドを使用して、それらを環境にアップロードします。

- `REPORT_NAME` (オプション)は、レポート・ファイルの名前です。デフォルトのレポート名は `SnapshotCompare.html` です。
[downloadFile](#) コマンドを使用して、レポートをダウンロードします。

例

- `epmAutomate snapshotCompareReport "Artifact Snapshot" Backup_22-09-08.zip reportName=Snapshot_Diffs.html`
- `epmAutomate snapshotCompareReport backup_snapshot_22-Aug-08.zip backup_Snapshot_22-Sep-08.zip reportName=Sep_22_snapshot_compare_report.html`

sortMember

エンティティ、勘定科目、シナリオ、バージョンのディメンションのメンバー、およびカスタム・ディメンションのメンバーをソートします。

このコマンドは、メンバーをアプリケーションへロードした後にディメンション・メンバーをソートするのに便利です。

ノート:

このコマンドを使用して期間、年および通貨のディメンションのメンバーをソートすることはできません。

適用対象

Planning、Planning モジュール、フリーフォーム、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning。

必要な役割

サービス管理者

使用方法

`epmautomate sortMember Member [type=children|descendants] [order=ascending|descending]`。ここで:

- `Member` は、子孫または子をソートする親メンバーの名前です。
- `type` は、必要に応じて、ソートするメンバーを指定します。使用可能な値は次のとおりです。
 - `descendants` は、`Member` の値として指定する親メンバーのサブメンバー(子および子孫)をすべてソートします
 - `children` (デフォルト値)は、`Member` の値として指定する親メンバーの直下にあるレベルのメンバーのみをソートします。
- `order` は、必要に応じて、ソート順を識別します。使用可能な値は次のとおりです。
 - `ascending`: これがデフォルトのソート順です。
 - `descending`

例

- エンティティ・ディメンションの子を昇順でソートします: `epmautomate sortMember Entity`
- エンティティ・ディメンションのサブメンバーをすべて降順でソートします: `epmautomate sortMember Entity type=descendants order=descending`

unassignRole

このコマンドで使用される ANSI または UTF-8 でエンコードされた CSV ファイルにログイン ID が含まれている、このコマンドを実行するユーザーを含む、現在ユーザーに割り当てられた役割を削除します。このコマンドを使用して、事前定義済役割またはアプリケーション役割の割当てを削除できます。

CSV ファイルの形式は次のとおりです。

```
User Login
jane.doe@example.com
jdoe
```

サービス管理者は、[uploadFile](#) コマンドを使用して、ファイルを環境にアップロードします。

ノート:

空白文字を含む役割名は二重引用符で囲みます。

完了すると、失敗した各エントリに関する情報がコンソールに出力されます。この情報を確認して、CSV ファイルの一部のエントリでコマンドの実行が失敗した理由を理解してください。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- 事前定義済役割割当てを削除するには:
 - サービス管理者または任意の事前定義済役割
 - アイデンティティ・ドメイン管理者と任意の事前定義済役割
- アプリケーション役割を削除する場合:
 - サービス管理者
 - 任意の事前定義済役割およびアクセス制御 - 管理アプリケーション役割

使用方法

epmautomate unassignRole *FILE_NAME* *ROLE*。ここで:

- *FILE_NAME* は、役割の割当てが取り消されるユーザーのログイン ID を含む CSV ファイルの名前です CSV 拡張子は小文字で指定してください。
ユーザー・ログイン値では大文字と小文字が区別されません。たとえば、jane.doe@example.com は、Jane.Doe@Example.com など、大文字と小文字のすべてのバリエーションと同じであるものとして処理されます。
- *ROLE* は次のいずれかを識別します役割名では大文字と小文字が区別されません。
 - 事前定義済役割へのユーザーの割当てを削除する場合、*ROLE* はサービスに適用可能な事前定義済役割を識別する必要があります。 *管理者用スタート・ガイド*の事前定義済役割の理解を参照してください。
 - アプリケーション役割へのユーザーの割当てを削除する場合、*ROLE* では、現在の環境のアプリケーションに属する役割を指定する必要があります。アプリケーション役割は、「アクセス制御」の「**役割**」タブにリストされます。各ビジネス・プロセスのアプリケーション役割の説明については、*Oracle Enterprise Performance Management Cloud* *アクセス制御の管理*の次のトピックを参照してください:
 - * Account Reconciliation
 - * Enterprise Profitability and Cost Management
 - * Planning、フリーフォーム、Financial Consolidation and Close および Tax Reporting
 - * Profitability and Cost Management
 - * Oracle Enterprise Data Management
 - * Narrative Reporting

例

- 事前定義済アイデンティティ・ドメイン役割からユーザーの割当てを解除します。
epmautomate unassignRole remove_roles.csv "Service Administrator"
- アプリケーション役割からユーザーの割当てを解除します。

```
epmautomate unassignRole example_file.csv "Task List Access Manager"
```

updateGuidedLearningSettings

環境で Oracle Guided Learning (OGL)を有効にするために必要な値を設定します。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Enterprise Profitability and Cost Management、Sales Planning および Strategic Workforce Planning

必要な役割

サービス管理者

使用方法

```
epmautomate updateGuidedLearningSettings [oglAppId=APP_ID]  
[oglServerUrl=OGL_SERVER_URL]。ここで:
```

- OglAppId はオプションで、**OGL アプリケーション ID** です。これは、**OGL サーバーのガイド (コンテンツ)の論理グループの ID** です。この値には最大 **100** 文字まで使用できます。
- OglServerUrl はオプションで、アクティブな **OGL アカウントがある OGL サーバーの URL** です。デフォルトは `https://guidedlearning.oracle.com` です。この URL にはプロトコルを指定する必要があり、最大 **300** 文字まで使用できます。

Note:

前述のパラメータの少なくとも 1 つは必須です。

例

```
epmautomate updateGuidedLearnignSettings "oglAppId=123xyz" "oglServerUrl=https://  
guidedlearning.oracle.com"
```

updateUsers

環境にアップロードされた ANSI または UTF-8 でエンコードされたカンマ区切り値(CSV)ファイルで識別される新しい値を使用して、アイデンティティ・ドメインの Oracle Fusion Cloud Enterprise Performance Management ユーザーの電子メール、名、姓などの属性を変更します。

サービス管理者は、[uploadFile](#) コマンドを使用して、ファイルを環境にアップロードします。CSV ファイルのすべての列は必須です。各列に有効なエントリを入力する必要があります。このコマンドは、これらの必須値の定義を検証し、欠落している値または無効な値を特定するエラー・メッセージを表示します。入力ファイルの形式は次のとおりです。

```
First Name,Last Name,Email,User Login  
Jane,Doe,jane.doe@example.com,jdoe  
John,Doe,john.doe@example.com,john.doe@example.com
```

CSV ファイルのユーザー・ログイン値がアイデンティティ・ドメインに存在するアカウントと一致する場合、コマンドは入力ファイルの値と一致するようにユーザー・アカウントを変更し

ます。ユーザー・アカウントはアイデンティティ・ドメインがサポートするすべての環境に共通するため、更新されたユーザー情報はアイデンティティ・ドメインを共有するすべての環境で使用できます。ユーザーに割り当てられた事前定義済アプリケーション固有の役割は、このコマンドの影響を受けません

 **Note:**

- このコマンドを使用してユーザー・ログイン値を変更することはできません。
- 独自のアカウント属性を変更することは許可されていません。
- マルチバイト文字を含む入力ファイルでは、**UTF-8** 文字エンコードを使用する必要があります。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Strategic Workforce Planning および Sales Planning。

必要な役割

アイデンティティ・ドメイン管理者と任意の事前定義済役割

使用方法

`epmautomate updateUsers FILE_NAME`。ここで、`FILE_NAME` は変更するユーザー情報を含む CSV ファイルの名前です。


例

```
epmautomate updateUsers update_user_info.csv
```

upgrade

最新バージョンの EPM 自動化を自動的にダウンロードし、サイレント・インストールします。

`login` コマンドを実行してセッションを開始すると、現在インストールされているバージョンが EPM 自動化によって識別されます。インストールされているバージョンが利用可能な最新バージョンでない場合は、より新しいバージョンが利用できるというメッセージが表示されます。

 **ノート:**

Windows 管理者によってデプロイされた EPM 自動化は、ログインしているユーザーが Windows 管理者である場合にのみアップグレードできます。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability

and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザー
- ユーザー
- 参照者

使用方法

```
epmautomate upgrade
```

例

```
epmautomate upgrade
```

uploadFile

ローカル・コンピュータからサービスにファイルをアップロードします。このコマンドは、データ、メタデータ、ルール定義、ディメンション定義、マップ済トランザクション、テンプレートおよびバックアップ・スナップショットが含まれるファイルをアップロードする場合に使用します。

このコマンドは、環境内の既存のファイルを上書きしません。アップロードされるファイルの名前がアップロード場所のファイルの名前と同じである場合は、エラーが表示されます。

このコマンドでは、バイナリ・ファイル(たとえば、拡張子が.exe、.com、.cmd、.bin、.bat、.msi、.vbs などのファイル)および許可されていない拡張子のファイルはアップロードできません。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- パワー・ユーザーおよび移行 - 管理アプリケーション役割(Oracle Enterprise Data Management Cloud および Profitability and Cost Management)

使用方法

```
epmautomate uploadFile "FILE_NAME" [UPLOAD_LOCATION]。ここで:
```

- *FILE_NAME* はファイルの名前です。EPM 自動化を実行しているディレクトリにファイルがない場合は絶対パスを指定します。
- *UPLOAD_LOCATION* はオプションで、ファイルをアップロードする Oracle Fusion Cloud Enterprise Performance Management の場所です。デフォルトのアップロード場所にファイルをアップロードする場合は、アップロード場所を指定しないでください。詳細は、[デ](#)

フォルトのファイルの場所を参照してください。サポートされている値は、次のとおりです:

- `inbox` は、受信ボックスにファイルをアップロードします。**Profitability and Cost Management** 以外の **Cloud EPM** ビジネス・プロセスは、処理するファイルをこの場所で見つけます。
- `profitinbox` は、**Profitability and Cost Management** で処理するファイルをアップロードします。
- `to_be_imported` は、環境の次の日次メンテナンスでインポートする **Narrative Reporting** スナップショットをアップロードします。
- `inbox/directory_name` は、データ管理で処理するために受信ボックス内のディレクトリにファイルをアップロードします。
- `outbox` は、**Profitability and Cost Management** 以外のビジネス・プロセスによって使用される送信ボックスにファイルをアップロードします。
- `profitoutbox` は、**Profitability and Cost Management** によって使用される送信ボックスにファイルをアップロードします。

例

- スナップショットをデフォルトの場所にアップロードします:
`epmautomate uploadFile "C:/snapshots/backup_snapshot.zip"`
- ファイルをデータ管理の受信ボックスにアップロードします:
`epmautomate uploadFile "C:/pbcsdata/quarterlydata.csv" inbox`
- 受信ボックス(データ管理)のフォルダにファイルをアップロードします:
`epmautomate uploadFile "C:/fdmee_data/data.zip" inbox/dm_folder`
- **profitinbox (Profitability and Cost Management)** にファイルをアップロードします:
`epmautomate uploadFile "C:/profitability_data/data.zip" profitinbox`
- **Narrative Reporting** スナップショットを `C:\temp` ディレクトリから `to_be_imported` の場所にアップロードします:
`epmautomate uploadFile "C:\temp\EPRCS_Backup.tar.gz" to_be_imported`

userAuditReport

ユーザー監査レポート(.CSV ファイル)を生成し、デフォルトのダウンロード場所に格納します。

ユーザー監査レポートには、指定期間にわたって環境にサインインしたユーザーに関する情報が含まれています(最長過去 120 日間)。ユーザー・ログイン ID、ユーザーのログイン元となるコンピュータの IP アドレス、およびユーザーが環境にアクセスした日時(July 28, 2022 18:43:21 UTC など)がリストされます。

ノート:

ユーザー監査レポートには、**Oracle Fusion Cloud Enterprise Performance Management** 環境に 5 分以内に複数回ログインしたユーザーのログイン・エントリが 1 つのみリストされます。

	A	B	C
1	Type Of Report	User Audit Report	
2	Report Generated Date	6/1/2020	
3			
4	User Name	IP Address	Access Date and Time
5			
6	jdoe@example.com	111.22.23.6	June 1, 2020 22:28:00 UTC
7	jane.doe@example.com	111.22.23.6	June 1, 2020 21:40:56 UTC

[downloadFile](#) を使用して、生成されたレポートをコンピュータにダウンロードします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- 任意の事前定義済役割およびアクセス制御 - 管理アプリケーション役割
- 任意の事前定義済役割およびアクセス制御 - 表示アプリケーション役割

使用方法

`epmautomate userAuditReport FROM_DATE TO_DATE REPORT_NAME`。ここで:

- `FROM_DATE` は、監査レポートが生成される期間の開始日(YYYY-MM-DD 形式)を示します
- `TO_DATE` は、監査レポートが生成される期間の終了日(YYYY-MM-DD 形式)を示します
- `REPORT_NAME` は、レポート・ファイルの名前です

ノート:

このレポートは、過去 120 日間についてのみ生成できます。

例

```
epmautomate userAuditReport 2016-10-15 2016-12-15 myAuditReport.CSV
```

userGroupReport

ユーザーが割り当てられているアクセス制御のグループが記載されたレポート(CSV ファイル)を生成し、デフォルトのダウンロード場所に格納します。

レポートには、グループへのユーザー割当てが直接(グループのメンバーとして)か、間接(ネストされたグループの子であるグループのメンバーとして)かが示されます。

レポートでは、ユーザーのログイン名、名、姓、電子メール・アドレス、割り当てられたグループおよび割当てのタイプが次のフォーマットで識別されます。これは、アクセス制御「ユーザー・グループ・レポート」タブから作成されるレポートの CSV バージョンと同一です。たと

例えば、jdoe は、ネストされたグループ Test2 の子であるグループ Test1 のメンバーであるとして、このシナリオでは、jdoe に関する次の情報がレポートに表示されます：

```
User Login,First Name,Last Name,Email,Direct,Group
jdoe, John, Doe, jdoe@example.com, Yes, test1
jdoe, John, Doe, jdoe@example.com, No, test2
```

[downloadFile](#) を使用して、生成されたレポートをコンピュータにダウンロードします。

適用対象

Planning、Planning モジュール、フリーフォーム、Financial Consolidation and Close、Tax Reporting、Account Reconciliation、Profitability and Cost Management、Enterprise Profitability and Cost Management、Oracle Fusion Cloud Enterprise Data Management、Narrative Reporting、Sales Planning および Strategic Workforce Planning。

必要な役割

- サービス管理者
- 任意の事前定義済役割およびアクセス制御 - 管理アプリケーション役割
- 任意の事前定義済役割およびアクセス制御 - 表示アプリケーション役割

使用方法

`epmautomate userGroupReport REPORT_NAME`。ここで、`REPORT_NAME` はレポート・ファイルの名前です。

例

```
epmautomate userGroupReport UsgGrpReport.CSV
```

validateConsolidationMetadata

環境のメタデータを検証して、データベース・リフレッシュおよび連結にエラーがないことを確認します。

[importMetadata](#) コマンドを使用してメタデータをインポートした後、このコマンドを実行してメタデータを検証し、[refreshCube](#) コマンドを実行したときにデータベース・リフレッシュにエラーがないことを確認します。連結メタデータが正しくない場合、連結も失敗する可能性があります。

このコマンドは、コマンドが実行されたコンソールに、0 (ゼロ) または検証エラーの件数を表示します。検証エラーは CSV ファイルに書き込まれ、このファイルを使用してメタデータ・エラーを修正できます。[downloadFile](#) コマンドを使用して、結果の CSV ファイルをローカル・サーバーにダウンロードします。

適用対象

Financial Consolidation and Close

必要な役割

サービス管理者

使用方法

`epmautomate validateConsolidationMetadata LOG_FILE_NAME`。ここで、`LOG_FILE_NAME` は、このコマンドで識別されたエラーに関する情報を含むファイルの名前を識別します。

例

```
epmautomate validateConsolidationMetadata validation_error.csv
```

validateModel

Enterprise Profitability and Cost Management モデルを検証し、検証出力をファイルに書き込みます。

適用対象

Enterprise Profitability and Cost Management

必要な役割

サービス管理者

使用方法

```
epmautomate validateModel "modelName" FILE_NAME.txt [messageType=All|Warning|Error] [ruleStatus=All|Enabled|Disabled]。ここで:
```

- `modelName` は、検証する **Enterprise Profitability and Cost Management** モデルの名前です。この値は、二重引用符で囲まれている必要があります。
- `FILE_NAME` は、コマンドによってモデル検証出力が書き込まれるテキスト・ファイルの一意の名前です。このファイルは送信ボックスに作成され、[downloadFile](#) コマンドを使用してダウンロードできます。
- `messageType` (オプション)は、モデル検証出力に含まれる情報のステータスです。使用可能なパラメータ値は次のとおりです:
 - `All` は、エラーと警告の両方を検証出力ファイルに書き込みます。
 - `Error` は、エラーのみを検証出力ファイルに記録します。これがデフォルト値です。
 - `Warning` は、モデル検証の警告のみを検証出力ファイルに記録します。
- `ruleStatus` は、オプションで、次のいずれかを使用します:
 - `All` は、選択したモデルですべてのルールを検証します。これがデフォルト値です。
 - `Enabled` は、使用可能なルールのみを検証します。
 - `Disabled` は、使用不可なルールのみを検証します。

例

```
epmautomate validateModel "10 Actuals Allocation Process" results.txt  
messageType=All ruleStatus=Enabled
```

終了コード

EPM 自動化では、操作のステータスを示す終了コードとメッセージを返します。終了コードは5つのコード番号に分類され、各コードが多数のエラー条件を示すことがあります。付属するメッセージを確認して、エラーを引き起こした固有の条件を特定します。

また、コマンドの実行が失敗するたびに、EPM 自動化によってログ・ファイル (`COMMANDNAME_TIMESTAMP.log`、たとえば `uploadfile_16_11_2016_11_27_10.log`) が作成されます。ログ・ファイルは EPM 自動化を実行したコンピュータ上に作成されます。

終了コード1のエラー

コマンドが実行できませんでした EPM 自動化はこの終了コードを使用して、HTTP ステータス・コード 200 および 400 に関連するメッセージを示します。これらのコードは、EPM 自動化が使用する REST API によって返されます。

操作を実行するための権限が不十分です このエラーは、サービスへのサインインに使用された資格証明のユーザーに、試行した操作を実行するために十分な権限がない場合に表示されます。

操作を実行するための十分な権限があるアカウントを使用してサインインしてください。通常、サービスで操作を実行できるのはサービス管理者のみです。

リソースが存在しません このエラーが表示されるのは、削除またはダウンロードしようとするファイルまたはスナップショットがサービスに存在していない場合です。

`listfiles` コマンドを使用して、ファイル名と場所を確認します。

無効なスナップショット *SNAPSHOT* このエラーが表示されるのは、エクスポートまたはインポート操作で指定したスナップショットをサービスが検証できない場合です。

有効なスナップショットを使用していることを確認します。

内部サーバー・エラー。ファイルを削除できません: *FILE_NAME*。詳細を含めて「フィードバックの提供」を発行してください このエラーが表示されるのは、サーバー・エラーのためにファイルまたはスナップショットをサービスから削除できなかった場合です。

この問題は、`Feedback` コマンドまたは「フィードバックの提供」機能を使用してオラクル社に報告してください。

無効なファイル: *FILE_NAME* このエラーが表示されるのは、削除またはダウンロードしようとしているファイルまたはスナップショットがサービスに存在していない場合、またはファイル名が必要な形式でない場合です。

`listfiles` コマンドを使用して、ファイル名と場所を確認します。

再作成が長い間実行しています。サポートに連絡してください このエラーが表示されるのは、開始した再作成操作が1時間以内に完了しない場合です。

この問題は、`feedback` コマンドまたは「フィードバックの提供」機能を使用してオラクル社に報告してください。

リセット・サービスが長い間実行しています。サポートに連絡してください このエラーが表示されるのは、開始したリセット・サービス操作が1時間以内に完了しない場合です。

この問題は、`feedback` コマンドまたは「フィードバックの提供」機能を使用してオラクル社に報告してください。

操作を実行できません。別のインスタンスが進行中です。しばらくしてから再試行してください このエラーが表示されるのは、`copysnapshotfrominstance` コマンドを実行しようとしたときに、同じコマンドの別のインスタンスがアクティブになっている 場合です。

`copysnapshotfrominstance` コマンドが完了するまで待ってから、コマンドを再び実行してみてください。

操作を実行できません。別のメンテナンス・スクリプトが進行中です。しばらくしてから再試行してください このエラーが表示されるのは、`copysnapshotfrominstance`、`recreate` または `resetservice` コマンドを実行しようとしたときに、日次メンテナンスまたはサービス・リセット・プロセスが実行している 場合です。

メンテナンスまたはリセット・プロセスが完了してから操作を再実行します。

ソース・インスタンスにログインできません: `SOURCE_URL` このエラーが表示されるのは、EPM 自動化が `copysnapshotfrominstance` コマンドを開始するためにソース環境にサインインできない 場合です

ソース環境へのアクセスに使用される資格証明、アイデンティティ・ドメインおよび URL が有効であることを確認します。

内部サーバー・エラー。ソース・インスタンスからスナップショットをコピーできません。詳細を含めて「フィードバックの提供」を発行してください このエラーが表示されるのは、EPM 自動化が `copysnapshotfrominstance` コマンドを実行中に予期しない問題が発生した場合 です。

この問題は、`feedback` コマンドまたは「フィードバックの提供」機能を使用してオラクル社に報告してください。

内部サーバー・エラー。詳細を含めて「フィードバックの提供」を発行してください このエラーは、内部サーバー条件が多数あり、オラクル社による修正処理が必要であることを示します。

この問題は、`feedback` コマンドまたは「フィードバックの提供」機能を使用してオラクル社に報告してください。

スナップショット `SNAPSHOT_NAME` はすでに存在します。スナップショットを削除してコマンドを再実行してください このエラーが表示されるのは、スナップショットのダウンロードまたはアップロードを行う場所に、同じ名前の別のスナップショットが存在している 場合です。

既存のスナップショットを削除または移動してからコマンドを再試行します。

スナップショットの抽出でエラーが発生しました。正しいスナップショットを使用して再試行してください このエラーが表示されるのは、`importsnapshot` コマンドの実行時に EPM 自動化がスナップショットの内容を抽出できない 場合です。

スナップショットが有効であることを確認してください。

内部サーバー・エラー。ファイルを書込み用に開くことができません。詳細を含めて「フィードバックの提供」を発行してください このエラーが表示されるのは、エラーによって CSV ファイルが作成または更新される場合です(たとえば監査レポートの生成時)。

この問題は、`feedback` コマンドまたは「フィードバックの提供」機能を使用してオラクル社に報告してください。

一致するレコードが見つかりませんでした。他の日付範囲を選択してください このエラーが表示されるのは、`userauditreport` コマンドを実行して監査レポートを生成するときに日付範囲に監査データがない 場合です。

有効な日付範囲を指定してから `userauditreport` コマンドを再実行します。サービスでは過去 365 日の監査履歴のみが保持されることに注意してください。

同じ名前のファイルが存在しています: `FILE_NAME`。別のファイル名を選択してください このエラーが表示されるのは、指定したのと同じ監査レポート名のレポートがサービスに存在している場合です。

サービスから既存のファイルを削除するか、別のファイル名を指定してから、`userauditreport` コマンドを再実行します。

操作がステータス\$1 で失敗しました。「フィードバックの提供」を発行してください このメッセージは、リセット・サービスまたは再作成サービス・プロセスの失敗を引き起こす内部サーバー・エラーを示します。

この問題は、`feedback` コマンドまたは「フィードバックの提供」機能を使用してオラクル社に報告してください。

EPMCSS-20643: ユーザーの追加に失敗しました。ファイル `FILE_NAME.csv` が見つかりません。有効なファイル名を指定してください このエラーが表示されるのは、追加するユーザーに関する情報が含まれている指定した CSV ファイルが受信ボックスにない場合です。

`listfiles` コマンドを使用して、ファイル名と場所を確認します。ファイルが受信ボックスにない場合は、`uploadFile` コマンドを使用してファイルをアップロードします。

EPMCSS-20644: ユーザーの削除に失敗しました。ファイル `FILE_NAME.csv` が見つかりません。有効なファイル名を指定してください このエラーが表示されるのは、削除するユーザーに関する情報が含まれている指定した CSV ファイルが受信ボックスにない場合です。

`listfiles` コマンドを使用して、ファイル名と場所を確認します。ファイルが受信ボックスにない場合は、`uploadFile` コマンドを使用してファイルをアップロードします。

20645: ユーザーへの役割の割当に失敗しました。役割名 `role` が無効です。有効な役割名を指定してください このエラーが表示されるのは、CSV ファイルに指定された役割がサポートされていない場合です。

ファイルに指定された役割名が、Service Administrator、Power User、User または Viewer であることを確認します。

`listfiles` コマンドを使用して、ファイル名と場所を確認します。ファイルが受信ボックスにない場合は、`uploadFile` コマンドを使用してファイルをアップロードします。

終了コード 6 のエラー

サービスを使用できません HTTP Error 404 のためにサービスが使用できません。

EPM 自動化を実行しているコンピュータ上のブラウザからサービスにアクセスして、サービスを使用できるかどうかを確認します。なんらかの理由でサービスが停止している場合は、しばらく待ってから再試行するか Oracle サポートに問い合わせてください。

読取り/書込みタイムアウト このエラーが表示されるのは、読取り/書込み操作中に、ネットワーク速度の低下またはファイアウォールの問題のためにクライアント・ソケットがタイムアウトした場合です(ソケットのタイムアウトは 15 分)。

ネットワークのスループットが高いときに、失敗したコマンドを再実行します。ファイアウォールの設定が失敗の原因である場合は、ネットワーク管理者に連絡してください。

終了コード7のエラー

EPM 自動化によってこのエラーが表示されるのは、コマンドを実行できない場合です。エラー・メッセージ(無効なコマンドなど)によって、エラーが発生した理由が示されます。

パスワード・ファイル *FILE_NAME* を開くことができません 暗号化されたパスワード・ファイル(*PASSWORD_FILE.EPW* など)が無効です。指定した場所でファイルが EPM 自動化で見つからなかったか、ファイルが必要な形式ではありません。

ファイル名とパスを確認してください。形式が無効なためにファイルを解析できない場合は、[encrypt](#) コマンドを使用してファイルを再作成します。

パスワード・ファイル *FILE_NAME* を解析できません 形式が無効なため、またはファイルが壊れているため、暗号化されたパスワード・ファイルを EPM 自動化で解析できませんでした。

[encrypt](#) コマンドを使用して、ファイルを再作成します。

URL に接続できません。根本的な原因: *MESSAGE* このエラーが表示されるのは、不正な URL が原因で接続を確立できない場合です。根本原因として表示されるメッセージに、正しくない URL を使用したために引き起こされた問題の詳細が示されます。

- 有効な URL を使用していることを確認します
- プロキシ設定で、インターネットに接続するときプロキシ・サーバーによる認証が必要とされている場合は、プロキシ・サーバーのユーザー名、ドメイン、およびパスワードを指定(またはプロキシ・サーバー・パスワードを含む暗号化パスワード・ファイルを使用)してサインインします。手助けが必要な場合はネットワーク管理者に連絡します。

URL に接続できません サポートされていないプロトコル 指定した URL で使用されるプロトコルがサポートされていないため、ログイン・コマンドが失敗しました。付随のエラー・メッセージに、サポートされないプロトコルが示されます。

ログイン・コマンドで使用する URL は安全なプロトコル(HTTPS)を使用することを確認してください。

セッションが認証されていません。他のコマンドを実行する前に login コマンドを実行してください login コマンドを使用してセッションを確立する前に、コマンドを実行しようとしてしました。

[login](#) コマンドを実行して、環境に対する保護された接続を確立してから、他のコマンドを実行します。

無効なパラメータ このメッセージは、コマンドのパラメータの順序が正しくないか、必須コマンド・パラメータの値がないために引き起こされた、コマンドのユーザー・エラーを示します。

コマンドのパラメータや指定順序を確認して修正します。

***COMMAND_NAME* コマンドは *SERVICE_TYPE* でサポートされません** ビジネス・プロセスでコマンドがサポートされないため、接続した環境に対して EPM 自動化はコマンドを実行できませんでした。

各ビジネス・プロセスでサポートされるコマンドの一覧については、[コマンド・リファレンス](#)を参照してください。

ファイルが次の場所にありません: *PATH* upload コマンドや replay コマンドなどを使用して処理しようとしているファイルを EPM 自動化が見つけれませんでした。

ファイル名とパスが正確であることを確認してください。

読取り用にファイルを開くことができません: *PATH* 指定したファイルを EPM 自動化が読み取ることができません。

ファイルが必要な形式であることを確認します。EPM 自動化を実行するユーザーにそのファイルへの読取りアクセス権があることを確認します。

書込み用にファイルを開くことができません: : *PATH* 指定したファイルに EPM 自動化が書き込むことができません。

ファイルが他のプロセスでロックされていないことを確認します。EPM 自動化を実行するユーザーにそのファイルへの書込みアクセス権があることを確認します。

無効なコマンド サポートされないコマンドが EPM 自動化で検出されました。

EPM 自動化がコマンドをサポートしていることを確認してください。また、コマンド名のスペリングが正しいことを確認します。

日付の形式が無効です 無効な日付形式が検出されました。

サポートされている日付形式でレポート生成日付を指定します。

FROMDATE DATE が TODATE DATE よりも後になることはできません 開始日より前の日付の終了日が EPM 自動化で検出されました。

指定した日付範囲の to date が from date よりも後の日付であることを確認します。

1 日の最大フィードバック数 (6) を超えました このエラーが表示されるのは、feedback コマンドを使用して送信できるフィードバック数を超えたときです。

同じ名前のファイルがすでにダウンロード・パス *PATH* に存在します。ファイルを削除してコマンドを再実行してください このエラーが表示されるのは、ダウンロードするファイルの名前に一致するファイルがすでに存在する場所にファイルをダウンロードしようとした場合です。

既存のファイルの削除、名前変更または移動を行ってから、コマンドを再実行します。

ファイルが空です。 *PATH* このエラーが表示されるのは、リプレイ・ファイルの中身がない場合です。

リプレイ・ファイル(CSV ファイル)に、資格証明(ユーザー名とパスワード)と、replay コマンドの実行に使用される HAR ファイルの名前が含まれることを確認します。

ローカルホストを解決できないためパスワードを暗号化できません。ホスト名が IP アドレスに正しくマップされていることを確認してください このエラーが表示されるのは、コンピュータ上のホスト・ファイルにアドレス 127.0.0.1 の localhost ではなくサーバー名が含まれているため、EPM 自動化がローカルホスト定義を MAC アドレスに解決できない場合です。

ホスト・ファイルに 127.0.0.1 のサーバー名として localhost が指定されていることを確認します

スナップショット名が無効です このエラーは、名前変更するスナップショットの名前を指定しない場合に表示されます。

環境で使用可能なスナップショットの名前を指定します。

新規スナップショット名が無効です このエラーは、スナップショットの新しい名前を指定しない場合に表示されます。

スナップショットの新しい名前を指定します。

無効なスナップショット名: {0}。文字 \\/*?"<>| は使用できません このエラーは、スナップショット名に、スペース、\ (バックスラッシュ)、/ (スラッシュ)、* (アスタリスク)、? (疑問符)、" (引用符)、< (次より小さい)、> (次より大きい)などの特殊文字が含まれている場合に表示されます。

これらの特殊文字を含まない新しいスナップショット名を指定します。

スナップショットの名前を変更できません: {0}。別のプロセスがアクセスしている可能性があります。しばらくしてから再試行してください このエラーが表示されるのは、スナップショットが別のプロセスで使用されているために、EPM 自動化でそのスナップショットに対する排他ロックを取得できない場合です。

スナップショットを使用している現在の操作が終了するまで待ってから再試行します。

スナップショット {0} はすでに存在します。スナップショットを削除してコマンドを再実行してください このエラーは、新しいスナップショットの名前が環境内の既存のスナップショットの名前と同じ場合に表示されます。

別のスナップショット名を使用するか、`deletefile` コマンドを使用して既存のスナップショットを削除します。

終了コード 9 のエラー

資格証明が無効です このエラーが表示されるのは、`login` コマンドで使用されたユーザー名またはパスワードが正しくないときです。

接続しようとする環境の有効な資格証明を指定します。

コマンドの実行中に認証に失敗しました。再試行してください このエラーが表示されるのは、`login` 以外のコマンドの実行中に Basic 認証が失敗した場合です。このエラーは、コマンドの実行が再試行(最大 3 回)されたときに HTTP コールで発生することもあります。

終了コード 11 のエラー

内部サーバー・エラー。手動リセット・サービスにより、Oracle EPM Cloud Service 環境は現在使用できません。 このエラーが表示されるのは、環境のリセットが進行中のときに EPM 自動化コマンドが実行された場合です。

内部サーバー・エラー: MESSAGE このエラーが表示されるのは、HTTP 接続に関連しない不明な例外が EPM 自動化で検出された場合です。サーバー・エラー 503 および 500 が含まれます。

URL に接続できません: MESSAGE このエラーが表示されるのは、サーバーが使用できないときです。エラー・メッセージに、コマンド失敗の原因となった例外が示されます。

サーバーが使用できない場合は、Oracle サポートに問い合せてください。メッセージに URL の問題が示された場合は、使用する URL が有効であることを確認します。

3

コマンド実行のサンプル・シナリオ

EPM 自動化を使用すると、Oracle Fusion Cloud Enterprise Performance Management の一般的なタスクの多くを自動化できます。

- [すべてのサービスのサンプル・シナリオ](#)
- [Planning、連結、Tax Reporting および Enterprise Profitability and Cost Management 用のサンプル・シナリオ](#)
- [Account Reconciliation のサンプル・シナリオ](#)
- [Profitability and Cost Management のサンプル・シナリオ](#)
- [Oracle Enterprise Data Management Cloud のサンプル・シナリオ](#)

サンプル・スクリプトのコピーについて

このドキュメントの PDF バージョンからサンプル・スクリプトをコピーしないでください。改行とフッター情報によってスクリプトが使用不能にならないように、スクリプトを HTML バージョンの [EPM 自動化の操作](#) からコピーすることをお勧めします。

すべてのサービスのサンプル・シナリオ

これらのシナリオでは、Oracle Fusion Cloud Enterprise Performance Management 環境で特定の操作を実行するために使用できる一般的な一連のコマンドを示します。

次も参照:

- [アプリケーション・スナップショットのコンピュータへのバックアップ](#)
このシナリオでは、日次サービス・メンテナンス中に作成されたスナップショットをローカル・コンピュータにバックアップするプロセスの自動化方法を説明します。
- [ユーザーへの日次メンテナンス完了の通知](#)
Oracle Fusion Cloud Enterprise Performance Management 環境の日次メンテナンスに要する時間は、多くの場合、予定されている 1 時間よりもはるかに短くなります。
- [Oracle Object Storage との間でのスナップショットのコピー](#)
- [ユーザーの作成および事前定義済役割への割当て](#)
この項のスクリプトを使用してユーザーを作成し、アイデンティティ・ドメインに事前定義されている役割にそのユーザーを割り当てます。
- [ライセンスされたユーザー\(役割に割り当てられたユーザー\)の数のカウント](#)
この項のスクリプトを使用して役割割当レポートを生成し、環境のユーザー数をカウントします。
- [役割に割り当てられたユーザーの監査レポートの作成](#)
この項のスクリプトを使用して、環境内の事前定義済役割に割り当てられたユーザーの監査レポートの作成プロセスを自動化し、オプションで受信者にレポートを電子メールで送信します。

- **役割割当ての作成と監査レポートの取消し**
この項の PowerShell スクリプトを使用して、環境内の役割割当ておよび役割取消しの詳細を示す監査レポートの作成プロセスを自動化します。
- **個人情報保護法に準拠するためのアクセス・ログとアクティビティ・レポートのマスク**
この項のスクリプトを使用して、個人情報保護法に準拠するためにアクティビティ・レポートまたはアクセス・ログの情報をマスクするプロセスを自動化し、オプションで受信者にレポートを電子メールで送信します。
- **アクティビティ・レポートのローカル・コンピュータへのダウンロード自動化**
この項のスクリプトを使用して、環境からローカル・コンピュータへのアクティビティ・レポートのダウンロードを自動化します。
- **環境からのアクセス・ログのダウンロード**
この項のスクリプトを使用して、環境からローカル・コンピュータへのアクセス・ログのダウンロード・プロセスを自動化します。
- **環境のクローニングの自動化**
この項のスクリプトを使用して、環境のクローニングを自動化します。
- **プライマリ環境での日次メンテナンス完了後の、プライマリ環境からスタンバイ環境への毎日のクローニング**
スタンバイ環境をプライマリ環境にあわせて最新の状態に保つには、プライマリ環境で日次メンテナンスが完了したらすぐに、次のスクリプトを使用して Oracle Fusion Cloud Enterprise Performance Management プライマリ環境をスタンバイ環境にクローニングします。
- **環境からの不要なファイルの削除**
これらのスクリプトを使用して、環境から不要なファイルを削除します。
- **環境からのファイルの検索およびダウンロード**
この項のサンプル・スクリプトを使用し、テキスト文字列をワイルドカードとして使用して、Oracle Fusion Cloud Enterprise Performance Management 環境から 1 つ以上のファイルをダウンロードするプロセスを自動化します。
- **監査用の古い Cloud EPM 環境の再作成**
この項のスクリプトを使用して、Oracle Fusion Cloud Enterprise Performance Management 環境の最新のスナップショットのライブラリを保持するためのセルフサービス・ソリューションを作成します。最新のスナップショットのライブラリをアップグレードおよび保持する目的に特化した環境が必要です。
- **データベース・アクセスの監査およびコンプライアンスの自動化**
この項では、PowerShell および Bash シェル・スクリプトを使用し、EPM 自動化コマンドを利用して、手動データベース・アクセス全体の監査およびコンプライアンス・データを収集します。
- **ユーザーおよび事前定義済役割割当てのレプリケート**
この項のスクリプトは、環境のユーザーおよび事前定義済役割割当てを別の環境に移行する際に役立ちます。
- **四半期の Cloud EPM アップグレード頻度の作成**
これらのスクリプトを使用して、Cloud EPM 環境が 2 週間のテスト・サイクルを使用して四半期ベースで更新されるように、更新をスキップするセルフサービス・ソリューションを作成します。この場合、本番環境はテスト環境の 2 週間後に更新されます。
- **6 週間のテスト・サイクルを使用した四半期の Cloud EPM アップグレード頻度の作成**
この項のスクリプトを使用して、Oracle Fusion Cloud Enterprise Performance Management 環境が 6 週間のテスト・サイクルを使用して四半期ベースで更新されるように、更新をスキップするセルフサービス・ソリューションを作成します。この場合、本番環境はテスト環境の 6 週間後に更新されます。

アプリケーション・スナップショットのコンピュータへのバックアップ

このシナリオでは、日次サービス・メンテナンス中に作成されたスナップショットをローカル・コンピュータにバックアップするプロセスの自動化方法を説明します。

- メンテナンス・ウィンドウ中に作成されたアプリケーションのスナップショット(Artifact Snapshot)をダウンロードします
- タイムスタンプを付加してダウンロードしたスナップショットの名前を変更します
- 必要に応じて最も古いバックアップを削除することで、バックアップを 10 個を維持します

Note:

- このスクリプトを使用して **Narrative Reporting** をバックアップすることはできません
- このスクリプトを使用のために再利用する場合、必要に応じてランタイム・パラメータの値(url、user、password および NumberOfBackups)を変更してください。

Windows タスク・スケジューラを使用してスクリプトをスケジュールする方法の詳細は、[スクリプトの実行の自動化](#)を参照してください。

Windows のサンプル・スクリプト

スナップショットのダウンロードを自動化する次のようなスクリプトが含まれるバッチ(.bat)またはシェル(.sh)ファイルを作成します。

```
@echo off
rem Sample script to download and maintain 10 maintenance backups
rem Update the following parameters

SET url=https://example.oraclecloud.com
SET user=ServiceAdmin
SET password=Example.epw
SET SnapshotName="Artifact Snapshot"
SET NumberOfBackups=10

rem EPM Automate commands
call epmautomate login %user% %password% %url%
IF %ERRORLEVEL% NEQ 0 goto :ERROR
    call epmautomate downloadfile %SnapshotName%
IF %ERRORLEVEL% NEQ 0 goto :ERROR
    call epmautomate logout
IF %ERRORLEVEL% NEQ 0 goto :ERROR

rem Rename downloaded Artifact Snapshot, keep the last 10 backups
Set Timestamp=%date:~4,2%_%date:~7,2%_%date:~10,2%%
Set Second=%time:~0,2%%time:~3,2%
ren %SnapshotName%.zip %SnapshotName%_%Timestamp%_%Second%.zip
```

```

SET Count=0
FOR %%A IN (%SnapshotName%*.*) DO SET /A Count += 1
IF %Count% gtr %NumberOfBackups% FOR %%A IN (%SnapshotName%*.*) DO del "%%A"
&& GOTO EOF
:EOF

echo Scheduled Task Completed successfully
exit /b %errorlevel%
:ERROR
echo Failed with error #%errorlevel%.
exit /b %errorlevel%

```

Linux/UNIX のサンプル・スクリプト

スナップショットのダウンロードを自動化する次のようなスクリプトが含まれるシェル(.sh) ファイルを作成します。パスワードに特殊文字が含まれている場合は、[特殊文字の処理](#)を参照してください。

```

#!/bin/sh
# Sample script to download and maintain 10 maintenance backups
# Update the following seven parameters

url=https://example.oraclecloud.com
user=serviceAdmin
password=/home/user1/epmautomate/bin/example.epw
snapshotname="Artifact Snapshot"
numberofbackups=10
epmautomatescript=/home/user1/epmautomate/bin/epmautomate.sh
javahome=/home/user1/jdk1.8.0_191/

export JAVA_HOME=${javahome}

printResult()
{
    op="$1"
    opoutput="$2"
    returncode="$3"

    if [ "${returncode}" -ne 0 ]
    then
        echo "Command failed. Error code: ${returncode}. ${opoutput}"
    else
        echo "${opoutput}"
    fi
}

processCommand()
{
    op="$1"
    date=`date`

    echo "Running ${epmautomatescript} ${op}"
    operationoutput=`eval "$epmautomatescript $op"`
    printResult "$op" "$operationoutput" "$?"
}

```

```

op="login ${user} ${password} ${url}"
processCommand "${op}"

op="downloadfile \"${snapshotname}\""
processCommand "${op}"

op="logout"
processCommand "${op}"

# Renames the downloaded artifacts, keeps the last 10 backups
timestamp=`date +%m_%d_%Y_%I%M`
mv "${snapshotname}.zip" "${snapshotname}_${timestamp}.zip"

((numberofbackups+=1))
ls -tp ${snapshotname}*.zip | grep -v '/$' | tail -n +${numberofbackups} |
xargs -d '\n' -r rm --

```

ユーザーへの日次メンテナンス完了の通知

Oracle Fusion Cloud Enterprise Performance Management 環境の日次メンテナンスに要する時間は、多くの場合、予定されている 1 時間よりもはるかに短くなります。

環境の実際の日次メンテナンス所要時間は、アクティビティ・レポートの操作メトリック・セクションにある日次メンテナンス所要時間(分単位)メトリックの値として記録されます。時間全体を待たずに環境を使用する場合は、このスクリプトのカスタム・バージョンを使用して、日次メンテナンスが完了してユーザーがアクティビティを再開できることをユーザーに通知します。

Windows スクリプト

次の PowerShell スクリプトをコピーして、daily_maintenance_completed.ps1 を作成します。スクリプトを更新して使用方法の詳細は、[スクリプトの実行](#)を参照してください。

```

# Daily Maintenance Completed Notification script
#
# Update the following parameters
# -----
$emailaddresses=user1@oracle.com,user2@oracle.com
# -----

$username=$args[0]
$password=$args[1]
$url=$args[2]

if (($args.count) -ne 3) {
    echo "Usage: ./daily_maintenance_completed.ps1 <USERNAME> <PASSWORD>
<URL>"
    exit 1
}

$amw_time=""

function getDailyMaintenanceStartTime {
    $amwstring=$(epmautomate.bat getDailyMaintenanceStartTime)

```

```

$elements=$amwstring.split(' ')
$amwtime=$elements[0]
return $amwtime
}

function goToSleep ($amw_time){
    $current_mdy=Get-Date -AsUTC -UFormat "%m/%d/%Y"
    $current_date_time=Get-Date -AsUTC -UFormat "%m/%d/%Y %H:%M:%S"
    $current_epoch=Get-Date -Date $current_date_time -UFormat "%s"
    $target_date_time=[DateTime]"${current_mdy} ${amw_time}"
    $target_epoch=Get-Date -Date $target_date_time -UFormat "%s"
    $sleep_seconds=$target_epoch - $current_epoch

    # Today's AMW start time has already passed, so add 24 hours to
sleep_seconds
    if ($sleep_seconds -lt 0) {
        $sleep_seconds=$sleep_seconds + 86400
    }

    $sleep_ts=New-TimeSpan -Seconds ${sleep_seconds}
    $sleep_hms="${sleep_ts}" -replace '^\\d+?\\.\\.'

    echo "Current time is ${current_date_time}. Sleeping for ${sleep_hms},
until daily maintenance start time of ${amw_time}."
    Start-Sleep -Seconds $sleep_seconds
}

function attemptLogin {
    $serverdown=$False
    while ($true) {
        epmautomate.bat login ${username} ${password} ${url}
        if ($?) { # login succeeded
            if ($serverdown) { # server has been brought down
                echo "Daily maintenance processing has completed ..."
                break
            } else { # server has not yet been brought down
                echo "Daily maintenance processing has not yet started.
Sleeping for 2 minutes before the next check ..."
                Start-Sleep -Seconds 120
            }
        } else { # login failed
            if ($serverdown) { # server has been brought down
                echo "Waiting for daily maintenance processing to complete.
Sleeping for 2 minutes before the next check ..."
                Start-Sleep -Seconds 120
            } else { # server has not yet been brought down
                echo "Daily maintenance processing is now beginning. Sleeping
for 2 minutes before the next check ..."
                Start-Sleep -Seconds 120
            }
            $serverdown=$True
        }
    }
}

function sendNotification {

```

```

$servername=$url.split("/") [2];
$subject="Daily maintenance processing has completed"
$formattedmessage="Daily maintenance processing has completed for server $
{servername}"
$emailaddresses=${emailaddresses}.replace(',',';')

echo "Mailing report"
epmautomate.bat sendmail "${emailaddresses}" "${subject}" Body="$
{formattedmessage}"
}

echo "Beginning daily maintenance completion notification script."
echo "Logging into server ..."
epmautomate.bat login ${username} ${password} ${url}
$amwtime=getDailyMaintenanceStartTime
goToSleep ($amwtime)
attemptLogin
sendNotification
echo "Logging out of server ..."
epmautomate.bat logout
echo "Script processing has completed."

```

Linux/UNIX のスクリプト

次のスクリプトをコピーして、daily_maintenance_completed.sh を作成します。スクリプトを更新して使用方法の詳細は、[スクリプトの実行](#)を参照してください。

```

#!/bin/bash
# Update the following parameters
# -----
epmautomatescript="LOCATION_EPM_AUTOMATE_EXECUTABLE"
javahome="LOCATION_JAVA_HOME"
emailaddresses=EMAIL_ADDRESS_1,EMAIL_ADDRESS_2,EMAIL_ADDRESS_N
# -----

username="$1"
password="$2"
url="$3"

export JAVA_HOME=${javahome}

if [ "$#" -ne 3 ]; then
    echo "Usage: ./daily_maintenance_completed.sh <USERNAME> <PASSWORD> <URL>"
    exit 1
fi

amw_time=""

getDailyMaintenanceStartTime() {
    amw_time=${${epmautomatescript} getDailyMaintenanceStartTime | cut -d' ' -
f1)
}

goToSleep() {
    current_mdy=$(date -u +%m/%d/%Y)

```

```

current_date_time=$(date -u)
current_epoch=$(date +%s)
target_epoch=$(date -d "${current_mdy} ${amw_time}" +%s)
sleep_seconds=$((target_epoch - current_epoch))

# Today's AMW start time has already passed, so add 24 hours to
sleep_seconds
if [[ ${sleep_seconds} -lt 0 ]]
then
    sleep_seconds=$((sleep_seconds + 86400))
fi

sleep_hms=$(date -d@${sleep_seconds} -u +%H:%M:%S)

echo "Current time is ${current_date_time}. Sleeping for ${sleep_hms},
until daily maintenance start time of ${amw_time}."
sleep $sleep_seconds
}

attemptLogin() {
    local serverdown=1
    while true
    do
        ${epmautomatescript} login ${username} ${password} ${url}
        if [[ $? -eq 0 ]] # login succeeded
        then
            if [[ ${serverdown} -eq 0 ]] # server has been brought down
            then
                echo "Daily maintenance processing has completed"
                break
            else # server has not yet been brought down
                echo "Daily maintenance processing has not yet started.
Sleeping for 2 minutes before the next check ..."
                sleep 120
            fi
        else # login failed
            if [[ ${serverdown} -eq 0 ]] # server has been brought down
            then
                echo "Waiting for daily maintenance processing to complete.
Sleeping for 2 minutes before the next check ..."
                sleep 120
            else # server has not yet been brought down
                echo "Daily maintenance processing is now beginning. Sleeping
for 2 minutes before the next check ..."
                sleep 120
                serverdown=0
            fi
        fi
    done
}

sendNotification()
{
    local servername=$(echo "${url}" | cut -d '/' -f3- | rev | cut -d ':' -f2-
| rev)
    local subject="Daily maintenance processing has completed"

```

```

    local formattedmessage="Daily maintenance processing has completed for
server ${servername}"
    local emailaddresses=$(echo ${emailaddresses} | sed "s/,;/g")

    echo "Mailing report"
    ${epmautomatescript} sendmail "${emailaddresses}" "${subject}" Body="${
formattedmessage}"
}

echo "Beginning daily maintenance completion notification script."
echo "Logging into server ..."
${epmautomatescript} login ${username} ${password} ${url}
getDailyMaintenanceStartTime
goToSleep
attemptLogin
sendNotification
echo "Logging out of server ..."
${epmautomatescript} logout
echo "Script processing has completed."

```

サーバー側 Groovy スクリプト

次のコードをコピーして、daily_maintenance_completed **Groovy** スクリプトを作成します。スクリプトを更新して使用方法の詳細は、[スクリプトの実行](#)を参照してください。

```

// Daily Maintenance Completed Notification script

// Update the following parameters
// -----
String username="USERNAME"
String password="PASSWORD"
String url="URL OF THE ENVIRONMENT"
String emailaddresses="EMAIL_ADDRESS_1,EMAIL_ADDRESS_2,EMAIL_ADDRESS_N"
// -----

def LogMessage(String message) {
    def date = new Date()
    def sdf = new SimpleDateFormat("MM/dd/yyyy HH:mm:ss")
    println('[ ' + sdf.format(date) + ' ] ' + message);
}

def LogOperationStatus(EpmAutomateStatus opstatus) {
    def returncode = opstatus.getStatus()
    if (returncode != 0){
        LogMessage(opstatus.getOutput())
    }
    LogMessage('return code: ' + returncode)
}

def getDailyMaintenanceStartTime(EpmAutomate automate) {
    LogMessage("Operation: getDailyMaintenanceStartTime")
    EpmAutomateStatus amwtimestatus =
automate.execute('getDailyMaintenanceStartTime')
    LogOperationStatus(amwtimestatus)
    def amwstring=(amwtimestatus.getOutput())
}

```

```

def elements=amwstring.split(' ')
def amwtime=elements[0]
return amwtime
}

def goToSleep(String amw_time){
def date = new Date()
def current_mdy = new SimpleDateFormat("MM/dd/yyyy")
def current_date_time = new SimpleDateFormat("MM/dd/yyyy HH:mm:ss")
float current_epoch = date.getTime() / 1000
def pattern = "MM/dd/yyyy HH:mm:ss"
def input = current_mdy.format(date) + " " + amw_time + ":00"
def target_date_time = Date.parse(pattern, input)
float target_epoch = target_date_time.getTime() / 1000
int sleep_seconds = Math.round(target_epoch - current_epoch)

//Today's AMW start time has already passed, so add 24 hours to
sleep_seconds
if (sleep_seconds < 0) {
sleep_seconds = sleep_seconds + 86400
}

def sleep_milliseconds = sleep_seconds * 1000
LogMessage("Current time is " + current_date_time.format(date) + ".
Sleeping until daily maintenance start time of " + amw_time + ":00.")
sleep(sleep_milliseconds)
}

def attemptLogin(EpmAutomate automate, String username, String password,
String url) {
def serverdown=1
while (true) {
LogMessage("Operation: login " + username + " " + password + " " +
url)
EpmAutomateStatus status =
automate.execute('login',username,password,url)
def returncode = status.getStatus()
if (returncode == 0) {
if (serverdown == 0){
LogMessage("Daily maintenance processing has completed ...")
break
} else {
LogMessage("Daily maintenance processing has not yet started.
Sleeping for 2 minutes before the next check ...")
sleep(120000)
}
} else {
if (serverdown == 0){
LogMessage("Waiting for daily maintenance processing to
complete. Sleeping for 2 minutes before the next check ...")
sleep(120000)
} else {
LogMessage("Daily maintenance processing is now beginning.
Sleeping for 2 minutes before the next check ...")
sleep(120000)
serverdown=0
}
}
}
}

```

```

    }
  }
}

def sendNotification(EpmAutomate automate, String url, String emailaddresses)
{
  def servername=url.tokenize("/")[-1];
  def subject="Daily maintenance processing has completed"
  def formattedmessage="Daily maintenance processing has completed for
server " + servername
  def emailaddressesformatted = emailaddresses.replaceAll(',',';')

  LogMessage("Operation: sendmail " + emailaddressesformatted + " " +
subject + " Body=" + formattedmessage)
  EpmAutomateStatus status =
automate.execute('sendmail',emailaddressesformatted,subject,'Body=' +
formattedmessage)
  LogOperationStatus(status)
}

LogMessage("Beginning daily maintenance completion notification script.")

EpmAutomate automate = getEpmAutomate()

LogMessage("Operation: login " + username + " " + password + " " + url)
EpmAutomateStatus status = automate.execute('login',username,password,url)
LogOperationStatus(status)

String amwtime = getDailyMaintenanceStartTime(automate)
goToSleep (amwtime)
attemptLogin(automate,username,password,url)
sendNotification(automate,url,emailaddresses)

LogMessage("Operation: logout ")
status = automate.execute('logout')
LogOperationStatus(status)

LogMessage ("Script processing has completed.")

```

スクリプトの実行

Windows および Linux/UNIX

1. 前述の項のスクリプトをコピーして、daily_maintenance_completed.ps1 または daily_maintenance_completed.sh を作成します。
2. スクリプトを更新します:
 - **Windows:** 日次メンテナンスが完了したときに通知する必要がある電子メール・アドレスのカンマ区切りリストを指定して、emailaddresses の値を更新します。
 - **Linux/UNIX:** 次の変数を更新します:
 - epmautomatescript (EPM 自動化の実行可能ファイルの場所を指定)。例:
epmautomatescript="/home/utils/EPMAutomate/bin/epmautomate.sh"

- javahome (EPM 自動化で使用される JDK がインストールされているディレクトリを指定)。例: `"/home/user1/jdk1.8.0_191"`
 - emailaddresses (日次メンテナンスが完了したときに通知する必要がある電子メール・アドレスのカンマ区切りリストを指定)。例:
`jdoe@example.com,jane_doe@example.com`
3. コマンド・ウィンドウまたはコンソールで、`daily_maintenance_completed` スクリプトが格納されたフォルダにナビゲートします。
 4. 次のコマンドを実行します。
 - **Windows:** `./daily_maintenance_completed.ps1 USERNAME PASSWORD URL`
 - **Linux/UNIX:** `./daily_maintenance_completed.sh USERNAME PASSWORD URL`、ここで:
 - USERNAME は、サービス管理者のユーザー名です
 - PASSWORD は、サービス管理者のパスワードです
 - URL は、Cloud EPM 環境の URL です

サーバー側 Groovy:

1. 前述の項のスクリプトをコピーして、`daily_maintenance_completed.groovy` Groovy スクリプトを作成します。
2. 次の値を更新します。
 - `username` (サービス管理者のユーザー名を指定)。
 - `password` (サービス管理者のパスワードを指定)
 - `url` (日次メンテナンス完了通知を送信する必要がある Cloud EPM 環境の URL を指定)。次に例を示します。例: `https://testExample-idDomain.pbcs.us1.oraclecloud.com`
 - `emailaddresses` (日次メンテナンスが完了したときに通知する必要がある電子メール・アドレスのカンマ区切りリストを指定)。
3. Cloud EPM ビジネス・プロセスで Groovy 画面を使用するか、`runBusinessRule` を使用してスクリプトの実行を自動化します。詳細は、次の情報ソースを参照してください:
 - [EPM 自動化をインストールしないコマンドの実行](#)
 - *Planning の管理* の Groovy ルールの使用

Oracle Object Storage との間でのスナップショットのコピー

このトピックには、これらのタスクを完了するためのサンプル・スクリプトが含まれています。

- **Artifact Snapshot (メンテナンス・スナップショット)** を **Oracle Fusion Cloud Enterprise Performance Management** から **Oracle Object Storage** バケットにコピーし、スナップショットがコピーされた日付を追加して名前を変更します。
- **バックアップ・スナップショット** を **Oracle Object Storage** バケットから **Cloud EPM** にコピーします。

このセクションのスクリプトは、スナップショットを保持するために **Oracle Object Storage** にバケットがすでに作成されていることを前提としています。これらのスクリプトを実行する前に、次のプレースホルダーを更新して使用するよう、スクリプトをカスタマイズしてください。

Table 3-1 パラメータとその値

プレースホルダー	必要な値
JAVA_HOME	EPM 自動化で使用される JDK がインストールされているディレクトリ。 例: ./home/JDK/bin
epmautomateExe	EPM 自動化がインストールされているディレクトリ。 例: ./home/utills/EPMAutomate/bin
cloudServiceUser	Cloud EPM サービス管理者のユーザー ID。 例: John.doe@example.com
cloudServicePassword	サービス管理者のパスワードまたはパスワード・ファイルの場所。パスワードに特殊文字が含まれている場合は、 特殊文字の処理 を参照してください。 例: ex_PWD_213
cloudServiceUrl	アーティファクト・スナップショットのコピー元の Cloud EPM 環境の URL。 例: https://test-cloud-id_Dom.pbcs.us1.oraclecloud.com
objectStorageUser	Oracle Object Storage のユーザーのユーザー ID。 スナップショットを Object Storage にコピーするには、このユーザーはスナップショットのコピー先のバケットへの書き込みアクセス権を持っている必要があります。スナップショットを Object Storage からコピーするには、このユーザーはスナップショットのコピー元のバケットへの読取りアクセス権を持っている必要があります。 例: jDoe
objectStoragePassword	objectStorageUser のパスワード。 例: example_PWD
objectStorageBucketUrl	スナップショットのコピー先の Oracle Object Storage バケットの URL。URL 形式については、次の情報ソースを参照してください。 <ul style="list-style-type: none"> copyToObjectStorage copyFromObjectStorage 例: https://swiftobjectstorage.us-ashburn-1.oraclecloud.com/v1/axaxnpcrorw5/bucket-20210301-1359
snapshot	Oracle Object Storage バケットからコピーするスナップショットの名前。 例: Artifact Snapshot20210429.zip

サンプル EPM 自動化スクリプト。スナップショットを Cloud EPM から Oracle Object Storage にコピー

このスクリプトをカスタマイズして実行し、名前を変更してから、アーティファクト・スナップショットを Cloud EPM から Oracle Object Storage バケットにコピーします。

```
#!/bin/sh
export JAVA_HOME=<path_to_jdk>
epmautomateExe=<path_to_epmautomate_executable>
cloudServiceUser=<cloud_service_user>
cloudServicePassword=<cloud_service_password>
cloudServiceUrl=<cloud_service_url>
# User with write access to Object Storage bucket
objectStorageUser=<object_storage_user>
```

```
objectStoragePassword=<object_storage_password>
objectStorageBucketUrl=<object_storage_bucket>
currentDate=`date +%Y%m%d`
sourceSnapshot="Artifact Snapshot"
targetSnapshot="${sourceSnapshot} ${currentDate}"
$pmautomateExe login ${cloudServiceUser} ${cloudServicePassword} $
{cloudServiceUrl}
$pmautomateExe renamesnapshot "${sourceSnapshot}" "${targetSnapshot}"
$pmautomateExe copyToObjectStorage "${targetSnapshot}" ${objectStorageUser} $
{objectStoragePassword} "${objectStorageBucketUrl}/${targetSnapshot}"
$pmautomateExe logout
exit 0
```

サンプル EPM 自動化スクリプト。スナップショットを Oracle Object Storage から Cloud EPM にコピー

このスクリプトをカスタマイズして実行し、特定の日付のアーティファクト・スナップショットを Oracle Object Storage バケットから Cloud EPM にコピーします。

```
#!/bin/sh
export JAVA_HOME=<path_to_jdk>
epmautomateExe=<path_to_epmautomate_executable>
cloudServiceUser=<cloud_service_user>
cloudServicePassword=<cloud_service_password>
cloudServiceUrl=<cloud_service_url>
# User with read access to Object Storage bucket
objectStorageUser=<object_storage_user>
objectStoragePassword=<object_storage_password>
objectStorageBucketUrl=<object_storage_bucket>
snapshot=<desired_snapshot>
$pmautomateExe login ${cloudServiceUser} ${cloudServicePassword} $
{cloudServiceUrl}
$pmautomateExe copyFromObjectStorage ${objectStorageUser} $
{objectStoragePassword} "${objectStorageBucketUrl}/${snapshot}"
$pmautomateExe logout
exit 0
```

ユーザーの作成および事前定義済役割への割当て

この項のスクリプトを使用してユーザーを作成し、アイデンティティ・ドメインに事前定義されている役割にそのユーザーを割り当てます。

これらのスクリプトは、EPM 自動化コマンドを使用して次のアクティビティを完了します：

- アイデンティティ・ドメイン管理者の役割も持つサービス管理者として環境にサインインします。
- グループとメンバーシップ情報を環境からエクスポートして、指定したスナップショット (例: example_snapshot.zip) を再生成します。このスナップショットを作成するために、「移行」を使用して「グループとメンバーシップ」アーティファクトをすでにエクスポート済であると仮定します。
- スナップショット(example_snapshot.zip)をローカル・ディレクトリにダウンロードします。
- スナップショット(example_snapshot.zip)を環境から削除します。

- 環境からサインアウトします。
- カスタム・コードを追加した操作を実行します。次の操作が含まれる場合があります:
 - example_snapshot.zip の内容を抽出します
 - 新規ユーザー情報を「名」、「姓」、「電子メール」、「ユーザー・ログイン」の書式で HSS-Shared Services\resource\External Directory\Users.csv に追加します。詳細は、『Oracle Cloud スタート・ガイド』のユーザー・アカウントのバッチのインポートを参照してください。
 - 新規ユーザー(「名」、「姓」、「電子メール」、「ユーザー・ログイン」の書式)の役割割当てに関する情報を HSS-Shared Services\resource\External Directory\Roles\内の適切な役割ファイルに追加します。たとえば、サービス管理者の役割への割当ては <service_name> Service Administrator.csv に追加する必要がありますが、参照者の役割への割当ては HSS-Shared Services\resource\External Directory\Roles\ - HSS-Shared Services ディレクトリおよびその内容を圧縮することで、更新されたファイルでスナップショットを再作成します。
- アイデンティティ・ドメイン管理者の役割も保持するサービス管理者として環境にサインインします。
- 変更された example_snapshot.zip を環境にアップロードします。
- example_snapshot.zip を環境にインポートします。
- アップロードされた example_snapshot.zip を環境から削除します。
- サインアウトします。

Windows のサンプル・スクリプト

次のスクリプトをコピーして、createUsersAndAssignRoles.ps1 という名前のファイルを作成します。それをローカル・ディレクトリに保存します。

```
$inputproperties = ConvertFrom-StringData(Get-Content ./input.properties -raw)
$username="$($inputproperties.username) "
$passwordfile="$($inputproperties.passwordfile) "
$serviceURL="$($inputproperties.serviceURL) "
$snapshotName="$($inputproperties.snapshotName) "
$userspassword="$($inputproperties.userspassword) "
$resetPassword="$($inputproperties.resetPassword) "

epmautomate login ${username} ${passwordfile} ${serviceURL}
epmautomate exportsnapshot ${snapshotName}
epmautomate downloadfile ${snapshotName}.zip
epmautomate deletefile ${snapshotName}.zip
epmautomate logout

# Add custom code to extract the contents of example_snapshot.zip
# Add custom code to append new user information to HSS-Shared
Services\resource\External Directory\Users.csv
# Optional: Add custom code to append role information to the appropriate
role file(s) in HSS-Shared Services\resource\External Directory\Roles\
# Add custom code to zip HSS-Shared Services and its contents as
```

```
example_snapshot.zip
```

```
epmautomate login ${username} ${passwordfile} ${serviceURL}
epmautomate uploadfile ${snapshotName}.zip
epmautomate importsnapshot ${snapshotName} userPassword=${userspassword}
resetPassword=${resetPassword}
epmautomate deletefile ${snapshotName}.zip
epmautomate logout
```

Linux/UNIX のサンプル・スクリプト

次のスクリプトをコピーして、createUsersAndAssignRoles.sh という名前のファイルを作成します。それをローカル・ディレクトリに保存します。

```
#!/bin/bash

. ./input.properties
export JAVA_HOME=${javahome}
${epmautomatescript} login "${username}" "${passwordfile}" "${serviceURL}"
${epmautomatescript} exportsnapshot "${snapshotName}"
${epmautomatescript} downloadfile "${snapshotName}.zip"
${epmautomatescript} deletefile "${snapshotName}.zip"
${epmautomatescript} logout

# Add custom code to extract the contents of example_snapshot.zip
# Add custom code to append new user information to HSS-Shared
Services\resource\External Directory\Users.csv
# Optional: Add custom code to append role information to the appropriate
role file(s) in HSS-Shared Services\resource\External Directory\Roles\
# Add custom code to zip HSS-Shared Services and its contents as
example_snapshot.zip

${epmautomatescript} login "${username}" "${passwordfile}" "${serviceURL}"
${epmautomatescript} uploadfile "${snapshotName}.zip"
${epmautomatescript} importsnapshot "${snapshotName}" "userPassword=${
userspassword}" "resetPassword=${resetPassword}"
${epmautomatescript} deletefile "${snapshotName}.zip"
${epmautomatescript} logout
```

input.properties ファイルのサンプル

createUsersAndAssignRoles スクリプトを実行するには、input.properties ファイルを作成し、環境の情報でファイルを更新します。createUsersAndAssignRoles.ps1 または createUsersAndAssignRoles.sh が格納されているディレクトリにファイルを保存します。

Windows

```
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
snapshotName=SNAPSHOT_NAME
userspassword=TEMP_IDM_PASSWORD
resetPassword=true
```

Linux/UNIX

```
javahome=JAVA_HOME
epmautomatescript=EPM_AUTOMATE_LOCATION
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
snapshotName=SNAPSHOT_NAME
userspassword=TEMP_IDM_PASSWORD
resetPassword=true
```

表 3-2 input.properties のパラメータ

パラメータ	説明
javahome	JAVA_HOME の場所。Linux/UNIX の場合のみ。
epmautomatescript	EPM 自動化の実行可能ファイル(epmautomate.sh)の絶対パス。Linux/UNIX の場合のみ。
username	アイデンティティ・ドメイン管理者の役割も保持するサービス管理者のユーザー名。
password	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
serviceURL	スナップショットを生成する環境の URL。
snapshotName	生成するスナップショットの名前。このスナップショットを作成するために、「移行」を使用して「グループとメンバーシップ」アーティファクトをすでにエクスポート済であると仮定します。
userspassword	新規ユーザーの初期パスワード。
resetPassword	新規ユーザーが最初のログインの際にパスワードをリセットする必要があるかどうか。新規ユーザーの初回ログイン時にパスワードを強制的に変更する場合は、この値を true に設定します。

スクリプトの実行

1. 前述の項のスクリプトをコピーして、createUsersAndAssignRoles.ps1 または createUsersAndAssignRoles.sh を作成します。
2. カスタム・コードを追加して、次の操作を実行します:
 - スナップショットの内容を抽出します
 - 新規ユーザー情報を HSS-Shared Services\resource\External Directory\Users.csv に追加します。
 - オプションで、新規ユーザー(「名」、「姓」、「電子メール」、「ユーザー・ログイン」の書式)の役割割当てに関する情報を HSS-Shared Services\resource\External Directory\Roles\内の適切な役割ファイルに追加します。
 - 更新されたファイルを使用してスナップショットを再作成します。
3. input.properties ファイルを作成して、createUsersAndAssignRoles スクリプトが配置されているディレクトリに保存します。このファイルのコンテンツは、オペレーティング・システムによって異なります。input.properties ファイルのサンプルを参照してください。このディレクトリの書込み権限があることを確認してください。Windows の場合、スクリプトを実行できるよう、「管理者として実行」オプションを使用して PowerShell を開始する必要があります。

4. スクリプトを起動します。
 - **Windows PowerShell:** createUsersAndAssignRoles.ps1 を実行します。
 - **Linux/UNIX:** ./createUsersAndAssignRoles.sh を実行します。

ライセンスされたユーザー(役割に割り当てられたユーザー)の数のカウント

この項のスクリプトを使用して役割割当レポートを生成し、環境のユーザー数をカウントします。

次のスクリプトをコピーすることで、provisionedUsersCount.bat を作成します。

ノート:

- provisionedUsersCount.bat を実行するための入力パラメータは、username、password/password_file、service_url および report_file_name です。たとえば、コマンド・プロンプトで、次のようなコマンドを入力します。
provisionedUsersCount.bat jdoe password.epw https://example.oraclecloud.com myRole_assign.CSV
- パスワードに特殊文字が含まれている場合は、[特殊文字の処理](#)を参照してください。

```
@echo off

set paramRequiredMessage=Syntax: provisionedUsersCount.bat USERNAME PASSWORD/
PASSWORD_FILE URL REPORT_FILE_NAME

if "%~1" == "" (
    echo User Name is missing.
    echo %paramRequiredMessage%
    exit /b 1
)

if "%~2" == "" (
    echo Password or Password_File is missing.
    echo %paramRequiredMessage%
    exit /b 1
)

if "%~3" == "" (
    echo URL is missing.
    echo %paramRequiredMessage%
    exit /b 1
)

if "%~4" == "" (
    echo Role Assignment Report File Name is missing.
    echo %paramRequiredMessage%
    exit /b 1
)
```

```

)

call epmautomate.bat login %~1 "%~2" %~3
REM call epmautomate.bat login %~1 "%~2" %~3

if %errorlevel% NEQ 0 exit /b 1
    call epmautomate.bat roleAssignmentReport "%4"
if %errorlevel% NEQ 0 exit /b 1
    call epmautomate.bat downloadFile "%4"
if %errorlevel% NEQ 0 exit /b 1

set filePath="%cd%\%4"

if exist %filePath% (
    SETLOCAL EnableDelayedExpansion
    set /a lineCount=0
    set /a userCount=0
    set userHeaderFound=false
    for /f "tokens=*" %%A in ( 'type %filePath%' ) do (
        set /a lineCount+=1
        set line=%%A

        REM Fetch username from role assignment information row
        if !userHeaderFound!==true (
            for /f "delims=" %%i in ("!line!") do (
                set userName=%%i
            )
            if NOT !userName! == "" (
                if !userCount! gtr 0 if NOT !userName! == !lastUserName! (
                    set /a userCount+=1
                    set users[!userCount!]=!userName!
                )
                if !userCount! == 0 (
                    set /a userCount+=1
                    set users[!userCount!]=!userName!
                )
            )
            set lastUserName=!userName!
        )
    )

    REM Check for headers of Role Assignment Report
    if "!line!"=="User Login,First Name,Last Name,Email,Role,Granted
through Group" (
        set userHeaderFound=true
    )
    if "!line!"=="User Login,First Name,Last Name,Email,Roles,Granted
Through" (
        set userHeaderFound=true
    )
)

echo Number of Users: !userCount!
for /l %%n in (1,1,!userCount!) do (
    REM echo !users[%%n]!
)
endlocal

```

```
) else (
    echo Invalid provisioning report file path - %filePath%.
)
```

役割に割り当てられたユーザーの監査レポートの作成

この項のスクリプトを使用して、環境内の事前定義済役割に割り当てられたユーザーの監査レポートの作成プロセスを自動化し、オプションで受信者にレポートを電子メールで送信します。

この監査レポートには、このレポートが最後に生成された以降に変更された、事前定義済の役割またはグループに割り当てられたユーザーが表示されます。日次監査レポートを作成するには、次のスクリプトを毎日実行します。

次のスクリプトをコピーすることで、provisioningAuditReport.bat を作成します。このラッパー・バッチ・スクリプトでは PowerShell スクリプトの provisioningAuditReport.ps1 が呼び出され、これに対するソース・コードはこのシナリオの後半で指定されます。

ノート:

- provisioningAuditReport.bat を実行するための入力パラメータは次のとおりです: username、password または password_file、service_url および report_email_to_address (オプションで、レポートを電子メール・アドレスに送信する場合のみ必要)。
- パスワードに特殊文字が含まれている場合は、[特殊文字の処理](#)を参照してください。

```
@echo off
set paramRequiredMessage=Syntax: provisioningAuditReport.bat USERNAME
PASSWORD/PASSWORD_FILE URL [REPORT_EMAIL_TO_ADDRESS]

if "%~1" == "" (
    echo User Name is missing.
    echo %paramRequiredMessage%
    exit /b 1
)
if "%~2" == "" (
    echo Password or Password_File is missing.
    echo %paramRequiredMessage%
    exit /b 1
)
if "%~3" == "" (
    echo URL is missing.
    echo %paramRequiredMessage%
    exit /b 1
)

PowerShell.exe -File provisioningAuditReport.ps1 %*
```

provisioningAuditReport.bat によって、provisioningAuditReport.ps1(これは次のスクリプトをコピーすることで作成します)が呼び出されます。

provisioningAuditReport.ps1 によって 監査レポートが作成されます。これを、provisioningAuditReport.bat が存在するのと同じディレクトリに配置します。

```

$username=$args[0]
$password=$args[1]
$url=$args[2]
$reportemailtoaddress=$args[3]

$date=$(get-date -f dd_MM_yy_HH_mm_ss)
$datedefaultformat=$(get-date)
$logdir="./logs/"
$logfile="$logdir/epmautomate-provisionauditreport-" + $date + ".log"
$reportdir="./reports/"
$provisionreport="provreport-audittest-" + $date + ".csv"
$provisionreporttemp="./provreport-audittest-temp.csv"
$provisionreportunique="./provreport-audittest-unique.csv"
$provisionreportbaselineunique="./provreport-audittest-baseline-unique.csv"

function EchoAndLogMessage
{
    $message=$args[0]
    echo "$message"
    echo "$message" >> $logfile
}

function Init
{
    $logdirexists=Test-Path $logdir
    if (!$logdirexists) {
        mkdir $logdir 2>&1 | out-null
    }

    $logfileexists=Test-Path $logfile
    if ($logfileexists) {
        rm $logfile 2>&1 | out-null
    }

    $reportdirexists=Test-Path $reportdir
    if (!$reportdirexists) {
        mkdir $reportdir 2>&1 | out-null
    }
}

function PostProcess
{
    rm $provisionreporttemp
    mv -Force $provisionreportunique $provisionreportbaselineunique
}

function ProcessCommand
{
    $op=$args
    echo "EPM Automate operation: epmautomate.bat $op" >> $logfile
}

```

```

    epmautomate.bat $op >> $logfile 2>&1
    if ($LASTEXITCODE -ne 0) {
        echo "EPM Automate operation failed: epmautomate.bat $op. See $logfile
for details."
        exit
    }
}

function RunEpmAutomateCommands
{
    EchoAndLogMessage "Running EPM Automate commands to generate the
provisioning report."
    ProcessCommand login $username $password $url
    ProcessCommand provisionreport $provisionreport
    ProcessCommand downloadfile $provisionreport
    ProcessCommand deletefile $provisionreport
    ProcessCommand logout
}

function CreateProvisionReportTempFile
{
    # Loop through iteration csv file and parse
    Get-Content $provisionreport | ForEach-Object {
        $elements=$_split(',')
        echo "$($elements[0]), $($elements[2])" >> $provisionreporttemp
    }
}

function CreateUniqueElementsFile
{
    gc $provisionreporttemp | sort | get-unique > $provisionreportunique
}

function CheckBaselineAndCreateAuditReport
{
    $provisionreportbaselineuniqueexists=Test-
Path $provisionreportbaselineunique
    if (!$provisionreportbaselineuniqueexists) {
        EchoAndLogMessage "No existing provisioning report, so comparison with a
baseline is not possible. Audit report will be created at the next test run."
    } else {
        CreateAuditReport
    }
}

function EmailAuditReport
{
    $auditreport=$args[0]
    $elements=$auditreport.split('/')
    $auditreportname=$elements[2]

    if (${reportemailtoaddress} -match "@") {
        EchoAndLogMessage "Emailing audit report"
        ProcessCommand login $username $password $url
        ProcessCommand uploadFile $auditreport
        ProcessCommand sendMail $reportemailtoaddress "Provisioning Audit

```

```

Report" Body="Provisioning Audit Report is attached."
Attachments=$auditreportname
    ProcessCommand deleteFile $auditreportname
    ProcessCommand logout
}
}

function CreateAuditReport
{
    $auditreport=$reportdir + "auditreport-"+ $date + ".txt"
    $additions = @()
    $deletions = @()

    EchoAndLogMessage "Comparing previous provisioning report with the current
report."
    $compare=compare-object (get-content $provisionreportunique) (get-
content $provisionreportbaselineunique)

    $compare | foreach {
        if ($_.sideindicator -eq '<=')
        {
            $additions += $_.inputobject
        } elseif ($_.sideindicator -eq '>') {
            $deletions += $_.inputobject
        }
    }

    echo "Provisioning Audit Report for $datedefaultformat" > $auditreport
    echo "-----" >> $auditreport

    if ($additions.count -ne 0)
    {
        echo " " >> $auditreport
        echo "Additions:" >> $auditreport
        foreach($element in $additions) { echo "$element" >> $auditreport }
    }

    if ($deletions.count -ne 0)
    {
        echo " " >> $auditreport
        echo "Deletions:" >> $auditreport
        foreach($element in $deletions) { echo "$element" >> $auditreport }
    }

    if (($additions.count -eq 0) -and ($deletions.count -eq 0))
    {
        echo " " >> $auditreport
        echo "No changes from last audit report." >> $auditreport
    }

    EchoAndLogMessage "Provisioning audit report has been
generated: $auditreport."
    EmailAuditReport $auditreport
}

Init

```

```
EchoAndLogMessage "Starting EPMAutomate provisioning audit reporting"
RunEpmAutomateCommands
CreateProvisionReportTempFile
CreateUniqueElementsFile
CheckBaselineAndCreateAuditReport
PostProcess
EchoAndLogMessage "EPMAutomate provisioning audit reporting completed"
```

役割割当ての作成と監査レポートの取消し

この項の PowerShell スクリプトを使用して、環境内の役割割当ておよび役割取消しの詳細を示す監査レポートの作成プロセスを自動化します。

次のスクリプトをコピーすることで、AuditReportRoleAssignment.bat を作成します。このラッパー・バッチ・スクリプトでは PowerShell スクリプトの AuditReportRoleAssignment.ps1 が呼び出され、これに対するソース・コードはこのシナリオの後半で指定されます。

ノート:

- AuditReportRoleAssignment.bat を実行するための入力パラメータは、username、password または password_file、および service_url です。
- パスワードに特殊文字が含まれている場合は、[特殊文字の処理](#)を参照してください。

スクリプト: AuditReportRoleAssignment.bat

```
@echo off
set paramRequiredMessage=Syntax: AuditReportRoleAssignment.bat USERNAME
PASSWORD/PASSWORD_FILE URL

if "%~1" == "" (
    echo User Name is missing.
    echo %paramRequiredMessage%
    exit /b 1
)
if "%~2" == "" (
    echo Password or Password_File is missing.
    echo %paramRequiredMessage%
    exit /b 1
)
if "%~3" == "" (
    echo URL is missing.
    echo %paramRequiredMessage%
    exit /b 1
)

PowerShell.exe -File AuditReportRoleAssignment.ps1 %*
```

スクリプト: AuditReportRoleAssignment.ps1

```
# EPM Automate Role Assignment Audit Report Script
$username=$args[0]
$password=$args[1]
$url=$args[2]

# Generic variables
$date=$(get-date -f dd_MM_yy_HH_mm_ss)
$datedefaultformat=$(get-date)
$logdir="./logs/"
$logfile="$logdir/epmautomate-provisionauditreport-" + $date + ".log"
$reportdir="./reports/"
$provisionreport="provreport-audittest-" + $date + ".csv"
$provisionreporttemp="./provreport-audittest-temp.csv"
$provisionreportunique="./provreport-audittest-unique.csv"
$provisionreportbaselineunique="./provreport-audittest-baseline-unique.csv"

function EchoAndLogMessage
{
    $message=$args[0]
    echo "$message"
    echo "$message" >> $logfile
}

function Init
{
    $logdirexists=Test-Path $logdir
    if (!$logdirexists) {
        mkdir $logdir 2>&1 | out-null
    }
    $logfileexists=Test-Path $logfile
    if ($logfileexists) {
        rm $logfile 2>&1 | out-null
    }
    $reportdirexists=Test-Path $reportdir
    if (!$reportdirexists) {
        mkdir $reportdir 2>&1 | out-null
    }
}

function PostProcess
{
    rm $provisionreporttemp
    mv -Force $provisionreportunique $provisionreportbaselineunique
}

function ProcessCommand
{
    $op=$args
    echo "EPM Automate operation: epmautomate.bat $op" >> $logfile
    epmautomate.bat $op >> $logfile 2>&1
    if ($LASTEXITCODE -ne 0) {
        echo "EPM Automate operation failed: epmautomate.bat $op.
See $logfile for details."
        exit
    }
}
```

```

    }
}

function RunEpmAutomateCommands
{
    EchoAndLogMessage "Running EPM Automate commands to generate the audit
report."
    ProcessCommand login $username $password $url
    ProcessCommand provisionreport $provisionreport
    ProcessCommand downloadfile $provisionreport
    ProcessCommand deletefile $provisionreport
    ProcessCommand logout
}

function CreateProvisionReportTempFile
{
    # Loop through iteration csv file and parse
    Get-Content $provisionreport | ForEach-Object {
        $elements=$_.split(',')
        echo "$($elements[0]), $($elements[2])" >> $provisionreporttemp
    }
}

function CreateUniqueElementsFile
{
    gc $provisionreporttemp | sort | get-unique > $provisionreportunique
}

function CheckBaselineAndCreateAuditReport
{
    $provisionreportbaselineuniqueexists=Test-
Path $provisionreportbaselineunique
    if (!$provisionreportbaselineuniqueexists) {
        EchoAndLogMessage "Could not find a baseline audit report to compare
with. Audit report will be created next time you run test."
    } else {
        CreateAuditReport
    }
}

function CreateAuditReport
{
    $auditreport=$reportdir + "auditreport-"+ $date + ".txt"
    $additions = @()
    $deletions = @()
    EchoAndLogMessage "Comparing previous audit report with the current one."
    $compare=compare-object (get-content $provisionreportunique) (get-
content $provisionreportbaselineunique)
    $compare | foreach {
        if ($_.sideindicator -eq '<=')
        {
            $additions += $_.inputobject
        } elseif ($_.sideindicator -eq '>') {
            $deletions += $_.inputobject
        }
    }
    echo "Provisioning Audit Report for $datedefaultformat" > $auditreport
}

```

```

echo "-----" >> $auditreport
if ($additions.count -ne 0)
{
    echo " " >> $auditreport
    echo "Additions:" >> $auditreport
    foreach($selement in $additions) { echo "$selement" >> $auditreport }
}
if ($deletions.count -ne 0)
{
    echo " " >> $auditreport
    echo "Deletions:" >> $auditreport
    foreach($selement in $deletions) { echo "$selement" >> $auditreport }
}
if (($additions.count -eq 0) -and ($deletions.count -eq 0))
{
    echo " " >> $auditreport
    echo "No changes from last audit report." >> $auditreport
}
EchoAndLogMessage "Role audit report generated: $auditreport."
}

Init
EchoAndLogMessage "Starting EPMAutomate role audit report generation"
RunEpmAutomateCommands
CreateProvisionReportTempFile
CreateUniqueElementsFile
CheckBaselineAndCreateAuditReport
PostProcess
EchoAndLogMessage "EPMAutomate role audit report completed"

```

個人情報保護法に準拠するためのアクセス・ログとアクティビティ・レポートのマスク

この項のスクリプトを使用して、個人情報保護法に準拠するためにアクティビティ・レポートまたはアクセス・ログの情報をマスクするプロセスを自動化し、オプションで受信者にレポートを電子メールで送信します。

一部の国には厳格な個人情報保護法があるため、アクティビティ・レポートとアクセス・ログで使用可能な情報がサービス管理者に表示されないようにしてユーザーの個人情報を保護する必要があります。

anonymizeData.bat を使用して、アクティビティ・レポートまたはアクセス・ログ内の情報をマスクして個人情報保護法に準拠し、オプションで電子メールで情報を送信します。情報をマスクするには、Windows スケジューラを使用してこのスクリプトまたはバリエーションをスケジューリングし、毎日各環境の日次メンテナンス・プロセスの完了後すぐに行われるようにします。

次の情報ソースを使用します:

- [管理者用スタート・ガイドのアクティビティ・レポートとアクセス・ログを使用した使用状況のモニタリング](#)
- [スクリプトの実行の自動化](#)

次の手順に示されている **Windows** スクリプトをコピーして `anonymizeData.bat` を手動で作成し、**Windows** スケジューラを使用してスケジュールします。スケジュールに **Windows** を使用していない場合は、同様のプラットフォームに適したスクリプトを作成および実行できます。

`anonymizeData.bat` は、次の手順で説明するように作成および更新する、`anonymizeData.ps1` スクリプトを実行するラッパー・スクリプトです。

使用するパスワードに特殊文字が含まれている場合は、[特殊文字の処理](#)を参照してください

1. 次のスクリプトが含まれる `anonymizeData.bat` という名前のバッチ(BAT)ファイルを作成し、`C:\automate_scripts` などのわかりやすい場所に保存します。

```
@echo off
set paramRequiredMessage=Syntax: anonymizeData.bat USERNAME PASSWORD/
PASSWORD_FILE URL [EMAIL_TO_ADDRESS]

if "%~1" == "" (
    echo User Name is missing.
    echo %paramRequiredMessage%
    exit /b 1
)
if "%~2" == "" (
    echo Password or Password_File is missing.
    echo %paramRequiredMessage%
    exit /b 1
)
if "%~3" == "" (
    echo URL is missing.
    echo %paramRequiredMessage%
    exit /b 1
)

PowerShell.exe -File anonymizeData.ps1 %*
```

2. 次のスクリプトが含まれる `anonymizeData.ps1` という名前の **PowerShell** スクリプト(PS1)ファイルを作成し、`C:\automate_scripts` などのわかりやすい場所に保存します。

```
# Anonymize data script

$username=$args[0]
$password=$args[1]
$url=$args[2]
$emailtoaddress=$args[3]

# Generic variables
$date=$(get-date -f dd_MM_yy_HH_mm_ss)
$datedefaultformat=$(get-date)
$logdir="./logs/"
$logfile="$logdir/anonymize-data-" + $date + ".log"
$filelist="filelist.txt"

function LogMessage
{
    $message=$args[0]

    echo "$message" >> $logfile
```

```

}

function EchoAndLogMessage
{
    $message=$args[0]

    echo "$message"
    echo "$message" >> $logfile
}

function Init
{
    $logdirexists=Test-Path $logdir
    if (!(($logdirexists)) {
        mkdir $logdir 2>&1 | out-null
    }

    $logfileexists=Test-Path $logfile
    if ($logfileexists) {
        rm $logfile 2>&1 | out-null
    }

    $filelistexists=Test-Path $filelist
    if ($filelistexists) {
        rm $filelist 2>&1 | out-null
    }
}

function ProcessCommand
{
    $op=$args
    echo "EPM Automate operation: epmautomate.bat $op" >> $logfile
    if ($op -eq 'listfiles') {
        epmautomate.bat $op | where {$_ -like ' apr/*/access_log.zip'} | Tee-
Object -FilePath $filelist | Out-File $logfile -Append 2>&1
    } else {
        epmautomate.bat $op >> $logfile 2>&1
        if ($LASTEXITCODE -ne 0) {
            echo "EPM Automate operation failed: epmautomate.bat $op.
See $logfile for details."
            #exit
        }
    }
}

function RunEpmAutomateCommands
{
    EchoAndLogMessage "Running EPM Automate commands to anonymize data in
the access logs and activity reports."
    ProcessCommand login $username $password $url
    ProcessCommand listfiles
    ProcessFiles
    ProcessCommand logout
}

function ProcessActivityReport
{

```

```

$activityreport=$args[0]
$user=$args[1]

$activityreportexists=Test-Path "$activityreport"
if ($activityreportexists) {
    LogMessage "Removing User ID: $user from activity
report $activityreport"
    (Get-Content "$activityreport").replace("$user", 'XXXXX') | Set-
Content "$activityreport"
    $txt = [io.file]::ReadAllText("$activityreport") -replace
" `r`n", "`n"
    [io.file]::WriteAllText("$activityreport", $txt)
    #Get-ChildItem -File -Recurse | % { $x = get-content -raw -
path $activityreport; $x -replace "`r`n", "`n" | set-content -
path $activityreport }
}
}

function AnonymizeData
{
    $aprdir=$args[0]
    $datestampdir=$args[1]
    $path="$aprdir/$datestampdir"
    $accesslogzipped="access_log.zip"
    $accesslog="access_log.csv"
    $accesslogupdated=$accesslog + ".tmp"
    $activityreportfile="$datestampdir" + ".html"
    $userArray = @()

    expand-Archive -Path "$path/$accesslogzipped" -DestinationPath $path
    rm $path/$accesslogzipped 2>&1 | out-null
    $accesslogexists=Test-Path "$path/$accesslog"
    if ($accesslogexists) {
        EchoAndLogMessage "Processing access log: $path/$accesslog"
        Get-Content $path/$accesslog | ForEach-Object {
            $elements=[regex]::Split( $_ , ', (?=(?:[^\"]|"[^"]*" )*)' )
            $date=$elements[0]
            $time=$elements[1]
            $uri=$elements[2]
            $duration=$elements[3]
            $bytes=$elements[4]
            $ip=$elements[5]
            $user=$elements[6]
            $screen=$elements[7]
            $action=$elements[8]
            $object=$elements[9]
            if ($date -like 'Date') {
                echo "$_" >> $path/$accesslogupdated
            } else {
                if ($user -notlike '-') {
                    LogMessage "Removing instance of User ID: $user
from $path/$accesslog."
                    echo
"$date,$time,$uri,$duration,$bytes,$ip,XXXXX,$screen,$action,$object"
>> $path/$accesslogupdated
                    $userArray += $user
                }
            }
        }
    }
}

```

```

        } else {
            echo
"$date,$time,$uri,$duration,$bytes,$ip,$user,$screen,$action,$object"
>> $path/$accesslogupdated
        }
    }
}
#Get-ChildItem -File -Recurse | % { $x = get-content -raw -
path $path/$accesslogupdated; $x -replace "`r`n","`n" | set-content -
path $path/$accesslogupdated }
$txt = [io.file]::ReadAllText("$path/$accesslogupdated") -replace
"`r`n","`n"
[io.file]::WriteAllText("$path/$accesslogupdated", $txt)
mv -Force $path/$accesslogupdated $path/$accesslog
Compress-Archive -Path $path/$accesslog $path/$accesslogzipped
rm $path/$accesslog 2>&1 | out-null
}

EchoAndLogMessage "Processing activity
report: $path/$activityreportfile"
$userArray = $userArray | Select-Object -Unique
foreach ($element in $userArray) {
    ProcessActivityReport "$path/$activityreportfile"
"$element"
}
}

function ProcessFiles
{
    # Loop through iteration csv file and parse
    Get-Content $filelist | ForEach-Object {
        $fullpath=$_trim()
        $elements=$fullpath.split('/')
        $aprdir=$elements[0]
        $datestampdir=$elements[1]
        $accesslogfile="access_log.zip"
        $activityreportfile="$datestampdir" + ".html"
        $datestampdirexists=Test-Path "$aprdir/$datestampdir"
        $accesslog="$aprdir/$datestampdir/$accesslogfile"
        $activityreport="$aprdir/$datestampdir/$activityreportfile"

        echo "fullpath: $fullpath" >> $logfile
        echo "aprdir: $aprdir, datestampdir: $datestampdir" >> $logfile
        if (!$datestampdirexists) {
            mkdir "$aprdir/$datestampdir" -ea 0 2>&1 | out-null
            ProcessCommand downloadfile "$accesslog"
            ProcessCommand downloadfile "$activityreport"
            mv "$accesslogfile" "$aprdir/$datestampdir"
            mv "$activityreportfile" "$aprdir/$datestampdir"
            AnonymizeData "$aprdir" "$datestampdir"
            ProcessCommand deletefile "$accesslog"
            ProcessCommand deletefile "$activityreport"
            ProcessCommand uploadfile "$accesslog" "$aprdir/$datestampdir"
            ProcessCommand uploadfile "$activityreport"
"$aprdir/$datestampdir"
        } else {

```

```

        EchoAndLogMessage "Files in directory $aprrdir/$datestampdir
were processed earlier. Skipping these files."
    }
}

function callSendMail
{
    $elements=$logfile.split('/')
    $logfile=$elements[3]

    if (${emailtoaddress} -match "@") {
        epmautomate.bat login ${username} ${password} ${url}
        epmautomate.bat uploadFile "$logfile"
        epmautomate.bat sendMail $emailtoaddress "Mask Access Logs and
Activity Reports results" Body="The results of running the Mask Access
Logs and Activity Reports script are attached." Attachments=$logfile
        epmautomate.bat deleteFile "$logfile"
        epmautomate.bat logout
    }
}

Init
EchoAndLogMessage "Starting the anonymize data script"
RunEpmAutomateCommands
EchoAndLogMessage "Anonymize data script completed"
EchoAndLogMessage "Refer to logfile: $logfile for details."
callSendMail

```

3. Windows スケジューラを使用して、anonymizeData.bat をスケジュールします。詳細なステップは、[スクリプトの実行の自動化](#)を参照してください。

次のパラメータ値を指定して anonymizeData.bat を実行する必要があります


- サービス管理者のユーザー名
- サービス管理者のパスワードまたは暗号化されたパスワード・ファイルが使用可能な場所
- アクセス・ログおよびアクティビティ・レポートをマスクするサービス環境の URL
- **オプション:** レポートが送信される電子メール・アドレス。この値が指定されている場合のみ、レポートが電子メールで送信されます。

アクティビティ・レポートのローカル・コンピュータへのダウンロード自動化

この項のスクリプトを使用して、環境からローカル・コンピュータへのアクティビティ・レポートのダウンロードを自動化します。

syncAprReports.bat を使用して、アクティビティ・レポートをダウンロードします。Windows スケジューラを使用してバッチ・ファイルをスケジュールし、アクティビティ・レポートのダウンロードを自動化できます。アクティビティ・レポートの詳細は、[管理者用スタート・ガイド](#)のアクティビティ・レポートとアクセス・ログを使用した使用状況のモニタリングを参照してください。

syncAprReports.bat を手動で作成するには、次の手順で指定するスクリプトをコピーして、接続パラメータを更新します。このスクリプトは環境をチェックし、ローカル・コンピュータのダウンロード・ディレクトリで使用可能なものより新しいレポートのみをダウンロードします。

 **ノート:**

- スクリプトは **Windows** コンピュータからのみ実行できます。
- このスクリプトは、ユーザーがフィードバックを送信すると生成されるフィードバック・アクティビティ・レポートはダウンロードしません。
- 使用するパスワードに特殊文字が含まれている場合は、[特殊文字の処理](#)を参照してください

1. 次のスクリプトが含まれる syncAprReports.bat という名前のバッチ(.BAT)ファイルを作成し、C:\automate_scripts などのわかりやすい場所に保存します。

```
@echo off
title APR
setlocal DisableDelayedExpansion

REM To hardcode the values in the script replace %1, %2, %3, and %4, with
the actual values.
REM Example:
REM set apr_dir="C:\Oracle\apr"
REM set username="serviceAdmin"
REM set password="Ex@mp!e!"
REM set url="https://test-example.stg-pbcs.us1.oraclecloud.com"
set apr_dir=%1
set username=%2
set password=%3
set url=%4
setlocal EnableDelayedExpansion
set epmautomate_dir=%cd%
set lastfile=
set argC=0
for %%x in (*) do Set /A argC+=1
if %argC% neq 0 (
    if %argC% neq 3 (
        if %argC% neq 4 (
            goto :usage
        )
    )
)
goto :login
:usage
echo.
echo Invalid syntax. Please check the parameters.
echo Syntax:
echo 1) syncAprReports.bat APR_FolderPath_on_client username password url
echo     or
echo 2) set the parameters in the file and use below syntax
echo     syncAprReports.bat
```

```

goto :end

:login
setlocal DisableDelayedExpansion
for /f "delims=" %%i in ('epmautomate login %username% %password% %url%')
do set result=%%i
if "Login successful" neq "%result%" (
    echo Login Failed
    goto :end
)

if not exist %apr_dir% (
    echo.
    echo apr folder does not exist
    GOTO :end
)
cd /D %apr_dir%
for /f "delims=" %%D in ('dir /a:d /b /o:-n') do (
    REM AFTER: for /f "delims=" %%D in ('dir /a-d /b /s /o:-n') do (
        set "lastFile=%%~nD"
        goto :next
    )
)

:next
setlocal EnableDelayedExpansion
echo.
echo Most Recent APR on client is %lastFile%

set "output_cnt=0"
cd /D %epmautomate_dir%
for /F "delims=" %%f in ('epmautomate listfiles') do (

    cd /D !apr_dir!
    set "line=%%f"
    for /f "tokens=* delims=" %%a in ("!line!") do set line=%%a
    if "!line:~0,3!" equ "apr" (

        if "!line:~4,8!" neq "Feedback" (

            set isValidFile=false
            if "!line:~-5!" equ ".html" set isValidFile=true
            if "!line:~-5!" equ ".json" set isValidFile=true

            if "!isValidFile!" equ "true" (

                if "%lastFile%" lss "!line:~4,19!" (

                    if "!line:~4,19!" neq "!dirname!" (

                        set apr_dir=!apr_dir:!="
                        set /a output_cnt+=1
                        set "output[!output_cnt!]=!apr_dir!\!"

                    line:~4,19!"

                        set "dirname=!line:~4,19!"

                        REM start downloading

```


表 3-3 (続き) syncAprReports.bat に含めるパラメータ値

パラメータ	必要な値
set username=%2	環境にサインインしてアクティビティ・レポートをダウンロードするために使用する Oracle Fusion Cloud Enterprise Performance Management のユーザー名です。 例: set username="ServiceAdmin"
set password=%3	username 変数で指定したユーザーの暗号化されたパスワードを保存するファイルの名前と場所です。ユーザーのパスワードをプレーン・テキストで指定することもできます(推奨されません)。暗号化されたパスワード・ファイルの作成の詳細は、 encrypt コマンドを参照してください。 例: set password="C:\mySecuredir\password.epw" set password="Ex@mp1e1"
set url=%4	環境の URL。 例: set url="https://test-example.stg-pbcs.us1.oraclecloud.com"

3. Windows スケジューラを使用して、syncAprReports.bat をスケジュールします。詳細なステップは、[スクリプトの実行の自動化](#)を参照してください。

環境からのアクセス・ログのダウンロード

この項のスクリプトを使用して、環境からローカル・コンピュータへのアクセス・ログのダウンロード・プロセスを自動化します。

Windows スケジューラを使用して syncAccessLog.bat をスケジュールし、ログ・ファイルのダウンロードを自動化できます。アプリケーション管理を使用してアクセス・ログをダウンロードする手順は、[管理者用スタート・ガイド](#)のアクティビティ・レポートとアクセス・ログの表示およびダウンロードに関する項を参照してください。

次のスクリプトは環境をチェックし、ローカル・コンピュータのダウンロード・ディレクトリで使用可能なものより新しいログ・ファイルのみをダウンロードします。これは Windows のスクリプトです。同様のシェル・スクリプトを Linux/UNIX 環境用に作成できます。

1. 次のスクリプトが含まれる syncAccessLog.bat という名前のバッチ(.BAT)ファイルを作成し、C:\automate_scripts などのわかりやすい場所に保存します。

ノート:

パスワードに特殊文字が含まれている場合は、[特殊文字の処理](#)を参照してください。

```
@echo off
title APR
setlocal DisableDelayedExpansion
```

```
REM To hardcode the values in the script replace %1, %2, %3, and %4 with
the actual values.
```

```
REM Example:
```

```
REM set apr_dir="C:\Oracle\apr"
```

```

REM set username="serviceAdmin"
REM set password="C:\mySecuredir\password.epw"
REM set url="https://test-cloudpln.pbcs.us1.oraclecloud.com"
set apr_dir=%1
set username=%2
set password=%3
set url=%4

setlocal EnableDelayedExpansion
set epmautomate_dir=%cd%
set lastfile=
REM if [%1]==[] goto :usage
REM if [%2]==[] goto :usage
REM if [%3]==[] goto :usage

set argC=0
for %%x in (*) do Set /A argC+=1
if %argC% neq 0 (
    if %argC% neq 3 (
        if %argC% neq 4 (
            goto :usage
        )
    )
)
goto :login

:usage
echo.
echo Invalid syntax. Please check the parameters.
echo Syntax:
echo 1) syncAccessLog.bat APR_FolderPath_on_client username password url
echo    or
echo 2) set the parameters in the file and use below syntax
echo syncAccessLog.bat
goto :end

:login
setlocal DisableDelayedExpansion
REM for /f "delims=" %%i in ('epmautomate login %2 %3 %4') do set result=%%i
for /f "delims=" %%i in ('epmautomate login %username% %password% %url%')
do set result=%%i

if not exist %apr_dir% (
echo.
echo apr folder does not exist
GOTO :end
)
cd /D %apr_dir%
for /f "delims=" %%D in ('dir /a:d /b /o:-n') do (
REM AFTER: for /f "delims=" %%D in ('dir /a-d /b /s /o:-n') do (
    set "lastFile=%%~nD"
    goto :next
)
)

:next

```

```

setlocal EnableDelayedExpansion
echo.
echo Most Recent Access Log on client is %lastFile%

set "output_cnt=0"
cd /D %epmautomate_dir%
for /F "delims=" %%f in ('epmautomate listfiles') do (

    cd /D !apr_dir!
    set "line=%%f"
    for /f "tokens=* delims=" %%a in ("!line!") do set line=%%a
    if "!line:~0,3!" equ "apr" (
        if "!line:~-4!" equ ".zip" (
            if "%lastFile%" lss "!line:~4,19!" (
                if "!line:~4,19!" neq "!dirname!" (
                    set apr_dir=!apr_dir:"=!
                    set /a output_cnt+=1
                    set "output[!output_cnt!]=!apr_dir!\!line:~4,19!"
                    set "dirname=!line:~4,19!"

                    REM start downloading
                    mkdir "!dirname!"
                    cd /D !dirname!
                    echo downloading !line!
                    set "downloadDir=!apr_dir!\!dirname!"
                    cd /D %epmautomate_dir%
                    for /f "delims=" %%i in ('epmautomate downloadfile "!line!"')
                do set result1=%%i
                    move "!line:~24!" "!downloadDir!" > nul
                    echo !result1!
                    REM end downloading

            ) else (
                REM start downloading
                cd /D !dirname!
                echo downloading !line!
                set apr_dir=!apr_dir:"=!
                set "downloadDir=!apr_dir!\!dirname!"
                cd /D %epmautomate_dir%
                for /f "delims=" %%i in ('epmautomate downloadfile "!line!"')
            do set result1=%%i
                    move "!line:~24!" "!downloadDir!" > nul
                    echo !result1!
                    REM end downloading

            )
        ) else (
            REM TO-DO
        )
    )
)

echo.
echo %output_cnt% access logs downloaded
for /L %%n in (1 1 !output_cnt!) DO echo !output[%%n]!
goto :end

```

```
:end
cd /D %epmautomate_dir%
endlocal
```

2. syncAccessLog.bat を変更して、次の表のパラメータの値を設定します。これらの値を使用して環境にアクセスし、アクセス・ログをダウンロードします。

表 3-4 syncAccessLog.bat に含める変数値

変数	必要な値
set apr_dir=%1	アクセス・ログのダウンロード先のディレクトリを指定します。 例: set apr_dir="C:\Oracle\apr"
set username=%2	環境にサインインしてアクセス・ログをダウンロードするために使用する Oracle Fusion Cloud Enterprise Performance Management のユーザー名です。 例: set username="ServiceAdmin"
set password=%3	username 変数で指定したユーザーの暗号化されたパスワードを保存するファイルの名前と場所です。ユーザーのパスワードをプレーン・テキストで指定することもできます(推奨されません)。暗号化されたパスワード・ファイルの作成の詳細は、 encrypt コマンドを参照してください。 例: set password="C:\mySecuredir\password.epw" set password="P@ssword1"
set url=%4	環境の URL。 例: set url="https://test-cloudpln.pbcs.us1.oraclecloud.com"

3. Windows スケジューラを使用して、syncAccessLog.bat をスケジューリングします。詳細なステップは、[スクリプトの実行の自動化](#)を参照してください。

環境のクローニングの自動化

この項のスクリプトを使用して、環境のクローニングを自動化します。

環境をクローニングする次のようなスクリプトが含まれるバッチ(.bat)またはシェル(.sh)ファイルを作成します。後述するサンプル・スクリプトでは次のアクティビティを処理します:

- ソース環境にサインインします。
- Artifact Snapshot (ソース環境の最新の日次メンテナンス時に作成されたスナップショット)またはソース環境で使用可能な別のスナップショットを使用して、ターゲット環境をソースのクローンに変換します。
- オプションで、ソース環境で照合するユーザーおよびその事前定義済役割とアプリケーション役割の割当てを作成します。
- オプションで、ソース環境の日次メンテナンス開始時間と一致するように日次メンテナンス開始時間を変更します。
- サインアウトします。

クローニング・プロセスの詳細は、[移行の管理の Cloud EPM 環境のクローニング](#) を参照してください。

Windows タスク・スケジューラを使用してスクリプトをスケジュールする方法の詳細は、[スクリプトの実行の自動化](#)を参照してください。

Windows

1. 次のスクリプトが含まれる cloneEnvironment.bat という名前のバッチ(.BAT)ファイルを作成し、C:\automate_scripts などのわかりやすい場所に保存します。

```
@echo off
set paramRequiredMessage=Syntax: cloneEnvironment.bat "SOURCE USERNAME"
"SOURCE PASSWORD FILE" "SOURCE URL" "TARGET USERNAME" "TARGET PASSWORD
FILE" "TARGET URL"

set usersandpredefinedroles="false"
set snapshotname="Artifact Snapshot"
set dailymaintenancestarttime="true"
set dirpath=%~dp0
cd %dirpath:~0,-1%

if "%~1" == "" (
    echo Source User Name is missing.
    echo %paramRequiredMessage%
    exit /b 1
)
if "%~2" == "" (
    echo Source Password File is missing.
    echo %paramRequiredMessage%
    exit /b 1
)
if "%~3" == "" (
    echo Source URL is missing.
    echo %paramRequiredMessage%
    exit /b 1
)
if "%~4" == "" (
    echo Target User Name is missing.
    echo %paramRequiredMessage%
    exit /b 1
)
if "%~5" == "" (
    echo Target Password File is missing.
    echo %paramRequiredMessage%
    exit /b 1
)
if "%~6" == "" (
    echo Target URL is missing.
    echo %paramRequiredMessage%
    exit /b 1
)

PowerShell.exe -File cloneEnvironment.ps1 %~1 %~2 %~3 %~4 %~5 %~6
%usersandpredefinedroles% %snapshotname% %dailymaintenancestarttime%
```

2. cloneEnvironment.bat を変更して、次のパラメータの値を設定します:

表 3-5 cloneEnvironment.bat で設定されるパラメータ

パラメータ	説明
usersandpredefinedroles	ユーザーおよびその事前定義済役割とアプリケーション役割の割当てをクローニングするには、このパラメータの値を true に設定します。 ユーザーおよび役割割当てをクローニングするには、スクリプトを実行するユーザーにターゲット環境のサービス管理者役割およびアイデンティティ・ドメイン管理者が必要です。
snapshotname	クローニングで使用するソース環境でのスナップショットの名前。
dailymaintenancestarttime	クローニングされる環境の日次メンテナンス開始時間をソース環境の日次メンテナンス開始時間に変更するには、このパラメータの値を true に設定します。この値を false に設定すると、クローニングされる環境の現在の日次メンテナンス開始時間が保持されます。

3. 次のスクリプトが含まれる cloneEnvironment.ps1 という名前の PowerShell スクリプトを作成し、cloneEnvironment.bat を保存したディレクトリ(C:\automate_scripts など)に保存します。

```
# Clone Environment script

$source_username=$args[0]
$source_password=$args[1]
$source_url=$args[2]
$target_username=$args[3]
$target_password=$args[4]
$target_url=$args[5]
$usersandpredefinedroles=$args[6]
$snapshotname=$args[7]
$dailymaintenancestarttime=$args[8]

epmautomate.bat login "${source_username}" "${source_password}" "${source_url}"
epmautomate.bat cloneEnvironment "${target_username}" "${target_password}" "${target_url}" UsersAndPreDefinedRoles="${usersandpredefinedroles}" SnapshotName="${snapshotname}" DailyMaintenanceStartTime="{dailymaintenancestarttime}"
epmautomate.bat logout
```

4. このコマンドを使用して、cloneEnvironment.bat を実行します:

```
cloneEnvironment.bat "SOURCE USERNAME" "SOURCE PASSWORD FILE" "SOURCE URL" "TARGET USERNAME" "TARGET PASSWORD FILE" "TARGET URL"
```

次に例を示します。

```
cloneEnvironment.bat jdoe@example.com C:\mySecuredir\example_pwd.epw https://source_example.oraclecloud.com jdoe@example.com C:\mySecuredir\example_pwd2.epw https://target_example.oraclecloud.com.
```

Linux

1. 次のスクリプトが含まれる `cloneEnvironment.sh` という名前のシェル・スクリプトを作成し、わかりやすい場所に保存します。

```
#!/bin/bash

# Update the following parameters
# -----
epmautomatescript=/home/user1/epmautomate/bin/epmautomate.sh
javahome=/home/user1/jdk1.8.0_191/
usersandpredefinedroles="false"
snapshotname="Artifact Snapshot"
dailymaintenancestarttime="true"
# -----

source_username="$1"
source_password="$2"
source_url="$3"
target_username="$4"
target_password="$5"
target_url="$6"

export JAVA_HOME=${javahome}

if [ "$#" -ne 6 ]; then
    echo "Usage: ./cloneEnvironment.sh <SOURCE USERNAME> <SOURCE PASSWORD FILE> <SOURCE URL> <TARGET USERNAME> <TARGET PASSWORD FILE> <TARGET URL>"
    exit 1
fi

${epmautomatescript} login "${source_username}" "${source_password}" "${source_url}"
${epmautomatescript} cloneEnvironment "${target_username}" "${target_password}" "${target_url}" UsersAndPreDefinedRoles="${usersandpredefinedroles}" SnapshotName="${snapshotname}" DailyMaintenanceStartTime="${dailymaintenancestarttime}"
${epmautomatescript} logout
```

2. `cloneEnvironment.sh` を変更して、次のパラメータの値を設定します:

表 3-6 cloneEnvironment.sh で設定されるパラメータ

パラメータ	説明
<code>epmautomatescript</code>	EPM 自動化の実行可能ファイル(<code>epmautomate.sh</code>)の絶対パス。
<code>javahome</code>	JAVA_HOME の場所。

表 3-6 (続き) cloneEnvironment.sh で設定されるパラメータ

パラメータ	説明
usersandpredefinedroles	ユーザーおよびその事前定義済役割とアプリケーション役割の割当てをクローニングするには、このパラメータの値を true に設定します。 ユーザーおよび役割割当てをクローニングするには、スクリプトを実行するユーザーにターゲット環境のサービス管理者役割およびアイデンティティ・ドメイン管理者が必要です。
snapshotname	クローニングで使用するソース環境でのスナップショットの名前。
dailymaintenancestarttime	クローニングされる環境の日次メンテナンス開始時間をソース環境の日次メンテナンス開始時間に変更するには、このパラメータの値を true に設定します。この値を false に設定すると、クローニングされる環境の現在の日次メンテナンス開始時間が保持されます。

- cloneEnvironment.sh を実行します。

```
./cloneEnvironment.sh "SOURCE USERNAME" "SOURCE PASSWORD FILE" "SOURCE URL" "TARGET USERNAME" "TARGET PASSWORD FILE" "TARGET URL"
```

次に例を示します。

```
./cloneEnvironment.sh jdoe@example.com ./home/secure/example_pwd.epw https://source_example.oraclecloud.com jdoe@example.com ./home/secure/example_pwd.epw2 https://target_example.oraclecloud.com.
```

プライマリ環境での日次メンテナンス完了後の、プライマリ環境からスタンバイ環境への毎日のクローニング

スタンバイ環境をプライマリ環境にあわせて最新の状態に保つには、プライマリ環境で日次メンテナンスが完了したらすぐに、次のスクリプトを使用して Oracle Fusion Cloud Enterprise Performance Management プライマリ環境をスタンバイ環境にクローニングします。

次のカスタム・スクリプトは、その日にスケジュールされた日次メンテナンスが完了しているかどうかを識別してから、環境をクローニングします。

Windows スクリプト

次の PowerShell スクリプトをコピーすることで、dailyclone.ps1 を作成します。

```
# Clone Environment script
#
# Update the following parameters
# -----
$users_and_predefined_roles="false"
$daily_maintenance_start_time="true"
$data_management="true"
$app_audit="true"
$job_console="true"
$stored_snapshots_and_files="false"
```

```

# -----

$source_username=$args[0]
$source_password=$args[1]
$source_url=$args[2]
$target_username=$args[3]
$target_password=$args[4]
$target_url=$args[5]

if (($args.count) -ne 6) {
    echo "Usage: ./dailyclone.ps1 <SOURCE USERNAME> <SOURCE PASSWORD> <SOURCE
URL> <TARGET USERNAME> <TARGET PASSWORD> <TARGET URL>"
    exit 1
}

$amw_time=""

function getDailyMaintenanceStartTime {
    $amwstring=$(epmautomate.bat getDailyMaintenanceStartTime)
    $elements=$amwstring.split(' ')
    $amwtime=$elements[0]
    return $amwtime
}

function goToSleep ($amw_time){
    $current_mdy=Get-Date -AsUTC -UFormat "%m/%d/%Y"
    $current_date_time=Get-Date -AsUTC -UFormat "%m/%d/%Y %H:%M:%S"
    $current_epoch=Get-Date -Date $current_date_time -UFormat "%s"
    $target_date_time=[DateTime]"${current_mdy} ${amw_time}"
    $target_epoch=Get-Date -Date $target_date_time -UFormat "%s"
    $sleep_seconds=$target_epoch - $current_epoch

    # Today's AMW start time has already passed, so add 24 hours to
sleep_seconds
    if ($sleep_seconds -lt 0) {
        $sleep_seconds=$sleep_seconds + 86400
    }

    $sleep_ts=New-TimeSpan -Seconds ${sleep_seconds}
    $sleep_hms="${sleep_ts}" -replace '^\\d+?\\.\\.'

    echo "Current time is ${current_date_time}. Sleeping for ${sleep_hms},
until daily maintenance start time of ${amw_time}."
    Start-Sleep -Seconds $sleep_seconds
}

function deleteArtifactSnapshotIfExists {
    if (artifactSnapshotExists) {
        $command_del=$(epmautomate.bat deletefile "Artifact Snapshot")
    }
}

function artifactSnapshotExists {
    $filelist=$(epmautomate.bat listfiles)
    if ("$filelist".contains("Artifact Snapshot")) {
        return $true
    }
}

```

```

        else
            return $false
        }
    }

function cloneEnvironment {
    echo "Checking to see if daily maintenance processing has completed ..."
    while ($true) {
        if (artifactSnapshotExists) {
            echo "Daily maintenance processing has completed ..."
            break
        } else {
            echo "Sleeping for 30 seconds before the next check to see if
daily maintenance processing has completed ..."
            Start-Sleep -Seconds 30
        }
    }

    echo "Encrypting target password ..."
    epmautomate.bat encrypt "${target_password}" "oracleKey"
"target_password.epw"

    echo "Cloning environment ..."
    epmautomate.bat cloneEnvironment "${target_username}"
"target_password.epw" "${target_url}" "SnapshotName=Artifact Snapshot"
"UsersAndPreDefinedRoles=${users_and_predefined_roles}" "DataManagement=${
{data_management}}" "appAudit=${app_audit}" "jobConsole=${job_console}"
"storedSnapshotsAndFiles=${stored_snapshots_and_files}"
"dailyMaintenanceStartTime=${daily_maintenance_start_time}"
}

echo "Beginning clone environment script."
echo "Logging into server ..."
epmautomate.bat login ${source_username} ${source_password} ${source_url}
$amwtime=getDailyMaintenanceStartTime
goToSleep ($amwtime)
deleteArtifactSnapshotIfExists
cloneEnvironment
echo "Logging out of server ..."
epmautomate.bat logout
echo "Clone environment script processing has completed."

```

Linux/UNIX のスクリプト

次のスクリプトをコピーすることで、dailyclone.sh を作成します。

```

#!/bin/bash

# Update the following parameters
# -----
epmautomatescript="LOCATION_EPM_AUTOMATE_EXECUTABLE"
javahome="LOCATION_JAVA_HOME"
users_and_predefined_roles="false"
data_management="true"
app_audit="true"
job_console="true"

```

```

stored_snapshots_and_files="false"
daily_maintenance_start_time="true"
# -----

source_username="$1"
source_password="$2"
source_url="$3"
target_username="$4"
target_password="$5"
target_url="$6"

export JAVA_HOME=${javahome}

if [ "$#" -ne 6 ]; then
    echo "Usage: ./dailyclone.sh SOURCE_USERNAME SOURCE_PASSWORD SOURCE_URL
TARGET_USERNAME TARGET_PASSWORD TARGET_URL"
    exit 1
fi

amw_time=""

getDailyMaintenanceStartTime() {
    amw_time=$((${epmautomatescript} getDailyMaintenanceStartTime | cut -d' ' -
f1)
}

goToSleep() {
    current_mdy=$(date -u +%m/%d/%Y)
    current_date_time=$(date -u)
    current_epoch=$(date +%s)
    target_epoch=$(date -d "${current_mdy} ${amw_time}" +%s)
    sleep_seconds=$((target_epoch - current_epoch))

    # Today's AMW start time has already passed, so add 24 hours to
sleep_seconds
    if [[ ${sleep_seconds} -lt 0 ]]
    then
        sleep_seconds=$((sleep_seconds + 86400))
    fi

    sleep_hms=$(date -d@${sleep_seconds} -u +%H:%M:%S)

    echo "Current time is ${current_date_time}. Sleeping for ${sleep_hms},
until daily maintenance start time of ${amw_time}."
    sleep $sleep_seconds
}

deleteArtifactSnapshotIfExists() {
    found=1
    filelist=$((${epmautomatescript} listfiles)
    if [[ ${filelist} == *"Artifact Snapshot"* ]]
    then
        command_del=$((${epmautomatescript} deletefile "Artifact Snapshot")
    fi
}

```

```

artifactSnapshotExists() {
    found=1

    filelist=$((${epmautomatescript} listfiles)
    if [[ ${filelist} == *"Artifact Snapshot"* ]]
    then
        found=0
    else
        found=1
    fi

    echo ${found}
}

cloneEnvironment() {
    local found=1

    while true
    do
        found=$(artifactSnapshotExists)
        if [[ ${found} -eq 0 ]]
        then
            echo "Daily maintenance processing has completed ..."
            break
        else
            echo "Sleeping for 30 seconds before the next check to see if
daily maintenance processing has completed ..."
            sleep 30
        fi
    done

    echo "Encrypting target password ..."
    ${epmautomatescript} encrypt "${target_password}" "oracleKey"
"target_password.epw"

    echo "Cloning environment ..."
    ${epmautomatescript} cloneEnvironment "${target_username}"
"target_password.epw" "${target_url}" "SnapshotName=Artifact Snapshot"
"UsersAndPreDefinedRoles=${users_and_predefined_roles}" "DataManagement=${
{data_management}" "appAudit=${app_audit}" "jobConsole=${job_console}"
"storedSnapshotsAndFiles=${stored_snapshots_and_files}"
"dailyMaintenanceStartTime=${daily_maintenance_start_time}"
}

    echo "Beginning clone environment script."
    echo "Logging into server ..."
    ${epmautomatescript} login ${source_username} ${source_password} ${source_url}
getDailyMaintenanceStartTime
goToSleep
deleteArtifactSnapshotIfExists
cloneEnvironment
    echo "Logging out of server ..."
    ${epmautomatescript} logout
    echo "Clone environment script processing has completed."
}

```

Groovy スクリプト

次のコードをコピーすることで、dailyclone Groovy スクリプトを作成します。

```
// Clone Environment script

// Update the following parameters
String source_username="USERNAME"
String source_password="PASSWORD!"
String source_url="URL OF THE SOURCE ENVIRONMENT"
String target_username="USERNAME"
String target_password="PASSWORD"
String target_url="URL OF THE TARGET ENVIRONMENT"
String snapshot_name="Artifact Snapshot"

// -----

// Do not update the following parameters
String users_and_predefined_roles="false"
String data_management="true"
String app_audit="true"
String job_console="true"
String stored_snapshots_and_files="false"
String daily_maintenance_start_time="true"
// -----

def LogMessage(String message) {
    def date = new Date()
    def sdf = new SimpleDateFormat("MM/dd/yyyy HH:mm:ss")
    println('[ ' + sdf.format(date) + ' ] ' + message);
}

def LogOperationStatus(EpmAutomateStatus opstatus) {
    def returncode = opstatus.getStatus()
    if (returncode != 0){
        LogMessage(opstatus.getOutput())
    }
    LogMessage('return code: ' + returncode)
}

def getDailyMaintenanceStartTime(EpmAutomate automate) {
    LogMessage("Operation: getDailyMaintenanceStartTime")
    EpmAutomateStatus amwtimestatus =
    automate.execute('getDailyMaintenanceStartTime')
    LogOperationStatus(amwtimestatus)
    def amwstring=(amwtimestatus.getOutput())
    def elements=amwstring.split(' ')
    def amwtime=elements[0]
    return amwtime
}

def goToSleep(String amw_time){
    def date = new Date()
    def current_mdy = new SimpleDateFormat("MM/dd/yyyy")
    def current_date_time = new SimpleDateFormat("MM/dd/yyyy HH:mm:ss")
```

```

float current_epoch = date.getTime() / 1000
def pattern = "MM/dd/yyyy HH:mm:ss"
def input = current_mdy.format(date) + " " + amw_time + ":00"
def target_date_time = Date.parse(pattern, input)
float target_epoch = target_date_time.getTime() / 1000
int sleep_seconds = Math.round(target_epoch - current_epoch)

//Today's AMW start time has already passed, so add 24 hours to
sleep_seconds
if (sleep_seconds < 0) {
    sleep_seconds = sleep_seconds + 86400
}

def sleep_milliseconds = sleep_seconds * 1000
LogMessage("Current time is " + current_date_time.format(date) + ".
Sleeping until daily maintenance start time of " + amw_time + ":00.")
sleep(sleep_milliseconds)
}

boolean artifactSnapshotExists(EpmAutomate automate,String snapshot_name) {
    LogMessage("Operation: listfiles")
    EpmAutomateStatus listfilesstatus = automate.execute('listfiles')
    String filelist=(listfilesstatus.getItemsList())
    LogOperationStatus(listfilesstatus)

    if (filelist.contains(snapshot_name)) {
        return true
    } else {
        return false
    }
}

def deleteArtifactSnapshotIfExists(EpmAutomate automate,String snapshot_name){
    if (artifactSnapshotExists(automate,snapshot_name)) {
        LogMessage("Operation: deletefile " + snapshot_name)
        EpmAutomateStatus deletefilestatus =
automate.execute('deletefile',snapshot_name)
        LogOperationStatus(deletefilestatus)
    }
}

def waitForDailyMaintenanceToComplete(EpmAutomate automate,String
snapshot_name) {
    LogMessage("Checking to see if daily maintenance processing has
completed ...")
    while (true) {
        if (artifactSnapshotExists(automate,snapshot_name)) {
            LogMessage("Daily maintenance processing has completed ...")
            break
        } else {
            LogMessage("Sleeping for 30 seconds before the next check to see
if daily maintenance processing has completed ...")
            sleep(30000)
        }
    }
}
}

```

```

LogMessage("Clone environment script processing beginning ...")

EpmAutomate automate = getEpmAutomate()

LogMessage("Operation: login " + source_username + " " + source_password + "
" + source_url)
EpmAutomateStatus status =
automate.execute('login',source_username,source_password,source_url)
LogOperationStatus(status)

String amwtime = getDailyMaintenanceStartTime(automate)
goToSleep (amwtime)
deleteArtifactSnapshotIfExists(automate,snapshot_name)
waitForDailyMaintenanceToComplete(automate,snapshot_name)

LogMessage("Operation: encrypt " + target_password + " oracleKey
target_password.epw")
status =
automate.execute('encrypt',target_password,'oracleKey','target_password.epw')
LogOperationStatus(status)

LogMessage("Operation: cloneEnvironment " + target_username + "
target_password.epw " + target_url + " SnapshotName=" + snapshot_name + "
UsersAndPreDefinedRoles=" + users_and_predefined_roles + " DataManagement=" +
data_management + " appAudit=" + app_audit + " jobConsole=" + job_console + "
storedSnapshotsAndFiles=" + stored_snapshots_and_files + "
dailyMaintenanceStartTime=" + daily_maintenance_start_time)
EpmAutomateStatus cloneenvironmentstatus =
automate.execute('cloneEnvironment',target_username,"target_password.epw",targ
et_url,"SnapshotName=" + snapshot_name,"UsersAndPreDefinedRoles=" +
users_and_predefined_roles,"DataManagement=" + data_management,"appAudit=" +
app_audit,"jobConsole=" + job_console,"storedSnapshotsAndFiles=" +
stored_snapshots_and_files,"dailyMaintenanceStartTime=" +
daily_maintenance_start_time)
LogOperationStatus(cloneenvironmentstatus)

LogMessage("Operation: logout ")
status = automate.execute('logout')
LogOperationStatus(status)

LogMessage ("Clone environment script processing has completed.")

```

スクリプトの実行

1. 前のセクションのいずれかからスクリプトをコピーして、dailyclone.ps1、dailyclone.sh または dailyclone Groovy スクリプトを作成します。
2. スクリプトを更新します:
dailyclone.sh の更新:
 - epmautomatescript (EPM 自動化の実行可能ファイルの場所を指定)。例:
epmautomatescript="/home/utills/EPMAutomate/bin/epmautomate.sh"
 - javahome (EPM 自動化で使用される JDK がインストールされているディレクトリを指定)。例: "/home/user1/jdk1.8.0_191"

dailyclone.ps1 および dailyclone.sh の更新:

- `users_and_predefined_roles`: ユーザーとその事前定義済役割割当てをクローニングするには、この値を `true` に設定します(アクセス制御グループは常にクローニングされます)。
- `data_management`: データ統合レコードをクローニングしない場合は、この値を `false` に設定します。データ統合レコードは、ソース環境とターゲット環境の両方が同じ月次更新の場合、またはターゲット環境がソース環境よりも 1 つ新しい更新である場合のみクローニングできることに注意してください。たとえば、22.01 のデータ管理レコードは、別の 22.01 環境または 22.02 環境のみにクローニングできます。
Narrative Reporting 環境および **Oracle Fusion Cloud Enterprise Data Management** 環境では無視されます。
- `app_audit`: **Planning**、フリーフォームおよび **Enterprise Profitability and Cost Management** アプリケーションの、アプリケーション監査データをクローニングしない場合は、この値を `false` に設定します。
Financial Consolidation and Close および **Tax Reporting** の監査情報は常にクローニングされます。
- `job_console`: ジョブ・コンソール・データをクローニングしない場合は、この値を `false` に設定します。
- `stored_snapshots_and_files`: ソース環境の受信ボックスおよび送信ボックスの最上位レベルにあるフォルダ(サブフォルダはクローニングされません)の内容をクローニングする場合は、この値を `true` に設定します。
- `daily_maintenance_start_time`: クローニングされたターゲット環境のメンテナンス開始時間をソース環境の時間にリセットしない場合は、この値を `false` に設定します。

dailyclone Groovy スクリプトの更新:

- `source_username` は、サービス管理者のユーザー名です。ユーザー役割および事前定義済役割をクローニングするには、アイデンティティ・ドメイン管理者役割が必要です。
 - `source_password` は、`SOURCE_USERNAME` によって識別されるユーザーのパスワードです。
 - `source_url` は、クローニングする環境の URL です。
 - `target_username` は、サービス管理者のユーザー名です。ユーザー役割および事前定義済役割をクローニングするには、アイデンティティ・ドメイン管理者役割が必要です。
 - `target_password` は、`TARGET_USERNAME` で識別されるユーザーのパスワードです。
 - `target_url` は、ターゲット環境の URL です。
3. `dailyclone.ps1`、`dailyclone.sh` または `dailyclone Groovy` スクリプトを実行します:
- #### **Windows**
- コマンド・ウィンドウで、`dailyclone.ps1` が格納されているフォルダに移動します。
 - 次のコマンドを実行します: `./dailyclone.ps1 SOURCE_USERNAME SOURCE_PASSWORD SOURCE_URL TARGET_USERNAME TARGET_PASSWORD TARGET_URL`。次の表で、これらの値の説明を参照してください。

Table 3-7 パラメータの説明

パラメータ	説明
SOURCE_USERNAME	サービス管理者のユーザー名。さらに、ユーザー役割および事前定義済役割をクローニングするには、このユーザーにアイデンティティ・ドメイン管理者役割が必要です。
SOURCE_PASSWORD	SOURCE_USERNAME によって識別されるユーザーのパスワード。
SOURCE_URL	クローニングする環境の URL。
TARGET_USERNAME	ターゲット環境のサービス管理者のユーザー名。さらに、ユーザー役割および事前定義済役割をクローニングするには、このユーザーにアイデンティティ・ドメイン管理者役割が必要です。
TARGET_PASSWORD	TARGET_USERNAME によって識別されるユーザーのパスワード。
TARGET_URL	ターゲット環境の URL。

Linux/UNIX

- UNIX または Linux シェルで、dailyclone.sh が格納されているディレクトリに移動します。
- 次のコマンドを実行します: `./dailyclone.sh SOURCE_USERNAME SOURCE_PASSWORD SOURCE_URL TARGET_USERNAME TARGET_PASSWORD TARGET_URL`。前の表で、これらの値の説明を参照してください。

Groovy

EPM 自動化をインストールしないコマンドの実行を参照してください。

環境からの不要なファイルの削除

これらのスクリプトを使用して、環境から不要なファイルを削除します。

これらのスクリプトでは、次のステップが実行されます:

- 環境にサインインします。
- 環境のファイルとスナップショットをリストします。
- `input.properties` で指定されたファイルを削除します。
- サインアウトします。

Windows のサンプル・スクリプト

次のスクリプトをコピーして、`removeUnnecessaryFiles.ps1` という名前のファイルを作成します。それをローカル・ディレクトリに保存します。

```
$inputproperties = ConvertFrom-StringData(Get-Content ./input.properties -raw)
$username="$($inputproperties.username) "
$passwordfile="$($inputproperties.passwordfile) "
$serviceURL="$($inputproperties.serviceURL) "
$file1="$($inputproperties.file1) "
$file2="$($inputproperties.file2) "

epmautomate login ${username} ${passwordfile} ${serviceURL}
epmautomate listfiles
epmautomate deletefile ${file1}
```

```
epmautomate deletefile ${file2}
epmautomate logout
```

Linux/UNIX のサンプル・スクリプト

次のスクリプトをコピーして、removeUnnecessaryFiles.sh という名前のファイルを作成します。それをローカル・ディレクトリに保存します。

```
#!/bin/bash
. ./input.properties
export JAVA_HOME=${javahome}
${epmautomatescript} login "${username}" "${passwordfile}" "${serviceURL}"
${epmautomatescript} listfiles
${epmautomatescript} deletefile "${file1}"
${epmautomatescript} deletefile "${file2}"
${epmautomatescript} logout
```

input.properties ファイルの作成

removeUnnecessaryFiles スクリプトを実行するには、input.properties ファイルを作成し、環境の情報でファイルを更新します。removeUnnecessaryFiles.ps1 または removeUnnecessaryFiles.sh が格納されているディレクトリにファイルを保存します。

Windows

```
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
file1=FILE_NAME
file2=FILE_NAME
```

Linux/UNIX

```
javahome=JAVA_HOME
epmautomatescript=EPM_AUTOMATE_LOCATION
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
file1=FILE_NAME
file2=FILE_NAME
```

表 3-8 input.properties のパラメータ

パラメータ	説明
javahome	JAVA_HOME の場所。Linux/UNIX の場合のみ。
epmautomatescript	EPM 自動化の実行可能ファイル(epmautomate.sh)の絶対パス。Linux/UNIX の場合のみ。
username	アイデンティティ・ドメイン管理者の役割も保持するサービス管理者のユーザー名。
password	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
serviceURL	スナップショットを生成する環境の URL。

表 3-8 (続き) input.properties のパラメータ

パラメータ	説明
file1 および file2	環境から削除するファイルまたはスナップショットの名前。ファイルが送信ボックスにない場合は、ファイルのパスおよび名前を指定します。

スクリプトの実行

1. 前述の項のスクリプトをコピーして、removeUnnecessaryFiles.ps1 または removeUnnecessaryFiles.sh を作成します。
2. input.properties ファイルを作成して、removeUnnecessaryFiles スクリプトが配置されているディレクトリに保存します。このファイルのコンテンツは、オペレーティング・システムによって異なります。[input.properties ファイルの作成](#)を参照してください。このディレクトリの書き込み権限があることを確認してください。Windows の場合、スクリプトを実行できるように、「**管理者として実行**」オプションを使用して PowerShell を開始する必要があります。
3. スクリプトを起動します。
 - **Windows PowerShell:** removeUnnecessaryFiles.ps1 を実行します。
 - **Linux/UNIX:** ./removeUnnecessaryFiles.sh を実行します。

環境からのファイルの検索およびダウンロード

この項のサンプル・スクリプトを使用し、テキスト文字列をワイルドカードとして使用して、Oracle Fusion Cloud Enterprise Performance Management 環境から 1 つ以上のファイルをダウンロードするプロセスを自動化します。

次のスクリプトでは、FILENAME パラメータの値として指定する文字列を、listfiles コマンドを使用して表示されたファイル名と照合してから、文字列に一致するファイルを自動的にダウンロードすることができます。

必ず適切な検索文字列を FILENAME パラメータに割り当ててください。たとえば、FILENAME="Scheduler Output/epm"は、ユーザー環境での listfiles コマンド出力のファイル名に対して Scheduler Output/epm という文字列に一致して、ダウンロードするファイルが特定されます。

このスクリプトを実行するための入力パラメータは、username、password または password_file、および service_url です。

ノート:

パスワードに特殊文字が含まれている場合は、[特殊文字の処理](#)を参照してください。

Windows

```
@echo off
setlocal EnableExtensions EnableDelayedExpansion
```

```

set USERNAME="username"
set PASSWORD="password"
set URL="url"

call epmautomate login %USERNAME% %PASSWORD% %URL%
set FILENAME="Scheduler Output/epm"
for /f "tokens=*" %%i in ('epmautomate listfiles ^| findstr /b /r /c:"^
*%FILENAME%" ') do (
    call epmautomate downloadfile "%i"
)
call epmautomate logout
endlocal

```

Linux/UNIX

```

#!/bin/sh
USERNAME="username"
PASSWORD="password"
URL="url"

./epmautomate.sh login $USERNAME $PASSWORD $URL
FILENAME='Scheduler Output/epm'
#echo $FILENAME
./epmautomate.sh listfiles | grep "^ $FILENAME" | while read -r line ; do
echo "Processing $line"
./epmautomate.sh downloadfile "$line"
done
./epmautomate.sh logout

```

監査用の古い Cloud EPM 環境の再作成

この項のスクリプトを使用して、Oracle Fusion Cloud Enterprise Performance Management 環境の最新のスナップショットのライブラリを保持するためのセルフサービス・ソリューションを作成します。最新のスナップショットのライブラリをアップグレードおよび保持する目的に特化した環境が必要です。

Cloud EPM は、1 回の月次サイクルのみについてスナップショットの互換性をサポートしています。メンテナンス・スナップショットは、テスト環境から本番環境に移行することも、この逆に移行することもできます。ただし、一部の顧客は、監査要件のために、最新の環境で複数年のスナップショットを復元し、短時間のうちにアプリケーションにアクセスすることが必要な場合があります。

このスクリプトは、使用可能なスナップショットを変換し、Cloud EPM の最新のパッチ・レベルとの互換性を確保するために、毎月 1 回実行されるようスケジュールする必要があります。本番環境内のすべての問題を確実に解決するために、月の第 3 金曜日の後にスクリプトを実行することをお勧めします。

ノート:

このスクリプトを使用して、Narrative Reporting、Account Reconciliation および Oracle Fusion Cloud Enterprise Data Management のスナップショットを更新することはできません。

スクリプトの仕組み

顧客によって格納されるスクリプトごとに、アップグレード・スクリプトによって EPM 自動化を使用して次のタスクが実行されます。

1. `input.properties` ファイル内の情報を使用して、環境にログインします
2. `recreate` コマンドを使用して、環境を改善します
3. 環境にスナップショットをインポートします
4. 環境に対して日次メンテナンスを実行します。この結果、スナップショットが Cloud EPM の現在のパッチ・レベルと互換性を持つフォーマットに変換されます。
5. Artifact Snapshot (メンテナンス・スナップショット)をフォルダにダウンロードします。snapshots/18.05 からスナップショットをアップロードすることによって 18.05 環境を再作成した場合、Artifact Snapshot は snapshots/18.06 にダウンロードされます。
6. 古い環境の再作成の結果を電子メール・アドレス(指定してある場合)に送信します。

スクリプトの実行

1. `input.properties` ファイルを作成し、環境の情報を使用して更新します。ファイルをローカル・ディレクトリに保存します。このディレクトリは、ここでは `parentsnapshotdirectory` と呼びます。このファイルのコンテンツは、オペレーティング・システムによって異なります。このディレクトリの書込み権限があることを確認してください。Windows の場合、スクリプトを実行できるよう、「**管理者として実行**」オプションを使用して PowerShell を開始する必要があります。
2. `upgradeSnapshots.ps1` (Windows PowerShell)または `upgradeSnapshots.sh` (Linux/UNIX) スクリプトを作成し、`input.properties` が格納されている `parentsnapshotdirectory` 内に保存します。
3. サブディレクトリ(`snapshots` など)を `parentsnapshotdirectory` 内に作成します。
4. 前の手順で作成したディレクトリ(`snapshots`)内で、Cloud EPM の現在のパッチ・レベルとの互換性を確保するために変換する月次スナップショットのサブディレクトリを作成します。YY.MM 形式でディレクトリに名前を付けます。たとえば、2018 年 5 月のスナップショットを格納するためのディレクトリの場合は、18.05 とします。
5. スナップショットを適切なサブディレクトリにコピーします。たとえば、2018 年 5 月のスナップショットは `snapshots/18.05` にコピーします。
6. スクリプトを起動します。
 - Linux/UNIX: `./upgradeSnapshots.sh` を実行します。
 - Windows PowerShell: `upgradeSnapshots.ps1` を実行します。

Windows

ここでのスクリプトをコピーすることにより、`input.properties` および `upgradeSnapshots.ps1` スクリプトを作成します。

`input.properties` の作成

```
username=exampleAdmin
userpassword=examplePassword
serviceurl=exapleURL
proxyserverusername=proxyServerUserName
proxyserverpassword=proxyPassword
proxyserverdomain=proxyDoamin
parentsnapshotdirectory=C:/some_directory/snapshots
emailtoaddress=exampleAdmin@oracle.com
```

`input.properties` の更新

ノート:

`authentication at proxy server` が **Windows** のネットワーク環境に対して有効でない場合、プロパティ `proxyserverusername`、`proxyserverpassword` および `proxyserverdomain` を `input.properties` ファイルから削除します。

表 3-9 `input.properties` のパラメータ

パラメータ	説明
<code>username</code>	サービス管理者のユーザー名
<code>userpassword</code>	サービス管理者のパスワード。
<code>serviceurl</code>	このアクティビティに使用する環境の URL。
<code>proxyserverusername</code>	インターネットへのアクセスを制御するプロキシ・サーバーとの安全なセッションを認証するユーザー名。
<code>proxyserverpassword</code>	プロキシ・サーバーに対してユーザーを認証するパスワード。
<code>proxyserverdomain</code>	プロキシ・サーバーに定義されているドメインの名前。
<code>parentsnapshotdirectory</code>	処理対象のスナップショットが格納されるディレクトリの親ディレクトリとして使用するディレクトリの絶対パス。ディレクトリの区切りとしてスラッシュ(/)を使用してください。
<code>emailtoaddress</code>	オプションで、古い環境の再作成の結果が送信される電子メール・アドレス。この値が指定されている場合のみ、結果が電子メールで送信されます。 例: <code>john.doe@example.com</code>

ノート:

パスワードに特殊文字が含まれている場合は、[特殊文字の処理](#)を参照してください。

upgradeSnapshots.ps1 の作成

このサンプル・スクリプトを使用して、upgradeSnapshots.ps1 を作成します

```
# Script for recreating an old Cloud EPM environment

# read in key/value pairs from input.properties file
$inputproperties=ConvertFrom-StringData(Get-Content ./input.properties -raw)

# Global variables
$parentsnapshotdirectory="$($inputproperties.parentsnapshotdirectory)"
$username="$($inputproperties.username)"
$userpassword="$($inputproperties.userpassword)"
$serviceurl="$($inputproperties.serviceurl)"
$proxyserverusername="$($inputproperties.proxyserverusername)"
$proxyserverpassword="$($inputproperties.proxyserverpassword)"
$proxyserverdomain="$($inputproperties.proxyserverdomain)"
$emailtoaddress="$($inputproperties.emailtoaddress)"
$operationmessage="EPM Automate operation:"
$operationfailuremessage="EPM Automate operation failed:"
$operationsuccessmessage="EPM Automate operation completed successfully:"
$epmautomatescript="epmautomate.bat"

$workingdir="$pwd"
$logdir="$workingdir/logs/"
$logfile="$logdir/epmautomate-upgradesnapshots.log"

function LogMessage
{
    $message=$args[0]
    $_mydate=$(get-date -f dd_MM_yy_HH_mm_ss)

    echo "[$_mydate] $message" >> $logfile
}

function LogAndEchoMessage
{
    $message=$args[0]
    $_mydate=$(get-date -f dd_MM_yy_HH_mm_ss)

    echo "[$_mydate] $message" | Tee-Object -Append -FilePath $logfile
}

function LogOutput
{
    $_mydate=$(get-date -f dd_MM_yy_HH_mm_ss)
    $op=$args[0]
    $opoutput=$args[1]
    $returncode=$args[2]

    #If error
    if ($returncode -ne 0) {
        $failmessage="[$_mydate] $operationfailuremessage $op"
        LogMessage $failmessage
        LogMessage $opoutput
        LogMessage "return code: $returncode"
    }
}
```

```

    } else {
        $successmessage="[$_mydate] $operationsuccessmessage $op"
        LogMessage $successmessage
        LogMessage $opoutput
        LogMessage "return code: $returncode"
    }
}

function ExecuteCommand
{
    $op=$args[0]
    $epmautomatecall="$epmautomatescript $op"
    $date=$(get-date -f dd_MM_yy_HH_mm_ss)

    LogMessage "$operationmessage $epmautomatecall"
    $operationoutput=iex "& $epmautomatecall" >> $logfile 2>&1
    LogOutput $op $operationoutput $LastExitCode
}

function ProcessCommand
{
    $command=$args[0]
    $date=$(get-date -f dd_MM_yy_HH_mm_ss)

    if (!(([string]::IsNullOrEmpty($command))) {
        if (!$command.StartsWith("#")) {
            ExecuteCommand $command
        }
    }
}

function Init
{
    $logdirexists=Test-Path $logdir
    if (!$logdirexists) {
        mkdir $logdir 2>&1 | out-null
    }

    # removing existing epmautomate debug logs
    rm ./*.log

    $logfileexists=Test-Path $logfile
    # remove existing log file
    if ($logfileexists) {
        rm $logfile
    }
}

function GetNextDate
{
    $latestyearmonth=$args[0]
    LogMessage "latest year.month: $latestyearmonth"
    $latestyear,$latestmonth=$latestyearmonth.split('\.')
    LogMessage "latest year: $latestyear"
    LogMessage "latest month: $latestmonth"
    $intlatelyear=[int]$latestyear
}

```

```

$intlATESTmonth=[int]$latestmonth

if ($intlATESTmonth -eq 12) {
    $intnextmonth=1
    $intnextyear=$intlATESTyear+1
} else {
    $intnextmonth=$intlATESTmonth+1
    $intnextyear=$intlATESTyear
}

$nextyear="{0:D2}" -f $intnextyear
$nextmonth="{0:D2}" -f $intnextmonth

echo "$nextyear.$nextmonth"
}

function ProcessSnapshot
{
    $snapshotfile=$args[0]
    LogMessage "snapshotfile: $snapshotfile"
    $nextdate=$args[1]
    LogMessage "nextdate: $nextdate"
    $snapshotfilename=$snapshotfile.split('/')[-1]
    LogMessage "snapshotfilename: $snapshotfilename"
    $snapshotname=$snapshotfilename.split('.')[0]
    LogMessage "snapshotname: $snapshotname"

    ProcessCommand
    "login $username $userpassword $serviceurl $proxyserverusername $proxyserverpa
ssword $proxyserverdomain"
    ProcessCommand "recreate -f"
    ProcessCommand "uploadfile $snapshotfile"
    ProcessCommand "importsnapshot $snapshotname"
    ProcessCommand "runDailyMaintenance skipNext=true -f"
    ProcessCommand "downloadfile 'Artifact Snapshot'"
    ProcessCommand "deletefile $snapshotname"
    ProcessCommand "logout"

    $nextdatedirexists=Test-Path $parentsnapshotdirectory/$nextdate
    if (!$nextdatedirexists) {
        mkdir $parentsnapshotdirectory/$nextdate 2>&1 | out-null
    }

    LogMessage "Renaming 'Artifact Snapshot.zip' to $snapshotname.zip and
moving to $parentsnapshotdirectory/$nextdate"
    mv $workingdir/'Artifact Snapshot.zip' $workingdir/$snapshotname.zip
>> $logfile 2>&1
    mv $workingdir/$snapshotname.zip $parentsnapshotdirectory/$nextdate
>> $logfile 2>&1
}

function callSendMail
{
    $logfile=$logfile -replace "\\\"", "/"
    $elements=$logfile.split('/')
    $logfile=$elements[-1]

```

```

        if (${emailtoaddress} -match "@") {
            epmautomate.bat login ${username} ${userpassword} ${serviceurl}
            epmautomate.bat uploadFile "$logfile"
            epmautomate.bat sendMail $emailtoaddress "Recreating An Old Cloud EPM
Environment results" Body="The results of recreating an old Cloud EPM
Environment are attached." Attachments=$logfile
            epmautomate.bat deleteFile "$logfile"
            epmautomate.bat logout
        }
    }

#----- main body of processing
date
Init
LogAndEchoMessage "Starting upgrade snapshots processing"
$snapshotdirs=@(Get-ChildItem -Directory "$parentsnapshotdirectory" -name)
LogMessage "snapshot directories: $snapshotdirs"
$latestreleasedate=$snapshotdirs[-1]
LogMessage "latest release date: $latestreleasedate"
$latestreleasesnapshotdir="$parentsnapshotdirectory/$latestreleasedate"
LogMessage "latest release snapshot dir: $latestreleasesnapshotdir"
$nextdate=$(GetNextDate "$latestreleasedate")
$snapshotfiles=@(Get-ChildItem -File "$latestreleasesnapshotdir")
if ($snapshotfiles.length -eq 0) {
    LogAndEchoMessage "No snapshot files found in
directory $latestreleasesnapshotdir. Exiting script."
    exit
}
foreach ($snapshotfile in $snapshotfiles) {
    LogAndEchoMessage "Processing snapshotfile: $snapshotfile"
    ProcessSnapshot $latestreleasesnapshotdir/$snapshotfile $nextdate
}
LogAndEchoMessage "Upgrade snapshots processing completed"
date
callSendMail

```

Linux/UNIX

次のスクリプトをコピーして upgradeSnapshots.sh および input.properties を作成します。

Linux/UNIX の input.properties の作成

ノート:

インターネットにアクセスするためにプロキシ・サーバーを使用するようネットワークが構成されていない場合、input.properties ファイルからプロパティ proxyserverusername、proxyserverpassword および proxyserverdomain を削除します。

```

username=exampleAdmin
userpassword=examplePassword

```

```
serviceurl=exapleURL
proxyserverusername=
proxyserverpassword=
proxyserverdomain=
jdkdir=/home/user1/jdk160_35
epmautomatescript=/home/exampleAdmin/epmautomate/bin/epmautomate.sh
parentsnapshotdirectory=/home/exampleAdmin/some_directory/snapshots
emailtoaddress=exampleAdmin@oracle.com
```

input.properties の更新

表 3-10 input.properties のパラメータ

パラメータ	説明
username	サービス管理者のユーザー名
userpassword	サービス管理者のパスワード。
serviceurl	このアクティビティに使用する環境の URL。
proxyserverusername	インターネットへのアクセスを制御するプロキシ・サーバーとの安全なセッションを認証するユーザー名。
proxyserverpassword	プロキシ・サーバーに対してユーザーを認証するパスワード。
proxyserverdomain	プロキシ・サーバーに定義されているドメインの名前。
jdkdir	JAVA_HOME の場所。
epmautomatescript	EPM 自動化の実行可能ファイル(epmautomate.sh)の絶対パス。
parentsnapshotdirectory	処理対象のスナップショットが格納されるディレクトリの親ディレクトリとして使用するディレクトリの絶対パス。
emailtoaddress	オプションで、古い環境の再作成の結果が送信される電子メール・アドレス。

ノート:

パスワードに特殊文字が含まれている場合は、[特殊文字の処理](#)を参照してください。

upgradeSnapshots.sh の作成

このサンプル・スクリプトを使用して、upgradeSnapshots.sh を作成します

```
#!/bin/sh

. ./input.properties
workingdir=$(pwd)
logdir="${workingdir}/logs"
logfile=epmautomate-upgradesnapshots.log
operationmessage="EPM Automate operation:"
operationfailuremessage="EPM Automate operation failed:"
operationsuccessmessage="EPM Automate operation completed successfully:"
logdebugmessages=true

if [ ! -d ${jdkdir} ]
then
```

```

    echo "Could not locate JDK/JRE. Please set value for "jdkdir" property in
input.properties file to a valid JDK/JRE location."
    exit
fi

if [ ! -f ${epmautomatescript} ]
then
    echo "Could not locate EPM Automate script. Please set value for
"epmautomatescript" property in the input.properties file."
    exit
fi

export JAVA_HOME=${jdkdir}

debugmessage() {
    # logdebugmessages is defined (or not) in testbase input.properties
    if [ "${logdebugmessages}" = "true" ]
    then
        logmessage "$1"
    fi
}

logmessage()
{
    local message=$1
    local _mydate=$(date)

    echo "[$_mydate] ${message}" >> "$logdir/$logfile"
}

echoandlogmessage()
{
    local message=$1
    local _mydate=$(date)

    echo "[$_mydate] ${message}" | tee -a ${logdir}/${logfile}
}

logoutput()
{
    date=`date`
    op="$1"
    opoutput="$2"
    returncode="$3"

    #If error
    #if grep -q "EPMAT-" <<< "$2"
    if [ $returncode -ne 0 ]
    then
        failmessage="[${date}] ${operationfailuremessage} ${op}"
        logmessage "${failmessage}"
        logmessage "${opoutput}"
        logmessage "return code: ${returncode}"
    else
        successmessage="${operationsuccessmessage} ${op}"
        logmessage "${successmessage}"
    fi
}

```

```

        logmessage "${opoutput}"
        logmessage "return code: ${returncode}"
    fi
}

getLatestReleaseSnapshotDir()
{
    local snapshotdirs=$(find ${parentsnapshotdirectory} -type d | sort)
    debugmessage "snapshot directories: ${snapshotdirs}"
    local latestreleasesnapshotdir=$(echo ${snapshotdirs} | rev | cut -
d' ' -f1 | rev)
    debugmessage "latest release snapshot dir: ${latestreleasesnapshotdir}"
    echo "${latestreleasesnapshotdir}"
}

getNextDate()
{
    local thisyearmonth=$1
    local thisyear=$(echo ${thisyearmonth} | cut -d'.' -f1)
    local thismonth=$(echo ${thisyearmonth} | cut -d'.' -f2)

    intthismonth=$(bc <<< ${thismonth})
    intthisyear=$(bc <<< ${thisyear})

    if [ ${intthismonth} -eq 12 ]
    then
        local intnextmonth=1
        local intnextyear=$((intthisyear+1))
    else
        local intnextmonth=$((intthismonth+1))
        local intnextyear=${intthisyear}
    fi

    nextmonth=$(printf "%02d\n" ${intnextmonth})
    nextyear=$(printf "%02d\n" ${intnextyear})

    debugmessage "next date: ${nextyear}.${nextmonth}"

    echo "${nextyear}.${nextmonth}"
}

init()
{
    if [ ! -d "$logdir" ]
    then
        mkdir $logdir
    fi

    # removing existing epmautomate debug logs
    if ls /*.log >/dev/null 2>&1
    then
        rm /*.log
    fi

    # remove existing log files
    if [ -f "${logdir}/${logfile}" ]

```

```

    then
        rm ${logdir}/${logfile}
    fi
}

processCommand()
{
    op="$1"
    date=`date`

    logmessage "$operationmessage $op"
    operationoutput=`eval "$epmautomatescript $op"`
    logoutput "$op" "$operationoutput" "$?"
}

processSnapshot()
{
    local snapshotfile="$1"
    local nextdate="$2"
    local snapshotname=$(echo "${snapshotfile}" | rev | cut -d'/' -f1 | rev |
cut -d'.' -f1)

    processCommand "login ${username} ${userpassword} ${serviceurl} $
(proxyserverusername) ${proxyserverpassword}"
    processCommand "recreate -f"
    processCommand "uploadfile ${snapshotfile}"
    processCommand "importsnapshot \"${snapshotname}\""
    processCommand "runDailyMaintenance skipNext=true -f"
    processCommand "downloadfile \"Artifact Snapshot\""
    processCommand "deletefile \"${snapshotname}\""
    processCommand "logout"

    if [ ! -d ${parentsnapshotdirectory}/${nextdate} ]
    then
        mkdir ${parentsnapshotdirectory}/${nextdate}
    fi
    runDailyMaintenance -f
    logmessage "Renaming \"Artifact Snapshot.zip\" to ${snapshotname}.zip and
moving to ${parentsnapshotdirectory}/${nextdate}"
    mv "${workingdir}/Artifact Snapshot.zip" "${workingdir}/${
snapshotname}.zip" >> "$logdir/$logfile" 2>&1
    mv "${workingdir}/${snapshotname}.zip" ${parentsnapshotdirectory}/${
nextdate} >> "$logdir/$logfile" 2>&1
}

callSendMail() {
    if [[ "${emailtoaddress}" == *@* ]]
    then
        ${epmautomatescript} login ${username} ${userpassword} ${serviceurl}
        ${epmautomatescript} uploadFile "$logdir/$logfile"
        ${epmautomatescript} sendMail $emailtoaddress "Recreating An Old
Cloud EPM Environment results" Body="The results of recreating an old Cloud
EPM Environment are attached" Attachments=$logfile
        ${epmautomatescript} deleteFile "$logfile"
        ${epmautomatescript} logout
    fi
}

```

```

    fi
}

#----- main body of processing
date
echoandlogmessage "Starting upgrade snapshots processing"
init
latestreleasesnapshotdir=$(getLatestReleaseSnapshotDir)
latestreleasedate=$(echo "${latestreleasesnapshotdir}" | rev | cut -d'/' -f1
| rev)
debugmessage "latest release date: ${latestreleasedate}"
nextdate=$(getNextDate ${latestreleasedate})

snapshotfiles=$(find ${latestreleasesnapshotdir} -type f -name \*.zip | tr
"\n" "|")
if [ ${#snapshotfiles} -eq 0 ]
then
    echoandlogmessage "No snapshot files found in directory $
{latestreleasesnapshotdir}"
fi

IFS="|"
for snapshotfile in $snapshotfiles
do
    echoandlogmessage "Processing snapshotfile: ${snapshotfile}"
    processSnapshot ${snapshotfile} ${nextdate}
done
unset IFS
echoandlogmessage "Upgrade snapshots processing completed."
callSendMail

```

データベース・アクセスの監査およびコンプライアンスの自動化

この項では、PowerShell および Bash シェル・スクリプトを使用し、EPM 自動化コマンドを利用して、手動データベース・アクセス全体の監査およびコンプライアンス・データを収集します。

これらのスクリプトを使用して、次のタスクを完了できます：

- その日のアクティビティ・レポートをダウンロードします
- レポートを解析して、環境に対する手動データベース・アクセスがレポートされているかどうかを判別します
- スクリプトを実行するディレクトリに対して、./reports/dataAccessAuditReport.txt を作成します。このレポートには、データベース・アクセスの時間と、実行された SQL コマンドがリストされます。これは累積ファイルで、最新の情報が最上部に表示されます。使用可能な情報は次のとおりです：
 - レポートが生成された日時
 - データベース・アクセスの詳細(使用可能な場合)。サービス・リクエストなしのデータベース・アクセスとサービス・リクエストありのデータベース・アクセスが別々のセクションにリストされます。

アクティビティ・レポートに手動データベース・アクセスがレポートされていない場合、レポートには No SQL statements executed と記載されます。

- オプションで、指定された電子メール・アドレスにレポートを送信します。

データ・アクセスの監査およびコンプライアンスを自動化するには:

1. 後続の項からいずれかのスクリプトをファイルにコピーし、ファイル・システムに保存します。ファイルに `parseActivityReport.ps1` (Windows の場合、[PowerShell スクリプト \(parseActivityReport.ps1\)](#)を参照)または `parseActivityReport.sh` (Linux/UNIX の場合、[Bash シェル・スクリプト \(parseActivityReport.sh\)](#)を参照)という名前を付けます。
2. **Windows のみ:** 次のスクリプトをファイルにコピーして、`parseActivityReport.bat` という名前のバッチ・ファイルを作成します。`parseActivityReport.ps1` が格納されているディレクトリにファイルを保存します。

```
@echo off
set paramRequiredMessage=Syntax: parseActivityReport.bat USERNAME PASSWORD/
PASSWORD_FILE URL [REPORT_EMAIL_TO_ADDRESS]

if "%~1" == "" (
    echo User Name is missing.
    echo %paramRequiredMessage%
    exit /b 1
)
if "%~2" == "" (
    echo Password or Password_File is missing.
    echo %paramRequiredMessage%
    exit /b 1
)
if "%~3" == "" (
    echo URL is missing.
    echo %paramRequiredMessage%
    exit /b 1
)

PowerShell.exe -File parseActivityReport.ps1 %*
```

3. `parseActivityReport.bat` (Windows)または `parseActivityReport.sh` (Linux/UNIX)を変更して、次の表にあるパラメータの値を設定します。

表 3-11 スクリプトに含める変数の値

変数	説明
<code>epmuser</code>	サービス管理者のユーザー名 例: Windows: <code>set epmuser="jDoe"</code> Linux/UNIX: <code>epmuser="jDoe"</code>

表 3-11 (続き) スクリプトに含める変数の値

変数	説明
epmpassword	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの場所暗号化されたパスワード・ファイルの作成の詳細は、 encrypt コマンドを参照してください。 パスワードに特殊文字が含まれている場合は、 特殊文字の処理 を参照してください。 例: Windows: set epmpassword = "Example" Linux/UNIX: epmpassword="Example"
epmurl	Oracle Fusion Cloud Enterprise Performance Management 環境の URL。 例: Windows: set epmurl="https://example.oraclecloud.com" Linux/UNIX: epmurl="https://example.oraclecloud.com"
report_email_to_addresses	オプションで、レポートが送信される電子メール・アドレス。この値が指定されている場合のみ、レポートが電子メールで送信されます。 例: john.doe@example.com

4. **parseActivityReport.sh の場合のみ:** 次の値がシステムに正確に設定されていることを確認します:

- JAVA_HOME
- epmautomatescript ディレクティブの値の更新による epmautomatescript.sh の場所

5. オペレーティング・システムで使用可能なスケジューラを使用して、parseActivityReport.bat (parseActivityReport.ps1 を実行する)または parseActivityReport.sh を毎日 1 回実行するようにスケジュールします。[スクリプトの実行の自動化](#)を参照してください。

PowerShell スクリプト(parseActivityReport.ps1)

```
# Parse Activity Report script

$epmuser=$args[0]
$epmpassword=$args[1]
$epmurl=$args[2]
$reportemailtoaddress=$args[3]

$logdir="./logs"
$logfile="$${logdir}/data_access.log"
$reportdir="./reports"
$reportfile="$${reportdir}/dataAccessAuditReport.txt"
$matchfile="$${reportdir}/matchfile.txt"
$nosrfile="$${reportdir}/data_access_nosr.csv"
$srfile="$${reportdir}/data_access_sr.csv"
$aprfilelist="$${reportdir}/aprfilelist.txt"
$activityreportfilelist="$${reportdir}/activityreportfiles.txt"
$activityreportregex='apr/[0-9]{4}-[0-9]{2}-[0-9]{2} [0-9]{2}_[0-9]{2}_[0-9]{2}/[0-9]{4}-[0-9]{2}-[0-9]{2} [0-9]{2}_[0-9]{2}_[0-9]{2}.html'

$global:activityreportfile=""
```

```

$NO_SQL_EXECUTED_STATEMENT="No SQL statements executed"
$$SQL_WITH_SR_EXECUTED_STATEMENT="SQL statements executed with an SR"
$$SQL_WITH_NO_SR_EXECUTED_STATEMENT="SQL statements executed without an SR"

function DownloadLatestActivityReport() {
    epmautomate.bat login ${epmuser} ${epmpassword} ${epmurl} >> ${logfile}
    epmautomate.bat listfiles > ${aprfilelist}
    foreach ($line in Get-Content $aprfilelist) {
        if ($line -match $activityreportregex){
            echo "$line" >> $activityreportfilelist
        }
    }
    $global:activityreportfile=Get-Content ${activityreportfilelist} -Tail 1
    $global:activityreportfile=$global:activityreportfile.trim()
    echo " "
    echo "Processing activity report file: $global:activityreportfile" | tee -
a ${logfile}
    epmautomate.bat downloadfile "$global:activityreportfile" >> ${logfile}
    epmautomate.bat logout >> ${logfile}
}

function deleteLine($file, $start, $end) {
    $i = 0
    $start--
    $end--
    (Get-Content $file) | where{
        ($i -lt $start -or $i -gt $end)
        $i++
    } > $file
    #(Get-Content $file)
}

function GenerateCsvs()
{
    $sqlregex='<DIV id="Database">.*?</DIV>'
    $activityreportfilename=Split-Path $global:activityreportfile -leaf

    echo "Creating CSV file: ${matchfile} from data in activityreportfile: $
(activityreportfilename)" >> ${logfile}
    # remove tab and newline characters
    $activityreportexists=Test-Path "$activityreportfilename"
    if ($activityreportexists) {
        (Get-Content "$activityreportfilename") -join ' ' | Set-Content
"$activityreportfilename"
        (Get-Content "$activityreportfilename") -replace "`t", "" | Set-
Content "$activityreportfilename"
    }

    # capture text matching regex
    $string=Get-Content $activityreportfilename
    $ans=$string -match $sqlregex

    if ($ans -eq "True") {
        $Matches.0 > $matchfile
        # remove HTML tags, etc.
    }
}

```

```

        (Get-Content "$matchfile") -replace "<tr", "`n<tr" | Set-Content
"$matchfile"
        (Get-Content "$matchfile") -replace "<tr[^>]*>", "" | Set-Content
"$matchfile"
        (Get-Content "$matchfile") -replace "<th[^>]*>", "" | Set-Content
"$matchfile"
        (Get-Content "$matchfile") -replace "<td[^>]*>", "|" | Set-Content
"$matchfile"
        (Get-Content "$matchfile") -replace "<br>", "" | Set-Content
"$matchfile"
        (Get-Content "$matchfile") -replace "</td>", "" | Set-Content
"$matchfile"
        (Get-Content "$matchfile") -replace "</tr>", "" | Set-Content
"$matchfile"
        (Get-Content "$matchfile") -replace "\s*</table>\s*</DIV>", "" | Set-
Content "$matchfile"
        deleteLine $matchfile 1 2

# create SR, NOSR CSV files
Get-Content $matchfile | ForEach-Object {
    $elements=$_split('|')
    $timeval=$elements[1].Trim()
    $srval=$elements[3].Trim()
    $sqlval=$elements[4].Trim()

    if ($srval -eq "") {
        echo "${timeval}|${sqlval}" >> ${nosrfile}
    } else {
        if ($sqlval -ne "") {
            echo "${srval}|${timeval}|${sqlval}" >> ${srfile}
        }
    }
}

} else { # no SQL statements in activity report
    echo "" >> ${reportfile}
    echo $(date) >> ${reportfile}
    echo "Processing activity report file: $global:activityreportfile"
>> ${reportfile}
    echo "${NO_SQL_EXECUTED_STATEMENT}" | tee -a ${reportfile}
    CleanUp
    EmailReportResults
    exit
}

}

function ReportResults() {
    echo $(date) >> ${reportfile}
    echo "Processing activity report file: $global:activityreportfile" >> $
{reportfile}
    $srfileexists=Test-Path $srfile
    if ($srfileexists) {
        echo "" | tee -a ${reportfile}
        echo "${SQL_WITH_SR_EXECUTED_STATEMENT}" | tee -a ${reportfile}
        echo "SR#          Time          SQL Statement" | tee -a ${reportfile}
        echo "----          ----          -----" | tee -a ${reportfile}
    }
}

```

```

    # Loop through csv file and parse
    Get-Content $srfile | ForEach-Object {
        $elements=$_split('|')
        $srval=$elements[0]
        $timeval=$elements[1]
        $sqlval=$elements[2]
        echo "${srval}    ${timeval}    ${sqlval}" | tee -a ${reportfile}
    }
}

$nosrfileexists=Test-Path $nosrfile
if ($nosrfileexists) {
    echo "" | tee -a ${reportfile}
    echo "${SQL_WITH_NO_SR_EXECUTED_STATEMENT}" | tee -a ${reportfile}
    echo "Time            SQL Statement" | tee -a ${reportfile}
    echo "-----            -----" | tee -a ${reportfile}

    # Loop through csv file and parse
    Get-Content $nosrfile | ForEach-Object {
        $elements=$_split('|')
        $timeval=$elements[0]
        $sqlval=$elements[1]
        echo "${timeval}    ${sqlval}" | tee -a ${reportfile}
    }
}

EmailReportResults
}

function EmailReportResults
{
    $elements=$reportfile.split('/')
    $reportfilename=$elements[2]

    if (${reportemailtoaddress} -match "@") {
        echo "Emailing Activity Report Results" | tee -a ${logfile}
        epmautomate.bat login ${epmuser} ${epmpassword} ${epmurl} >> ${logfile}
        epmautomate.bat uploadFile $reportfile >> ${logfile}
        epmautomate.bat sendMail $reportemailtoaddress "Database Access Audit
Report Results" Body="Database Access Audit Report Results are attached."
Attachments=$reportfilename >> ${logfile}
        epmautomate.bat deleteFile $reportfilename >> ${logfile}
        epmautomate.bat logout >> ${logfile}
    }
}

function Init
{
    $logdirexists=Test-Path $logdir
    if (!$logdirexists) {
        mkdir $logdir 2>&1 | out-null
    }

    $reportdirexists=Test-Path $reportdir
    if (!$reportdirexists) {
        mkdir $reportdir 2>&1 | out-null
    }
}

```

```
}

$logfileexists=Test-Path $logfile
if ($logfileexists) {
    rm $logfile 2>&1 | out-null
}

$matchfileexists=Test-Path $matchfile
if ($matchfileexists) {
    rm $matchfile 2>&1 | out-null
}

$nosrfileexists=Test-Path $nosrfile
if ($nosrfileexists) {
    rm $nosrfile 2>&1 | out-null
}

$srfileexists=Test-Path $srfile
if ($srfileexists) {
    rm $srfile 2>&1 | out-null
}

$aprfilelistexists=Test-Path $aprfilelist
if ($aprfilelistexists) {
    rm $aprfilelist 2>&1 | out-null
}

$activityreportfilelistexists=Test-Path $activityreportfilelist
if ($activityreportfilelistexists) {
    rm $activityreportfilelist 2>&1 | out-null
}
}

function CleanUp
{
    $matchfileexists=Test-Path $matchfile
    if ($matchfileexists) {
        rm $matchfile 2>&1 | out-null
    }

    $aprfilelistexists=Test-Path $aprfilelist
    if ($aprfilelistexists) {
        rm $aprfilelist 2>&1 | out-null
    }

    $activityreportfilelistexists=Test-Path $activityreportfilelist
    if ($activityreportfilelistexists) {
        rm $activityreportfilelist 2>&1 | out-null
    }
}

Init
DownloadLatestActivityReport
GenerateCsvs
ReportResults
CleanUp
```

Bash シェル・スクリプト(parseActivityReport.sh)

```
#!/bin/sh

export JAVA_HOME=/scratch/dteHome/autoWork/jdk1.8.0_191
epmautomatescript=/scratch/dteHome/autoWork/epmautomate/19.11.55/bin/
epmautomate.sh

epmuser="<EPM USER>"
epmpwd="<EPM PASSWORD>"
epmurl="<EPM URL>"
reportemailtoaddress="<EMAIL ADDRESS>"

logdir=./logs
logfile="${logdir}/data_access.log"
reportdir=./reports
reportfile="${reportdir}/dataAccessAuditReport.txt"
nosrfile="${reportdir}/data_access_nosr.csv"
srfile="${reportdir}/data_access_sr.csv"
matchfile="${reportdir}/match.out"
aprfilelist="${reportdir}/aprfilelist.txt"
activityreportfile=""
activityreportregex='apr/[0-9]{4}-[0-9]{2}-[0-9]{2} [0-9]{2}_[0-9]{2}_[0-9]{2}/[0-9]{4}-[0-9]{2}-[0-9]{2} [0-9]{2}_[0-9]{2}_[0-9]{2}.html'

NO_SQL_EXECUTED_STATEMENT="No SQL statements executed".
SQL_WITH_SR_EXECUTED_STATEMENT="SQL statements executed with an SR"
SQL_WITH_NO_SR_EXECUTED_STATEMENT="SQL statements executed without an SR"

cd "$(dirname "$0")"

generateCsvs()
{
    local sqlregex='<DIV id="Database">.*?</DIV>'
    local activityreportfilename=$(echo "${activityreportfile}" | rev | cut -d '/' -f1 | rev)

    echo "Creating CSV file: ${matchfile} from data in activityreportfile: ${activityreportfilename}" >> ${logfile}
    # remove tab and newline characters
    cat "${activityreportfilename}" | tr -d "\t\n\r" > ${matchfile}
    # capture text matching regex
    grep -Po "${sqlregex}" ${matchfile} > ${matchfile}.tmp

    # remove HTML tags, etc.
    sed -e 's/<tr/\n<tr/g' -e 's/<tr[^>]*>//g' -e 's/<th[^>]*>//g' -e 's/<td[^>]*>||/g' -e 's/<br>//g' -e 's|</td>||g' -e 's|</tr>||g' -e 's|[ ]*</table></DIV>||g' -e 's|[ ]*||g' -e 's/[ ]*||g' -e 's/<DIV id="Database">.*?<!-- Print Tables -->\n//g' ${matchfile}.tmp > ${matchfile}

    # create SR, NOSR CSV files
    while read line
    do
        timeval=$(echo "${line}" | cut -d'|' -f2)
        srval=$(echo "${line}" | cut -d'|' -f4)
    done
}

```

```

sqlval=$(echo "${line}" | cut -d'|' -f5)

if [[ "${srval}" == "" ]]
then
    echo "${timeval}|${sqlval}" >> ${nosrfile}
else
    if [[ "${sqlval}" != "" ]]
    then
        echo "${srval}|${timeval}|${sqlval}" >> ${srfile}
    fi
fi
done < ${matchfile}
}

reportResults() {
    echo $(date) >> ${reportfile}
    echo "Processing activity report file: $activityreportfile" >> $
{reportfile}
    if [[ -f ${srfile} ]]
    then
        echo "" | tee -a ${reportfile}
        echo "${SQL_WITH_SR_EXECUTED_STATEMENT}" | tee -a ${reportfile}
        echo "SR#          Time          SQL Statement" | tee -a ${reportfile}
        echo "---          ----          -" | tee -a ${reportfile}
        while read line
        do
            srval=$(echo "${line}" | cut -d'|' -f1)
            timeval=$(echo "${line}" | cut -d'|' -f2)
            sqlval=$(echo "${line}" | cut -d'|' -f3)
            echo "${srval}    ${timeval}    ${sqlval}" | tee -a ${reportfile}
        done < ${srfile}
    fi

    if [[ -f ${nosrfile} ]]
    then
        echo "" | tee -a ${reportfile}
        echo "${SQL_WITH_NO_SR_EXECUTED_STATEMENT}" | tee -a ${reportfile}
        echo "Time          SQL Statement" | tee -a ${reportfile}
        echo "----          --- -" | tee -a ${reportfile}
        while read line
        do
            timeval=$(echo "${line}" | cut -d'|' -f1)
            sqlval=$(echo "${line}" | cut -d'|' -f2)
            echo "${timeval}    ${sqlval}" | tee -a ${reportfile}
        done < ${nosrfile}
    fi

    if [[ ! -f ${srfile} ]] && [[ ! -f ${nosrfile} ]]
    then
        echo "" | tee -a ${reportfile}
        echo "${NO_SQL_EXECUTED_STATEMENT}" | tee -a ${reportfile}
    fi

    emailReportResults
}

```

```

downloadLatestActivityReport() {
    ${epmautomatescript} login ${epmuser} ${epmpwd} ${epmurl} >> ${logfile}
    ${epmautomatescript} listfiles > ${aprfilelist}
    activityreportfile=$(cat ${aprfilelist} | grep -P "$
{activityreportregex}" | tail -n 1 | sed -e 's/^ //' )
    echo " "
    echo "Processing activity report file: ${activityreportfile}" | tee -a $
{logfile}
    ${epmautomatescript} downloadfile "${activityreportfile}" >> ${logfile}
    ${epmautomatescript} logout >> ${logfile}
}

emailReportResults() {
    reportfilename=$(echo "${reportfile}" | cut -d'/' -f3)

    if [[ "${reportemailtoaddress}" == *@"* ]]
    then
        echo "Emailing Activity Report Results" | tee -a ${logfile}
        ${epmautomatescript} login ${epmuser} ${epmpwd} ${epmurl} >> $
{logfile}
        ${epmautomatescript} uploadFile "$reportfile" >> ${logfile}
        ${epmautomatescript} sendMail $reportemailtoaddress "Database Access
Audit Report Results" Body="Database Access Audit Report Results are
attached." Attachments=$reportfilename >> ${logfile}
        ${epmautomatescript} deleteFile "$reportfilename" >> ${logfile}
        ${epmautomatescript} logout >> ${logfile}
    fi
}

checkParams()
{
    if [ -z "$epmuser" ]
    then
        echo "Username is missing."
        echo "Syntax: parseActivityReport.sh USERNAME PASSWORD URL"
        exit 2
    fi

    if [ -z "$epmpwd" ]
    then
        echo "Password is missing."
        echo "Syntax: parseActivityReport.sh USERNAME PASSWORD URL"
        exit 2
    fi

    if [ -z "$epmurl" ]
    then
        echo "URL is missing."
        echo "Syntax: parseActivityReport.sh USERNAME PASSWORD URL"
        exit 2
    fi
}

init()
{
    checkParams
}

```

```
if [ ! -d "${logdir}" ]
then
    mkdir ${logdir}
fi

if [ ! -d "${reportdir}" ]
then
    mkdir ${reportdir}
fi

if [ ! -f "${epmautomatescript}" ]
then
    echo "Cannot locate EPMAutomate script: ${epmautomatescript}. Please
check setting and run script again. Exiting." | tee -a ${logfile}
    exit
fi

if [ -f "${srfile}" ]
then
    rm ${srfile}
fi

if [ -f "${nosrfile}" ]
then
    rm ${nosrfile}
fi

if [ -f "${matchfile}" ]
then
    rm ${matchfile}
fi

if [ -f "${aprfilelist}" ]
then
    rm ${aprfilelist}
fi
}

cleanup()
{
    if [ -f "${matchfile}" ]
    then
        rm ${matchfile}
    fi

    if [ -f "${matchfile}.tmp" ]
    then
        rm ${matchfile}.tmp
    fi

    if [ -f "${aprfilelist}" ]
    then
        rm ${aprfilelist}
    fi
}
```

```
init  
downloadLatestActivityReport  
generateCsvs  
reportResults  
cleanup
```

ユーザーおよび事前定義済役割割当てのレプリケート

この項のスクリプトは、環境のユーザーおよび事前定義済役割割当てを別の環境に移行する際に役立ちます。

スクリプトについて

2つの異なるスクリプトを使用します。1つはユーザーをアイデンティティ・ドメインにレプリケートし、もう1つはユーザーの事前定義済役割割当てをレプリケートします。これらのスクリプトを実行する順序は次のとおりです：

- ユーザーをレプリケートするスクリプト(`replicateusers`)を実行し、すべてのユーザーがターゲット・アイデンティティ・ドメインに作成されたことを確認します。このスクリプトを実行するユーザーには、両方の環境のアイデンティティ・ドメイン管理者とサービス管理者の役割が必要です。
- 役割割当てをレプリケートするスクリプト(`replicatepredefinedroles`)を実行します。

ノート:

- パスワードに特殊文字が含まれている場合は、[特殊文字の処理](#)を参照してください
- この項のスクリプトは、事前定義済役割(サービス管理者、パワー・ユーザー、ユーザーおよび参照者)に対してのみ動作します。

スクリプトの実行

必要なスクリプトおよびバッチ・ファイルの作成の詳細は、次のトピックを参照してください：

- [あるアイデンティティ・ドメインから別のアイデンティティ・ドメインへのユーザーのレプリケート](#)
- [ある環境から別の環境への事前定義済役割割当てのレプリケート](#)

Windows でのステップ

1. `replicateusers.bat`、`replicateusers.ps1`、`replicatepredefinedroles.bat` および `replicatepredefinedroles.ps1` を作成し、書込み権限と実行権限を持つローカル・ディレクトリに保存します。
2. 必要に応じて、ソース環境とターゲット環境およびインターネット・プロキシ・サーバーの情報でバッチ・ファイルを更新します。
3. `replicateusers.bat` を実行すると、`replicateusers.ps1` が実行されます。レプリケートされたユーザーに割り当てるデフォルトのパスワードを、次のようにコマンドライン・パラメータとして指定します：
`replicateusers.bat Pwd_for_users`

パスワードに特殊文字が含まれている場合は、適切なエスケープ文字を使用してください。
[特殊文字の処理](#)を参照してください。

4. replicatepredefinedroles.bat を実行して、ソース環境にあるものと同じ役割割当てを作成します。

Linux/UNIX でのステップ

1. replicateusers.sh および replicatepredefinedroles.sh スクリプトを作成し、書き込み権限と実行権限を持つローカル・ディレクトリに保存します。
2. 必要に応じて、ソース環境とターゲット環境およびインターネット・プロキシ・サーバーの情報で replicateusers.sh および replicatepredefinedroles.sh を更新します。
3. replicateusers.sh を実行します。レプリケートされたユーザーに割り当てるデフォルトのパスワードを、次のようにコマンドライン・パラメータとして指定します:
./replicateusers.sh Pwd_for_users

パスワードに特殊文字が含まれている場合は、適切なエスケープ文字を使用してください。
[特殊文字の処理](#)を参照してください。

4. replicatepredefinedroles.sh スクリプトを実行して、ソース環境にあるものと同じ役割割当てを作成します。

あるアイデンティティ・ドメインから別のアイデンティティ・ドメインへのユーザーのレプリケート

この項のスクリプトを使用して、1つのアイデンティティ・ドメインのユーザーを別のアイデンティティ・ドメインにクローニングします。これらのスクリプトを実行するには、ソース環境とターゲット環境のアイデンティティ・ドメイン管理者の役割とサービス管理者の役割が必要です。

Windows

この項のスクリプトをコピーすることにより、replicateusers.bat および replicateusers.ps1 を作成します。

1. このスクリプトをコピーすることにより、replicateusers.ps1 を作成します:

```
# Replicate users script

param(
    [string]$epmusersource,
    [string]$epmpwdsource,
    [string]$epmurlsource,
    [string]$epmidentitydomainsource,
    [string]$epmuserstarget,
    [string]$epmpwdtarget,
    [string]$epmurltarget,
    [string]$epmidentitydomaintarget,
    [string]$proxyserverusername,
    [string]$proxyserverpassword,
    [string]$proxyserverdomain,
    [string]$userpassword,
    [string]$resetpassword,
    [string]$emailtoaddress
)
```

```

$roleassignmentreport="roleassignmentreport.csv"
$usersreport="users.csv"

echo "Replicate users script started"

# delete existing reports
$roleassignmentreportexists=Test-Path $roleassignmentreport
if ($roleassignmentreportexists) {
    rm $roleassignmentreport 2>&1 | out-null
}

$usersreportexists=Test-Path $usersreport
if ($usersreportexists) {
    rm $usersreport 2>&1 | out-null
}

# epmautomate login Source App as an IDM Admin
echo "Logging into source application at ${epmurlsource}"
epmautomate login ${epmusersource} ${epmpwdsource} ${epmurlsource} $
{epmidentitydomainsource} ${proxyserverusername} ${proxyserverpassword} $
{proxyserverdomain}
echo "Creating role assignment report: ${roleassignmentreport}"
epmautomate roleAssignmentReport ${roleassignmentreport}
if (${emailtoaddress} -match "@") {
    epmautomate.bat sendMail $emailtoaddress "Role assignment report"
    Body="Role assignment report is attached."
    Attachments=$roleassignmentreport}
echo "Downloading role assignment report"
epmautomate downloadfile ${roleassignmentreport}
epmautomate deletefile ${roleassignmentreport}
epmautomate logout

# Create users report
Get-Content ${roleassignmentreport} | ForEach-Object {
    $user=$_split(',') [0]
    $firstname=$_split(',')[1]
    $lastname=$_split(',')[2]
    $email=$_split(',')[3]

    if ($firstname -eq "First Name") {
        return
    } else {
        echo "${firstname},${lastname},${email},${user}" >> ${usersreport}
    }
}

Get-Content -Path "${usersreport}" | Sort-Object -Unique > "$
{usersreport}.tmp"
mv -Force "${usersreport}.tmp" "${usersreport}"
$userheader="First Name,Last Name,Email,User Login"
"${userheader}`r`n" + (Get-Content $usersreport -Raw) | Set-
Content $usersreport

```

```
# epmautomate login Target App as an IDM Admin
echo "Logging into target application at ${epmurltarget}"
epmautomate login ${epmuserstarget} ${epmpwdtarget} ${epmurltarget} $
{epmidentitydomaintarget} ${proxyserverusername} ${proxyserverpassword} $
{proxyserverdomain}
epmautomate deletefile ${usersreport} | Out-Null
echo "Uploading file ${usersreport}"
epmautomate uploadfile ${usersreport}
echo "Adding users"
epmautomate addUsers ${usersreport} userPassword=${userpassword}
resetPassword=${resetpassword}
epmautomate deletefile ${usersreport}
epmautomate logout
rm deletefile*.log | Out-Null
echo "Replicate users script completed"
```

2. このスクリプトをコピーすることにより、replicateusers.bat を作成します:

```
@ECHO OFF
SET thisdir=%~dp0
SET scriptpath=%thisdir%replicateusers.ps1
SET paramRequiredMessage=Syntax: replicateusers.bat "USER_PASSWORD"

REM USER DEFINED VARIABLES
REM -----
set epusersource="<EPM USER FOR SOURCE ENVIRONMENT>"
set epmpwdsourc="<EPM PASSWORD FOR SOURCE ENVIRONMENT>"
set epmurlsource="<EPM URL FOR SOURCE ENVIRONMENT>"
set epmidentitydomainsourc="<EPM IDENTITY DOMAIN FOR SOURCE ENVIRONMENT>"
set epmuserstarget="<EPM USER FOR TARGET ENVIRONMENT>"
set epmpwdtarget="<EPM PASSWORD FOR TARGET ENVIRONMENT>"
set epmurltarget="<EPM URL FOR TARGET ENVIRONMENT>"
set epmidentitydomaintarget="<EPM IDENTITY DOMAIN FOR TARGET ENVIRONMENT>"
set proxyserverusername="<PROXY SERVER USER NAME>"
set proxyserverpassword="<PROXY SERVER PASSWORD>"
set proxyserverdomain="<PROXY SERVER DOMAIN>"
set resetpassword=false
set emailtoaddress="<EMAIL_TO_ADDRESS>"
REM -----

if "%~1" == "" (
    echo USER_PASSWORD is missing. This is used to set the default
password for the replicated users.
    echo %paramRequiredMessage%
    exit /b 1
)

PowerShell -NoProfile -ExecutionPolicy Bypass -Command "& '%scriptpath%' -
epusersource '%epusersource%' -epmpwdsourc '%epmpwdsourc%' -
epmurlsource '%epmurlsource%' -epmidentitydomainsourc
'%epmidentitydomainsourc%' -epmuserstarget '%epmuserstarget%' -epmpwdtarget
'%epmpwdtarget%' -epmurltarget '%epmurltarget%' -epmidentitydomaintarget
'%epmidentitydomaintarget%' -proxyserverusername '%proxyserverusername%' -
proxyserverpassword '%proxyserverpassword%' -proxyserverdomain
```

```
'%proxyserverdomain%' -userpassword '%~1' -resetpassword '%resetpassword%'  
-emailtoaddress '%emailtoaddress%'"
```

3. replicateusers.bat を更新します。指定する必要がある値は、次の表を参照してください。

パラメータ	説明
epmusersource	ソース環境のアイデンティティ・ドメイン管理者とサービス管理者の役割を持つユーザーのユーザー名。 例: Windows: set epmusersource="jDoe" Linux/UNIX: epmusersource="jDoe"
epmpwdsource	ユーザーのパスワードまたは暗号化されたパスワード・ファイルの絶対パス。 例: Windows: set epmpwdsource="Example" Linux/UNIX: epmpwdsource="Example"
epmurlsource	ユーザーのコピー元の環境の URL。 例: Windows: set epmurlsource="https://example.oraclecloud.com" Linux/UNIX: epmurlsource="https://example.oraclecloud.com"
epmidentitydomainsource	ソース環境で使用されるアイデンティティ・ドメインの名前。 例: Windows: set epmidentitydomainsource="example_source_dom" Linux/UNIX: epmidentitydomainsource="example_source_dom"
epmusertarget	ターゲット環境のアイデンティティ・ドメイン管理者とサービス管理者の役割を持つユーザーのユーザー名。 例: Windows: set epmusertarget="John.Doe" Linux/UNIX: set epmusertarget="John.Doe"
epmpwdtarget	ユーザーのパスワードまたは暗号化されたパスワード・ファイルの絶対パス。 例: Windows: set epmpwdtarget="Example1" Linux/UNIX: epmpwdtarget="Example1"
epmurltarget	ユーザーを作成する環境の URL。 例: Windows: set epmurltarget="https://example.oraclecloud.com" Linux/UNIX: epmurltarget="https://example.oraclecloud.com"

パラメータ	説明
epmidentitydomaintarget	ターゲット環境で使用されるアイデンティティ・ドメインの名前。 例: Windows: set epmidentitydomaintarget="example_source_dom" Linux/UNIX: epmidentitydomaintarget="example_target_dom"
proxyserverusername	インターネットへのアクセスを制御するプロキシ・サーバーとの安全なセッションを認証するユーザー名。使用しない場合は、このプロパティをすべて削除してください。 例: Windows: set proxyserverusername="Example" Linux/UNIX: proxyserverusername="Example"
proxyserverpassword	プロキシ・サーバーに対してユーザーを認証するパスワード。使用しない場合は、このプロパティをすべて削除してください。 例: Windows: set proxyserverpassword="examplePwd" Linux/UNIX: proxyserverpassword="examplePwd"
proxyserverdomain	プロキシ・サーバーに定義されているドメインの名前。使用しない場合は、このプロパティをすべて削除してください。 例: Windows: set proxyserverdomain="exampleDom" Linux/UNIX: proxyserverdomain="exampleDom"
emailtoaddress	オプションで、役割の割当レポートが送信される電子メール・アドレス。この値が指定されている場合のみ、レポートが電子メールで送信されます。 例: emailtoaddress=john.doe@example.com

Linux/UNIX

1. 次のスクリプトをコピーすることで、replicateusers.shを作成します。

```
#!/bin/sh

userpassword="$1"

# USER DEFINED VARIABLES
#-----
javahome="<<JAVA HOME>"
epmautomatescript="<<EPM AUTOMATE SCRIPT LOCATION>"
epmusersource="<<EPM USER FOR SOURCE ENVIRONMENT>"
epmpwdsource="<<EPM PASSWORD FOR SOURCE ENVIRONMENT>"
epmurlsource="<<EPM URL FOR SOURCE ENVIRONMENT>"
epmidentitydomainsource="<<EPM IDENTITY DOMAIN FOR SOURCE ENVIRONMENT>"
epmusertarget="<<EPM USER FOR TARGET ENVIRONMENT>"
epmpwdtarget="<<EPM PASSWORD FOR TARGET ENVIRONMENT>"
epmurltarget="<<EPM URL FOR TARGET ENVIRONMENT>"
epmidentitydomaintarget="<<EPM IDENTITY DOMAIN FOR TARGET ENVIRONMENT>"
proxyserverusername="<<PROXY SERVER USER NAME>"
proxyserverpassword="<<PROXY SERVER PASSWORD>"
proxyserverdomain="<<PROXY SERVER DOMAIN>"
resetpassword="false"
emailtoaddress="<<EMAIL TO ADDRESS>"
```

```

#-----

roleassignmentreport="roleassignmentreport.csv"
usersreport="users.csv"
paramrequiredmessage='Syntax: replicateusers.sh "USER_PASSWORD"'

export JAVA_HOME=${javahome}

if [ "${userpassword}" == "" ]
then
    echo "USER_PASSWORD is missing. This is used to set the default
password for the replicated users."
    echo "${paramrequiredmessage}"
    exit
fi

echo "Replicate users script started"

# epmautomate login Source App as an IDM Admin
echo "Logging into source application at ${epmurlsource}"
${epmautomatescript} login ${epmusersource} ${epmpwdsource} $
{epmurlsource} ${epmidentitydomainsource} ${proxyserverusername} $
{proxyserverpassword} ${proxyserverdomain}
echo "Creating role assignment report: ${roleassignmentreport}"
${epmautomatescript} roleAssignmentReport ${roleassignmentreport}
if [[ "${emailtoaddress}" == *@"* ]]
then
    ${epmautomatescript} sendMail $emailtoaddress "Role assignment report"
Body="Role assignment report is attached."
Attachments=${roleassignmentreport}
fi
echo "Downloading role assignment report"
${epmautomatescript} downloadfile ${roleassignmentreport}
${epmautomatescript} deletefile ${roleassignmentreport}
${epmautomatescript} logout

awk -F, '{print $2,""$3,""$4,""$1}' ${roleassignmentreport} | (read -r;
printf "%s\n" "$REPLY"; sort -u) > ${usersreport}

# epmautomate login Target App as an IDM Admin
echo "Logging into target application at ${epmurltarget}"
${epmautomatescript} login ${epmuserstarget} ${epmpwdtarget} $
{epmurltarget} ${epmidentitydomaintarget} ${proxyserverusername} $
{proxyserverpassword} ${proxyserverdomain}
${epmautomatescript} deletefile ${usersreport} > /dev/null 2>&1
echo "Uploading file ${usersreport}"
${epmautomatescript} uploadfile ${usersreport}
echo "Adding users"
${epmautomatescript} addUsers ${usersreport} userPassword=${userpassword}
resetPassword=${resetpassword}
${epmautomatescript} deletefile ${usersreport}
${epmautomatescript} logout
rm deletefile*.log > /dev/null 2>&1

echo "Replicate users script completed"

```

2. replicateusers.sh を更新します。指定する必要がある値の詳細は、前の表を参照してください。さらに、次のプロパティの値を指定する必要があります:
 - javahome: **Java** がインストールされているディレクトリへの絶対パス。
 - epmautomatescript: epmautomatescript.sh の場所。例: epmautomatescript="/home/user1/epmautomate/bin/epmautomate.sh"

ある環境から別の環境への事前定義済役割割当てのレプリケート

この項のスクリプトを使用して、事前定義済役割割当てをある環境から別の環境にクローニングします。これらのスクリプトを実行するには、両方の環境のサービス管理者の役割が必要です。

ノート:

このドキュメントの PDF バージョンを使用している場合: これらのスクリプトを使用不能にする改行とフッター情報を回避するために、スクリプトを [トピックの HTML バージョン](#) からコピーしてください。

Windows

1. 次のスクリプトをコピーすることで、replicatepredefineroles.ps1 を作成します。

```
# Replicate predefined roles script

param(
    [string]$epmusersource,
    [string]$epmpwdsource,
    [string]$epmurlsource,
    [string]$epmidentitydomainsource,
    [string]$epmusertarget,
    [string]$epmpwdtarget,
    [string]$epmurltarget,
    [string]$epmidentitydomaintarget,
    [string]$proxyserverusername,
    [string]$proxyserverpassword,
    [string]$proxyserverdomain,
    [string]$emailtoaddress
)

$roleassignmentreport="roleassignmentreport.csv"

function replicateroles
{
    # epmautomate login Source App as an IDM Admin
    echo "Logging into source application at ${epmurlsource}"
    epmautomate login ${epmusersource} ${epmpwdsource} ${epmurlsource} $
{epmidentitydomainsource} ${proxyserverusername} ${proxyserverpassword} $
{proxyserverdomain}
    echo "Creating role assignment report: ${roleassignmentreport}"
    epmautomate roleAssignmentReport ${roleassignmentreport}
    if (${emailtoaddress} -match "@") {
        epmautomate.bat sendMail $emailtoaddress "Role assignment report"
```

```

Body="Role assignment report is attached."
Attachments=$roleassignmentreport
}
echo "Downloading role assignment report"
epmautomate downloadfile ${roleassignmentreport}
epmautomate deletefile ${roleassignmentreport}
epmautomate logout

echo "Creating files to use with epmautomate assignRoles"

Get-Content ${roleassignmentreport} | ForEach-Object {
    $user=$_.split(',')[0]
    $rolename=$_.split(',')[4]

    if ($rolename -like '*User' -And $rolename -notlike '*Power User')
    {
        $rolenamearray=$rolename.split(" ")
        $arraysize=$rolenamearray.count
        $rolename="User"
        if ($arraysize.count -le 2) {
            echo "${user}" | Out-File -Append -Encoding "UTF8" "role-${rolename}.csv"
        }
    }
    elseif ($rolename -like '*Viewer') {
        $rolenamearray=$rolename.split(" ")
        $arraysize=$rolenamearray.count
        $rolename="Viewer"
        if ($arraysize -le 2) {
            echo "${user}" | Out-File -Append -Encoding "UTF8" "role-${rolename}.csv"
        }
    }
    elseif ($rolename -like '*Power User') {
        $rolenamearray=$rolename.split(" ")
        $arraysize=$rolenamearray.count
        $rolename="Power User"
        if ($arraysize -le 3) {
            echo "${user}" | Out-File -Append -Encoding "UTF8" "role-${rolename}.csv"
        }
    }
    elseif ($rolename -like '*Service Administrator') {
        $rolenamearray=$rolename.split(" ")
        $arraysize=$rolenamearray.count
        $rolename="Service Administrator"
        if ($arraysize -le 3) {
            echo "${user}" | Out-File -Append -Encoding "UTF8" "role-${rolename}.csv"
        }
    }
    elseif ($rolename -like 'Planner') {
        echo "${user}" | Out-File -Append -Encoding "UTF8" "role-User.csv"
    }
}
}
}

```

```

# Add header and format
$rolefiles = Get-ChildItem "role-*.csv"
foreach ($rolefile in $rolefiles) {
    $rolefilecontent = Get-Content "$rolefile"
    $headerline='User Login'
    Set-Content $rolefile -value $headerline,$rolefilecontent
    $txt = [io.file]::ReadAllText("$rolefile") -replace "`r`n","`n"
    [io.file]::WriteAllText("$rolefile", $txt)
}

# epmautomate login Target App as an IDM Admin
echo "Logging into target application at ${epmurltarget}"
epmautomate login ${epmuserstarget} ${epmpwdtarget} ${epmurltarget} $
{epmidentitydomaintarget} ${proxyserverusername} ${proxyserverpassword} $
{proxyserverdomain}

$rolefiles = Get-ChildItem "role-*.csv"
foreach ($rolefile in $rolefiles) {
    $rolenamecsv=$rolefile.BaseName.split('-')[1]
    $rolename=$rolenamecsv.split('.')[0]
    epmautomate deletefile "${rolefile}" | Out-Null
    echo "Uploading file ${rolefile}"
    epmautomate uploadfile "${rolefile}"
    echo "Assigning ${rolename} roles"
    epmautomate assignRole "role-${rolename}.csv" "${rolename}"
    epmautomate deletefile "role-${rolename}.csv"
}
epmautomate logout
rm deletefile*.log | Out-Null
}

function init
{
    # delete ${role}.csv files
    $rolefiles = Get-ChildItem "role-*.csv"
    foreach ($rolefile in $rolefiles) {
        $rolefileexists=Test-Path $rolefile
        if ($rolefileexists) {
            rm "${rolefile}"
        }
    }
}

echo "Replicate predefined roles script started"
init
replicateroles
echo "Replicate predefined roles script completed"

```

2. 次のスクリプトをコピーすることで、replicatepredefineroles.bat を作成します。

```

@ECHO OFF
SET thisdir=%~dp0
SET scriptpath=%thisdir%replicatepredefinedroles.ps1

REM USER DEFINED VARIABLES

```

```

REM -----
set epusersource="<EPM USER FOR SOURCE ENVIRONMENT>"
set epmpwdsourc="<EPM PASSWORD FOR SOURCE ENVIRONMENT>"
set epurlsource="<EPM URL FOR SOURCE ENVIRONMENT>"
set epidentitydomainsourc="<EPM IDENTITY DOMAIN FOR SOURCE ENVIRONMENT>"
set epusertarget="<EPM USER FOR TARGET ENVIRONMENT>"
set epmpwdtarget="<EPM PASSWORD FOR TARGET ENVIRONMENT>"
set epurltarget="<EPM URL FOR TARGET ENVIRONMENT>"
set epidentitydomaintarget="<EPM IDENTITY DOMAIN FOR TARGET ENVIRONMENT>"
set proxyserverusername="<PROXY SERVER USER NAME>"
set proxyserverpassword="<PROXY SERVER PASSWORD>"
set proxyserverdomain="<PROXY SERVER DOMAIN>"
set emailtoaddress="<EMAIL_TO_ADDRESS>"
REM -----

PowerShell -NoProfile -ExecutionPolicy Bypass -Command "& '%scriptpath%' -
epusersource '%epusersource%' -epmpwdsourc '%epmpwdsourc%' -
epurlsource '%epurlsource%' -epidentitydomainsourc
'%epidentitydomainsourc%' -epusertarget '%epusertarget%' -epmpwdtarget
'%epmpwdtarget%' -epurltarget '%epurltarget%' -epidentitydomaintarget
'%epidentitydomaintarget%' -proxyserverusername '%proxyserverusername%' -
proxyserverpassword '%proxyserverpassword%' -proxyserverdomain
'%proxyserverdomain%' -emailtoaddress '%emailtoaddress%'"

```

- 必要に応じて `replicatepredefineroles.bat` を更新します。このファイルのプロパティに設定する必要がある値の詳細は、次の表を参照してください。

replicatepredefineroles.bat の更新

パラメータ	説明
<code>epusersource</code>	ソース環境のアイデンティティ・ドメイン管理者とサービス管理者の役割を持つユーザーのユーザー名。 例: Windows: <code>set epusersource="jDoe"</code> Linux/UNIX: <code>epusersource="jDoe"</code>
<code>epmpwdsourc</code>	ユーザーのパスワードまたは暗号化されたパスワード・ファイルの絶対パス。 例: Windows: <code>set epmpwdsourc="Example"</code> Linux/UNIX: <code>epmpwdsourc="Example"</code>
<code>epurlsource</code>	ユーザーのコピー元の環境の URL。 例: Windows: <code>set epurlsource="https://example.oraclecloud.com"</code> Linux/UNIX: <code>epurlsource="https://example.oraclecloud.com"</code>
<code>epidentitydomainsourc e</code>	ソース環境で使用されるアイデンティティ・ドメインの名前。 例: Windows: <code>set epidentitydomainsourc="example_source_dom"</code> Linux/UNIX: <code>epidentitydomainsourc="example_source_dom"</code>

パラメータ	説明
epmuserstarget	ターゲット環境のアイデンティティ・ドメイン管理者とサービス管理者の役割を持つユーザーのユーザー名。 例: Windows: set epmuserstarget="John.Doe" Linux/UNIX: set epmuserstarget="John.Doe"
epmpwdtarget	ユーザーのパスワードまたは暗号化されたパスワード・ファイルの絶対パス。 例: Windows: set epmpwdtarget="Example1" Linux/UNIX: epmpwdtarget="Example1"
epmurltarget	ユーザーを作成する環境の URL。 例: Windows: set epmurltarget="https://example.oraclecloud.com" Linux/UNIX: epmurltarget="https://example.oraclecloud.com"
epmidentitydomaintarget	ターゲット環境で使用されるアイデンティティ・ドメインの名前。 例: Windows: set epmidentitydomaintarget="example_target_dom" Linux/UNIX: epmidentitydomaintarget="example_target_dom"
proxyserverusername	インターネットへのアクセスを制御するプロキシ・サーバーとの安全なセッションを認証するユーザー名。使用しない場合は、このプロパティをすべて削除してください。 例: Windows: set proxyserverusername="Example" Linux/UNIX: proxyserverusername="Example"
proxyserverpassword	プロキシ・サーバーに対してユーザーを認証するパスワード。使用しない場合は、このプロパティをすべて削除してください。 例: Windows: set proxyserverpassword="examplePwd" Linux/UNIX: proxyserverpassword="examplePwd"
proxyserverdomain	プロキシ・サーバーに定義されているドメインの名前。使用しない場合は、このプロパティをすべて削除してください。 例: Windows: set proxyserverdomain="exampleDom" Linux/UNIX: proxyserverdomain="exampleDom"
emailtoaddress	オプションで、役割の割当レポートが送信される電子メール・アドレス。この値が指定されている場合のみ、レポートが電子メールで送信されます。 例: emailtoaddress=john.doe@example.com

Linux/UNIX

1. 次のスクリプトをコピーすることで、replicatepredefineroles.sh を作成します。

```
#!/bin/sh
```

```

# USER DEFINED VARIABLES
#-----
javahome="<<JAVA HOME>"
epmautomatescript="<<EPM AUTOMATE SCRIPT LOCATION>"
epmusersource="<<EPM USER FOR SOURCE ENVIRONMENT>"
epmpwdsource="<<EPM PASSWORD FOR SOURCE ENVIRONMENT>"
epmurlsource="<<EPM URL FOR SOURCE ENVIRONMENT>"
epmidentitydomainsource="<<EPM IDENTITY DOMAIN FOR SOURCE ENVIRONMENT>"
epmuserstarget="<<EPM USER FOR TARGET ENVIRONMENT>"
epmpwdtarget="<<EPM PASSWORD FOR TARGET ENVIRONMENT>"
epmurltarget="<<EPM URL FOR TARGET ENVIRONMENT>"
epmidentitydomaintarget="<<EPM IDENTITY DOMAIN FOR TARGET ENVIRONMENT>"
proxyserverusername="<<PROXY SERVER USER NAME>"
proxyserverpassword="<<PROXY SERVER PASSWORD>"
proxyserverdomain="<<PROXY SERVER DOMAIN>"
emailtoaddress="<<EMAIL TO ADDRESS>"
#-----

roleassignmentreport="roleassignmentreport.csv"

export JAVA_HOME=${javahome}

replicateroles()
{
    # epmautomate login Source App as an DM Admin
    echo "Logging into source application at ${epmurlsource}"
    ${epmautomatescript} login ${epmusersource} ${epmpwdsource} $
{epmurlsource} ${epmidentitydomainsource} ${proxyserverusername} $
{proxyserverpassword} ${proxyserverdomain}
    echo "Creating role assignment report: ${roleassignmentreport}"
    ${epmautomatescript} roleAssignmentReport ${roleassignmentreport}
    if [[ "${emailtoaddress}" == *@"* ]]
    then
        ${epmautomatescript} sendMail $emailtoaddress "Role assignment
report" Body="Role assignment report is attached."
Attachments=$roleassignmentreport
    fi
    echo "Downloading role assignment report"
    ${epmautomatescript} downloadfile ${roleassignmentreport}
    ${epmautomatescript} deletefile ${roleassignmentreport}
    ${epmautomatescript} logout

    echo "Creating files to use with epmautomate assignRoles"
    while read line
    do
        user=$(echo "${line}" | cut -d',' -f1)
        rolename=$(echo "${line}" | cut -d',' -f5)

        if [[ "$rolename" == *User ]] && [[ "$rolename" != *Power
User ]]
        then
            count=$(echo "${rolename}" | wc -w);
            rolename="User"
            if [[ $count -le 2 ]]
            then

```

```

        echo "${user}" >> "role-${rolename}.csv"
    fi
elif [[ "$rolename" == *"Viewer" ]]
then
    count=$(echo "${rolename}" | wc -w);
    rolename="Viewer"
    if [[ $count -le 2 ]]
    then
        echo "${user}" >> "role-${rolename}.csv"
    fi
elif [[ "$rolename" == *"Power User" ]]
then
    count=$(echo "${rolename}" | wc -w);
    rolename="Power User"
    if [[ $count -le 3 ]]
    then
        echo "${user}" >> "role-${rolename}.csv"
    fi
elif [[ "$rolename" == *"Service Administrator" ]]
then
    count=$(echo "${rolename}" | wc -w);
    rolename="Service Administrator"
    if [[ $count -le 3 ]]
    then
        echo "${user}" >> "role-${rolename}.csv"
    fi
elif [[ "$rolename" == "Planner" ]]
then
    echo "${user}" >> "role-User.csv"
fi
done < ${roleassignmentreport}

# write header line
for f in role-*.csv
do
    sed -i 'liUser Login' "$f"
done

# epmautomate login Target App as an IDM Admin
echo "Logging into target application at ${epmurltarget}"
${epmautomatescript} login ${epmuserstarget} ${epmpwdtarget} $
{epmurltarget} ${epmidentitydomaintarget} ${proxyserverusername} $
{proxyserverpassword} ${proxyserverdomain}

for rolefile in role-*.csv
do
    rolenamecsv=$(echo "$rolefile" | cut -d'-' -f2)
    rolename=$(echo "$rolenamecsv" | cut -d'.' -f1)
    ${epmautomatescript} deletefile "${rolefile}" > /dev/null 2>&1
    echo "Uploading file ${rolefile}"
    ${epmautomatescript} uploadfile "${rolefile}"
    echo "Assigning roles"
    ${epmautomatescript} assignrole "${rolefile}" "${rolename}"
    ${epmautomatescript} deletefile "${rolefile}"
done

```

```

    ${epmautomatescript} logout
    rm deletedefile*.log > /dev/null 2>&1
}

init()
{
    # delete role-${role}.csv files
    for f in role-*.csv
    do
        rm "$f" > /dev/null 2>&1
    done
}

echo "Replicate predefined roles script started"
init
replicateroles
echo "Replicate predefined roles script completed"

```

2. `replicatepredefineroles.sh` を更新します。指定する必要がある値の詳細は、前の表を参照してください。さらに、次のプロパティの値を指定する必要があります。
 - `javahome`: **Java** がインストールされているディレクトリへの絶対パス。
 - `epmautomatescript`: `epmautomatescript.sh` の場所。例: `epmautomatescript="/home/user1/epmautomate/bin/epmautomate.sh"`

四半期の Cloud EPM アップグレード頻度の作成

これらのスクリプトを使用して、**Cloud EPM** 環境が 2 週間のテスト・サイクルを使用して四半期ベースで更新されるように、更新をスキップするセルフサービス・ソリューションを作成します。この場合、本番環境はテスト環境の 2 週間後に更新されます。

必要に応じて、このスクリプトは、2 か月ごとに更新をスキップするのにも使用できます。デフォルトでは、**Cloud EPM** は環境に対して月次更新を適用します。`skipUpdate` コマンドを使用して、環境に対する月次更新の適用をスキップしたり、現在の更新のスキップ要求を表示します。この項に含まれるスクリプトを使用して、`skipUpdate` コマンドの手動実行を自動化できます。これらのスクリプトは、更新が四半期または 2 か月ごとに適用されるよう、更新のスキップ・プロセスを自動化します。

Note:

1. 連続 2 か月を超えて更新をスキップすることはできません。たとえば、**Cloud EPM** 環境を 2 月、6 月および 11 月のみ更新しようとした場合、スクリプトによりエラーがスローされます。
2. その間に発生したすべての更新は、次回の更新時に環境に適用されます。たとえば、このスクリプトを使用して、2 月、5 月、8 月および 11 月にのみ発生する四半期ごとの更新をスケジュールするとします。この場合、たとえば、5 月の更新では、2 月の更新後にリリースされた適用可能なすべての **Cloud EPM** 月次更新およびパッチが環境に適用されます。更新が適用されると、メンテナンス・プロセスが通常より時間がかかる場合があります。
3. このスクリプトは、1 四半期ごとの更新頻度を設定します。1 年中更新頻度が構成されるようにするには、このスクリプトを月次ベースで実行します。

- [Windows スクリプトと手順](#)
- [UNIX/Linux スクリプトと手順](#)
- [Groovy スクリプト](#)

スクリプトの実行

1. Windows スクリプトと Linux/UNIX スクリプトを実行するには:
 - a. `input.properties` ファイルを作成し、環境の情報を使用して更新します。ファイルをローカル・ディレクトリに保存します。このファイルのコンテンツは、オペレーティング・システムによって異なります。このディレクトリの書き込み権限があることを確認してください。Windows の場合、スクリプトを実行できるよう、「**管理者として実行**」オプションを使用して PowerShell を開始する必要があります。
 - b. `skip_update.ps1` (Windows PowerShell) または `skip_update.sh` (Linux/UNIX) Bash スクリプトを作成し、`input.properties` があるディレクトリに保存します。
 - c. スクリプトを起動します。
 - Linux/UNIX: `./skip_update.sh` を実行します。
 - Windows PowerShell: `skip_update.ps1` を実行します。
2. Groovy スクリプトを実行するには、Cloud EPM ビジネス・プロセスで Groovy 画面を使用するか、[runBusinessRule](#) を使用してスクリプトの実行を自動化します。EPM 自動化を使用して Groovy スクリプトを実行する方法については、[EPM 自動化をインストールしないコマンドの実行](#)を参照してください。

Windows スクリプトと手順

この項のスクリプトをコピーすることにより、`input.properties` および `skip_update.ps1` を作成します。

1. 次のスクリプトをコピーすることにより、`input.properties` を作成します:

```
username=exampleAdmin
password=examplePassword.epw
url=exampleURL
updatemonths=02,05,08,11
```

2. パラメータ値を指定して、`input.properties` を更新します。

Table 3-12 `input.properties` のパラメータ

パラメータ	説明
<code>username</code>	サービス管理者のユーザー名
<code>password</code>	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
<code>url</code>	月次以外の更新頻度を設定する環境の URL。

Table 3-12 (Cont.) input.properties のパラメータ

パラメータ	説明
updatemonths	Oracle Fusion Cloud Enterprise Performance Management の更新時の url パラメータで識別される環境に適用する必要がある月のカンマ区切りリスト。例: updatemonths=02,05,08,11。 月は2桁で指定する必要があります(1月の01から12月の12まで)。1月から9月までは、先頭にゼロを付けてください。スクリプトは、updatemonths パラメータ値に含まれない月に対して、skipUpdate コマンドを実行します。たとえば、updatemonths=02,05,08,11 と指定すると、スクリプトは更新のスキップ・フラグを1月、3月、4月、6月、7月、9月、10月および12月に設定するため、2月、5月、8月および11月にのみ更新が行われます。

3. 次のスクリプトをコピーすることにより、skip_updates.ps1 を作成します:

```
# Skip Update PowerShell script

$inputproperties = ConvertFrom-StringData(Get-Content ./input.properties -
raw)
$username="$($inputproperties.username)"
$password="$($inputproperties.password)"
$url="$($inputproperties.url)"
$updatemonths="$($inputproperties.updatemonths)"

$monthsarr = ("01","02","03","04","05","06","07","08","09","10","11","12")
$global:monthsarrfromcurrent = @()
$global:yearsarrfromcurrent = @()
$updatemonthsarr = $updatemonths.Split(",")
$currentyear=Get-Date -Format yy
$currentmonth=Get-Date -Format MM
$nextyear=[int]$currentyear+1

function populateFromCurrentArrays() {
    $startposition = 0

    for ($i = 0; $i -le ($monthsarr.length - 1); $i++) {
        if ($currentmonth -eq $monthsarr[$i]) {
            $startposition=$i
            break
        }
    }

    for ($i = 0; $i -le ($monthsarr.length - 1); $i++) {
        if ($i -ge $startposition) {
            $global:monthsarrfromcurrent += $monthsarr[$i]
            $global:yearsarrfromcurrent += $currentyear
        }
    }

    for ($i = 0; $i -le ($monthsarr.length - 1); $i++) {
        if ($i -lt $startposition) {
            $global:monthsarrfromcurrent += $monthsarr[$i]
        }
    }
}
```

```

        $global:yearsarrfromcurrent += $nextyear
    }
}

function skipUpdateAdd($yearnumber, $monthnumber) {
    echo "Running: epmautomate.bat skipUpdate add version=${yearnumber}.${
{monthnumber} comment=`"adding skipUpdate`"
    epmautomate skipUpdate add version=${yearnumber}.${monthnumber}
comment="adding skipUpdate"
}

function processSkipUpdates() {
    $addcount = 0

    echo "Running: epmautomate.bat login ${username} ${password} ${url}"
    epmautomate login ${username} ${password} ${url}
    echo "Running: epmautomate.bat skipUpdate remove"
    epmautomate skipUpdate remove

    for ($i = 0; $i -le ($global:monthsarrfromcurrent.length - 1); $i++) {
        $match = 1

        if (${addcount} -eq 2) {
            echo "Two skip update add calls have been made. No more will
be attempted."
            break
        }

        for ($j = 0; $j -le ($updatemonthsarr.length - 1); $j++) {
            if ($global:monthsarrfromcurrent[$i] -eq $updatemonthsarr[$j]) {
                $match = 0
            }
        }

        if (${match} -eq 1) {

skipUpdateAdd $global:yearsarrfromcurrent[$i] $global:monthsarrfromcurrent[
$i]
                $addcount += 1
            }
        }

        echo "Running: epmautomate.bat skipUpdate list"
        epmautomate skipUpdate list
        echo "Running: epmautomate.bat logout"
        epmautomate logout
    }

function compareUpdateMonths($thismonth, $nextmonth) {
    $nextmonthorig=${nextmonth}

    if (${nextmonth} -lt ${thismonth}) {
        $nextmonth+=12
    }
}

```

```

$monthdiff = $nextmonth - $thismonth

if (${monthdiff} -gt 3) {
    echo "There are more than 2 months skipped from month ${thismonth}
to month ${nextmonthorig}. Please correct updatemonths in input.properties
so that there are not more than two months skipped between each update
month. Exiting."
    exit 1
}
}

function validateUpdateMonths() {
    for ($i = 0; $i -le ($updatemonthsarr.length - 1); $i++) {
        $nextint = $i + 1
        $thisupdatemonth = $updatemonthsarr[$i]
        $thisupdatemonthhint=[int]$thisupdatemonth
        $nextupdatemonth=$updatemonthsarr[$nextint]
        $nextupdatemonthhint=[int]$nextupdatemonth

        if (${nextupdatemonth} -eq "") {
            $nextupdatemonth=$updatemonthsarr[0]
            $nextupdatemonthhint=[int]$nextupdatemonth
        }

        compareUpdateMonths $thisupdatemonthhint $nextupdatemonthhint
    }
}

validateUpdateMonths
populateFromCurrentArrays
processSkipUpdates

```

UNIX/Linux スクリプトと手順

この項のスクリプトをコピーすることにより、input.properties および skip_update.sh を作成します。

1. 次のスクリプトをコピーすることにより、input.properties を作成します:

```

javahome=JAVA_HOME
epmautomatescript=EPM_AUTOMATE_LOCATION
username=exampleAdmin
password=examplePassword.epw
url=exampleURL
updatemonths=02,05,08,11

```

2. パラメータ値を指定して、input.properties を更新します。

Table 3-13 input.properties のパラメータ

パラメータ	説明
javahome	JAVA_HOME の場所。

Table 3-13 (Cont.) input.properties のパラメータ

パラメータ	説明
epmautomatescript	EPM 自動化の実行可能ファイル(epmautomate.sh)の絶対パス。
username	サービス管理者のユーザー名
password	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
url	月次以外の更新頻度を設定する環境の URL。
updatemonths	<p>Oracle Fusion Cloud Enterprise Performance Management の更新時の url パラメータで識別される環境に適用する必要がある月のカンマ区切りリスト。例: updatemonths=02,05,08,11。</p> <p>月は 2 桁で指定する必要があります。1 月から 9 月までは、先頭にゼロを付けてください。スクリプトは、updatemonths パラメータ値に含まれない月に対して、skipUpdate コマンドを実行します。たとえば、updatemonths=02,05,08,11 と指定すると、スクリプトは更新のスキップ・フラグを 1 月、3 月、4 月、6 月、7 月、9 月、10 月および 12 月に設定するため、2 月、5 月、8 月および 11 月にのみ更新が行われます。</p>

3. 次のスクリプトをコピーすることにより、skip_updates.sh を作成します:

```
#!/bin/sh

. ./input.properties
export JAVA_HOME=${javahome}

declare -a monthsarr=(01 02 03 04 05 06 07 08 09 10 11 12)
declare -a monthsarrfromcurrent
declare -a yearsarrfromcurrent
updatemonthsarr=( $(echo "${updatemonths}" | sed 's/,/ /g') )
currentyear=$(date +%y)
nextyear=$((currentyear+1))
currentmonth=$(date +%m)

populateFromCurrentArrays() {
  for i in ${!monthsarr[@]}
  do
    if [[ "${currentmonth}" == "${monthsarr[$i]}" ]]
    then
      startposition=$i
      break
    fi
  done

  for i in ${!monthsarr[@]}
  do
    if [[ $i -ge $startposition ]]
    then
      monthsarrfromcurrent=("${monthsarrfromcurrent[@]}" "${monthsarr[$i]}")
      yearsarrfromcurrent=("${yearsarrfromcurrent[@]}" "${currentyear}")
    fi
  done
}
```

```

        fi
    done

    for i in ${!monthsarr[@]}
    do
        if [[ ${i} -lt ${startposition} ]]
        then
            monthsarrfromcurrent=("${monthsarrfromcurrent[@]}" "${monthsarr[$i]}")
            yearsarrfromcurrent=("${yearsarrfromcurrent[@]}" "${nextyear}")
        fi
    done
}

skipUpdateAdd() {
    local yearnumber="$1"
    local monthnumber="$2"

    echo "Running: ${epmautomatescript} skipUpdate add version=${yearnumber}.${monthnumber} comment=\"adding skipUpdate\""
    ${epmautomatescript} skipUpdate add version=${yearnumber}.${monthnumber} comment="adding skipUpdate"
}

processSkipUpdates() {
    local addcount=0

    echo "Running: ${epmautomatescript} login ${username} ${password} ${url}"
    ${epmautomatescript} login ${username} ${password} ${url}
    echo "Running: ${epmautomatescript} skipUpdate remove"
    ${epmautomatescript} skipUpdate remove

    for i in ${!monthsarrfromcurrent[@]}
    do
        local match=1

        if [[ ${addcount} -eq 2 ]]
        then
            echo "Two skip update add calls have been made. No more will be attempted."
            break
        fi

        for j in ${!updatemonthsarr[@]}
        do
            if [[ "${monthsarrfromcurrent[$i]}" == "${updatemonthsarr[$j]}" ]]
            then
                match=0
                break
            fi
        done

        if [[ ${match} -eq 1 ]]
        then

```

```

        skipUpdateAdd ${yearsarrfromcurrent[$i]} "$
{monthsarrfromcurrent[$i]}"
        addcount=$((addcount+1))
    fi
done

echo "Running: ${epmautomatescript} skipUpdate list"
${epmautomatescript} skipUpdate list
echo "Running: ${epmautomatescript} logout"
${epmautomatescript} logout
}

compareUpdateMonths() {
    local thismonth=$1
    local nextmonth=$2
    local nextmonthorig=${nextmonth}

    if [[ ${nextmonth} -lt ${thismonth} ]]
    then
        nextmonth=$((nextmonth+12))
    fi

    monthdiff=$((nextmonth-thismonth))

    if [[ ${monthdiff} -gt 3 ]]
    then
        echo "There are more than 2 months skipped from month ${thismonth}
to month ${nextmonthorig}. Please correct updatemonths in input.properties
so that there are not more than two months skipped between each update
month. Exiting."
        exit 1
    fi
}

validateUpdateMonths() {
    for i in ${!updatemonthsarr[@]}
    do
        nextint=$((i+1))
        thisupdatemonth="${updatemonthsarr[$i]}"
        thisupdatemonthhint=${thisupdatemonth#0}
        nextupdatemonth="${updatemonthsarr[$nextint]}"
        nextupdatemonthhint=${nextupdatemonth#0}

        if [[ ${nextupdatemonth} == "" ]]
        then
            nextupdatemonth="${updatemonthsarr[0]}"
            nextupdatemonthhint=${nextupdatemonth#0}
        fi

        compareUpdateMonths ${thisupdatemonthhint} ${nextupdatemonthhint}
    done
}

validateUpdateMonths
populateFromCurrentArrays

```

processSkipUpdates

Groovy スクリプト

パスワードに特殊文字が含まれている場合は、[特殊文字の処理](#)を参照してください。また、使用環境にあわせて次のパラメータ値を確実に置き換えてください:

Table 3-14 変更するパラメータ

パラメータ	説明
user	サービス管理者のユーザー名
password	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
url	月次以外の更新頻度を設定する環境の URL。
updatemonths	Oracle Fusion Cloud Enterprise Performance Management の更新時の url パラメータで識別される環境に適用する必要がある月のカンマ区切りリスト。例: updatemonths=02,05,08,11。月は 2 桁で指定する必要があります(1 月の 01 から 12 月の 12 まで)。1 月から 9 月までは、先頭にゼロを付けてください。スクリプトは、updatemonths パラメータ値に含まれない月に対して、 skipUpdate コマンドを実行します。たとえば、updatemonths=02,05,08,11 と指定すると、スクリプトは更新のスキップ・フラグを 1 月、3 月、4 月、6 月、7 月、9 月、10 月および 12 月に設定するため、2 月、5 月、8 月および 11 月にのみ更新が行われます。

```
import java.text.SimpleDateFormat

String user = 'service_administrator'
String password = 'examplePWD'
String url = 'example_EPM_URL'
String updatemonths = '02,05,08,11'

def currentdate = new Date()
def yf = new SimpleDateFormat("yy")
def mf = new SimpleDateFormat("MM")
String[] monthsarr = ["01", "02", "03", "04", "05", "06", "07", "08", "09",
"10", "11", "12"]
List<String> monthsarrfromcurrent = new ArrayList<>()
List<String> yearsarrfromcurrent = new ArrayList<>()
String currentyear = yf.format(currentdate)
String nextyear = (currentyear.toInteger() + 1).toString()
String currentmonth = mf.format(currentdate)

String[] updateMonthsStringArr = updatemonths.split(',')
def updatemonthsarr = new int[updateMonthsStringArr.length]
for(int i = 0; i < updateMonthsStringArr.length; i++)
{
    updatemonthsarr[i] = Integer.parseInt(updateMonthsStringArr[i]);
}

def LogMessage(String message) {
```

```
def date = new Date()
def sdf = new SimpleDateFormat("MM/dd/yyyy HH:mm:ss")
println([' + sdf.format(date) + '][GROOVY] ' + message);
}

def LogOperationStatus(EpmAutomateStatus opstatus) {
def returncode = opstatus.getStatus()
if (returncode != 0){
    LogMessage(opstatus.getOutput())
}
LogMessage('return code: ' + returncode)
}

int CompareUpdateMonths(int thismonth, int nextmonth) {
int nextmonthorig = nextmonth

if (nextmonth < thismonth) {
    nextmonth = nextmonth + 12
}

int monthdiff = nextmonth - thismonth

if (monthdiff > 3) {
    LogMessage('There are more than 2 months skipped from month ' +
thismonth + ' to month ' + nextmonthorig + '. Please correct updatemonths so
that there are not more than two months skipped between each update month.
Exiting.')
    return 1
}

return 0
}

int ValidateUpdateMonths(int[] updatemonthsarr) {
for(int i = 0; i < updatemonthsarr.length; i++)
{
    int nextint = i + 1
    String nextupdatemonth = ""
    int nextupdatemonthint = 0
    String thisupdatemonth = updatemonthsarr[i]
    int thisupdatemonthint = thisupdatemonth.toInteger()

    if (nextint < updatemonthsarr.length) {
        nextupdatemonth = updatemonthsarr[nextint]
    } else {
        nextupdatemonth = updatemonthsarr[0]
    }

    nextupdatemonthint = nextupdatemonth.toInteger()

    int returncode = CompareUpdateMonths(thisupdatemonthint,
nextupdatemonthint)
    if (returncode > 0) {
        return 1
    }
}
}
```

```

        return 0
    }

    def SkipUpdateAdd(EpmAutomate automate, String yearnumber, String
monthnumber) {
        String yeardotmonth = yearnumber + '.' + monthnumber
        LogMessage('Running: epmautomate skipUpdate add version=' + yeardotmonth
+ ' comment=\"adding skipUpdate\"')
        EpmAutomateStatus status = automate.execute('skipupdate','add','version='
+ yeardotmonth,'comment=\"adding skipUpdate\"')
        LogOperationStatus(status)
    }

    LogMessage('Starting skip update processing')
    EpmAutomate automate = getEpmAutomate()

    // validate update months
    int returncode = ValidateUpdateMonths(updatemonthsarr)
    if (returncode != 0) {
        return 1
    }

    // populate arrays
    int startposition = 0
    for(int i = 0; i < monthsarr.length; i++)
    {
        if (currentmonth == monthsarr[i]) {
            startposition = i
            break
        }
    }

    for(int i = 0; i < monthsarr.length; i++)
    {
        if (i >= startposition) {
            monthsarrfromcurrent.add(monthsarr[i])
            yearsarrfromcurrent.add(currentyear)
        }
    }

    for(int i = 0; i < monthsarr.length; i++)
    {
        if (i <= startposition) {
            monthsarrfromcurrent.add(monthsarr[i])
            yearsarrfromcurrent.add(nextyear)
        }
    }

    // process skip updates
    LogMessage("Operation: encrypt " + password + " oracleKey password.epw")
    EpmAutomateStatus status =
    automate.execute('encrypt',password,"oracleKey","password.epw")
    LogOperationStatus(status)

    LogMessage("Operation: login " + user + " password.epw " + url)
    status = automate.execute('login',user,"password.epw",url)

```

```

LogOperationStatus(status)

LogMessage('Running: epmautomate skipUpdate remove')
status = automate.execute('skipupdate','remove')
LogOperationStatus(status)

int addcount = 0

for (int i = 0; i < monthsarrfromcurrent.size(); i++) {
    int match = 1

    if (addcount == 2){
        LogMessage('Two skip update add calls have been made. No more will be
attempted.')
        break
    }

    for(int j = 0; j < updatemonthsarr.length; j++) {

        if (Integer.parseInt(monthsarrfromcurrent.get(i)) ==
updatemonthsarr[j]) {
            match = 0
            break
        }
    }

    if (match == 1) {
        SkipUpdateAdd(automate, yearsarrfromcurrent.get(i),
monthsarrfromcurrent.get(i))
        addcount+=1
    }
}

LogMessage('Running: epmautomate skipUpdate list')
status = automate.execute('skipupdate','list')
LogOperationStatus(status)

LogMessage('Running: epmautomate logout')
status = automate.execute('logout')
LogOperationStatus(status)

LogMessage('Skip update processing completed')

```

6 週間のテスト・サイクルを使用した四半期の Cloud EPM アップグレード頻度の作成

この項のスクリプトを使用して、Oracle Fusion Cloud Enterprise Performance Management 環境が 6 週間のテスト・サイクルを使用して四半期ベースで更新されるように、更新をスキップするセルフサービス・ソリューションを作成します。この場合、本番環境はテスト環境の 6 週間後に更新されます。

デフォルトでは、Cloud EPM は環境に対して月次更新を適用します。skipUpdate コマンドを使用して、環境に対する月次更新の適用をスキップしたり、現在の更新のスキップ要求を表示します。この項に含まれるスクリプトを使用して、skipUpdate コマンドの手動実行を自動化でき

ます。これらのスクリプトは、6週間のテスト・サイクルを使用した四半期ベースで更新が適用されるように、更新のスキップ・プロセスを自動化します。

 **Note:**

1. 連続3か月を超えて更新をスキップすることはできません。Cloud EPM 環境を2月と10月のみ更新しようとした場合、このスクリプトによりエラーがスローされます。
2. その間に発生したすべての更新は、次回の更新時に環境に適用されます。たとえば、このスクリプトを使用して、2月、5月、8月および11月の更新時にのみ発生する四半期ごとの更新をスケジュールするとします。この場合、たとえば、5月の更新では、2月の更新後にリリースされた適用可能なすべての Cloud EPM 月次更新およびパッチが環境に適用されます。更新が適用されると、メンテナンス・プロセスが通常より時間がかかる場合があります。
3. このスクリプトは、1四半期のみ更新頻度を設定します。
サンプルのシナリオ: テスト環境の更新サイクルは、2月(24.02 更新)、5月(24.05 更新)、8月(24.08 更新)および11月(24.11 更新)の第1金曜日として確立されます。2月(24.02 更新)の第1金曜日にテスト環境を更新するのに使用されたバージョンを使用して、本番環境が3月(24.02 更新)の第3金曜日に更新されます。本番環境への同様の更新が、6月(24.05 更新)、9月(24.08 更新)および12月(24.11 更新)の第3週に発生します。このシナリオでは、本番環境は現在の更新には更新されませんが、現在テスト環境にある更新を使用して更新されます。

Windows のサンプル・スクリプト

次のスクリプトをコピーして、skip_update.ps1 を作成します。それをローカル・ディレクトリに保存します。このスクリプトを実行する方法については、[スクリプトの実行](#)を参照してください。

```
# Skip Update PowerShell script

$inputproperties = ConvertFrom-StringData (Get-Content ./input.properties -raw)
$username="$($inputproperties.username) "
$password="$($inputproperties.password) "
$url="$($inputproperties.url) "
$updateversions="$($inputproperties.updateversions) "
$podtype="$($inputproperties.podtype) "
$proxyserverusername="$($inputproperties.proxyserverusername) "
$proxyserverpassword="$($inputproperties.proxyserverpassword) "
$proxyserverdomain="$($inputproperties.proxyserverdomain) "

echo "Starting skip_update.ps1 script."

$monthsarr = ("01","02","03","04","05","06","07","08","09","10","11","12")
$global:monthsarrfromcurrent = @()
$global:yearsarrfromcurrent = @()
$updateversionsarr = $updateversions.Split(",")
$currentyear=Get-Date -Format yy
$currentmonth=Get-Date -Format MM
$nextyear=[int]$currentyear+1
```

```

function populateFromCurrentArrays() {
    $startposition = 0

    for ($i = 0; $i -le ($monthsarr.length - 1); $i++) {
        if (${currentmonth} -eq $monthsarr[$i]) {
            if (${podtype} -eq "prod") {
                if (${updateversionsarr} -contains ${currentmonth}) {
                    $startposition=$i-2
                } else {
                    $startposition=$i-1
                }
            } else {
                if (${updateversionsarr} -contains ${currentmonth}) {
                    $startposition=$i
                } else {
                    $startposition=$i-1
                }
            }
        }
        break
    }

    if (${startposition} -lt 0) {
        $startposition=$startposition+12
    }

    for ($i = 0; $i -le ($monthsarr.length - 1); $i++) {
        if (${i} -ge ${startposition}) {
            $global:monthsarrfromcurrent += $monthsarr[$i]
            $global:yearsarrfromcurrent += $currentyear
        }
    }

    for ($i = 0; $i -le ($monthsarr.length - 1); $i++) {
        if (${i} -lt ${startposition}) {
            $global:monthsarrfromcurrent += $monthsarr[$i]
            $global:yearsarrfromcurrent += $nextyear
        }
    }
}

function skipUpdateAdd($yearnumber, $monthnumber) {
    echo "Running: epmautomate.bat skipUpdate add version=${yearnumber}.${
monthnumber} comment=`"adding skipUpdate`""
    epmautomate skipUpdate add version=${yearnumber}.${monthnumber}
comment="adding skipUpdate"
}

function processSkipUpdates() {
    $addcount = 0
    $countlimit = 0

    if (${podtype} -eq "prod") {
        $countlimit = 3
    } else {
        $countlimit = 2
    }
}

```

```

    }

    if ((${proxyserverusername} -eq "") -And (${proxyserverpassword} -eq "") -
And (${proxyserverdomain} -eq "")) {
        echo "Running: epmautomate.bat login ${username} ${password} ${url}"
        epmautomate login ${username} ${password} ${url}
    } else {
        echo "Running: epmautomate.bat login ${username} ${password} ${url}
ProxyServerUserName=${proxyserverusername} ProxyServerPassword=$
${proxyserverpassword} ProxyServerDomain=${proxyserverdomain}"
        epmautomate login ${username} ${password} ${url} ProxyServerUserName=$
${proxyserverusername} ProxyServerPassword=${proxyserverpassword}
ProxyServerDomain=${proxyserverdomain}
    }

    echo "Running: epmautomate.bat skipUpdate remove"
    epmautomate skipUpdate remove

    for ($i = 0; $i -le ($global:monthsarrfromcurrent.length - 1); $i++) {
        $match = 1

        if (${addcount} -eq ${countlimit}) {
            echo "Update calls are completed. No more will be attempted."
            break
        }

        for ($j = 0; $j -le ($updateversionsarr.length - 1); $j++) {
            if ((${currentmonth} -eq $updateversionsarr[$j]) -And (${addcount} -
gt 0)) {
                $match = 1
                break
            }

            if (($global:monthsarrfromcurrent[$i] -eq $updateversionsarr[$j]) -
And (${addcount} -eq 0)){
                $match = 0
                break
            }
        }

        if (${match} -eq 1) {
            skipUpdateAdd $global:yearsarrfromcurrent[$i] $global:monthsarrfromcurrent[$i]
                $addcount += 1
        }
    }

    echo "Running: epmautomate.bat skipUpdate list"
    epmautomate skipUpdate list
    echo "Running: epmautomate.bat logout"
    epmautomate logout
}

function compareUpdateMonths($thismonth, $nextmonth) {
    $nextmonthorig=${nextmonth}

```

```

    if (${nextmonth} -lt ${thismonth}) {
        $nextmonth+=12
    }

    $monthdiff = $nextmonth - $thismonth

    if (${monthdiff} -gt 4) {
        echo "There are more than 3 versions skipped from version $
{thismonth} to version ${nextmonthorig}. Please correct updateversions in
input.properties so that there are not more than three versions skipped
between each update version. Exiting."
        exit 1
    }
}

function validateUpdateVersions() {
    for ($i = 0; $i -le ($updateversionsarr.length - 1); $i++) {
        $nextint = $i + 1
        $thisupdatemonth = $updateversionsarr[$i]
        $thisupdatemonthhint=[int]$thisupdatemonth
        $nextupdatemonth=$updateversionsarr[$nextint]
        $nextupdatemonthhint=[int]$nextupdatemonth

        if (${nextupdatemonth} -eq "") {
            $nextupdatemonth=$updateversionsarr[0]
            $nextupdatemonthhint=[int]$nextupdatemonth
        }

        compareUpdateMonths $thisupdatemonthhint $nextupdatemonthhint
    }
}

validateUpdateVersions
populateFromCurrentArrays
processSkipUpdates

```

Linux/UNIX のサンプル・スクリプト

次のスクリプトをコピーして、skip_update.sh を作成します。それをローカル・ディレクトリに保存します。このスクリプトを実行する方法については、[スクリプトの実行](#)を参照してください:

```

#!/bin/sh

. ./input.properties

echo "Starting skip_update.sh script."

export JAVA_HOME=${javahome}

declare -a monthsarr=(01 02 03 04 05 06 07 08 09 10 11 12)
declare -a monthsarrfromcurrent
declare -a yearsarrfromcurrent
updateversionsarr=( $(echo "${updateversions}" | sed 's/,/ /g') )
currentyear=$(date +%y)

```

```

nextyear=$((currentyear+1))
currentmonth=$(date +%m)

populateFromCurrentArrays() {
    local startposition=0

    for i in ${!monthsarr[@]}
    do
        if [[ "${currentmonth}" == "${monthsarr[$i]}" ]]
        then
            if [[ "${podtype}" == "prod" ]]
            then
                if [[ ${updateversionsarr[@]} =~ ${currentmonth} ]]
                then
                    startposition=$((i-2))
                else
                    startposition=$((i-1))
                fi
                break
            else
                if [[ ${updateversionsarr[@]} =~ ${currentmonth} ]]
                then
                    startposition=$i
                else
                    startposition=$((i-1))
                fi
                break
            fi
        fi
    done

    if [[ ${startposition} -lt 0 ]]
    then
        startposition=$((startposition+12))
    fi

    for i in ${!monthsarr[@]}
    do
        if [[ $i -ge ${startposition} ]]
        then
            monthsarrfromcurrent=("${monthsarrfromcurrent[@]}" "$
{monthsarr[$i]}")
            yearsarrfromcurrent=("${yearsarrfromcurrent[@]}" "${currentyear}")
        fi
    done

    for i in ${!monthsarr[@]}
    do
        if [[ $i -lt ${startposition} ]]
        then
            monthsarrfromcurrent=("${monthsarrfromcurrent[@]}" "$
{monthsarr[$i]}")
            yearsarrfromcurrent=("${yearsarrfromcurrent[@]}" "${nextyear}")
        fi
    done
}

```

```

skipUpdateAdd() {
    local yearnumber="$1"
    local monthnumber="$2"

    echo "Running: ${epmautomatescript} skipUpdate add version=${yearnumber}.${
{monthnumber} comment=\"adding skipUpdate\""
    ${epmautomatescript} skipUpdate add version=${yearnumber}.${monthnumber}
comment="adding skipUpdate"
}

processSkipUpdates() {
    local addcount=0
    local countlimit=0

    if [[ "${podtype}" == "prod" ]]
    then
        countlimit=3
    else
        countlimit=2
    fi

    if [[ "${proxyserverusername}" == "" ]] && [[ "${proxyserverpassword}" ==
"" ]] && [[ "${proxyserverdomain}" == "" ]]
    then
        echo "Running: ${epmautomatescript} login ${username} ${password} $
{url}"
        ${epmautomatescript} login ${username} ${password} ${url}
    else
        echo "Running: ${epmautomatescript} login ${username} ${password} $
{url} ProxyServerUserName=${proxyserverusername} ProxyServerPassword=$
{proxyserverpassword} ProxyServerDomain=${proxyserverdomain}"
        ${epmautomatescript} login ${username} ${password} ${url}
ProxyServerUserName=${proxyserverusername} ProxyServerPassword=$
{proxyserverpassword} ProxyServerDomain=${proxyserverdomain}
    fi
    echo "Running: ${epmautomatescript} skipUpdate remove"
    ${epmautomatescript} skipUpdate remove

    for i in ${!monthsarrfromcurrent[@]}
    do
        local match=1

        if [[ ${addcount} -eq ${countlimit} ]]
        then
            echo "Update add calls are completed. No more will be attempted."
            break
        fi

        for j in ${!updateversionsarr[@]}
        do
            if [[ "${currentmonth}" == "${updateversionsarr[$j]}" ]] && [[ $
{addcount} -gt 0 ]]
            then
                match=1
                break
            fi
        done
    done
}

```

```

        fi

        if [[ "${monthsarrfromcurrent[$i]}" == "${updateversionsarr[$j]}" ]] && [[ ${addcount} -eq 0 ]]
        then
            match=0
            break
        fi
    done

    if [[ ${match} -eq 1 ]]
    then
        skipUpdateAdd ${yearsarrfromcurrent[$i]} "${monthsarrfromcurrent[$i]}"
        addcount=$((addcount+1))
    fi
done

echo "Running: ${epmautomatescript} skipUpdate list"
${epmautomatescript} skipUpdate list
echo "Running: ${epmautomatescript} logout"
${epmautomatescript} logout
}

compareUpdateMonths() {
    local thismonth=$1
    local nextmonth=$2
    local nextmonthorig=${nextmonth}

    if [[ ${nextmonth} -lt ${thismonth} ]]
    then
        nextmonth=$((nextmonth+12))
    fi

    monthdiff=$((nextmonth-thismonth))

    if [[ ${monthdiff} -gt 4 ]]
    then
        echo "There are more than 3 versions skipped from version ${thismonth} to version ${nextmonthorig}. Please correct updateversions in input.properties so that there are not more than three versions skipped between each update version. Exiting."
        exit 1
    fi
}

validateUpdateVersions() {
    for i in ${!updateversionsarr[@]}
    do
        nextint=$((i+1))
        thisupdatemonth="${updateversionsarr[$i]}"
        thisupdatemonthint=${thisupdatemonth#0}
        nextupdatemonth="${updateversionsarr[$nextint]}"
        nextupdatemonthint=${nextupdatemonth#0}

        if [[ ${nextupdatemonth} == "" ]]

```

```

        then
            nextupdatemonth="${updateversionsarr[0]}"
            nextupdatemonthhint=${nextupdatemonth#0}
        fi

        compareUpdateMonths ${thisupdatemonthhint} ${nextupdatemonthhint}
    done
}

validateUpdateVersions
populateFromCurrentArrays
processSkipUpdates

```

Groovy スクリプト

次のスクリプトをコピーして更新することにより、`skip_update.groovy` **Groovy** スクリプトを作成します。このスクリプトを実行する方法については、[スクリプトの実行](#)を参照してください:

この **Groovy** スクリプトの次の変数を更新します:

- `username`: 月次以外の更新頻度を設定する環境のサービス管理者のユーザー名。
- `password`: サービス管理者のパスワード、または暗号化されたパスワード・ファイルの名前と場所。
- `url`: 月次以外の更新頻度を設定する環境の URL。
- `updateversions`: `url` パラメータで識別される環境に適用される **Cloud EPM** 更新のカンマ区切りリスト。例: `updateversions=02,05,08,11`。
バージョンは 2 桁で指定する必要があります。更新 **01** から **09** までは先頭にゼロを付けてください。スクリプトは、`updateversions` パラメータ値に含まれない更新に対して、`skipUpdate` コマンドを実行します。たとえば、`updateversions=02,05,08,11` と指定すると、スクリプトは更新のスキップ・フラグを **01** (1 月)、**03** (3 月)、**04** (4 月)、**06** (6 月)、**07** (7 月)、**09** (9 月)、**10** (10 月)および **12** (12 月)更新に設定します。この場合、**Cloud EPM** の更新 **02** (2 月)、**05** (5 月)、**08** (8 月)および **11** (11 月)が環境に適用されます。
- `podtype`: **Cloud EPM** 環境タイプ。有効な値は、`test` および `prod` です。
- `proxyserverusername`: インターネットへのアクセスを制御するプロキシ・サーバーとの安全なセッションを認証するユーザー名。
- `proxyserverpassword`: プロキシ・サーバーに対してユーザーを認証するパスワード。
- `proxyserverdomain`: プロキシ・サーバーに定義されているドメインの名前。

Note:

プロキシ・サーバーを使用しない場合は、`proxyserverusername`、`proxyserverpassword` および `proxyserverdomain` パラメータの値を指定しないでください。

```

import java.text.SimpleDateFormat

String username = 'service_administrator'
String password = 'examplePWD'

```

```

String url = 'example_EPM_URL'
String updateversions = '01,04,07,10'
String podtype = 'test'
String proxyserverusername = ''
String proxyserverpassword = ''
String proxyserverdomain = ''

def currentdate = new Date()
def yf = new SimpleDateFormat("yy")
def mf = new SimpleDateFormat("MM")
String[] monthsarr = ["01", "02", "03", "04", "05", "06", "07", "08", "09",
"10", "11", "12"]
List<String> monthsarrfromcurrent = new ArrayList<>()
List<String> yearsarrfromcurrent = new ArrayList<>()
String currentyear = yf.format(currentdate)
String nextyear = (currentyear.toInteger() + 1).toString()
String currentmonth = mf.format(currentdate)

String[] updateVersionsStringArr = updateversions.split(',');
def updateversionsarr = new int[updateVersionsStringArr.length];
for(int i = 0; i < updateVersionsStringArr.length; i++)
{
    updateversionsarr[i] = Integer.parseInt(updateVersionsStringArr[i]);
}

def LogMessage(String message) {
    def date = new Date()
    def sdf = new SimpleDateFormat("MM/dd/yyyy HH:mm:ss")
    println('[ ' + sdf.format(date) + '][GROOVY] ' + message);
}

def LogOperationStatus(EpmAutomateStatus opstatus) {
    def returncode = opstatus.getStatus()
    LogMessage(opstatus.getOutput())
    LogMessage('return code: ' + returncode)
}

int CompareUpdateMonths(int thismonth, int nextmonth) {
    int nextmonthorig = nextmonth

    if (nextmonth < thismonth) {
        nextmonth = nextmonth + 12
    }

    int monthdiff = nextmonth - thismonth

    if (monthdiff > 4) {
        LogMessage('There are more than 3 versions skipped from version ' +
thismonth + ' to version ' + nextmonthorig + '. Please correct updateversions
so that there are not more than three versions skipped between each update
version. Exiting.')
        return 1
    }

    return 0
}

```

```

int ValidateUpdateMonths(int[] updateversionsarr) {
    for(int i = 0; i < updateversionsarr.length; i++)
    {
        int nextint = i + 1
        String nextupdatemonth = ""
        int nextupdatemonthint = 0
        String thisupdatemonth = updateversionsarr[i]
        int thisupdatemonthint = thisupdatemonth.toInteger()

        if (nextint < updateversionsarr.length) {
            nextupdatemonth = updateversionsarr[nextint]
        } else {
            nextupdatemonth = updateversionsarr[0]
        }

        nextupdatemonthint = nextupdatemonth.toInteger()

        int returncode = CompareUpdateMonths(thisupdatemonthint,
nextupdatemonthint)
        if (returncode > 0) {
            return 1
        }
    }
    return 0
}

def SkipUpdateAdd(EpmAutomate automate, String yearnumber, String
monthnumber) {
    String yeardotmonth = yearnumber + '.' + monthnumber
    LogMessage('Running: epmautomate skipUpdate add version=' + yeardotmonth
+ ' comment=\\"adding skipUpdate\\"')
    EpmAutomateStatus status = automate.execute('skipupdate','add','version='
+ yeardotmonth,'comment=\\"adding skipUpdate\\"')
    LogOperationStatus(status)
}

LogMessage('Starting skip update processing')
EpmAutomate automate = getEpmAutomate()

// validate update months
int returncode = ValidateUpdateMonths(updateversionsarr)
if (returncode != 0) {
    return 1
}

// populate arrays
int startposition = 0
for(int i = 0; i < monthsarr.length; i++)
{
    if (currentmonth == monthsarr[i]) {
        if (podtype.equals("prod")) {
            if (updateVersionsStringArr.contains(currentmonth)) {
                startposition = (i-2)
            } else {
                startposition = (i-1)
            }
        }
    }
}

```

```

    }
    } else {
        if (updateVersionsStringArr.contains(currentmonth)) {
            startposition = i
        } else {
            startposition = (i-1)
        }
    }
}

break
}
}

if (startposition < 0) {
    startposition = startposition + 12
}

for(int i = 0; i < monthsarr.length; i++)
{
    if (i >= startposition) {
        monthsarrfromcurrent.add(monthsarr[i])
        yearsarrfromcurrent.add(currentyear)
    }
}

for(int i = 0; i < monthsarr.length; i++)
{
    if (i <= startposition) {
        monthsarrfromcurrent.add(monthsarr[i])
        yearsarrfromcurrent.add(nextyear)
    }
}

// process skip updates
LogMessage("Operation: encrypt " + password + " oracleKey password.epw")
EpmAutomateStatus status =
automate.execute('encrypt',password,"oracleKey","password.epw")
LogOperationStatus(status)

if ((proxyserverusername != null && proxyserverusername != '') &&
(proxyserverpassword != null && proxyserverpassword != '') &&
(proxyserverdomain != null && proxyserverdomain != '')) {
    LogMessage("Operation: login " + username + " password.epw " + url + "
ProxyServerUserName=" + proxyserverusername + " ProxyServerPassword=" +
proxyserverpassword + " ProxyServerDomain=" + proxyserverdomain)
    status =
automate.execute('login',username,"password.epw",url,"ProxyServerUserName=" +
proxyserverusername,"ProxyServerPassword=" +
proxyserverpassword,"ProxyServerDomain=" + proxyserverdomain)
    LogOperationStatus(status)
} else {
    LogMessage("Operation: login " + username + " password.epw " + url)
    status = automate.execute('login',username,"password.epw",url)
    LogOperationStatus(status)
}
}
LogMessage('Running: epmautomate skipUpdate remove')

```

```
status = automate.execute('skipupdate','remove')
LogOperationStatus(status)

int addcount = 0
int countlimit = 0

if (podtype.equals("prod")) {
    countlimit = 3
} else {
    countlimit = 2
}

for (int i = 0; i < monthsarrfromcurrent.size(); i++) {
    int match = 1

    if (addcount == countlimit){
        LogMessage('Update add calls are completed. No more will be
attempted.')
        break
    }

    for(int j = 0; j < updateversionsarr.length; j++) {

        if ((Integer.parseInt(currentmonth) == updateversionsarr[j]) &&
(addcount > 0)) {
            match = 1
            break
        }

        if ((Integer.parseInt(monthsarrfromcurrent.get(i)) ==
updateversionsarr[j]) && (addcount == 0)) {
            match = 0
            break
        }
    }

    if (match == 1) {
        SkipUpdateAdd(automate, yearsarrfromcurrent.get(i),
monthsarrfromcurrent.get(i))
        addcount+=1
    }
}

LogMessage('Running: epmautomate skipUpdate list')
status = automate.execute('skipupdate','list')
LogOperationStatus(status)
println(status.getItemsList())

LogMessage('Running: epmautomate logout')
status = automate.execute('logout')
LogOperationStatus(status)

LogMessage('Skip update processing completed')
```

skip_update Windows および Linux/UNIX スクリプトを実行するための input.properties ファイルの作成

skip_update.ps1 または skip_update.sh を実行するには、input.properties ファイルを作成し、環境の情報でファイルを更新します。ファイルをローカル・ディレクトリに保存します。このファイルのコンテンツは、オペレーティング・システムによって異なります。

Windows

```
username=exampleAdmin
password=examplePassword.epw
url=exampleURL
updateversions=01,04,07,10
podtype=test
```

Linux/UNIX

```
javahome=JAVA_HOME
epmautomatescript=EPM_AUTOMATE_LOCATION
username=exampleAdmin
password=examplePassword.epw
url=exampleURL
updateversions=01,04,07,10
podtype=test
```

Table 3-15 input.properties のパラメータ

パラメータ	説明
javahome	JAVA_HOME の場所。Linux/UNIX の場合のみ。
epmautomatescript	EPM 自動化の実行可能ファイル(epmautomate.sh)の絶対パス。Linux/UNIX の場合のみ。
username	サービス管理者のユーザー名
password	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
url	月次以外の更新頻度を設定する環境の URL。
updateversions	updateversions:url パラメータで識別される環境に適用される Cloud EPM 更新のカンマ区切りリスト。例: updateversions=02,05,08,11。 バージョンは 2 桁で指定する必要があります。更新 01 から 09 までは先頭にゼロを付けてください。スクリプトは、updateversions パラメータ値に含まれない更新に対して、skipUpdate コマンドを実行します。たとえば、updateversions=02,05,08,11 と指定すると、スクリプトは更新のスキップ・フラグを 01 (1 月)、03 (3 月)、04 (4 月)、06 (6 月)、07 (7 月)、09 (9 月)、10 (10 月)および 12 (12 月)更新に設定します。この場合、Cloud EPM の更新 02 (2 月)、05 (5 月)、08 (8 月)および 11 (11 月)が環境に適用されます。
podtype	Cloud EPM 環境タイプ。有効な値は、test および prod です。

スクリプトの実行

1. Windows および Linux/UNIX の場合:

- 前述の項のスクリプトをコピーして、skip_update.ps1 または skip_update.sh を作成します。
- input.properties ファイルを作成して、skip_update スクリプトが配置されているディレクトリに保存します。このファイルのコンテンツは、オペレーティング・システムによって異なります。[skip_update Windows および Linux/UNIX スクリプトを実行するための input.properties ファイルの作成](#)を参照してください。このディレクトリの書き込み権限があることを確認してください。Windows の場合、スクリプトを実行できるよう、「**管理者として実行**」オプションを使用して PowerShell を開始する必要があります。
- スクリプトを起動します。
 - **Windows PowerShell:** skip_update.ps1 を実行します。
 - **Linux/UNIX:** ./skip_update.sh を実行します。

2. Groovy:

- skip_update.groovy Groovy スクリプトを作成し、必要に応じて更新します。
- Cloud EPM ビジネス・プロセスで Groovy 画面を使用するか、[runBusinessRule](#) を使用してスクリプトの実行を自動化します。EPM 自動化を使用して Groovy スクリプトを実行する方法については、[EPM 自動化をインストールしないコマンドの実行](#)を参照してください。

Planning、連結、Tax Reporting および Enterprise Profitability and Cost Management 用のサンプル・シナリオ

この項にあるスクリプトは、Planning (Planning モジュールを含む)、Financial Consolidation and Close、Tax Reporting および Enterprise Profitability and Cost Management の環境でのタスクの自動化に役立ちます。

次も参照:

- [集約ストレージ・キューブからの大量のセル・エクスポートの自動化](#)
この項では、PowerShell または Bash スクリプトを使用して、集約ストレージ(ASO)キューブから大量のセルをエクスポートします。
- [アプリケーションへのメタデータのインポート](#)
これらのスクリプトを使用して、アプリケーション・メタデータをファイルから手動でインポートします。
- [データのインポート、計算スクリプトの実行、ブロック・ストレージ・データベースから集約ストレージ・データベースへのデータのコピー](#)
これらのスクリプトを使用して、ファイルからデータをインポートしてキューブをリフレッシュし、ビジネス・ルールを実行してキューブを計算し、データを ASO キューブにプッシュします。
- [メタデータおよびデータのエクスポートおよびダウンロード](#)
これらのスクリプトを使用して、アプリケーション・メタデータおよびデータをエクスポートした後、エクスポート・ファイルをローカル・ディレクトリにダウンロードします。
- [アプリケーション・データのエクスポートおよびダウンロード](#)
これらのスクリプトを使用して、アプリケーション・データをエクスポートした後、ローカル・ディレクトリにダウンロードします。

- アプリケーション監査レコードのアーカイブの自動化**
 この項の **Windows** スクリプトと **Linux** スクリプトを使用して、アプリケーション監査データをローカル・コンピュータにエクスポートおよびアーカイブするプロセスを自動化します。
- 環境へのデータ・ファイルのアップロードおよびデータ・ロード・ルールの実行**
 これらのスクリプトを使用して、ファイルを環境にアップロードした後、データ・ルールを実行して、データをファイルからアプリケーションにインポートします。
- 日次データ統合の自動化**
 このシナリオでは、データ統合を定期的に自動化するサンプル・スクリプトの使用方法を調べます。

集約ストレージ・キューブからの大量のセル・エクスポートの自動化

この項では、**PowerShell** または **Bash** スクリプトを使用して、集約ストレージ(ASO)キューブから大量のセルをエクスポートします。

Oracle Essbase の `QUERYRESULTLIMIT` によって制限が課されるため、ユーザー・インタフェースから大量のデータをエクスポートすることはできません。この項で使用可能な **PowerShell** スクリプトは、エクスポート操作を指定のジョブ数に分割して各ジョブを実行し、エクスポートしたデータをダウンロードします。次に、複数のエクスポート・ファイルを1つのエクスポート・ファイルに結合して、ヘッダーが1つのみ存在するようにします。

ノート:

これらのスクリプトは、既存の「データのエクスポート」タイプのジョブを実行します。ジョブを作成する詳細な手順は、*Planning* の *管理* のジョブの管理を参照してください。

PowerShell スクリプト

```
$user = '<USERNAME>'
$pass = '<PASSWORD>'
$serverURL = '<URL>'
$applicationName = '<APPLICATIONNAME>'
$cubeName = '<CUBENAME>'
$splitDimension = '<DIMENSION_TO_SPLIT_THE_EXPORT>'
$topLevelMemberForExport = '<TOP_MEMBER_FOR_EXPORT>'
$exportJobName = '<EXPORT_JOB_NAME>'
$exportFilePrefix = '<PREFIX_FOR_EXPORT_FILE>'
$columnMembers = '<MEMBERS_ON_COLUMNS>'
$povMembers = '<POV_MEMBERS>'
$numberOfExportFiles = <NUMBER_OF_FILES_TO_SPLIT_THE_EXPORT>

$memberArray = @()
$exportFileArray = @()

function getLevel0 ($parent) {
    $parent.children.ForEach({
        if ( $_.children.count -eq 0 ) {
            $script:memberArray += $_.name
        }
    })
}
```

```

    }
    getLevel0($_)
  })
}

function findMember ($tree, $memberName) {
  $subtree = ""
  if ($tree.name -eq $memberName){
    return $tree
  } else {
    $tree.children.ForEach({
      #Write-Host $_.name
      if ($subtree -eq ""){ $subtree = findMember $_ $memberName}
    })
    return $subtree
  }
}

#putting together base64 encoded authentication header based un user and
password
$encodedCredentials =
[Convert]::ToBase64String([System.Text.Encoding]::ASCII.GetBytes(($user) +
":" + $($pass)))
$headers = @{ Authorization = "Basic $encodedCredentials" }

#test login
$testRequest = $serverURL + '/HyperionPlanning/rest/v3/applications'

try {
  $response = Invoke-RestMethod -Uri $testRequest -Method Get -
Headers $headers -UseBasicParsing
}
catch {
  Write-Host $_
  return
}

#retrieve dimension hierarchy from application
Write-Host "Retrieving member list for split dimension " $splitDimension
$request = $serverURL + '/HyperionPlanning/rest/v3/internal/applications/'
+ $applicationName + '/plantypes/' + $cubeName + '/dimensions/'
+ $splitDimension
try {
  $response = Invoke-RestMethod -Uri $request -Method Get -Headers $headers
-UseBasicParsing
}
catch {
  Write-Host $_
  return
}
Write-Host $splitDimension " member list retrieved"

#search for the top of the export hierarchy
Write-Host "Searching for member " $stopLevelMemberForExport " in hierarchy"
$member = findMember $response $stopLevelMemberForExport
if ( $member.name -ne $stopLevelMemberForExport ) {

```

```

        Write-Host $stopLevelMemberForExport " not found in hierarchy, exiting ..."
        return 128
    }
    Write-Host "Found member " $stopLevelMemberForExport " in hierarchy"

    #retrieve level 0 members in export hierarchy
    Write-Host "Retrieving Level 0 members for hierarchy"
    getLevel0($member)
    if ( $memberArray.Length -eq 0 ) {
        Write-Host "no level 0 members found in hierarchy, exiting ..."
        return 128
    }
    Write-Host $memberArray.Length " Level 0 members for export hierarchy
    retrieved"

    $request = $serverURL + '/HyperionPlanning/rest/v3/applications/'
    + $applicationName + '/jobs'

    #splitting member list into the number of export files
    $numberOfEntitiesPerFile =
    [math]::truncate($memberArray.Length / $numberOfExportFiles)
    for ($i = 1; $i -le $numberOfExportFiles; $i++) {
        $memberList = ""
        $firstMember = ($i - 1) * $numberOfEntitiesPerFile
        if ($i -lt $numberOfExportFiles) {
            $lastMember = $i * $numberOfEntitiesPerFile
        } else {
            $lastMember = $i * $numberOfEntitiesPerFile + $memberArray.Length
        }
        % $numberOfExportFiles
        for ($j = $firstMember; $j -lt $lastMember; $j++) {
            $memberList += $memberArray[$j]
            if ($j -lt $lastMember - 1) {$memberList += ","} #avoid adding a
            comma (,) after the last member of each set
        }

        $jobDetails='
        {
        "jobType":"EXPORT_DATA","jobName":"' + $exportJobName + '",
        "parameters":{
            "exportFileName":"Export-' + $i + '.zip",
            "rowMembers":"' + $memberList + '",
            "columnMembers":"' + $columnMembers + '",
            "povMembers":"' + $povMembers + '"
        }
        }'

        #start export job
        try{
            $response = Invoke-RestMethod -Uri $request -Method Post -
            Headers $headers -Body $jobDetails -ContentType "application/json"
        } catch {
            Write-Host $_
            return
        }
    }

```

```

Write-Host "Started export job " $i " out of " $numberOfExportFiles

#checking job status, continue once jos is completed
$statusRequest = $serverURL + '/HyperionPlanning/rest/v3/applications/'
+ $applicationName + '/jobs/' + $response.jobId
$statusResponse = Invoke-RestMethod -Uri $statusRequest -Method Get -
Headers $headers -UseBasicParsing

while ( $statusResponse.descriptiveStatus -eq "Processing" ) {
    Write-Host $statusResponse.descriptiveStatus
    Start-Sleep -s 10
    $statusResponse = Invoke-RestMethod -Uri $statusRequest -Method Get -
Headers $headers -UseBasicParsing
}
Write-Host $statusResponse.descriptiveStatus

Write-Host "Downloading export file ..."
$downloadRequest = $serverURL + '/interop/rest/11.1.2.3.600/
applicationsnapshots/Export-' + $i + '.zip/contents'
$statusResponse = Invoke-RestMethod -Uri $downloadRequest -Method Get -
Headers $headers -OutFile "$exportFilePrefix-$i.zip"

Write-Host "Expanding archive ..."
Expand-Archive -Force -LiteralPath "$exportFilePrefix-$i.zip" -
DestinationPath "$exportFilePrefix-$i"
Remove-Item "$exportFilePrefix-$i.zip"

Get-ChildItem -Path "$exportFilePrefix-$i" -File -Name | ForEach-Object
{ $exportFileArray += "$exportFilePrefix-$i\" + $_ }
}

Write-Host "creating outputfile ..."
#write header to outputfile
Get-Content $exportFileArray[0] | Select-Object -First 1 | Out-File
"$exportFilePrefix.csv"

#write content to outputfile skipping header
ForEach ($exportFile in $exportFileArray) {
    Get-Content $exportFile | Select-Object -Skip 1 | Out-File -Append
"$exportFilePrefix.csv"
}

Compress-Archive -LiteralPath "$exportFilePrefix.csv" -DestinationPath
"$exportFilePrefix.zip"

Write-Host "cleaning up ..."
Remove-Item "$exportFilePrefix-*" -Recurse
Remove-Item "$exportFilePrefix.csv"

```

Bash スクリプト

```

#!/bin/bash

user='<USERNAME>'

```

```

pass='<PASSWORD>'
serverURL='<URL>'
applicationName='<APPLICATIONNAME>'
cubeName='<CUBENAME>'
splitDimension='<DIMENSION_TO_SPLIT_THE_EXPORT>'
topLevelMemberForExport='<TOP_MEMBER_FOR_EXPORT>'
exportJobName='<EXPORT_JOB_NAME>'
exportFilePrefix='<PREFIX_FOR_EXPORT_FILE>'
columnMembers='<MEMBERS_ON_COLUMNS>'
povMembers='<POV_MEMBERS>'
numberOfExportFiles='<NUMBER_OF_FILES_TO_SPLIT_THE_EXPORT>'

getRowMembers() {
    local memberList="$1"
    local firstMember=$2
    local lastMember=$3
    local nameCount=0
    local rowMember=""
    local rowMembers=""

    while IFS= read -r line
    do
        if [[ "${line}" == *"name"* ]]
        then
            if [[ ${nameCount} -ge ${firstMember} ]] && [[ ${nameCount} -lt $
{lastMember} ]]
            then
                rowMember=$(echo "${line}" | cut -d':' -f2- | sed s'/',,/'g')
                rowMembers="${rowMembers}${rowMember},"
            fi
            ((nameCount+=1))
        fi
    done <<< "${memberList}"
    rowMembers=$(echo "${rowMembers}" | rev | cut -d',' -f2- | rev)
    echo "${rowMembers}"
}

getLevel0()
{
    local memberList="$1"
    local names=$(echo "${memberList}" | jq 'recurse (try .children[])
| .name' | sed -e 's"/,/'g')
    local elements=""

    formerIFS=$IFS
    IFS=$'\n'
    namesarr=( $names )
    IFS=$formerIFS

    for i in ${!namesarr[@]}
    do
        testelement=$(echo "${memberList}" | jq --arg currentName "$
{namesarr[i]}" 'recurse (try .children[]) | select(.name==$currentName)')
        if [[ "${testelement}" != *"children"* ]]
        then
            elements="${elements}${testelement}"
        fi
    done
}

```

```

        fi
    done

    echo "${elements}"
}

#test login
header="Content-Type: application/x-www-form-urlencoded"
applicationsRequest="${serverURL}/HyperionPlanning/rest/v3/applications"
response=$(curl -X "GET" -s -w "%{http_code}" -u "${user}:${pass}" -H "${header}" "${applicationsRequest}")
http_response_code=$(echo "${response}" | rev | cut -d'}' -f1 | rev)

if [ ${http_response_code} -ne 200 ]
then
    echo "${response}"
    exit
fi

#retrieve dimension hierarchy from application
echo "Retrieving member list for split dimension ${splitDimension}"
splitDimensionRequest="${serverURL}/HyperionPlanning/rest/v3/internal/
applications/${applicationName}/plantypes/${cubeName}/dimensions/${
splitDimension}"
response=$(curl -X GET -s -w "%{http_code}" -u "${user}:${pass}" -o "response-
memberlist.txt" -D "respHeader-memberlist.txt" -H "${header}" "${
splitDimensionRequest}")
http_response_code=$(echo "${response}" | rev | cut -d'}' -f1 | rev)

if [ ${http_response_code} -ne 200 ]
then
    echo "${response}"
    exit
fi

echo "${splitDimension} member list retrieved"

#search for the top of the export hierarchy
echo "Searching for member ${topLevelMemberForExport} in hierarchy"
memberList=$(cat response-memberlist.txt | jq --arg topLevelMember "${
topLevelMemberForExport}" 'recurse(try .children[]) | select (.name
== $topLevelMember)')
if [[ "${memberList}" == "" ]]
then
    echo "${topLevelMemberForExport} not found in hierarchy, exiting ..."
    exit 128
fi

echo "Found member ${topLevelMemberForExport} in hierarchy"

#retrieve level 0 members in export hierarchy
echo "Retrieving Level 0 members for hierarchy"
totalCount=$(echo "${memberList}" | grep "name" | wc -l)
grepChildrenCount=$(echo "${memberList}" | grep "children" | wc -l)
levelZeroCount=$((totalCount-grepChildrenCount))

```

```

if [[ "${levelZeroCount}" -eq 0 ]]
then
    echo "no level 0 members found in hierarchy, exiting ..."
    exit 128
fi

echo "${levelZeroCount} Level 0 members for export hierarchy retrieved"

#splitting member list into the number of export files
numberOfEntitiesPerFile=$((levelZeroCount/numberOfExportFiles))
jobsRequest="${serverURL}/HyperionPlanning/rest/v3/applications/${
applicationName}/jobs"
header="Content-Type: application/json"

for ((i = 1 ; i <= ${numberOfExportFiles}; i++))
do
    firstMember=$((i-1)*numberOfEntitiesPerFile)
    if [[ $i -lt ${numberOfExportFiles} ]]
    then
        lastMember=$((i*numberOfEntitiesPerFile))
    else
        lastMember=$((i*numberOfEntitiesPerFile+levelZeroCount%numberOfExportFiles))
    fi

    elements=$(getLevel0 "${memberList}")
    rowMembers=$(getRowMembers "${elements}" ${firstMember} ${lastMember})

    response=$(curl -X POST -s -w "%{http_code}" -u "${user}:${pass}" -o
"response-job.txt" -D "respHeader-job.txt" -H "${header}" "${jobsRequest}" -d
'{"jobType":"EXPORT_DATA","jobName":"${exportJobName}","parameters":
{"exportFileName":"Export-${i}.zip","rowMembers":"${
rowMembers}","columnMembers":"${columnMembers}","povMembers":"${
povMembers}"})

    echo "Started export job " $i " out of " $numberOfExportFiles
    jobId=$(cat response-job.txt | grep -o "jobId:[^, }]*" | cut -d':' -f2)
    descriptiveStatus=$(cat response-job.txt | grep -o "descriptiveStatus":
[^, }]*" | cut -d':' -f2 | sed -e 's//g')
    jobIdRequest="${serverURL}/HyperionPlanning/rest/v3/applications/${
applicationName}/jobs/${jobId}"
    response=$(curl -X GET -s -w "%{http_code}" -u "${user}:${pass}" -o
"response-jobstatus.txt" -D "respHeader-jobstatus.txt" -H "${header}" "${
jobIdRequest}")

    jobId=$(cat response-jobstatus.txt | grep -o "jobId:[^, }]*" | cut -
d':' -f2)
    descriptiveStatus=$(cat response-jobstatus.txt | grep -o
"descriptiveStatus":[^, }]*" | cut -d':' -f2 | sed -e 's//g')

    while [[ "${descriptiveStatus}" == "Processing" ]]
    do
        echo "${descriptiveStatus}"
        sleep 10
        response=$(curl -X GET -s -w "%{http_code}" -u "${user}:${pass}" -o
"response-jobstatus.txt" -D "respHeader-jobstatus.txt" -H "${header}" "${

```

```

{jobIdRequest}")
    descriptiveStatus=$(cat response-jobstatus.txt | grep -o
'"descriptiveStatus":[^\, ]*' | cut -d':' -f2 | sed -e 's//g')
    done

    echo "${descriptiveStatus}"

    echo "Downloading export file ..."
    contentsRequest="${serverURL}/interop/rest/11.1.2.3.600/
applicationsnapshots/Export-${i}.zip/contents"
    curl -X GET -s -w "%{http_code}" -u "${user}:${pass}" -D "respHeader-
download.txt" "${contentsRequest}" > "${exportFilePrefix}-${i}.zip"

    echo "Expanding archive ..."
    unzip "${exportFilePrefix}-${i}.zip" -d "${exportFilePrefix}-${i}"
    rm "${exportFilePrefix}-${i}.zip"

    echo "Writing to outputfile ..."
    if [[ -d "${exportFilePrefix}-${i}" ]]
    then
        find "${exportFilePrefix}-${i}" -name \*.csv | xargs cat | tail -n +2
    >> "${exportFilePrefix}.csv"
    fi
done
zip "${exportFilePrefix}.zip" "${exportFilePrefix}.csv"

echo "cleaning up ..."
find . -name "${exportFilePrefix}-*" | xargs rm -r
rm "${exportFilePrefix}.csv"

```

集約ストレージ(ASO)キューブから大量のセルをエクスポートするには:

1. PowerShell または Bash スクリプトをコピーして、ASOCellExport.ps1 や ASOCellExport.sh などのファイル・システムに保存します。
2. スクリプト・ファイルを変更して、パラメータ値を設定します。詳細は、次の表を参照してください。

表 3-16 PowerShell および Bash スクリプトに含める変数の値

変数	説明
user	<p>DOMAIN.USER 形式のサービス管理者のドメインおよびユーザー名。</p> <p>例:</p> <p>Windows: \$user = 'exampleDomain.jDoe'</p> <p>Linux/UNIX: user = 'exampleDomain.jDoe'</p>
pass	<p>サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの場所暗号化されたパスワード・ファイルの作成の詳細は、encrypt コマンドを参照してください。</p> <p>例:</p> <p>Windows: \$pass = 'Example'</p> <p>Linux/UNIX: pass = 'Example'</p>

表 3-16 (続き) PowerShell および Bash スクリプトに含める変数の値

変数	説明
serverURL	Oracle Fusion Cloud Enterprise Performance Management 環境の URL。 例: Windows: \$serverURL = 'https://example.oraclecloud.com' Linux/UNIX: serverURL = 'https://example.oraclecloud.com'
applicationName	Planning、Financial Consolidation and Close、Tax Reporting または Enterprise Profitability and Cost Management アプリケーションの名前。 例: Windows: \$applicationName = 'Vision' Linux/UNIX: applicationName = 'Vision'
cubeName	アプリケーションのキューブの名前。 例: Windows: \$cubeName = 'VisASO' Linux/UNIX: cubeName = 'VisASO'
splitDimension	エクスポートを複数のグループに分割するためにそのメンバーが使用されるディメンションの名前。 例: Windows: \$splitDimension = 'Account' Linux/UNIX: splitDimension = 'Account'
topLevelMemberForExport	レベル 0 のメンバーのリストが作成されるディメンション・サブ階層のメンバーの名前。 例: Windows: \$topLevelMemberForExport = 'Total Cash Flow' Linux/UNIX: topLevelMemberForExport = 'Total Cash Flow'
exportJobName	既存の「データのエクスポート」タイプのジョブの名前。このジョブに指定された設定は、スクリプトに設定したパラメータによって上書きされます。 例: Windows: \$exportJobName = 'ASO Cell Export' Linux/UNIX: exportJobName = 'ASO Cell Export'
exportFilePrefix	エクスポート・ジョブによって生成されるファイルを一意に識別するファイル名の接頭辞。 例: Windows: \$exportFilePrefix = 'cashflow' Linux/UNIX: exportFilePrefix = 'cashflow'
columnMembers	エクスポートに含めるメンバーの列。 例: Windows: \$columnMembers = 'Period' Linux/UNIX: columnMembers = 'Period'

表 3-16 (続き) PowerShell および Bash スクリプトに含める変数の値

変数	説明
povMembers	<p>エクスポートに含める視点。POV メンバーには他のすべてのディメンションが含まれる必要があり、次に示すように関数を含めることができます。</p> <p>ILvl0Descendants (YearTotal), ILvl0Descendants (Year), ILvl0Descendants (Scenario), ILvl0Descendants (Version), ILvl0Descendants (P_TP), ILvl0Descendants (AltYear)</p> <p>例: Windows: \$povMembers = 'YTD' Linux/UNIX: povMembers = 'YTD'</p>
numberOfExportFiles	<p>このエクスポート操作で実行するジョブの数。問合せ制限のためにエクスポートが失敗する場合は、この数を増やします。</p> <p>例: Windows: \$numberOfExportFiles = 3 Linux/UNIX: numberOfExportFiles = 3</p>

3. Windows スケジューラまたは cron ジョブを使用して、都合のよい時間にスクリプトを実行するようにスケジュールします。詳細なステップは、[スクリプトの実行の自動化](#)を参照してください。

アプリケーションへのメタデータのインポート

これらのスクリプトを使用して、アプリケーション・メタデータをファイルから手動でインポートします。

これらのスクリプトでは、次のアクティビティが実行されます:

- 環境にサインインします。
- メタデータをアップロードします。
- ジョブを使用して、アップロードされたファイルからアプリケーションにメタデータをインポートします。
- キューブをリフレッシュします。
- サインアウトします。

Windows のサンプル・スクリプト

次のスクリプトをコピーして、importMetadata.ps1 を作成します。それをローカル・ディレクトリに保存します。

```
$inputproperties = ConvertFrom-StringData (Get-Content ./input.properties -raw)
$username="$($inputproperties.username) "
$passwordfile="$($inputproperties.passwordfile) "
$serviceURL="$($inputproperties.serviceURL) "
$file1="$($inputproperties.file1) "
$jobName="$($inputproperties.jobName) "
```

```
epmautomate login ${username} ${passwordfile} ${serviceURL}
```

```

epmautomate uploadfile ${file1}
epmautomate importmetadata ${jobName} ${file1}
epmautomate refreshcube
epmautomate logout

```

Linux/UNIX のサンプル・スクリプト

次のスクリプトをコピーして、importMetadata.sh を作成します。それをローカル・ディレクトリに保存します。

```

#!/bin/bash
. ./input.properties
export JAVA_HOME=${javahome}
${epmautomatescript} login "${username}" "${passwordfile}" "${serviceURL}"
${epmautomatescript} uploadfile "${file1}"
${epmautomatescript} importmetadata "${jobName}" "${file1}"
${epmautomatescript} refreshcube
${epmautomatescript} logout

```

input.properties ファイルの作成

次のいずれかをコピーし、環境の情報で更新することにより、input.properties ファイルを作成します。importMetadata.ps1 または importMetadata.sh が格納されているディレクトリにファイルを保存します。

Windows

```

username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
File1=FILE_NAME.zip
jobName=JOB_NAME

```

Linux/UNIX

```

javahome=JAVA_HOME
epmautomatescript=EPM_AUTOMATE_LOCATION
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
File1=FILE_NAME.zip
jobName=JOB_NAME

```

表 3-17 input.properties のパラメータ

パラメータ	説明
javahome	JAVA_HOME の場所。Linux/UNIX の場合のみ。
epmautomatescript	EPM 自動化の実行可能ファイル(epmautomate.sh)の絶対パス。Linux/UNIX の場合のみ。
username	アイデンティティ・ドメイン管理者の役割も保持するサービス管理者のユーザー名。

表 3-17 (続き) input.properties のパラメータ

パラメータ	説明
password	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
serviceURL	スナップショットを生成する環境の URL。
File1	インポートするメタデータが格納された ZIP ファイルの名前。
JobName	メタデータのインポートに使用するジョブ。

スクリプトの実行

1. 前述の項のスクリプトをコピーして、importMetadata.ps1 または importMetadata.sh を作成します。
2. input.properties ファイルを作成し、importMetadata スクリプトが配置されているディレクトリに保存します。このファイルのコンテンツは、オペレーティング・システムによって異なります。[input.properties ファイルの作成](#)を参照してください。このディレクトリの書込み権限があることを確認してください。Windows の場合、スクリプトを実行できるよう、「**管理者として実行**」オプションを使用して PowerShell を開始する必要があります。
3. スクリプトを起動します。
 - **Windows PowerShell:** importMetadata.ps1 を実行します。
 - **Linux/UNIX:** ./importMetadata.sh を実行します。

データのインポート、計算スクリプトの実行、ブロック・ストレージ・データベースから集約ストレージ・データベースへのデータのコピー

これらのスクリプトを使用して、ファイルからデータをインポートしてキューブをリフレッシュし、ビジネス・ルールを実行してキューブを計算し、データを ASO キューブにプッシュします。

これらのスクリプトでは、次のアクションが実行されます:

- 環境にサインインします。
- ファイル data.csv をアップロードします。
- ジョブ loadingq1data を使用して data.csv からアプリケーションにデータをインポートします。
- キューブをリフレッシュします。
- ビジネス・ルールを実行してデータを変換します。
- ジョブを使用して、データを集約ストレージ・データベースにプッシュします。
- サインアウトします。

Windows のサンプル・スクリプト

このスクリプトをコピーして、importDataPlus.ps1 を作成します。それをローカル・ディレクトリに保存します。

```
$inputproperties = ConvertFrom-StringData (Get-Content ./input.properties -raw)
$username="$($inputproperties.username) "
$passwordfile="$($inputproperties.passwordfile) "
$serviceURL="$($inputproperties.serviceURL) "
$importDataJobName="$($inputproperties.importDataJobName) "
$businessRuleName="$($inputproperties.businessRuleName) "
$planTypeMapName="$($inputproperties.planTypeMapName) "
$params1Key="$($inputproperties.params1Key) "
$params1Value="$($inputproperties.params1Value) "
$params2Key="$($inputproperties.params2Key) "
$params2Value="$($inputproperties.params2Value) "
$clearData="$($inputproperties.clearData) "

epmautomate login ${username} ${passwordfile} ${serviceURL}
epmautomate uploadfile ${file1}
epmautomate importdata ${importDataJobName} ${file1}
epmautomate refreshcube
epmautomate runbusinessrule ${businessRuleName} ${params1Key}=${params1Value} ${params2Key}=${params2Value}
epmautomate runplantypemap ${planTypeMapName} clearData=${clearData}
epmautomate logout
```

Linux/UNIX のサンプル・スクリプト

このスクリプトをコピーして、importDataPlus.ps1 を作成します。それをローカル・ディレクトリに保存します。

```
#!/bin/bash
. ./input.properties
export JAVA_HOME=${javahome}
${epmautomatescript} login "${username}" "${passwordfile}" "${serviceURL}"
${epmautomatescript} uploadfile "${file1}"
${epmautomatescript} importdata "${importDataJobName}" "${file1}"
${epmautomatescript} refreshcube
${epmautomatescript} runbusinessrule "${businessRuleName}" "${params1Key}=${params1Value}" "${params2Key}=${params2Value}"
${epmautomatescript} runplantypemap "${planTypeMapName}" clearData=${clearData}
${epmautomatescript} logout
```

input.properties ファイルの作成

Windows

```
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
file1=FILE_NAME.csv
importDataJobName=FILE_NAME
```

```

businessRuleName=RULE_NAME
planTypeMapName=PLAN_TYPE_MAP_NAME
param1Key=RUN-TIME_PARAMETER_1
param1Value=RUN-TIME_PARAMETER_1_VALUE
param2Key=RUN-TIME_PARAMETER_2
param2Value=RUN-TIME_PARAMETER_2_VALUE
clearData=true

```

Linux/UNIX

```

javahome=JAVA_HOME
epmautomatescript=EPM_AUTOMATE_LOCATION
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
File1=FILE_NAME.csv
importDataJobName=FILE_NAME
businessRuleName=RULE_NAME
planTypeMapName=PLAN_TYPE_MAP_NAME
param1Key=RUN-TIME_PARAMETER_1
param1Value=RUN-TIME_PARAMETER_1_VALUE
param2Key=RUN-TIME_PARAMETER_2
param2Value=RUN-TIME_PARAMETER_2_VALUE
clearData=true

```

表 3-18 input.properties のパラメータ

パラメータ	説明
javahome	JAVA_HOME の場所。Linux/UNIX の場合のみ。
epmautomatescript	EPM 自動化の実行可能ファイル(epmautomate.sh)の絶対パス。Linux/UNIX の場合のみ。
username	アイデンティティ・ドメイン管理者の役割も保持するサービス管理者のユーザー名。
password	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
serviceURL	スナップショットを生成する環境の URL。
File1	データをアプリケーションにロードするインポート・ファイル。
importDataJobName	データのインポートに使用するジョブの名前。
businessRuleName	インポートされたデータ上で実行されるビジネス・ルール
planTypeMapName	BSO データベースから ASO データベースに、または BSO データベースから別の BSO データベースにデータをコピーするのに使用するプラン・タイプ・マップ。
param1Key	ビジネス・ルールを実行するための実行時プロンプト 1。
param1Value	実行時プロンプト 1 の値。
param2Key	ビジネス・ルールを実行するための実行時プロンプト 2。
param2Value	実行時プロンプト 2 の値。
clearData	受入れデータベース内のデータが削除されるかどうかを示します。データを保持するには、false を指定します。

スクリプトの実行

1. 前述の項のスクリプトをコピーして、importDataPlus.ps1 または importDataPlus.sh を作成します。
2. input.properties ファイルを作成して、importDataPlus スクリプトが配置されているディレクトリに保存します。このファイルのコンテンツは、オペレーティング・システムによって異なります。[input.properties ファイルの作成](#)を参照してください。このディレクトリの書込み権限があることを確認してください。Windows の場合、スクリプトを実行できるように、「**管理者として実行**」オプションを使用して PowerShell を開始する必要があります。
3. スクリプトを起動します。
 - **Windows PowerShell:** importDataPlus.ps1 を実行します。
 - **Linux/UNIX:** ./importDataPlus.sh を実行します。

メタデータおよびデータのエクスポートおよびダウンロード

これらのスクリプトを使用して、アプリケーション・メタデータおよびデータをエクスポートした後、エクスポート・ファイルをローカル・ディレクトリにダウンロードします。

これらのスクリプトでは、次のアクティビティを完了します:

- 環境にサインインします。
- 指定したジョブを使用して、メタデータを zip ファイルにエクスポートします。
- 指定したジョブを使用して、アプリケーション・データを zip ファイルにエクスポートします。
- 受信/送信ボックスの内容をリストします。
- エクスポートされたデータ・ファイルをローカル・コンピュータにダウンロードします。
- サインアウトします。

Windows のサンプル・スクリプト

次のスクリプトをコピーして、exportDownloadMetadataAndData.ps1 を作成します。それをローカル・ディレクトリに保存します。

```
$inputproperties = ConvertFrom-StringData(Get-Content ./input.properties -raw)
$username="$($inputproperties.username) "
$passwordfile="$($inputproperties.passwordfile) "
$serviceURL="$($inputproperties.serviceURL) "
$exportFile1="$($inputproperties.exportFile1) "
$exportFile2="$($inputproperties.exportFile2) "
$exportMetaJobName="$($inputproperties.exportMetaJobName) "
$exportDataJobName="$($inputproperties.exportDataJobName) "

epmautomate login ${username} ${passwordfile} ${serviceURL}
epmautomate exportmetadata ${exportMetaJobName} ${exportFile1}
epmautomate exportdata ${exportDataJobName} ${exportFile2}
epmautomate listfiles
epmautomate downloadfile ${exportFile1}
```

```
epmautomate downloadfile f${exportFile2}
epmautomate logout
```

Linux/UNIX のサンプル・スクリプト

次のスクリプトをコピーして、exportDownloadMetadataAndData.sh を作成します。それをローカル・ディレクトリに保存します。

```
#!/bin/bash
. ./input.properties
export JAVA_HOME=${javahome}
${epmautomatescript} login "${username}" "${passwordfile}" "${serviceURL}"
${epmautomatescript} exportmetadata "${exportMetaJobName}" "${exportFile1}"
${epmautomatescript} exportdata "${exportDataJobName}" "${exportFile2}"
${epmautomatescript} listfiles
${epmautomatescript} downloadfile "${exportFile1}"
${epmautomatescript} downloadfile "${exportFile2}"
${epmautomatescript} logout
```

プロパティ・ファイルの作成

次のいずれかをコピーし、環境の情報で更新することにより、input.properties ファイルを作成します。exportDownloadMetadataAndData.ps1 または exportDownloadMetadataAndData.sh が格納されているディレクトリにファイルを保存します。

Windows

```
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
exportFile1=FILE_NAME1.zip
exportFile2=FILE_NAME2.zip
exportMetaJobName=METADATA_EXPORT_JOB_NAME
exportDataJobName=DATA_EXPORT_JOB_NAME
```

Linux/UNIX

```
javahome=JAVA_HOME
epmautomatescript=EPM_AUTOMATE_LOCATION
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
exportFile1=FILE_NAME1.zip
exportFile2=FILE_NAME2.zip
exportMetaJobName=METADATA_EXPORT_JOB_NAME
exportDataJobName=DATA_EXPORT_JOB_NAME
```

表 3-19 input.properties のパラメータ

パラメータ	説明
javahome	JAVA_HOME の場所。Linux/UNIX の場合のみ。

表 3-19 (続き) input.properties のパラメータ

パラメータ	説明
epmautomatescript	EPM 自動化の実行可能ファイル(epmautomate.sh)の絶対パス。 Linux/UNIX の場合のみ。
username	アイデンティティ・ドメイン管理者の役割も保持するサービス管理者のユーザー名。
password	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
serviceURL	スナップショットを生成する環境の URL。
exportFile1	メタデータのエクスポート先のファイルの名前。
exportFile2	データのエクスポート先のファイルの名前。
exportDataJobName1	メタデータのエクスポートに使用するジョブ。
exportDataJobName2	データのエクスポートに使用するジョブ。

スクリプトの実行

1. 前述の項のスクリプトをコピーして、exportDownloadMetadataAndData.ps1 または exportDownloadMetadataAndData.sh を作成します。
2. input.properties ファイルを作成して、exportDownloadMetadataAndData スクリプトが配置されているディレクトリに保存します。このファイルのコンテンツは、オペレーティング・システムによって異なります。[プロパティ・ファイルの作成](#)を参照してください。このディレクトリの書込み権限があることを確認してください。Windows の場合、スクリプトを実行できるよう、「**管理者として実行**」オプションを使用して PowerShell を開始する必要があります。
3. スクリプトを起動します。
 - **Windows PowerShell:** exportDownloadMetadataAndData.ps1 を実行します。
 - **Linux/UNIX:** ./exportDownloadMetadataAndData.sh を実行します。

アプリケーション・データのエクスポートおよびダウンロード

これらのスクリプトを使用して、アプリケーション・データをエクスポートした後、ローカル・ディレクトリにダウンロードします。

これらのスクリプトは、次の操作を実行します:

- 環境にサインインします。
- 指定したジョブを使用して、2 セットのデータをバックアップします。
- エクスポートされたデータ・ファイルをダウンロードします。
- サインアウトします。

Windows のサンプル・スクリプト

このスクリプトをコピーして、`exportDownloadData.ps1` を作成します。それをローカル・ディレクトリに保存します。

```
$inputproperties = ConvertFrom-StringData(Get-Content ./input.properties -raw)
$username="$($inputproperties.username) "
$passwordfile="$($inputproperties.passwordfile) "
$serviceURL="$($inputproperties.serviceURL) "
$exportFile1="$($inputproperties.exportFile1) "
$exportFile2="$($inputproperties.exportFile2) "
$exportDataJobName1="$($inputproperties.exportDataJobName1) "
$exportDataJobName2="$($inputproperties.exportDataJobName2) "

epmautomate login ${username} ${passwordfile} ${serviceURL}
epmautomate exportdata ${exportDataJobName1} ${exportFile1}
epmautomate exportdata ${exportDataJobName2} ${exportFile2}
epmautomate listfiles
epmautomate downloadfile ${exportFile1}
epmautomate downloadfile ${exportFile2}
epmautomate logout
```

Linux/UNIX のサンプル・スクリプト

このスクリプトをコピーして、`exportDownloadData.sh` を作成します。それをローカル・ディレクトリに保存します。

```
#!/bin/bash
. ./input.properties
export JAVA_HOME=${javahome}
${epmautomatescript} login "${username}" "${passwordfile}" "${serviceURL}"
${epmautomatescript} exportdata "${exportDataJobName1}" "${exportFile1}"
${epmautomatescript} exportdata "${exportDataJobName2}" "${exportFile2}"
${epmautomatescript} listfiles
${epmautomatescript} downloadfile "${exportFile1}"
${epmautomatescript} downloadfile "${exportFile2}"
${epmautomatescript} logout
```

input.properties ファイルの作成

次のいずれかをコピーし、環境の情報で更新することにより、`input.properties` ファイルを作成します。`exportDownloadData.ps1` または `exportDownloadData.sh` が格納されているディレクトリにファイルを保存します。

Windows

```
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
exportFile1=FILE_NAME.zip
exportFile2=FILE_NAME.zip
exportDataJobName1=JOB_NAME
exportDataJobName2=FILE_NAME
```

Linux/UNIX

```

javahome=JAVA_HOME
epmautomatescript=EPM_AUTOMATE_LOCATION
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
exportFile1=FILE_NAME.zip
exportFile2=FILE_NAME.zip
exportDataJobName1=FILE_NAME
exportDataJobName2=FILE_NAME

```

表 3-20 input.properties のパラメータ

パラメータ	説明
javahome	JAVA_HOME の場所。Linux/UNIX の場合のみ。
epmautomatescript	EPM 自動化の実行可能ファイル(epmautomate.sh)の絶対パス。Linux/UNIX の場合のみ。
username	アイデンティティ・ドメイン管理者の役割も保持するサービス管理者のユーザー名。
password	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
serviceURL	スナップショットを生成する環境の URL。
exportFile1 および exportFile2	データのエクスポート先のファイルの名前。
exportDataJobName1 および exportDataJobName2	データのエクスポートに使用するジョブ。

スクリプトの実行

1. 前述の項のスクリプトをコピーして、exportDownloadData.ps1 または exportDownloadData.sh を作成します。
2. input.properties ファイルを作成して、exportDownloadData スクリプトが配置されているディレクトリに保存します。このファイルのコンテンツは、オペレーティング・システムによって異なります。表 1 を参照してください。このディレクトリの書込み権限があることを確認してください。Windows の場合、スクリプトを実行できるよう、「**管理者として実行**」オプションを使用して PowerShell を開始する必要があります。
3. スクリプトを起動します。
 - **Windows PowerShell:** exportDownloadData.ps1 を実行します。
 - **Linux/UNIX:** ./exportDownloadData.sh を実行します。

アプリケーション監査レコードのアーカイブの自動化

この項の Windows スクリプトと Linux スクリプトを使用して、アプリケーション監査データをローカル・コンピュータにエクスポートおよびアーカイブするプロセスを自動化します。

アプリケーション監査データは 365 日間のみ保持されます。これらのスクリプトをカスタマイズし、180 日ごとに 1 回、またはデータ保持ポリシーに規定されているとおりに実行して、365 日より前の履歴監査データが失われないようにします。

 **Note:**

これらのスクリプトは、ローカル・ストレージにデータをアーカイブするように調整されています。これらを変更して、エクスポートされた監査データ・ファイルをネットワーク・ストレージまたはストレージ・クラウド(Oracle Object Storage など)にアーカイブできます。

Table 3-21 パラメータとその値

パラメータ	値
url	環境の URL。 例: <ul style="list-style-type: none"> Windows: set url=https://example-epmidm.epm.usphoenix-1.ocs.oraclecloud.com/epmcloud Linux: url=https://example-epmidm.epm.usphoenix-1.ocs.oraclecloud.com/epmcloud
user	環境にサインインして監査データをダウンロードするサービス管理者のユーザー名。 例: <ul style="list-style-type: none"> Windows: set user=ExampleAdmin Linux: user=ExampleAdmin
password	サービス管理者のパスワード(非推奨)または暗号化されたパスワード・ファイルの名前と場所。暗号化されたパスワード・ファイルの作成の詳細は、 encrypt コマンドを参照してください。 例: <ul style="list-style-type: none"> Windows: set password="C:\mySecuredir\password.epw" Linux: password="/home/user1/mySecuredir/password.epw"
AuditFileName	監査データ・ファイルの名前。このファイルを一意にするために、スクリプトによって監査データのエクスポートのタイムスタンプがこのファイル名に追加されます。 例: <ul style="list-style-type: none"> Windows: set AuditFileName=AuditData Linux: AuditFileName=AuditData
NumberOfBackups	ストレージに保持するバックアップ・ファイルの数。デフォルトは 10 です。その後は必要に応じて最も古いバックアップが置き換えられます。 例: <ul style="list-style-type: none"> Windows: set NumberOfBackups=20 Linux: NumberOfBackups=20
Linux スクリプトの場合のみ	
epmautomatescript	EPM 自動化がインストールされている場所。 例: /home/user1/epmautomate/bin/epmautomate.sh
javahome	JAVA_HOME の場所。 例: /home/user1/jdk1.8.0_191/

Windows スクリプト

ローカル・コンピュータへの監査データのエクスポートとダウンロードを自動化する次のようなスクリプトが含まれるバッチ・ファイル(AuditExport.bat など)を作成します。

```
@echo off
rem Sample script to download and maintain 10 audit data backups
rem Update the following parameters

SET url=https://example.oraclecloud.com
SET user=ServiceAdmin
SET password=Example.epw
SET AuditFileName="AuditBackup"
SET NumberOfBackups=10

rem EPM Automate commands
call epmautomate login %user% %password% %url%
    IF %ERRORLEVEL% NEQ 0 goto :ERROR
        call epmautomate exportAppAudit %AuditFileName% nDays=180
    IF %ERRORLEVEL% NEQ 0 goto :ERROR
        call epmautomate downloadfile %AuditFileName%.zip
    IF %ERRORLEVEL% NEQ 0 goto :ERROR
        call epmautomate logout
    IF %ERRORLEVEL% NEQ 0 goto :ERROR

rem Rename downloaded audit data backup, keep the last 10 backups
Set Timestamp=%date:~4,2%_%date:~7,2%_%date:~10,2%
Set Second=%time:~0,2%%time:~3,2%
ren %AuditFileName%.zip %AuditFileName%_%Timestamp%_%Second%.zip
SET Count=0
FOR %%A IN (%AuditFileName%*.*) DO SET /A Count += 1
IF %Count% gtr %NumberOfBackups% FOR %%A IN (%AuditFileName%*.*) DO del "%A"
&& GOTO EOF
:EOF

echo Scheduled Task Completed successfully
exit /b %errorlevel%
:ERROR
echo Failed with error #%errorlevel%.
exit /b %errorlevel%
```

Linux スクリプト

ローカル・コンピュータへの監査データのエクスポートとダウンロードを自動化する次のようなスクリプトが含まれるシェル・スクリプト(AuditExport.sh など)を作成します。

```
#!/bin/sh
# Sample script to export, download and maintain 10 audit data backups
# Update the following seven parameters
url=https://example.oraclecloud.com
user=serviceAdmin
password=/home/user1/epmautomate/bin/example.epw
auditfilename="AuditBackup"
```

```

numberofbackups=10
epmautomatescript=/home/user1/epmautomate/bin/epmautomate.sh
javahome=/home/user1/jdk1.8.0_191/

export JAVA_HOME=${javahome}

printResult()
{
    op="$1"
    opoutput="$2"
    returncode="$3"

    if [ "${returncode}" -ne 0 ]
    then
        echo "Command failed. Error code: ${returncode}. ${opoutput}"
    else
        echo "${opoutput}"
    fi
}

processCommand()
{
    op="$1"
    date=`date`

    echo "Running ${epmautomatescript} ${op}"
    operationoutput=`eval "$epmautomatescript $op"`
    printResult "$op" "$operationoutput" "$?"
}

op="login ${user} ${password} ${url}"
processCommand "${op}"
op="exportAppAudit \"${auditfilename}\" -nDays=180"
processCommand "${op}"
op="downloadfile \"${auditfilename}.zip\""
processCommand "${op}"
op="logout"
processCommand "${op}"
# Rename the downloaded audit data backup, keep the last 10 backups
timestamp=`date +%m_%d_%Y_%I%M`
mv "${auditfilename}.zip" "${auditfilename}_${timestamp}.zip"

((numberofbackups+=1))
ls -tp ${auditfilename}*.zip | grep -v '/$' | tail -n +${numberofbackups} |
xargs -d '\n' -r rm --

```

環境へのデータ・ファイルのアップロードおよびデータ・ロード・ルールの実行

これらのスクリプトを使用して、ファイルを環境にアップロードした後、データ・ルールを実行して、データをファイルからアプリケーションにインポートします。

前提条件

- データ管理内の定義:

- データ・ロード・ルール定義 VisionActual データ・ルールによって入力ファイルのファイル・パスが指定されないことを前提としています。
- 期間定義 Mar-15 から Jun-15 まで
- データが格納されている、適切にフォーマットされたデータ・ファイル(GLActual.dat)。

データをインポートしてデータ・ロード・ルールを実行するには、コマンドを実行して次のステップを完了します。

- 環境にサインインします。
- Mar-15 から Jun-15 までの期間のデータが含まれているファイル GLActual.dat をデータ管理のフォルダ inbox/Vision にアップロードします。
- データ・ロード・ルール VisionActual、開始期間 Mar-15、終了期間 Jun-15、インポート・モード REPLACE を使用して、GLActual.dat からデータ管理にデータをインポートします。
- STORE_DATA オプションを指定してデータをエクスポートし、データ管理のステージング表のデータと既存のアプリケーション・データをマージします。
- サインアウトします。

Windows のサンプル・スクリプト

次のスクリプトをコピーして、runDataLoadRule.ps1 を作成します。それをローカル・ディレクトリに保存します。

```
$inputproperties = ConvertFrom-StringData(Get-Content ./input.properties -raw)
$username="$($inputproperties.username) "
$passwordfile="$($inputproperties.passwordfile) "
$serviceURL="$($inputproperties.serviceURL) "
$dataFile="$($inputproperties.dataFile) "
$dataRuleName="$($inputproperties.dataRuleName) "
$startPeriod="$($inputproperties.startPeriod) "
$endPeriod="$($inputproperties.endPeriod) "
$importMode="$($inputproperties.importMode) "
$exportMode="$($inputproperties.exportMode) "

epmautomate login ${username} ${passwordfile} ${serviceURL}
epmautomate uploadfile ${datafile} ${dataFileUploadLocation}
epmautomate rundatarule ${dataRuleName} ${startPeriod} ${endPeriod} $
{importMode} ${exportMode} ${dataFileUploadLocation}/${dataFile}
epmautomate logout
```

Linux/UNIX のサンプル・スクリプト

次のスクリプトをコピーして、runDataLoadRule.sh を作成します。それをローカル・ディレクトリに保存します。

```
#!/bin/bash
. ./input.properties
export JAVA_HOME=${javahome}
${epmautomatescript} login "${username}" "${passwordfile}" "${serviceURL}"
${epmautomatescript} uploadfile "${datafile}" "${dataFileUploadLocation}"
${epmautomatescript} rundatarule "${dataRuleName}" "${startPeriod}" "$
{endPeriod}" "${importMode}" "${exportMode}" "${dataFileUploadLocation}/${
```

```
{dataFile}"
${epmautomatescript} logout
```

input.properties ファイルの作成

次のいずれかをコピーし、環境の情報で更新することにより、input.properties ファイルを作成します。runDataLoadRule.ps1 または runDataLoadRule.sh が格納されているディレクトリにファイルを保存します。

Windows

```
username=serviceAdmin
passwordfile=./password.epw
serviceURL=https://example.oraclecloud.com
dataFile=GLActual.dat
dataFileUploadLocation=UPLOAD_LOCATION
dataRuleName=RULE_NAME
startPeriod=START_PERIOD
endPeriod=END_PERIOD
importMode=IMPORT_MODE
exportMode=EXPORT_MODE
```

Linux/UNIX


```
javahome=JAVA_HOME
epmautomatescript=EPM_AUTOMATE_LOCATION
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURLdataFile=GLActual.dat
dataFileUploadLocation=UPLOAD_LOCATION
dataRuleName=RULE_NAME
startPeriod=START_PERIOD
endPeriod=END_PERIOD
importMode=IMPORT_MODE
exportMode=EXPORT_MODE
```

表 3-22 input.properties のパラメータ

パラメータ	説明
javahome	JAVA_HOME の場所。Linux/UNIX の場合のみ。
epmautomatescript	EPM 自動化の実行可能ファイル(epmautomate.sh)の絶対パス。 Linux/UNIX の場合のみ。
username	アイデンティティ・ドメイン管理者の役割も保持するサービス管理者のユーザー名。
password	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
serviceURL	スナップショットを生成する環境の URL。
dataFile	データ・ルールを使用してインポートされるデータが格納されているファイル。
dataFileUploadLocation	データ・ファイルのアップロード先の場所。
dataRuleName	データ統合に定義されたデータ・ロード・ルールの名前。

表 3-22 (続き) input.properties のパラメータ

パラメータ	説明
startPeriod	データがロードされる最初の期間。この期間名は、データ統合の期間マッピングに定義されている必要があります。
endPeriod	複数期間データ・ロードの場合の、データがロードされる最後の期間。単一期間ロードの場合は、開始期間と同じ期間を使用します。この期間名は、データ統合の期間マッピングに定義されている必要があります。
importMode	データをデータ統合にインポートするモード。APPEND、REPLACE または RECALCULATE を使用します。ステージング表へのデータ・インポートをスキップするには、NONE を使用します。
exportMode	データをアプリケーションにエクスポートするモード。データ統合を使用します。STORE_DATA、ADD_DATA、SUBTRACT_DATA または REPLACE_DATA を使用します。データ統合からアプリケーションへのデータ・エクスポートをスキップするには、NONE を使用します。

 **ノート:**

Financial Consolidation and Close では、MERGE および NONE モードのみをサポートしています。

スクリプトの実行

1. 前述の項のスクリプトをコピーして、runDataLoadRule.ps1 または runDataLoadRule.sh を作成します。
2. input.properties ファイルを作成して、runDataLoadRule スクリプトが配置されているディレクトリに保存します。このファイルのコンテンツは、オペレーティング・システムによって異なります。[input.properties ファイルの作成](#)を参照してください。このディレクトリの書込み権限があることを確認してください。Windows の場合、スクリプトを実行できるよう、「**管理者として実行**」オプションを使用して PowerShell を開始する必要があります。
3. スクリプトを起動します。
 - **Windows PowerShell:** runDataLoadRule.ps1 を実行します。
 - **Linux/UNIX:** ./runDataLoadRule.sh を実行します。

日次データ統合の自動化

このシナリオでは、データ統合を定期的に自動化するサンプル・スクリプトの使用方法を調べます。

データ統合関連のアクティビティを自動化する次のようなスクリプトが含まれるバッチ(.bat)またはシェル(.sh)ファイルを作成します。後述する Windows 用のサンプル・スクリプトでは次のアクティビティを完了することで毎日のアプリケーション・データ統合を自動化します。

- 環境にサインインします。
- 存在する場合は DailyPlanData を削除します。

- DailyPlanData をサービスにアップロードします。
- プラン・タイプ Plan1 でビジネス・ルール Clear Plan Targets を実行します。
- ジョブ名 LoadDailyPlan を使用してデータをインポートします。
- ビジネス・ルール Balance Sheet - Plan を実行します。
- ビジネス・ルール Allocate Plan Targets を実行します。
- 存在する場合は DailyTarget.zip を削除します。
- ジョブ名 ExportDailyTarget を使用してデータを DailyTarget.zip にエクスポートします。
- DailyTarget.zip をサーバーにダウンロードしてタイムスタンプを付加します。
- 環境からサインアウトします。

ノート:

このスクリプトを使用のために再利用する場合、必ず SET url および SET user パラメータの値を変更してください。さらに、dataimportfilename、dataexportfilename、importdatajobname、exportdatajobname、br_clear、br_calculatebalancesheet および br_allocatetarget パラメータの値を要件に合うように変更できます

Windows タスク・スケジューラを使用してスクリプトをスケジュールする方法の詳細は、[スクリプトの実行の自動化](#)を参照してください。

```
@echo off

rem Sample Script to demonstrate daily data integration with
rem Cloud EPM application.
rem This script uploads Plan data, clears target numbers,
rem runs a business rule to calculate balance sheet data, and
rem recalculates target numbers on the Vision demo application

rem Please update these parameters
SET url=https://example.oraclecloud.com
SET user=serviceAdmin
SET dataimportfilename=DailyPlanData.csv
SET dataexportfilename=DailyTarget
SET importdatajobname=LoadDailyPlan
SET exportdatajobname=ExportDailyTarget
SET br_clear=Clear Plan Targets
SET br_calculatebalancesheet=Balance Sheet - Plan
SET br_allocatetarget=Allocate Plan Targets

SET password=%1

rem Executing EPM Automate commands

CD /D %~dp0
call epmautomate login %user% %password% %url%
```

```

IF %ERRORLEVEL% NEQ 0 goto :ERROR

for /f %%i in ('call epmautomate listfiles') do if %%i==%dataimportfilename%
(call epmautomate deletefile %%i)
IF %ERRORLEVEL% NEQ 0 goto :ERROR

call epmautomate uploadfile %dataimportfilename%
IF %ERRORLEVEL% NEQ 0 goto :ERROR

call epmautomate runbusinessrule "%br_clear%"
IF %ERRORLEVEL% NEQ 0 goto :ERROR

call epmautomate importdata "%importdatajobname%"
IF %ERRORLEVEL% NEQ 0 goto :ERROR

call epmautomate runbusinessrule "%br_calculatebalancesheet%"
IF %ERRORLEVEL% NEQ 0 goto :ERROR

call epmautomate runbusinessrule "%br_allocatetarget%"
"TargetVersion=Baseline"
IF %ERRORLEVEL% NEQ 0 goto :ERROR

for /f %%i in ('call epmautomate listfiles') do if %
%i=="%dataexportfilename%.zip" (call epmautomate deletefile %%i)
IF %ERRORLEVEL% NEQ 0 goto :ERROR

call epmautomate exportdata %exportdatajobname% "%dataexportfilename%.zip"
IF %ERRORLEVEL% NEQ 0 goto :ERROR

call epmautomate downloadfile "%dataexportfilename%.zip"
IF %ERRORLEVEL% NEQ 0 goto :ERROR

rem Section to rename the file

Set Timestamp=%date:~4,2%_%date:~7,2%_%date:~10,4%_%time:~1,1%%time:~3,2%
ren "%dataexportfilename%.zip" "%dataexportfilename%_%Timestamp%.zip"

call epmautomate logout
IF %ERRORLEVEL% NEQ 0 goto :ERROR

:EOF
echo Scheduled Task Completed successfully
exit /b %errorlevel%

:ERROR
echo Failed with error #%errorlevel%.
exit /b %errorlevel%

```

Account Reconciliation のサンプル・シナリオ

次も参照:

- [期間への事前フォーマット済残高のロード](#)
これらのスクリプトを使用して、マッピングされたデータをアップロード済ファイルから Account Reconciliation 環境にインポートします。

- **バックアップ・スナップショットのアップロードとインポート**
これらのスクリプトを使用して、バックアップ・スナップショットを Account Reconciliation 環境にアップロードおよびインポートします。
- **照合した古いトランザクションのアーカイブおよびアーカイブしたトランザクションのページ**
この項のスクリプトを使用して、サポートおよび調整の詳細を含めて、指定の経過期間以上経過した照合済トランザクションをアーカイブし、アーカイブしたトランザクションを Account Reconciliation からパージします。アーカイブした照合済トランザクションは ZIP ファイルに記録されます。

期間への事前フォーマット済残高のロード

これらのスクリプトを使用して、マッピングされたデータをアップロード済ファイルから Account Reconciliation 環境にインポートします。

Windows のサンプル・スクリプト

次のスクリプトをコピーして、runPreformattedBalances.ps1 という名前のファイルを作成します。それをローカル・ディレクトリに保存します。

```
$inputproperties = ConvertFrom-StringData(Get-Content ./input.properties -raw)
$username="$($inputproperties.username) "
$passwordfile="$($inputproperties.passwordfile) "
$serviceURL="$($inputproperties.serviceURL) "
$dataFile="$($inputproperties.dataFile) "
$period="$($inputproperties.period) "
$balanceType="$($inputproperties.balanceType) "
$currencyBucket="$($inputproperties.currencyBucket) "

$elements=$dataFile.split('/')
$dataFileName=$elements[-1]

epmautomate login ${username} ${passwordfile} ${serviceURL}
epmautomate uploadfile ${dataFile}
epmautomate importpremappedbalances ${period} ${dataFileName} ${balanceType} $
{currencyBucket}
epmautomate deletefile ${dataFileName}
epmautomate logout
```

Linux/UNIX のサンプル・スクリプト

次のスクリプトをコピーして、runPreformattedBalances.sh という名前のファイルを作成します。それをローカル・ディレクトリに保存します。

```
#!/bin/bash

. ./input.properties

export JAVA_HOME=${javahome}

dataFileName=$(echo "${dataFile}" | rev | cut -d '/' -f1 | rev)

${epmautomatescript} login "${username}" "${passwordfile}" "${serviceURL}"
${epmautomatescript} uploadfile "${dataFile}"
```

```

${epmautomatescript} importpremappedbalances "${period}" "${dataFileName}" "$
{balanceType}" "${currencyBucket}"
${epmautomatescript} deletefile "${dataFileName}"
${epmautomatescript} logout

```

input.properties ファイルのサンプル

runPreformattedBalances スクリプトを実行するには、input.properties ファイルを作成し、環境の情報でファイルを更新します。runPreformattedBalances.sh または runPreformattedBalances.ps1 が格納されているディレクトリにファイルを保存します。

Windows

```

username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
dataFile=DATA_FILE_NAME.csv
period=PERIOD_NAME
balanceType=BALANCE_TYPE
currencyBucket=CURRENCY_BUCKET

```

Linux/UNIX

```

javahome=JAVA_HOME
epmautomatescript=EPM_AUTOMATE_LOCATION
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
dataFile=DATA_FILE_NAME.csv
period=PERIOD_NAME
balanceType=BALANCE_TYPE
currencyBucket=CURRENCY_BUCKET

```

表 3-23 input.properties のパラメータ

パラメータ	説明
javahome	JAVA_HOME の場所。Linux/UNIX の場合のみ。
epmautomatescript	EPM 自動化の実行可能ファイル(epmautomate.sh)の絶対パス。Linux/UNIX の場合のみ。
username	サービス管理者のユーザー名
password	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
serviceURL	事前フォーマット済残高をロードするアプリケーションをホストする環境の URL。
dataFile	アプリケーションにロードする事前フォーマット済残高(通常は一般会計から作成)が格納されている CSV ファイル。このファイルは、 uploadFile コマンドを使用して環境にアップロードされている必要があります。
period	事前フォーマット済残高をアップロードする照合期間。
balanceType	dataFile に格納されている事前フォーマット済残高のタイプ。
currencyBucket	事前フォーマット済残高の通貨バケット。

スクリプトの実行

1. 前述の項のスクリプトをコピーして、runPreformattedBalances.ps1 または runPreformattedBalances.sh を作成します。
2. **Windows および Linux/UNIX の場合:**
 - input.properties ファイルを作成し、runPreformattedBalances スクリプトが配置されているディレクトリに保存します。このファイルのコンテンツは、オペレーティング・システムによって異なります。表 1 を参照してください。このディレクトリの書き込み権限があることを確認してください。Windows の場合、スクリプトを実行できるよう、「**管理者として実行**」オプションを使用して PowerShell を開始する必要があります。
 - スクリプトを起動します。
 - **Windows PowerShell:** runPreformattedBalances.ps1 を実行します。
 - **Linux/UNIX:** ./runPreformattedBalances.sh を実行します。

バックアップ・スナップショットのアップロードとインポート

これらのスクリプトを使用して、バックアップ・スナップショットを Account Reconciliation 環境にアップロードおよびインポートします。

Windows のサンプル・スクリプト

次のスクリプトをコピーして、importBackupSnapshot.ps1 という名前のファイルを作成します。それをローカル・ディレクトリに保存します。

```
$inputproperties = ConvertFrom-StringData (Get-Content ./input.properties -raw)
$username="$($inputproperties.username)"
$passwordfile="$($inputproperties.passwordfile)"
$serviceURL="$($inputproperties.serviceURL)"
$snapshotName="$($inputproperties.snapshotName)"
$userPassword="$($inputproperties.userPassword)"

epmautomate login ${username} ${passwordfile} ${serviceURL}
epmautomate uploadfile ${snapshotName}.zip
epmautomate importsnapshot ${snapshotName} "userPassword=${userPassword}"
epmautomate deletefile ${snapshotName}.zip
epmautomate logout
```

Linux/UNIX のサンプル・スクリプト

次のスクリプトをコピーして、importBackupSnapshot.sh という名前のファイルを作成します。それをローカル・ディレクトリに保存します

```
#!/bin/bash

. ./input.properties
export JAVA_HOME=${javahome}
${epmautomatescript} login "${username}" "${passwordfile}" "${serviceURL}"
${epmautomatescript} uploadfile "${snapshotName}.zip"
${epmautomatescript} importsnapshot "${snapshotName}" "userPassword=${userPassword}"
```

```

${epmautomatescript} deletefile "${snapshotName}.zip"
${epmautomatescript} logout

```

input.properties ファイルのサンプル

importBackupSnapshot スクリプトを実行するには、input.properties ファイルを作成し、環境の情報でファイルを更新します。importBackupSnapshot.sh または importBackupSnapshot.ps1 が格納されているディレクトリにファイルを保存します。

Windows

```

username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
snapshotName=SNAPSHOT_NAME
userPassword=IDM_NEW_USER_PWD

```

Linux/UNIX

```

javahome=JAVA_HOME
epmautomatescript=EPM_AUTOMATE_LOCATION
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
snapshotName=SNAPSHOT_NAME
userPassword=IDM_NEW_USER_PWD

```

表 3-24 input.properties のパラメータ

パラメータ	説明
javahome	JAVA_HOME の場所。Linux/UNIX の場合のみ。
epmautomatescript	EPM 自動化の実行可能ファイル(epmautomate.sh)の絶対パス。Linux/UNIX の場合のみ。
username	サービス管理者のユーザー名
password	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
serviceURL	スナップショットをインポートする環境の URL。
snapshotName	アーティファクトおよびデータのインポート元スナップショットの名前。このスナップショットは、uploadFile コマンドを使用して環境にアップロードされている必要があります。
userPassword	このスナップショットのインポートの結果としてアイデンティティ・ドメインに作成されたすべての新規ユーザーに割り当てる必要があるデフォルトのパスワード。

スクリプトの実行

1. 前述の項のスクリプトをコピーして、importBackupSnapshot.ps1 または importBackupSnapshot.sh を作成します。
2. input.properties ファイルを作成し、runPreformattedBalances スクリプトが配置されているディレクトリに保存します。このファイルのコンテンツは、オペレーティング・システムによって異なります。input.properties ファイルのサンプルを参照してください。

このディレクトリの書込み権限があることを確認してください。Windows の場合、スクリプトを実行できるよう、「**管理者として実行**」オプションを使用して PowerShell を開始する必要があります。

3. スクリプトを起動します。
 - **Windows PowerShell:** `importBackupSnapshot.ps1` を実行します。
 - **Linux/UNIX:** `./importBackupSnapshot.sh` を実行します。

照合した古いトランザクションのアーカイブおよびアーカイブしたトランザクションのパージ

この項のスクリプトを使用して、サポートおよび調整の詳細を含めて、指定の経過期間以上経過した照合済トランザクションをアーカイブし、アーカイブしたトランザクションを **Account Reconciliation** からパージします。アーカイブした照合済トランザクションは ZIP ファイルに記録されます。

スクリプトの仕組み

1. `input.properties` ファイル内の情報を使用して、環境にログインします
2. 次の `archiveTmTransactions` コマンドを実行してアーカイブを作成します。生成された ZIP ファイルおよびログ・ファイルでは、デフォルト名の `Archive_Transactions_INTERCO_JOB_ID.zip` および `Archive_Transactions_INTERCO_JOB_ID.log` が使用されます

```
epmautomate archiveTmTransactions INTERCO 365 filterOperator=contains filterValue=14001
```

`input.properties` ファイルを変更して、コマンド・パラメータを変更できます。
3. ログ・ファイル、およびアーカイブ済トランザクションを含む ZIP ファイルをローカル・コンピュータにダウンロードします。一致するトランザクションが見つからない場合は、スクリプトでエラー・メッセージが表示されます。
4. アーカイブ済トランザクションを含む ZIP ファイルを **Oracle Object Store** にコピーします。
5. `purgeArchivedTmTransactions` コマンド(`archiveTmTransactions` ジョブのジョブ ID を指定)を実行して、アーカイブされた照合済トランザクションをアプリケーションから削除します。

スクリプトの実行

1. `input.properties` ファイルを作成し、環境の情報を使用して更新します。ファイルをローカル・ディレクトリに保存します。このディレクトリは、ここでは `parentsnapshotdirectory` と呼ばれます。このファイルのコンテンツは、オペレーティング・システムによって異なります。
このディレクトリの書込み権限があることを確認してください。Windows の場合、スクリプトを実行できるよう、「**管理者として実行**」オプションを使用して PowerShell を開始する必要があります。
2. `transaction_match.ps1` (Windows PowerShell)または `transaction_match.sh` (Linux/UNIX)スクリプトを作成し、`input.properties` が配置されている `parentsnapshotdirectory` 内に保存します。
3. スクリプトを起動します。

- Linux/UNIX: `./transaction_match.sh` を実行します。
- Windows PowerShell: `transaction_match.ps1` を実行します。

input.properties スクリプトの作成

次のスクリプトをコピーして更新することにより、`input.properties` を作成します。

```
javahome=JAVA_HOME
epmautomatescript=EPM_AUTOMATE_LOCATION
epmusername=exampleAdmin1
epmpassword=examplePassword1.epw
epmurl=exampleURL1
objectstorageusername=exampleAdmin2
objectstoragepassword=examplePassword2
objectstorageurl=exampleURL2
matchtype=INTERCO
age=365
filteroperator=contains
filtervalues=FilterValue=14001
proxyserverusername=myProxyserver
proxyserverpassword=myProxyserver_pwd
proxyserverdomain=myProxyDomain
```

ノート:

Windows のみ: 次のプロパティを `input.properties` ファイルから削除します:

- `javahome=JAVA_HOME`
- `epmautomatescript=EPM_AUTOMATE_LOCATION`

`authentication at proxy server` が Windows のネットワーク環境に対して有効でない場合は、次のプロパティを `input.properties` ファイルから削除します。

- `proxyserverusername`
- `proxyserverpassword`
- `proxyserverdomain`

表 3-25 `input.properties` のパラメータ

パラメータ	説明
<code>javahome</code>	EPM 自動化で使用される JDK がインストールされているディレクトリ。 Windows バージョンの <code>input.properties</code> からこのエントリを削除します。 例: <code>javahome=./home/JDK/bin</code>
<code>epmautomatescript</code>	EPM 自動化がインストールされているディレクトリ。 Windows バージョンの <code>input.properties</code> からこのエントリを削除します。 例: <code>epmautomatescript=./home/Utils/EPMAutomate/bin</code>

表 3-25 (続き) input.properties のパラメータ

パラメータ	説明
epmusername	サービス管理者、または照合済トランザクションをアーカイブする権限があるパワー・ユーザー、ユーザーまたは参照者のユーザー名。 例: epmusername=ServiceAdmin
epmuserpassword	epmusername として識別されるユーザーの暗号化されたパスワード・ファイル。 例: epmpassword=myPwd.epw
epmurl	照合済トランザクションがアーカイブされる環境の URL。 例: epmurl=https://test-cloudpln.pbcs.us1.oraclecloud.com
objectstorageusername	Oracle Object Storage Cloud への書込みに必要なアクセス権を持つユーザーの ID。 フェデレーション・アイデンティティ・プロバイダで作成されたユーザーの場合は、ユーザーの完全修飾名を指定します(たとえば、exampleIdP/jdoe や exampleIdP/john.doe@example.com。ここで、exampleIdP はフェデレーション・アイデンティティ・プロバイダの名前)。その他のユーザーの場合は、ユーザー ID を指定します。 例: epmusername=myIdP/jdoe
objectstoragepassword	objectstorageusername で識別されるユーザーに関連付けられている Swift パスワードまたは認証トークン。このパスワードは、ユーザーがオブジェクト・ストレージ・コンソールにサインインするのに使用するパスワードとは異なります。認証トークンは、 Oracle で生成されるトークンであり、たとえば、 Swift クライアントでの認証など、サード・パーティの API での認証に使用します。このトークンを作成する手順は、 Oracle Cloud Infrastructure ドキュメントの 認証トークンを作成するには を参照してください。 例: objectstoragepassword=jDoe_PWD
objectstorageurl	オプションのオブジェクト名が追加された Oracle Object Storage Cloud バケットの URL。 例: objectstorageurl=https://swiftobjectstorage.region_identifier.oraclecloud.com/v1/namespace/MT_Archives/2023_archives
matchtype	照合済トランザクションをアーカイブする照合タイプの識別子 (TextID)。 例: matchtype=cashrecon
age	トランザクションが照合されてからの日数。この値以上経過した照合済トランザクションがアーカイブされます。 例: age=180
filteroperator	アーカイブ対象の照合済トランザクションが格納されている勘定科目を識別するためのフィルタ条件。次のいずれかである必要があります: equals、not_equals、starts_with、ends_with、contains または not_contains。 この値を filterValue と組み合わせて、照合済トランザクションをアーカイブする勘定科目を識別します。 例: filteroperator=not_equals

表 3-25 (続き) input.properties のパラメータ

パラメータ	説明
filtervalues	アーカイブするトランザクションを識別する 1 つ以上のフィルタ値。 filterOperator として equals または not_equals が指定されると、 スペース区切りのリストを使用して複数の値を指定できます。複数の値 が指定されている場合、フィルタ演算子とフィルタ値の組合せに一致す る勘定科目のトランザクションがアーカイブ対象として選択されます。 例: filterValue=101-120 filterValue=140-202
proxyserverusername	インターネットへのアクセスを制御するプロキシ・サーバーとの安全な セッションを認証するユーザー名。 例: proxyserverusername=myProxyserver
proxyserverpassword	プロキシ・サーバーに対してユーザーを認証するパスワード。 例: proxyserverpassword=myProxyserver_pwd
proxyserverdomain	プロキシ・サーバーに定義されているドメインの名前。 例: proxyserverdomain=myProxyDomain

Windows スクリプト

次のスクリプトをコピーして、transaction_match.ps1 を作成します。input.properties が
格納されているフォルダに格納します。

```
# Archive and Purge Transaction Matching PowerShell script

$inputproperties = ConvertFrom-StringData (Get-Content ./input.properties -raw)
$epmusername="$($inputproperties.epmusername) "
$epmpassword="$($inputproperties.epmpassword) "
$epmurl="$($inputproperties.epmurl) "
$objectstorageusername="$($inputproperties.objectstorageusername) "
$objectstoragepassword="$($inputproperties.objectstoragepassword) "
$objectstorageurl="$($inputproperties.objectstorageurl) "
$matchtype="$($inputproperties.matchtype) "
$age="$($inputproperties.age) "
$filteroperator="$($inputproperties.filteroperator) "
$filtervalues="$($inputproperties.filtervalues) "
$proxyserverusername="$($inputproperties.proxyserverusername) "
$proxyserverpassword="$($inputproperties.proxyserverpassword) "
$proxyserverdomain="$($inputproperties.proxyserverdomain) "

echo "Running processes to archive and purge transaction matching
transactions ..."
if (${proxyserverusername}) {
    echo "Running epmautomate login ${epmusername} ${epmpassword} ${epmurl} $
{proxyserverusername} ${proxyserverpassword} ${proxyserverdomain}"
    epmautomate login ${epmusername} ${epmpassword} ${epmurl} $
{proxyserverusername} ${proxyserverpassword} ${proxyserverdomain}
} else {
    echo "Running epmautomate login ${epmusername} ${epmpassword} ${epmurl}"
    epmautomate login ${epmusername} ${epmpassword} ${epmurl}
}
echo "Running epmautomate archiveTmTransactions \"${matchtype}\" ${age}
filterOperator=${filteroperator} ${filtervalues}"
epmautomate archiveTmTransactions "${matchtype}" "${age}" "filterOperator=$
```

```

{filteroperator}" "${filtervalues}" > ./outfile.txt.tmp

$jobIdLine=Select-String -Path "outfile.txt.tmp" -Pattern "Job
ID"; $jobIdLine=($jobIdLine -split ":-)[-2]; $jobid=($jobIdLine -split " ")
[1];
$logfile="Archive_Transactions_${matchtype}_${jobid}.log"
$transactionsfile="Archive_Transactions_${matchtype}_${jobid}.zip"
epmautomate listfiles > ./files.txt.tmp
$transactionslogfound=Select-String -Path "./files.txt.tmp" -Pattern "$
{logfile}"
$transactionsfilefound=Select-String -Path "./files.txt.tmp" -Pattern "$
{transactionsfile}"
rm ./outfile.txt.tmp
rm ./files.txt.tmp

if (${transactionslogfound}) {
    echo "Running epmautomate downloadfile \"${logfile}\""
    epmautomate downloadfile "${logfile}"
    if (${transactionsfilefound}) {
        echo "Running epmautomate downloadfile ${transactionsfile}"
        epmautomate downloadfile "${transactionsfile}"
        if ($?) {
            echo "Running epmautomate copyToObjectStorage $
{transactionsfile} ${objectstorageusername} ${objectstoragepassword} $
{objectstorageurl}"
            epmautomate copyToObjectStorage "${transactionsfile}" "$
{objectstorageusername}" "${objectstoragepassword}" "${objectstorageurl}"
            if ($?) {
                echo "Running epmautomate purgeArchivedTMTransactions JOBID=$
{jobid}"
                epmautomate purgeArchivedTMTransactions "JobID=${jobid}"
            }
        }
        else {
            echo "EPMAutomate copyToObjectStorage failed. Purging of
archived matched transactions will not be attempted."
        }
    }
    else {
        echo "Download of transactions file ${transactionsfile} failed.
Copy to object storage, and purging of archived matched transactions will not
be attempted."
    }
}
else {
    echo "No matched transactions found. Archive file download, copy to
object storage, and purging of archived matched transactions will not be
attempted."
}
}
else {
    echo "Matchtype ID \"${matchtype}\" not found. Archive file download,
copy to object storage, and purging of archived matched transactions will not
be attempted."
}

echo "Running epmautomate logout"

```

```
epmautomate logout
echo "Script processing completed."
```

Linux/UNIX のスクリプト

次のスクリプトをコピーして、transaction_match.sh を作成します。input.properties が格納されているフォルダに格納します。

```
#!/bin/sh

. ./input.properties
export JAVA_HOME=${javahome}

echo "Running processes to archive and purge transaction matching
transactions..."
if [[ "${proxyserverusername}" != "" ]]
then
    echo "Running epmautomate login ${epmusername} ${epmpassword} ${epmurl}"
    ProxyServerUserName=${proxyserverusername} ProxyServerPassword=${
proxyserverpassword} ProxyServerDomain=${proxyserverdomain}"
    ${epmautomatescript} login "${epmusername}" "${epmpassword}" "${epmurl}"
    "ProxyServerUserName=${proxyserverusername}" "ProxyServerPassword=${
proxyserverpassword}" "ProxyServerDomain=${proxyserverdomain}"
else
    echo "Running epmautomate login ${epmusername} ${epmpassword} ${epmurl}"
    ${epmautomatescript} login "${epmusername}" "${epmpassword}" "${epmurl}"
fi
echo "Running epmautomate archiveTmTransactions \"${matchtype}\" ${age}
filterOperator=${filteroperator} ${filtervalues}"
${epmautomatescript} archiveTmTransactions "${matchtype}" "${age}"
"filterOperator=${filteroperator}" "${filtervalues}" > ./output.txt.tmp

jobid=$(grep "Job ID" ./output.txt.tmp | cut -d':' -f3 | cut -d' ' -f2)
logfile="Archive_Transactions_${matchtype}_${jobid}.log"
transactionsfile="Archive_Transactions_${matchtype}_${jobid}.zip"
${epmautomatescript} listfiles > ./files.txt.tmp
transactionslogfound=$(grep "${logfile}" ./files.txt.tmp | wc -l)
transactionsfilefound=$(grep "${transactionsfile}" ./files.txt.tmp | wc -l)
rm ./files.txt.tmp
rm ./output.txt.tmp

if [ ${transactionslogfound} -eq 0 ]
then
    echo "Matchtype ID \"${matchtype}\" not found. Archive file download,
copy to object storage, and purging of archived matched transactions will not
be attempted."
else
    echo "Running epmautomate downloadfile \"${logfile}\""
    ${epmautomatescript} downloadfile "${logfile}"
    if [ ${transactionsfilefound} -eq 0 ]
    then
        echo "No matched transactions found. Archive file download, copy to
object storage, and purging of archived matched transactions will not be
attempted."
    else
        echo "Running epmautomate downloadfile ${transactionsfile}"
```

```

    ${epmautomatescript} downloadfile "${transactionsfile}"
    if [ $? -eq 0 ]
    then
        echo "Running epmautomate copyToObjectStorage $
{transactionsfile} ${objectstorageusername} ${objectstoragepassword} $
{objectstorageurl}"
        ${epmautomatescript} copyToObjectStorage "${transactionsfile}" "$
{objectstorageusername}" "${objectstoragepassword}" "${objectstorageurl}"
        if [ $? -eq 0 ]
        then
            echo "Running epmautomate purgeArchivedTMTransactions JOBID=$
{jobid}"
            ${epmautomatescript} purgeArchivedTMTransactions "JobID=$
{jobid}"
        else
            echo "EPMAutomate copyToObjectStorage failed. Purging of
archived matched transactions will not be attempted."
            fi
        else
            echo "Download of transactions file ${transactionsfile} failed.
Copy to object storage, and purging of archived matched transactions will not
be attempted."
            fi
        fi
    fi
fi

echo "Running epmautomate logout"
${epmautomatescript} logout
echo "Script processing completed."

```

Profitability and Cost Management のサンプル・シナリオ

これらのシナリオでは、EPM 自動化のコマンドを使用して、Profitability and Cost Management の一般的なタスクを実行する方法について検討します。

次も参照:

- [アプリケーションへのメタデータのインポート](#)
 これらのスクリプトを使用して、メタデータ・ファイルをアップロードし、そのファイルからディメンション・メタデータを Profitability and Cost Management アプリケーションにインポートします。
- [データのインポートとプログラム・ルールの実行](#)
 これらのスクリプトを使用して、データ・ファイルをアップロードし、アップロードしたファイルからデータを Profitability and Cost Management ビジネス・プロセスにインポートします。

アプリケーションへのメタデータのインポート

これらのスクリプトを使用して、メタデータ・ファイルをアップロードし、そのファイルからディメンション・メタデータを Profitability and Cost Management アプリケーションにインポートします。

これらのスクリプトは、次の操作を実行します:

- 環境にサインインします。

- メタデータをアップロードします。
- アップロードされたファイルからアプリケーションにメタデータをインポートします。
- アプリケーションを有効にします。
- サインアウトします。

Windows スクリプト

このスクリプトをコピーして、importMetadata.ps1 を作成します。

```
$inputproperties = ConvertFrom-StringData (Get-Content ./input.properties -raw)
$username="$($inputproperties.username)"
$passwordfile="$($inputproperties.passwordfile)"
$serviceURL="$($inputproperties.serviceURL)"
$applicationName="$($inputproperties.applicationName)"
$dataFileName="$($inputproperties.dataFileName)"
$dataFileNameDestination="$($inputproperties.dataFileNameDestination)"

epmautomate login ${username} ${passwordfile} ${serviceURL}
epmautomate uploadfile "${dataFileName}" "${dataFileNameDestination}"
epmautomate loadaddimdata ${applicationName} dataFileName=${dataFileName}
epmautomate enableapp ${applicationName}
epmautomate logout
```

Linux/UNIX のスクリプト

このスクリプトをコピーして、importMetadata.sh を作成します。

```
#!/bin/bash
. ./input.properties
export JAVA_HOME=${javahome}
${epmautomatescript} login "${username}" "${passwordfile}" "${serviceURL}"
${epmautomatescript} uploadfile "${dataFileName}" "${dataFileNameDestination}"
${epmautomatescript} loadaddimdata "${applicationName}" "dataFileName=${dataFileName}"
${epmautomatescript} enableapp "${applicationName}"
${epmautomatescript} logout
```

input.properties ファイルの作成

importMetadata スクリプトを実行するには、input.properties ファイルを作成し、環境の情報でファイルを更新します。importMetadata.ps1 または importMetadata.sh が格納されているディレクトリにファイルを保存します。

Windows

```
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
applicationName=APPLICATION_NAME
dataFileName=DATA_FILE.txt
dataFileNameDestination=profitinbox
```

Linux/UNIX

```
javahome=JAVA_HOME
epmautomatescript=EPM_AUTOMATE_LOCATION
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
applicationName=APPLICATION_NAME
dataFileName=DATA_FILE.txt
dataFileNameDestination=profitinbox
```

表 3-26 input.properties のパラメータ

パラメータ	説明
javahome	JAVA_HOME の場所。Linux/UNIX の場合のみ。
epmautomatescript	EPM 自動化の実行可能ファイル(epmautomate.sh)の絶対パス。Linux/UNIX の場合のみ。
username	アイデンティティ・ドメイン管理者の役割も保持するサービス管理者のユーザー名。
password	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
serviceURL	スナップショットを生成する環境の URL。
applicationName	データのロード先となる Profitability and Cost Management の名前。
dataFileName	インポートするメタデータが格納されているファイルの名前。
dataFileNameDestination	メタデータ・ファイルのアップロード場所。

スクリプトの実行

1. 前述の項のスクリプトをコピーして、importMetadata.ps1 または importMetadata.sh を作成します。
2. input.properties ファイルを作成し、importMetadata スクリプトが配置されているディレクトリに保存します。このファイルのコンテンツは、オペレーティング・システムによって異なります。[input.properties ファイルの作成](#)を参照してください。このディレクトリの書込み権限があることを確認してください。Windows の場合、スクリプトを実行できるよう、「**管理者として実行**」オプションを使用して PowerShell を開始する必要があります。
3. スクリプトを起動します。
 - **Windows PowerShell:** importMetadata.ps1 を実行します。
 - **Linux/UNIX:** ./importMetadata.sh を実行します。

データのインポートとプログラム・ルールの実行

これらのスクリプトを使用して、データ・ファイルをアップロードし、アップロードしたファイルからデータを Profitability and Cost Management ビジネス・プロセスにインポートします。

これらのスクリプトは、次のステップを完了します:

- 環境にサインインします。

- ロードするデータが格納されているファイルをアップロードします。
- データ・ルールが格納されているルール・ファイルをアップロードします。
- データ・ファイルからアプリケーションにデータをロードして、既存の値を上書きします。
- ルール・ファイルのすべてのルールを実行します。
- サインアウトします。

Windows スクリプト

次のスクリプトをコピーして、importData.ps1 という名前のファイルを作成します。それをローカル・ディレクトリに保存します。

```
$inputproperties = ConvertFrom-StringData(Get-Content ./input.properties -raw)
$username="$($inputproperties.username) "
$passwordfile="$($inputproperties.passwordfile) "
$serviceURL="$($inputproperties.serviceURL) "
$applicationName="$($inputproperties.applicationName) "
$dataFileName="$($inputproperties.dataFileName) "
$rulesFileName="$($inputproperties.rulesFileName) "
$fileDestination="$($inputproperties.fileDestination) "
$clearAllDataFlag="$($inputproperties.clearAllDataFlag) "
$dataLoadValue="$($inputproperties.dataLoadValue) "

epmautomate login ${username} ${passwordfile} ${serviceURL}
epmautomate uploadfile "${dataFileName}" ${fileDestination}
epmautomate uploadfile "${rulesFileName}" ${fileDestination}
epmautomate loaddata ${applicationName} clearAllDataFlag=${clearAllDataFlag}
dataLoadValue=${dataLoadValue} rulesFileName="${rulesFileName}"
dataFileName="${dataFileName}"
epmautomate logout
```

Linux/UNIX のスクリプト

次のスクリプトをコピーして、importData.sh という名前のファイルを作成します。それをローカル・ディレクトリに保存します。

```
#!/bin/bash
. ./input.properties
export JAVA_HOME=${javahome}
${epmautomatescript} login "${username}" "${passwordfile}" "${serviceURL}"
${epmautomatescript} uploadfile "${dataFileName}" "${fileDestination}"
${epmautomatescript} uploadfile "${rulesFileName}" "${fileDestination}"
${epmautomatescript} loaddata "${applicationName}" "clearAllDataFlag=${clearAllDataFlag}" "dataLoadValue=${dataLoadValue}" rulesFileName="${rulesFileName}" dataFileName="${dataFileName}"
${epmautomatescript} logout
```

input.properties ファイルの作成

importData スクリプトを実行するには、input.properties ファイルを作成し、環境の情報でファイルを更新します。importData.ps1 または importData.sh が格納されているディレクトリにファイルを保存します。

Windows

```
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
applicationName=APPLICATION_NAME
dataFileName=DATA_FILE.txt
rulesFileName=RULE_FILE.txt
fileDestination=profitinbox
clearAllDataFlag=true
dataLoadValue=OVERWRITE_EXISTING_VALUES
```

Linux/UNIX

```
javahome=JAVA_HOME
epmautomatescript=EPM_AUTOMATE_LOCATION
username=exampleAdmin
passwordfile=examplePassword.epw
serviceURL=exampleURL
applicationName=APPLICATION_NAME
dataFileName=DATA_FILE.txt
rulesFileName=RULE_FILE.txt
fileDestination=profitinbox
clearAllDataFlag=true
dataLoadValue=OVERWRITE_EXISTING_VALUES
```

表 3-27 input.properties のパラメータ

パラメータ	説明
javahome	JAVA_HOME の場所。Linux/UNIX の場合のみ。
epmautomatescript	EPM 自動化の実行可能ファイル(epmautomate.sh)の絶対パス。Linux/UNIX の場合のみ。
username	アイデンティティ・ドメイン管理者の役割も保持するサービス管理者のユーザー名。
password	サービス管理者のパスワードまたは暗号化されたパスワード・ファイルの名前と場所。
serviceURL	スナップショットを生成する環境の URL。
applicationName	データのロード先となる Profitability and Cost Management の名前。
dataFileName	インポートするデータが格納されているファイルの名前。
rulesFileName	インポートされたデータに対して実行するルールが格納されているファイルの名前。
fileDestination	データ・ファイルとルール・ファイルをアップロードする場所。
clearAllDataFlag	アプリケーション・キューブの既存のデータをクリアするかどうかを指定します。既存のデータをクリアしない場合は、false に設定します。
dataLoadValue	既存のデータの処理方法を指定します。キューブの既存のデータを保持する場合は、ADD_TO_EXISTING を指定します。

スクリプトの実行

1. 前述の項のスクリプトをコピーして、importData.ps1 または importData.sh を作成します。
2. input.properties ファイルを作成し、importData スクリプトが配置されているディレクトリに保存します。このファイルのコンテンツは、オペレーティング・システムによって異なります。[input.properties ファイルの作成](#)を参照してください。
このディレクトリの書込み権限があることを確認してください。Windows の場合、スクリプトを実行できるよう、「**管理者として実行**」オプションを使用して PowerShell を開始する必要があります。
3. スクリプトを起動します。
 - **Windows PowerShell:** importData.ps1 を実行します。
 - **Linux/UNIX:** ./importData.sh を実行します。

Oracle Enterprise Data Management Cloud のサンプル・シナリオ

これらのサンプル・シナリオでは、Oracle Fusion Cloud Enterprise Data Management と Oracle Fusion Cloud Enterprise Performance Management 間でアプリケーションのディメンションを同期する EPM 自動化コマンドの使用方法を調べます。

次も参照:

- [Oracle Enterprise Data Management Cloud ディメンションおよびマッピングの Cloud EPM アプリケーションとの同期](#)
このサンプル・シナリオでは、Oracle Fusion Cloud Enterprise Data Management アプリケーションと Oracle Fusion Cloud Enterprise Performance Management アプリケーション間でのディメンションの同期方法を調べます。
- [Cloud EPM ディメンションの Oracle Enterprise Data Management Cloud アプリケーションとの同期](#)
このサンプル・シナリオでは、Oracle Fusion Cloud Enterprise Performance Management アプリケーションと Oracle Fusion Cloud Enterprise Data Management アプリケーション間でのディメンションの同期方法を調べます。

Oracle Enterprise Data Management Cloud ディメンションおよびマッピングの Cloud EPM アプリケーションとの同期

このサンプル・シナリオでは、Oracle Fusion Cloud Enterprise Data Management アプリケーションと Oracle Fusion Cloud Enterprise Performance Management アプリケーション間でのディメンションの同期方法を調べます。

この項のスクリプトを使用して、次のタスクを完了できます:

- Oracle Enterprise Data Management Cloud アプリケーションからディメンションをエクスポートします
- Oracle Enterprise Data Management Cloud アプリケーションのディメンションからマッピングをエクスポートします
- エクスポート・ファイルを Cloud EPM 環境にコピーします

- ディメンション・メタデータおよびマッピングを **Cloud EPM** アプリケーションにインポートします

Oracle Enterprise Data Management Cloud アプリケーションと **Cloud EPM** アプリケーション間でディメンションおよびマッピングを同期するには:

1. 次のスクリプトをコピーして、スクリプト・ファイルを作成します:

```
rem Integration example to sync application dimensions between Cloud EDM
and Cloud EPM
rem Windows script for demonstration purposes only; do not use in
production environments

set EDMUSER=userid
set EDMSVR=https://hostname
set EDMPWDFILE=example_EDM
set EDMAPP=appname
set EDMDIM=dimname
set EDMLOC=location

set EPMUSER=userid
set EPMSVR=https://hostname
set EPIMPJOB=importjobname
set PWDFILE=C:\Oracle\EPM.epw
set DIMFILE=dimension.csv
set MAPFILE=mapping.csv

rem Synchronizing EDM ---> EPM
rem Export Dimension and Mappings from EDM

call epmautomate login %EDMUSER% %EDMPWDFILE% %EDMSVR%
call epmautomate exportdimension %EDMAPP% %EDMDIM% %DIMFILE%
call epmautomate exportdimensionmapping %EDMAPP% %EDMDIM% %EDMLOC%
%MAPFILE%
call epmautomate logout

rem Log into the Cloud EPM environment
call epmautomate login %EPMUSER% %PWDFILE% %EPMSVR%

rem Copy exported files from EDM environment to EPM and import metadata
and mappings
call epmautomate copyfilefrominstance %DIMFILE% %EDMUSER% %EDMPWDFILE%
%EDMSVR% inbox/%DIMFILE%
call epmautomate importmetadata %EPIMPJOB%

call epmautomate copyfilefrominstance %MAPFILE% %EDMUSER% %EDMPWDFILE%
%EDMSVR% inbox/%MAPFILE%
call epmautomate importmapping %EDMDIM% %MAPFILE% REPLACE FALSE %EDMLOC%

call epmautomate logout
```

2. スクリプト・ファイルを変更して、必要なパラメータ値を設定します。パラメータの説明と例は、[スクリプト実行用のパラメータ](#)を参照してください。
3. 必要に応じてスクリプトを手動で実行するか、実行をスケジュールします。[スクリプトの実行の自動化](#)を参照してください。

スクリプト実行用のパラメータ

この項のスクリプト・ファイルには、次の表で説明するいくつかのパラメータ値を指定する必要があります。これらのすべてのパラメータがすべてのスクリプトで使用されるとはかぎりません。

表 3-28 スクリプト・ファイル用のパラメータ値

パラメータ	説明
EDMUSER	Oracle Enterprise Data Management Cloud サービス管理者のユーザー・ログイン ID。 例: EDMUSER=jdoe@example.com
EDMSVR	Oracle Enterprise Data Management Cloud 環境の URL。 例: EDMSVR=https:// example.oraclecloud.com
EDMPWDFILE	Oracle Enterprise Data Management Cloud サービス管理者の暗号化パスワード・ファイル(EPW)の名前と場所。 例: EDMPWDFILE=edm_jdoe.epw
EDMAPP	Oracle Enterprise Data Management Cloud アプリケーションのディメンションの名前。 例: EDMAPP=USOperations
EDMDIM	エクスポートまたはインポートするディメンションの名前。 例: EDMDIM=entity
EDMLOC	エクスポートする場所の名前。 例: EDMLOC=Loc1
EPMUSER	Cloud EPM サービス管理者のログイン名。 例: EPMUSER=john.doe@example.com
EPMSVR	Cloud EPM 環境の URL。 例: EPMSVR=https://example.oraclecloud.com
EPMIMPJOB	Cloud EPM 環境の既存のインポート・ジョブ・タイプ <code>import metadata</code> の名前。 例: EPMIMPJOB=imp_DIMMetadata
EPMEXPJOB	Cloud EPM 環境の既存のジョブ・タイプ <code>export metadata</code> の名前。 例: EPMEXPJOB=Exp_DIMMetadata
PWDFILE	Cloud EPM サービス管理者用の暗号化パスワード・ファイル(EPW)の名前と場所。 encrypt コマンドを参照してください。 例: PWDFILE=pwd_jdoe.epw
DIMFILE	エクスポートしたディメンション・データを保存するファイルの名前。 例: DIMFILE=entity_file.CSV
MAPFILE	エクスポートしたマッピング・データを保存するファイルの名前。 例: MAPFILE=map_file.CSV

Cloud EPM ディメンションの Oracle Enterprise Data Management Cloud アプリケーションとの同期

このサンプル・シナリオでは、Oracle Fusion Cloud Enterprise Performance Management アプリケーションと Oracle Fusion Cloud Enterprise Data Management アプリケーション間でのディメンションの同期方法を調べます。

この項のスクリプトを使用して、次のタスクを完了できます:

- Cloud EPM アプリケーションからメタデータ(ディメンション)をエクスポートします
- ディメンション・データが保存されたエクスポート・ファイルを Oracle Enterprise Data Management Cloud 環境にコピーします
- ディメンション・メタデータを Oracle Enterprise Data Management Cloud アプリケーションにインポートします

Cloud EPM アプリケーションと Oracle Enterprise Data Management Cloud アプリケーション間でディメンションを同期するには:

1. 次のスクリプトをコピーして、Windows スクリプト・ファイルを作成します:

```
rem Integration example to sync an application dimension between Cloud EPM
and Cloud EDM
rem Windows script for demonstration purposes only; do not use in
production environments

set EDMUSER=userid
set EDMSVR=https://hostname
set EDMPWDFILE=example_EDM.epw
set EDMAPP=appname
set EDMDIM=dimname

set EPMUSER=userid
set EPMSVR=https://hostname
set PWDFILE=example_epm.epw
set EPMEXPJOB=exportjobname

rem Synchronizing EPM ---> EDM

rem Export Metadata from EPM
call epmautomate login %EPMUSER% %PWDFILE% %EPMSVR%
call epmautomate exportmetadata %EPMEXPJOB%
call epmautomate logout

rem Import Dimension to EDM
rem Log into the EDM environment
call epmautomate login %EDMUSER% %EDMPWDFILE% %EDMSVR%
rem Copy exported metadata file into the EDM environment
call epmautomate copyfilefrominstance %EPMEXPJOB%.zip %EPMUSER% %PWDFILE%
%EPMSVR% %EPMEXPJOB%.zip
call epmautomate importdimension %EDMAPP% %EDMDIM% ReplaceNodes
%EPMEXPJOB%.zip
call epmautomate logout
```

スクリプト・ファイルを変更して、必要なパラメータ値を設定します。パラメータの説明と例は、[スクリプト実行用のパラメータ](#)を参照してください。

2. 必要に応じてスクリプトを手動で実行するか、実行をスケジュールします。[スクリプトの実行の自動化](#)を参照してください。

スクリプトの実行の自動化

サービス管理者は Windows タスク・スケジューラでスクリプトをスケジュールするか、EPM 自動化を使用してアクティビティを自動化する cron ジョブを使用します。

Windows タスク・スケジューラを使用して EPM 自動化のスクリプト実行をスケジュールするには:

1. 「スタート」、「コントロール パネル」、「管理ツール」の順にクリックします。
2. 「タスク スケジューラ」を開きます。
3. 「操作」、「基本タスクの作成」の順に選択します。
4. タスクの名前と説明(オプション)を入力し、「次」をクリックします。
5. 「タスク トリガー」でスクリプトを実行するスケジュールを選択して、「次」をクリックします。
6. 次の画面でその他のスケジュール・パラメータを指定して「次」をクリックします。
7. 「操作」で「プログラムの開始」が選択されていることを確認します。
8. 「プログラムの開始」で次のステップを実行します。
 - a. 「プログラム/スクリプト」でスケジュールを設定するスクリプトを参照および選択します。
 - b. 「引数の追加 (オプション)」に、SET user スクリプト・パラメータで指定されたサービス管理者のパスワードを入力します。
 - c. 「開始 (オプション)」に、EPM 自動化がインストールされている場所(通常、C:/Oracle/EPMAutomate/bin)を入力します。
 - d. 「次」をクリックします。
9. 「概要」で、「[完了]をクリックしたときに、このタスクの[プロパティ]ダイアログを開く」を選択してから「完了」をクリックします。
10. 「全般」で、次のセキュリティ・オプションを選択してから「OK」をクリックします。
 - ユーザーがログオンしているかどうかにかかわらず実行する
 - 最上位の特権で実行する

EPM 自動化アクティビティのモニタリング

初期化した操作のステータスを識別しやすくするために、EPM 自動化には EPM 自動化を実行するコンソールのステータス・コードが表示されます。

終了コードを参照してください。

ジョブ・コンソールを使用して、EPM 自動化を使用して実行するジョブをモニターします。詳細は、*Planning の管理*のジョブの管理を参照してください。

4

EPM 自動化をインストールしないコマンドの実行

Groovy を使用すると、Oracle Fusion Cloud Enterprise Performance Management で直接 select コマンドを実行できます。サーバー側のコマンドを実行するために EPM 自動化をインストールする必要はありません。

Note:

このシナリオでは、Groovy スクリプトは、クライアント・マシンではなく、Cloud EPM で直接実行されるように記述されています。

この章の内容:

- サーバー側のコマンド実行をサポートする環境
- 情報ソース
- サポートされているコマンド
- サーバー側の Groovy を使用して EPM 自動化を実行するために使用されるメソッド
- サーバー側の Groovy スクリプトを使用した環境のクローニング
- サーバー側の Groovy スクリプトを使用したアクティビティ・レポートの電子メール送信

サーバー側のコマンド実行をサポートする環境

サーバー側での EPM 自動化のコマンド実行に対する Groovy スクリプトのサポートは、次の環境でのみ使用できます:

- EPM Enterprise Cloud Service 環境にデプロイされた Planning および Planning モジュール・ビジネス・プロセス。
- Enterprise Planning and Budgeting Cloud
- Planning and Budgeting Cloud (Plus One オプション付き)
- フリーフォーム
- Enterprise Profitability and Cost Management
- EPM Enterprise Cloud Service 環境の Financial Consolidation and Close。
- EPM Enterprise Cloud Service 環境の Tax Reporting。
- Strategic Workforce Planning
- Sales Planning

EPM 自動化のコマンドを組み込んだ Groovy スクリプトは、前述の Oracle Fusion Cloud Enterprise Performance Management 環境でのみ記述および実行できます。一方、このような

環境で記述されたスクリプトは、任意の Cloud EPM 環境で EPM 自動化のコマンドを発行できます。たとえば、Planning EPM Enterprise Cloud Service 環境でスクリプトを作成し、Groovy スクリプトをサポートしていない Narrative Reporting 環境でコマンドを発行できます。

情報ソース

詳細は、*Calculation Manager* での設計の次のソースを参照してください:

- Groovy ビジネス・ルールについて
- Groovy ビジネス・ルールの作成

サポートされているコマンド

以下を除いて、すべての EPM 自動化コマンドは Groovy を介して実行できます:

- [downloadFile](#)
- [upgrade](#)
- [uploadFile](#)

Groovy スクリプトを実行している環境では、次のコマンドを実行できません:

- [recreate](#)
- [replay](#)
- [resetService](#)
- [restructureCube](#)

Note:

- [encrypt](#) コマンドを実行すると、暗号化されたパスワード・ファイルがサーバーに作成され、長期使用のために保持されます。[listFiles](#) コマンドを実行する場合は、暗号化されたパスワード・ファイルはリストされません。[downloadFile](#) コマンドを使用してサーバーからダウンロードすることはできません。
- [feedback](#) コマンドを機能させるには、添付ファイルとして使用されるすべてのファイルとスクリーンショットがデフォルトのアップロード場所で利用可能である必要があります。[デフォルトのアップロード場所](#)を参照してください。これは、[uploadFile](#) コマンドで場所を指定しない場合にファイルが保存される場所です。

サーバー側の Groovy を使用して EPM 自動化を実行するために使用されるメソッド

- `getEPMAutomate ()` この静的メソッドは、EPM 自動化コマンドを呼び出すために使用できる `EpmAutomate` クラスのインスタンスを提供します。
- `execute ()` `EpmAutomate` クラスのこのメソッドは、EPM 自動化コマンドを実行するために使用されます。EPM 自動化コマンド名を最初のパラメータとして渡し、コマンド・オブ

ションを後続のパラメータとして渡します。このメソッドは、EpmAutomateStatus クラスのインスタンスを返します。

- getStatus () EPMAutomateStatus クラスのこのメソッドは、コマンドによって返された実行ステータスを返します。戻り値 0 は成功を示し、ゼロ以外の値はコマンドの失敗を意味します。
- getOutput () EPMAutomateStatus クラスのこのメソッドは、コマンドの文字列出力を返します。たとえば、このメソッドを使用して、getApplicationAdminMode と getDailyMaintenanceStartTime コマンドの出力を返すことができます。コマンドの戻りステータスがゼロ以外の場合、このメソッドはエラー・メッセージを返します。
- getItemsList () EPMAutomateStatus クラスのこのメソッドは、コマンドの文字列出力を返します。たとえば、このメソッドを使用して、getSubstVar、listBackups、listFiles の各コマンドの出力を返すことができます。

サーバー側の Groovy スクリプトを使用した環境のクローニング

サーバー側の Groovy スクリプトに EPM 自動化コマンドを追加して、障害回復目的で環境をクローニングできます。これにより、オンプレミスのフットプリントなしで障害回復を設定できます。

パスワードに特殊文字が含まれている場合は、[特殊文字の処理](#)を参照してください。また、使用環境にあわせて次のパラメータ値を確実に置き換えてください:

Table 4-1 変更するパラメータ

パラメータ	説明
password	ソース環境でクローン操作を実行しているサービス管理者のパスワード。
targetpassword	ターゲット環境でクローン操作を実行しているサービス管理者のパスワード。
username	ソース環境のサービス管理者のユーザー ID。
targetusername	ターゲット環境のサービス管理者のユーザー ID。このユーザーは、ターゲット環境のアイデンティティ・ドメイン管理者の役割にも割り当てられている必要があります。
email_id	クローニング・プロセスに関する情報を送信する予定の電子メール・アドレス。

パスワードを暗号化するためのスクリプト

```
EpmAutomate automate = getEpmAutomate()

//Encrypt the password of a Service Administrator in the source environment
EpmAutomateStatus encryptstatus1 = automate.execute('encrypt',
'password','encryptionKey','sourcePassword.epw')
if(encryptstatus1.getStatus() != 0)
throwVetoException(encryptstatus1.getOutput())
println(encryptstatus1.getOutput())

//Encrypt the password of a Service Administrator in the target environment
//This user must also have the Identity Domain Administrator
```

```
//role in the target environment

EpmAutomateStatus encryptstatus2= automate.execute('encrypt',
'targetpassword',
'encryptionKey', 'targetPassword.epw')
if(encryptstatus2.getStatus() != 0)
throwVetoException(encryptstatus2.getOutput())
println(encryptstatus2.getOutput())
```

環境をクローニングするためのスクリプト

このスクリプトでは、前述のスクリプトを使用して作成した暗号化されたパスワード・ファイルを使用します。

```
EpmAutomate automate = getEpmAutomate()

//Log into the target environment
EpmAutomateStatus loginstatus = automate.execute('login',
'username','targetPassword.epw' , 'targeturl')
if(loginstatus.getStatus() != 0) throwVetoException(loginstatus.getOutput())
println(loginstatus.getOutput())

//Recreate the target environment
EpmAutomateStatus recreatestatus = automate.execute('recreate' , '-f' )
if(recreatestatus.getStatus() != 0)
throwVetoException(recreatestatus.getOutput())
println(recreatestatus.getOutput())

//Copy Artifact Snapshot from the source environment
//to the target environment
EpmAutomateStatus copystatus = automate.execute('copysnapshotfrominstance',
'Artifact Snapshot', 'sourceusername', 'sourcePassword.epw','source url')
if(copystatus.getStatus() != 0) throwVetoException(copystatus.getOutput())
println(copystatus.getOutput())

//import Artifact Snapshot into the target environment
EpmAutomateStatus importstatus = automate.execute('importsnapshot', 'Artifact
Snapshot')
println(importstatus.getOutput())

//Send an email to a designated user with the status of the
//snapshot import process
EpmAutomateStatus emailstatus = automate.execute('sendmail',
'email_id' , 'Status of DR' , 'BODY='+ importstatus.getOutput())
println(emailstatus.getOutput())

//Sign out of the target environment
EpmAutomateStatus logoutstatus = automate.execute('logout')
println(logoutstatus.getOutput())
```

サーバー側の Groovy スクリプトを使用したアクティビティ・レポートの電子メール送信

このスクリプトを使用して、アクティビティ・レポートを受信者のリストに電子メールで送信できます。このスクリプトは、毎日実行するようにスケジュールして、毎日のアクティビティ・レポートを取得できます。このスクリプトは、次の機能を実行します。

- Groovy スクリプトを実行するサービス管理者のパスワードを暗号化します。
- 暗号化されたパスワードを使用して、Oracle Fusion Cloud Enterprise Performance Management 環境にサインインします。
- 環境で利用可能なアクティビティ・レポートを受信者のリスト(通常はサービス管理者)に電子メールで送信します
- 環境からサインアウトします。

パスワードに特殊文字が含まれている場合は、[特殊文字の処理](#)を参照してください。また、使用環境にあわせて次のパラメータ値を確実に置き換えてください:

Table 4-2 変更するパラメータ

パラメータ	説明
user	環境にサインインするサービス管理者のユーザー ID。
password	サービス管理者のパスワード
url	アクティビティ・レポートの電子メール送信元となる Cloud EPM 環境の URL。
emailaddresses	アクティビティ・レポートの送信先となる電子メール・アドレスのセミコロン区切りリスト。

Groovy ルールの使用の詳細は、*Planning の管理*の Groovy ルールの使用を参照してください。

```
/*RTPS: {user} {password} {url} {emailaddresses}*/
import java.text.SimpleDateFormat

String user = 'service_administrator'
String password = 'examplePWD'
String url = 'example_EPM_URL'
String emailaddresses = 'service_administrator@oracle.com'

EpmAutomate automate = getEpmAutomate()

def LogMessage(String message) {
    def date = new Date()
    def sdf = new SimpleDateFormat("MM/dd/yyyy HH:mm:ss")
    println([' ' + sdf.format(date) + '][GROOVY] ' + message);
}

def LogOperationStatus(EpmAutomateStatus opstatus) {
    def returncode = opstatus.getStatus()
    LogMessage(opstatus.getOutput())
    LogMessage('return code: ' + returncode)
}
```

```
LogMessage('Starting mail activity report processing')

// encrypt
LogMessage("Operation: encrypt " + password + " oracleKey password.epw")
EpmAutomateStatus status =
automate.execute('encrypt',password,"oracleKey","password.epw")
LogOperationStatus(status)

// login
LogMessage("Operation: login " + user + " password.epw " + url)
status = automate.execute('login',user,"password.epw",url)
LogOperationStatus(status)

// listfiles
LogMessage('Operation: listfiles')
status = automate.execute('listfiles')
LogOperationStatus(status)

String filelist = status.getItemsList()
String[] str = filelist.split(',');
String reportfile = ''

for( String svalues : str ) {
    String[] ftr = svalues.split('/')
    for( String fvalues : ftr ) {
        if (fvalues.startsWith('2') && fvalues.endsWith('html')) {
            reportfile = fvalues
        }
    }
}

def reportdir = reportfile.tokenize(".")[0]
String reportpath = 'apr/' + reportdir + '/' + reportfile

// sendMail
LogMessage('Operation: sendMail ' + emailaddresses + ' Daily Activity Report
Body=Daily Activity Report Attachments=' + reportpath)
status = automate.execute('sendmail',emailaddresses,'Daily Activity
Report','Body=Daily Activity Report',"Attachments=${reportpath}")
LogOperationStatus(status)

// logout
LogMessage('Operation: logout')
status = automate.execute('logout')
LogOperationStatus(status)
```

5

Cloud EPM 環境のレプリケート

次のステップは、Oracle Fusion Cloud Enterprise Performance Management のセカンダリ環境の構成に含まれています。これにより、予期しない状況のためにプライマリ Oracle データ・センターが使用できなくなった場合に、サービスの可用性が保証されます。

ノート:

この付録で説明する手順は、Narrative Reporting には適用されません。

- [日次アーティファクト・レプリケーションの設定](#)
- [オンデマンド・レプリケーションの設定](#)
- [セカンダリ環境の構成](#)

日次レプリケーションの設定

環境をレプリケートするには、EPM 自動化を使用して、日次メンテナンス中に作成したアーティファクト・スナップショットをプライマリ環境からセカンダリ環境にコピーします。

各環境では、ルーチン・メンテナンスが毎日実行されています。このサービス・メンテナンスでは、サービス環境のコンテンツ(アイデンティティ・ドメインのユーザーと役割の割当てを含む既存のデータとアーティファクト)がバックアップされ、メンテナンス・スナップショットが作成されます。

日次サービス・レプリケーションを設定するには:

1. EPM 自動化の次のコマンドを含むスクリプト・ファイルを作成します。このスクリプトは、プライマリ環境のアプリケーション・スナップショットをセカンダリ環境にレプリケートします。

ノート:

ユーザー名、パスワード・ファイル、アイデンティティ・ドメイン名およびサービス URL を変更してください。暗号化されたパスワード・ファイルの作成方法の詳細は、[encrypt](#) コマンドを参照してください。

```
REM Sign in to the secondary instance
epmautomate login serviceAdmin secondaryPassword.epw secondary_URL
secondaryDomain
REM Delete the existing artifact snapshot
epmautomate deletefile "Artifact Snapshot"
REM Copy the snapshot from the primary instance
epmautomate copysnapshotfrominstance "Artifact Snapshot"
```

```
primaryPassword.epw primary_URL primaryDomain
REM Sign out of the secondary instance
epmautomate logout
```

2. スケジューラ (Windows タスク・スケジューラなど) を使用し、メンテナンス・ウィンドウが開始してから 2 時間後にスクリプト・ファイルが実行するようにスケジュール設定します。
3. プライマリ環境とセカンダリ環境の両方でメンテナンス・ウィンドウの開始時刻を同じに設定します。詳細は、*管理者用スタート・ガイド* でサービス・メンテナンス時間の設定を参照してください。

オンデマンド・レプリケーションの設定

RPO を短縮するために、プライマリ環境のオンデマンド・スナップショットを作成してから、セカンダリ環境にコピーできます。

たとえば、EPM 自動化スクリプトを作成して、日次レプリケーションの間に 6 時間おきに実行するようにスケジュール設定すると、RPO を 24 時間から 6 時間に短縮できます。

ノート:

オンデマンド・スナップショットの作成中に、プライマリ環境は数分間読取り専用モードになります。

オンデマンド・レプリケーションを設定するには:

1. EPM 自動化の次のコマンドを含むスクリプト・ファイルを作成します。このスクリプトは、プライマリ環境のアプリケーション・スナップショットをセカンダリ環境にレプリケートします。

ノート:

ユーザー名、パスワード・ファイル、アイデンティティ・ドメイン名およびサービス URL を変更してください。暗号化されたパスワード・ファイルの作成方法の詳細は、[encrypt](#) コマンドを参照してください。

```
REM Sign in to the primary instance
epmautomate login serviceAdmin primaryPassword.epw primary_URL
primaryDomain
REM Create a snapshot and then sign out
epmautomate exportsnapshot "Artifact Snapshot"
epmautomate logout
REM Sign in to the secondary instance
epmautomate login serviceAdmin secondaryPassword.epw secondary_URL
secondaryDomain
REM Copy the snapshot from the primary instance
epmautomate copysnapshotfrominstance "Artifact Snapshot"
primaryPassword.epw primary_URL primaryDomain
```

```
REM Sign out of the secondary instance
epmautomate logout
```

2. スケジューラ(Windows タスク・スケジューラなど)を使用して、望ましい RPO を実現するために必要に応じてスクリプト・ファイルの実行をスケジュール設定します。

セカンダリ環境の構成

セカンダリ環境を構成してアクティブ化します。

この手順を行うのは、プライマリ環境を長期間使用できないときにセカンダリ環境をアクティブ化する必要がある場合のみです。セカンダリ環境を構成する前に、*管理者用スタート・ガイド*の次のトピックを参照してください:

- 従来の EPM Cloud のスナップショットの移行パス
- EPM Standard Cloud Service および EPM Enterprise Cloud Service のスナップショットの移行パス

セカンダリ環境を構成するには:

1. EPM 自動化セッションを開始し、次のアクティビティを実行します。
 - サービス管理者とアイデンティティ・ドメイン管理者の両方の役割を持つアカウントを使用して、セカンダリ環境にサインインします。適切なユーザー名、パスワード、ドメイン名およびサービス URL を指定してください。
 - セカンダリ環境を再作成します。
 - プライマリ環境が **Planning**、**Tax Reporting**、**Financial Consolidation and Close** または **Enterprise Profitability and Cost Management** 環境である場合、次を使用します:


```
epmautomate recreate -f
```
 - プライマリ環境が **Planning**、**Tax Reporting**、**Financial Consolidation and Close** または **Enterprise Profitability and Cost Management** 環境でない場合、次を使用します:


```
epmautomate recreate -f TempServiceType=PRIMARY_APPLICATION_TYPE。
```

 ここで、**PRIMARY_APPLICATION_TYPE** は ARCS、EDMCS、PCMCS、EPRCS です。
 - スナップショットからアプリケーションとアイデンティティ・ドメイン・アーティファクトをインポートします。
 - サインアウトします。

これらのアクティビティは次のコマンドを実行して行うことができます。次のトピックを参照してください。

- [login](#) コマンド
- [recreate](#) コマンド
- [importSnapshot](#) コマンド

```
epmautomate login serviceAdmin secondaryPassword.epw secondary_URL
epmautomate recreate -f
epmautomate importsnapshot "Artifact Snapshot" importUsers=true
epmautomate logout
```

2. セカンダリ環境にサインインして、すべてのデータが使用できることを確認します。

3. セカンダリ環境の URL をすべてのユーザーに送信します。

A

simulateConcurrentUsage コマンドの実行の準備

[simulateConcurrentUsage](#) コマンドは、環境の負荷をシミュレートするために次の操作をサポートしています:

- フォームを開く
- フォームの保存
- ビジネス・ルールの実行
- ビジネス・ルールセットの実行
- データ・ルールの実行
- アド・ホック・グリッドを開く
- 管理レポートのブックの実行
- 管理レポートのレポートの実行

関係するステップ

1. [requirement.csv](#) ファイルを作成します。[requirement.csv](#) ファイルの作成を参照してください
2. [requirement.csv](#) に含まれる操作の詳細を指定した入力ファイルを作成します。参照:
 - [フォーム入力ファイルを開く](#)
 - [フォーム入力ファイルの保存](#)
 - [ビジネス・ルール入力ファイルの実行](#)
 - [ビジネス・ルールセット入力ファイルの実行](#)
 - [データ・ルール入力ファイルの実行](#)
 - [アド・ホック・グリッド入力ファイル](#)
 - [ブック入力ファイルの実行](#)
 - [レポート入力ファイルの実行](#)
3. ユーザー変数メンバー・マッピングを設定する必要がある場合は、ユーザー変数の詳細を格納した [UserVarMemberMapping.csv](#) を作成します。[UserVarMemberMapping.csv](#) ファイルの作成を参照してください
4. Smart View オプションを使用する必要がある場合は、オプションで、Oracle Smart View for Office オプションを [options.xml](#) にエクスポートします。詳細は、[options.xml](#) ファイルの作成を参照してください。
5. オプションで、[users.csv](#) ファイルを作成して、既存のユーザーを使用してシミュレーションを実行します。詳細は、[users.csv](#) ファイルの作成を参照してください。
6. 前述のステップで作成したファイルを格納した ZIP ファイルを作成し、環境にアップロードします。[入力 ZIP ファイルの作成および環境へのアップロード](#)を参照してください

- アップロードした ZIP ファイルを使用して、`simulateConcurrentUsage` コマンドを実行します。

requirement.csv ファイルの作成

最初に、テストするユース・ケースの詳細をリストした `requirement.csv` ファイルを作成します。この CSV ファイルの各行は、実行する操作のタイプ、アーティファクト名、同時ユーザー数、操作の詳細を指定した入力ファイル、および操作に関連する情報(ある場合)を識別します。たとえば、2つのフォームを開き、2つのフォームを保存し、2つのビジネス・ルールを実行するには、入力 CSV ファイルに 6 行を指定します。`requirement.csv` の最初の行には、次の情報を格納する必要があります:

```
#Type of Operation,Artifact Name,Number of Users,Input File,Additional Info
```

ファイルの後続する各行には、単一の操作とそのパラメータを格納します。操作によっては、これらのパラメータのすべての値が必須ではない場合があります。想定されるファイル・エンタリについて、次の表で説明します。



Note:

表で特に示されていないかぎり、値はすべて必須です。

Table A-1 requirement.csv の形式

フィールド	説明
操作のタイプ	次のいずれかである必要があります: <ul style="list-style-type: none"> Open Form Save Form Run Business Rule Run Business Ruleset Run Data Rule Ad Hoc Grid Execute Report Execute Book
アーティファクト名	この値は操作のタイプによって異なります: <ul style="list-style-type: none"> Open Form: 開くフォームの名前と場所。 Save Form: 保存するフォームの名前と場所。 Run Business Rule: ビジネス・ルールの名前。 Run Business Ruleset: ビジネス・ルールセットの名前。 Data Rule: データ・ルールの名前。 Ad Hoc Grid: 適用なし(空白のまま)。 Execute Report: レポートの名前と場所。 Execute Book: ブックの名前と場所。
ユーザー数	同時使用をシミュレートするユーザーの数。
入力ファイル	POV 値、実行時プロンプトまたは使用するその他ユース・ケース固有の値が格納された CSV ファイルの名前
追加情報	操作に必要な追加のパラメータ。アド・ホック・グリッドにのみ適用されます。その他のユース・ケースでは空白のままにします。

Table A-1 (Cont.) requirement.csv の形式

フィールド	説明
<p>ノート: アーティファクト名はアプリケーション内の名前と一致し、大/小文字も同じである必要があります。</p> <p>requirement.csv ファイルの例:</p> <pre># Type of Operation,Artifact Name,Number of Users,Input File,Additional Info Open Form,Library/Global Assumption/Revenue Forecast Assumptions,10,openform_input.csv, Save Form,Library/Global Assumption/ExchangeRates,5,saveform_input.csv, Run Business Rule,Run_FinStatement - Copy Budget to Prior Year Budget,4,runbusinessrule_input.csv, Run Business Ruleset,RollupUSSales,5,runbusinessruleset_input.csv, Run Data Rule,Delimited_file_DL,5,rundatarule_input.csv, Ad Hoc Grid,,3,runadhocgrid_input.csv,cube=FinStmt Execute Book,Review Books/Revenue Reports,10,book_input.csv Execute Report,Review Reports/Executive Report,10,report_input.csv,</pre>	

入力ファイルの作成

requirement.csv で識別される各ユース・ケースには、その実行に必要なすべてのパラメータを提供する、対応する入力ファイルが必要です。

入力ファイルには、理想的には、requirement.csv でこのユース・ケースに指定されたユーザーの数ごとに 1 つのエントリが含まれている必要があります。

入力ファイル内のエントリ数が、requirement.csv 内のそのユース・ケースの同時ユーザー数より少ない場合、EPM 自動化は、requirement.csv で指定された数のユーザーに対して操作が実行されるまで、入力ファイルのいくつかのエントリを繰り返してユース・ケースを実行します。

たとえば、ビジネス・ルールの実行操作の requirement.csv に、次のユース・ケース・エントリがあるとします:

```
Run Business Rule, Copy Budget,10,br_input_file.csv,
```

br_input_file.csv にも 10 個のエントリが含まれている必要があります。

br_input_file.csv に含まれているエントリが 6 個のみである場合、EPM 自動化は最初の 6 人のユーザーに対してそれらのエントリを使用します。次の 4 人のユーザーに対して、EPM 自動化は br_input_file.csv の最初の 4 個のエントリを再利用します。

入力ファイル内のエントリの数が、ユース・ケースに指定されたユーザーの数を超える場合、EPM 自動化は入力ファイルの最後の余分なエントリを無視します。

- [フォーム入力ファイルを開く](#)
- [フォーム入力ファイルの保存](#)
- [ビジネス・ルール入力ファイルの実行](#)
- [ビジネス・ルールセット入力ファイルの実行](#)
- [データ・ルール入力ファイルの実行](#)

- [アド・ホック・グリッド入力ファイル](#)
- [レポート入力ファイルの実行](#)
- [ブック入力ファイルの実行](#)

フォーム入力ファイルを開く

フォームを開く機能をサポートするために requirement.csv で参照されるこのファイルには、pov=[DIM 1:MEMBER 1],[DIM 2:MEMBER 2],[DIM 3:MEMBER 3],などの形式の POV エントリが含まれています。

ここで示されている DIM 1、DIM 2 などはディメンション名であり、MEMBER 1、MEMBER などは POV のディメンション・メンバー値です。

サンプル入力ファイル:

```
pov=[Account:APL_RATE_AED],[Scenario:Budget],[Years:FY20]
pov=[Account:APL_RATE_AED],[Scenario:Budget],[Years:FY19]
pov=[Account:APL_RATE_AED],[Scenario:Budget],[Years:FY18]
pov=[Account:APL_RATE_AED],[Scenario:Budget],[Years:FY17]
pov=[Account:APL_RATE_AED],[Scenario:Budget],[Years:FY16]
```

Note:

requirement.csv に指定したフォームにユーザー変数を設定する必要がある場合は、UserVarMemberMapping.csv も作成する必要があります。

[UserVarMemberMapping.csv ファイルの作成](#)を参照してください。

フォーム入力ファイルの保存

フォームの保存をサポートするために requirement.csv で参照されるこのファイルには、POV およびセル入力値が次の形式に含まれています。フォームにビジネス・ルールが関連付けられている場合、このファイルにはそれらのビジネス・ルール用の実行時プロンプトも含まれています:

サンプル入力ファイル:

```
pov=[DIM 1:MEMBER1],[DIM 2:MEMBER2],[DIM 3:MEMBER3],...;cells=[CELL COLUMN
HEADER 1 -> CELL COLUMN HEADER 2 -> CELL COLUMN HEADER 3 ->.. | CELL ROW
HEADER 1-> CELL ROW HEADER 2-> CELL ROW HEADER 3->..| CELL 1 DATA], [CELL
COLUMN HEADER 11 -> CELL COLUMN HEADER 22 -> CELL COLUMN HEADER 33 ->.. |
CELL ROW HEADER 11-> CELL ROW HEADER 22-> CELL ROW HEADER 33->..| CELL 2
DATA];rtp=[BUSINESS RULE NAME1[RTP1:VALUE1][RTP2:VALUE2]],[BUSINESS
RULE2[RTP3:VALUE3]]..
```

この例では:

- DIM はディメンションの名前を示し、MEMBER はディメンション・メンバー値を示します
- CELL COLUMN HEADER は列ヘッダーの名前を識別し、CELL ROW HEADER は行ヘッダーの名前を識別します

- **BUSINESS RULE NAME** はビジネス・ルールの名前を示し、**RTP** は実行時プロンプト名を示し、**VALUE** はその値を示します。フォームにビジネス・ルールが関連付けられていない場合や、フォームにあるデフォルト実行時プロンプト値を使用する場合には、**RTP** は必要ありません。

次に例を示します。

```
pov=[Version View:Working],[Sales Entity:International Sales];cells=[FY16-  
>x-----x->Pct|P293:Maintenance->4120: Support|1];rtp=[Services Revenue -  
Forecast[Department:000][Scenario:Plan]],[Allocate Plan  
Targets[TargetVersion:Baseline]]
```

Note:

requirement.csv に指定したフォームにユーザー変数を設定する必要がある場合は、UserVarMemberMapping.csv も作成する必要があります。
[UserVarMemberMapping.csv ファイルの作成](#)を参照してください。

フォームにスマート・プッシュが関連付けられている場合は自動的に実行されます。

ビジネス・ルール入力ファイルの実行

ビジネス・ルールの実行をサポートするために requirement.csv で参照されるこのファイルには、rtp=[*PARAMETER:VALUE*],[*PARAMETER:VALUE*]などの形式のランタイム・パラメータの値が含まれています。この形式で、*PARAMETER* は実行時プロンプト名を識別し、*VALUE* はその値を識別します。

ルールを実行するために必要な数の実行時プロンプトを確実に追加してください。実行時プロンプトとその値を指定しない場合は、デフォルト値が使用されます。このコマンドは、ルールで定義されているものと完全一致しない実行時プロンプトを無視します。ビジネス・ルールにランタイム・パラメータが必要ない場合は、rtp=[]を指定します。

サンプル入力ファイル:

```
rtp=[Period:Q1],[Entity:USA]  
rtp=[Period:Q2],[Entity:USA]  
rtp=[Period:Q3],[Entity:USA]  
rtp=[Period:Q4],[Entity:USA]
```

ビジネス・ルールセット入力ファイルの実行

ビジネス・ルールセットの実行をサポートするために requirement.csv で参照されるこのファイルには、rtp=[*PARAMETER:VALUE*],[*PARAMETER:VALUE*]などの形式のランタイム・パラメータの値が含まれています。この形式で、*PARAMETER:VALUE* は、ランタイム・パラメータ名とその値を識別します。

PARAMETER:VALUE のペアを使用して、ルールセットで必要とされるだけの実行時プロンプトを指定します。ランタイム・パラメータに値を指定しない場合は、デフォルト値が使用されます。このコマンドは、ルールセットで定義されているものと完全一致しない実行時プロンプトを無視します。ビジネス・ルールセットにランタイム・パラメータ値が必要ない場合は、rtp=[]を指定します。

サンプル入力ファイル:

```
rpt=[Period:Q1],[Entity:USA]
rtp=[Period:Q2],[Entity:USA]
rtp=[Period:Q3],[Entity:USA]
rtp=[Period:Q4],[Entity:USA]
```

データ・ルール入力ファイルの実行

データ・ルールの実行をサポートするために `requirement.csv` で参照されるこのファイルには、開始期間、終了期間、インポート・モード、エクスポート・モード、および環境で使用可能なオプションのインポート・ファイル名を指定してください。ファイル名が指定されていない場合は、データ・ルールで指定されたファイル名が使用されます。各行の形式:

```
startperiod=START PERIOD;endperiod=END
PERIOD;importmode=IMPORT_MODE;exportmode=EXPORT_MODE;filename=FILE NAME
```

サンプル入力ファイル:

```
startperiod=Dec-15;endperiod=Dec-15;importmode=REPLACE;exportmode=STORE_DATA;filename=comma_delim_file1.csv
startperiod=Dec-16;endperiod=Dec-16;importmode=REPLACE;exportmode=STORE_DATA;filename=comma_delim_file2.csv
startperiod=Dec-17;endperiod=Dec-17;importmode=REPLACE;exportmode=STORE_DATA;filename=comma_delim_file3.csv
startperiod=Dec-18;endperiod=Dec-18;importmode=REPLACE;exportmode=STORE_DATA;filename=comma_delim_file4.csv
startperiod=Dec-19;endperiod=Dec-19;importmode=REPLACE;exportmode=STORE_DATA;filename=comma_delim_file5.csv
```

アド・ホック・グリッド入力ファイル

`simulateConcurrentUsage` コマンドでは、ネイティブ・モードと標準モードの両方のアド・ホック・グリッドがサポートされます。ネイティブ・モードのグリッドの場合、POV は Oracle Smart View for Office プラグイン・ツール・バーの一部として表示されます。標準モードの場合、POV はスプレッドシート自体の一部になり、スプレッドシートの最初の行を占めます。

アド・ホック・グリッドを開く機能をサポートするために `requirement.csv` で参照されるアド・ホック・グリッド入力ファイルには、開くグリッドを指定します。ファイルの各行を次の形式にする必要があります。

```
filename=xlsx filename#sheet name; pov=[DIM 1:MEMBER 1],[DIM 2:MEMBER 2]..;
rows=[ROW HEADER 1, ROW HEADER 2,..]; cols=[COL HEADER 1, COL HEADER 2,..]
```

サンプル入力ファイル:

```
fileName=dropdown.xlsx#sheet4;pov=[HSP_View:BaseData],[Scenario:Forecast],
[Product:No Product],[Entity:Sales Mid-Atlantic];rows = [ Account ]; cols=
[Year, Period, Version]
```

 **Note:**

入力 CSV ファイル内の `fileName` で識別されるファイルには、指定されたシートにアド・ホック・グリッド定義が格納されている必要があります。たとえば、前述のサンプル入力ファイルでは、アド・ホック・グリッド定義のソースとして `dropdown.xlsx` の `sheet4` を指定しています。この Excel ファイルは、`requirement.csv` および入力 CSV ファイルとともに、`simulateConcurrentUsage` コマンドの実行に使用される `INPUT_FILE.zip` で使用できる必要があります。

レポート入力ファイルの実行

管理レポートを開く機能をサポートするために `requirement.csv` で参照されるこのファイルには、開くレポートを指定します。ファイルの各行を次の形式にする必要があります。

```
format=[REPORT_FORMAT];globalPov=[DIM 1:MEMBER 1],[DIM 2:MEMBER 2],...;prompts=[PROMPT 1:VALUE 1],[PROMPT 2:VALUE 2],...
```

`globalPov` および `prompts` はオプションです。

 **Note:**

- サポートされている形式は `pdf` および `embedded` です。
- `globalPov` ディメンションまたはそのメンバーの名前にコロン(:)またはセミコロン(;)`が含まれている場合は、その直前にエスケープ文字\を使用します。たとえば、ディメンション名 Version:View は、Version\\:View と指定してください`

グローバル POV [`Version View:Working`],[`Sales Entity:International Sales`] およびプロンプト [`Actual;Budget`],[`Year:2018`] を使用してレポートから PDF を生成するサンプル入力ファイル:

```
format=pdf;globalPov=[Version View:Working],[Sales Entity:International Sales];prompts=[Actual:Budget],[Year:2018]
```

ブック入力ファイルの実行

レポートでブックを開く機能をサポートするために `requirement.csv` で参照されるこのファイルには、開くブックを指定します。ファイルの各行を次の形式にする必要があります。

```
format=BOOK_FORMAT または
```

```
format=BOOK_FORMAT;globalPov=[DIM 1:MEMBER 1],[DIM 2:MEMBER 2],...
```

グローバル POV [`Version View:Working`],[`Sales Entity:International Sales`] が含まれているブックから PDF を生成するサンプル入力ファイル:

```
format=pdf;globalPov=[Version View:Working],[Sales Entity:International Sales]
```

 **Note:**

- サポートされているブック形式は PDF および XLSX です。
- globalPov ディメンションまたはそのメンバーの名前にコロン(:)またはセミコロン(;)が含まれている場合は、その直前にエスケープ文字\\を使用します。たとえば、ディメンション名 Version:View は、Version\\:View と指定してください

UserVarMemberMapping.csv ファイルの作成

フォームを開くまたはフォームを保存するコース・ケースの入力ファイルで指定したフォームにユーザー変数を設定する必要がある場合は、このファイルが必要です。このファイルは、他のコース・ケースには必要ありません。

このファイルの最初の行はヘッダー #Dimension, User Variable, Member です

後続するエントリには、ディメンション、ユーザー変数およびディメンション・メンバーのマッピングを格納します。

UserVarMemberMapping.csv ファイルのサンプル:

```
#Dimension, User Variable, Member
Account, Account View, Revenue Driver Assumptions
Entity, Entity, No Entity
Entity, Entity View, Total Entity
HSP_View, HSP_View, BaseData
Market Size, Market View, Large Market
Period, Period, Jan
```

options.xml ファイルの作成

行の抑制、欠落ブロックの抑制およびページ・メンバーのインデントなどの Oracle Smart View for Office オプションは、フォームを開く、フォームの保存およびアド・ホック・グリッドのコース・ケースをシミュレートしているときに使用できます。これらの各オプションにより、環境に異なるレベルの負荷がかかります。

Smart View オプション使用の負荷をシミュレートする場合は、入力 ZIP ファイルに options.xml ファイルを含めることができます。

Smart View オプションをエクスポートして、options.xml を作成します。options.xml を生成するには、「SmartView」タブから「オプション」をクリックし、次に「OK」をクリックしたら「オプションのエクスポート」をクリックします。

options.xml の使用は必須ではありません。入力 ZIP ファイルにこのファイルが含まれない場合は、シミュレーションにデフォルト・オプションが使用されます。

users.csv ファイルの作成

simulateConcurrentUsage コマンドのモード 4 では、入力 ZIP ファイルに格納された users.csv ファイルで識別される既存のユーザーを使用してコマンドを実行できます。このモードでは、シミュレーション用のユーザーは作成されません。

users.csv ファイルの形式は loginname,password です:

```
jdoe,jdoe_pwd
john.doe@example.com,john_doe_pwd
```

users.csv の loginname の値は、アイデンティティ・ドメインに定義されているログイン名と一致している必要があります。また、識別されたユーザーは、操作を実行するために必要な事前定義済役割とアプリケーション役割を持っている必要があります。

入力 ZIP ファイルの作成および環境へのアップロード

7 Zip などのツールを使用して、requirement.csv、対応するユース・ケース入力ファイル、オプションで UserVarMemberMapping.csv、users.csv および options.xml(必要な場合)を格納した単一の ZIP ファイルを作成します。

`uploadFile` コマンドを使用して、シミュレーションを実行する環境の受信ボックスに結果の ZIP ファイルをアップロードします(コマンド構文の例: `epmautomate uploadFile "C:/uploads/INPUT_FILE.zip" inbox`)。

同時使用のシミュレート・レポートのサンプル

同時使用のシミュレートは、デフォルトでは、`simulateConcurrentUsage` コマンドを実行するユーザーに送信されます。電子メールの受信者を指定すると、レポートはそれらの受信者にのみ電子メールで送信されます。

Simulate Concurrent Usage report

Operation #	Operation	Artifact Name	Users	Iterations	Min. Duration	Max. Duration	Avg. Duration	Return Status
1	Open Form	Set Services Revenue Forecast Assumptions	2	1	00:00:00.17	00:00:00.21	00:00:00.19	Passed
2	Save Form	Set Headcount and Salary Forecast Assumptions	1	1	00:00:00.45	00:00:00.45	00:00:00.45	Passed
3	Run Business Rule	Operating Expense Adj Plan	2	1	00:00:00.60	00:00:00.61	00:00:00.60	Passed
4	Run Data Rule	test	2	1	00:00:03.00	00:00:03.29	00:00:03.14	Passed
5	Ad Hoc Grid		1	1	00:00:00.13	00:00:00.13	00:00:00.13	Passed
6	Execute Book	Book1	1	1	00:00:10.21	00:00:10.21	00:00:10.21	Passed
7	Execute Report	Variance Explanations1	1	1	00:00:04.95	00:00:04.95	00:00:04.95	Passed
8	Save Form	Set Services Revenue Forecast Assumptions	1	1	00:00:05.56	00:00:05.56	00:00:05.56	Passed
9	Run Business Rule	Allocate Plan Targets	2	1	00:00:02.62	00:00:02.62	00:00:02.62	Passed
10	Run Business Ruleset	Revenue Plan	2	1	00:00:00.60	00:00:00.61	00:00:00.61	Passed

このレポートでは、次のことが識別されます:

列	説明
操作番号	requirement.csv のユース・ケースのシーケンス番号
操作	requirement.csv に指定されている操作のタイプ
アーティファクト名	requirement.csv に指定されているアーティファクト名
ユーザー	requirement.csv に指定されているユーザーの数
反復	iterations パラメータで指定された、ユース・ケースが実行された回数
最小期間	1 人のユーザーがこのユース・ケースを実行するのに要した最小時間

列	説明
最大期間	1人のユーザーがこのコース・ケースを実行するのに要した最大時間
平均期間	1人のユーザーがこのコース・ケースを実行するのに要した平均時間
戻りステータス	コース・ケースのステータス。コース・ケースの実行が正常に完了しなかった場合は Failed が表示されます

B

replay コマンドの実行準備

replay コマンドを使用して、負荷がかかった状況でパフォーマンス・テストを行い、指定された負荷の影響をサービスが受ける際のユーザー・エクスペリエンスを許容できるかどうかを確認します。テスト環境をロードする前に、いくつかのステップを完了する必要があります。

この付録では、サービス管理者が EPM 自動化の replay コマンドを実行する前に完了しておく必要があるステップについて説明します。

- [replay コマンドについて](#)
- [前提条件](#)
- [HAR ファイルの作成](#)
- [リプレイ・ファイルの作成](#)
- [トレース・ファイルの生成](#)
- [サンプル・リプレイ・セッション](#)

replay コマンドについて

replay コマンドを使用すると、環境に対する Oracle Smart View for Office、Oracle Fusion Cloud Enterprise Performance Management の REST API または EPM 自動化の負荷がリプレイされます。これにより、高負荷の状況でのパフォーマンス・テストが可能になり、環境が指定された負荷の影響下にある場合にユーザー・エクスペリエンスを許容できるかどうかを確認できます。

たとえば、高負荷のテスト環境でユーザー・エクスペリエンスをテストして、アプリケーションをテスト環境から本番環境に移行した後もユーザー・エクスペリエンスが問題なく実行されることを確認できます。

前提条件

リプレイ・ファイルを使用してコマンドを実行すると、EPM 自動化はリプレイ・ファイルの各行を並列実行して、サービスに負荷を与え、サービスに負荷がかかっている状況でもユーザー・エクスペリエンスが許容できることを確認するテストを実行できます。

- 環境で影響力の大きな処理を必要とするフォームを特定します。該当する可能性が高いのは、大容量のデータを扱うフォーム、または複雑な計算を含むフォームです。たとえば、予測の発行に使用されるフォームや、アドホック・レポートまたは静的レポートの作成に関連するプロセスによって、サービスに対する高い負荷が生成されます。同様に、ビジネス・ルールの実行、レポートの実行、リソースが集中する REST API の実行、EPM 自動化コマンド(たとえば、runBusinessrule、runDataRule、exportData、exportMetadata、restructureCube)などのアクティビティは、環境に大きな負荷が生じる場合があり、負荷テストの候補になる可能性があります。
- 必要な場合には Fiddler をインストールします。EPM 自動化には、Oracle Smart View for Office、Oracle Fusion Cloud Enterprise Performance Management の REST API または EPM 自動化と Cloud EPM 環境との相互作用の記録が格納されている HTTP アーカイブ形

式(HAR) 1.1 ファイルが必要です。通常は、Fiddler を使用して、Cloud EPM との相互作用のログをキャプチャする HAR ファイルを生成します。

- 前に特定した影響力の大きいアクティビティを実行します。Smart View を使用して、フォームの起動と保存、ビジネス・ルールの実行、レポートの作成などのアクティビティを実行し、Fiddler を使用してアクティビティの詳細をキャプチャして HAR ファイルにエクスポートします。同様に、REST API および EPM 自動化コマンドを実行し、Fiddler で詳細をキャプチャします。詳細は、[HAR ファイルの作成](#)を参照してください。
- 資格証明(ユーザー名とパスワード)と実行する HAR ファイルの名前を含むリプレイ CSV ファイルを作成します。このファイルの各行には、複数の同時ユーザー・セッションをシミュレーションするために一意のユーザーのユーザー名とパスワードを含めることができます。詳細は、[リプレイ・ファイルの作成](#)を参照してください。

HAR ファイルを実行するために資格証明が行に指定されるユーザーは、HAR ファイルの作成に使用されたセッションを実行したユーザーである必要はありません。ただし、このユーザーはこれらのアクティビティを環境で実行する権限を持つ必要があります。

replay コマンドを実行するステップの詳細は、[サンプル・リプレイ・セッション](#)を参照してください。

HAR ファイルの作成

HAR ファイルは、Oracle Smart View for Office、REST API または EPM 自動化と Oracle Fusion Cloud Enterprise Performance Management との相互作用のトレースをキャプチャします。

Fiddler はすべての HTTP(S)トラフィックの情報をキャプチャするため、HAR ファイルを作成するときは、不必要なトレースを Fiddler に追加する可能性のあるアクティビティを控えてください。

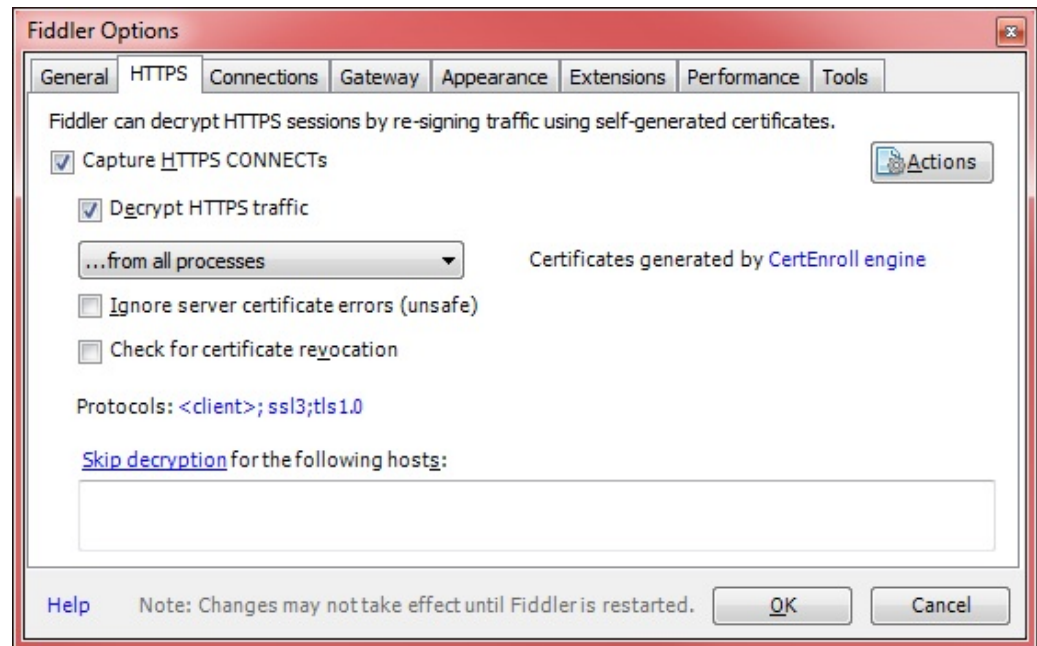
HAR ファイルを作成するには:

1. Fiddler を起動します。
2. すべてのプロセスの HTTPS トラフィックを復号化するように Fiddler が構成されていることを確認します。
 - a. 「ツール」、「オプション」、HTTPS の順に選択します。
 - b. 「Decrypt HTTPS traffic」を選択します(選択されていない場合)。

Fiddler によって、HTTPS トラフィックの捕捉に使用されるルート証明書が表示されます。通常はこの証明書を信頼することができます。

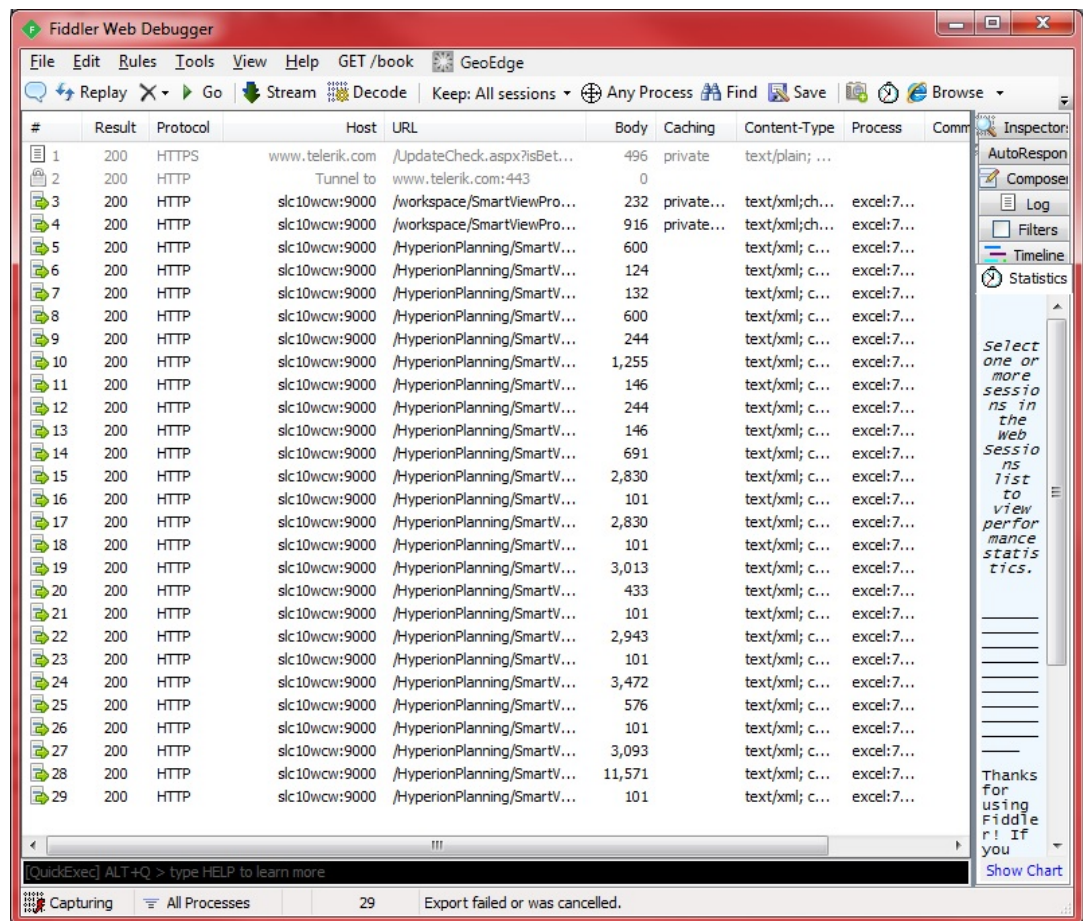


- c. 信頼された証明機関のリストにルート証明書を追加するには「Yes」をクリックします。追加しない場合は「No」を選択します。
- d. **オプション:** 前のステップで「No」を選択した場合は、「Ignore server certificate errors」を選択して、HTTPS トラフィックの復号化に関連する Fiddler のセキュリティ警告を表示しないようにすることができます。
- e. 「OK」をクリックします。



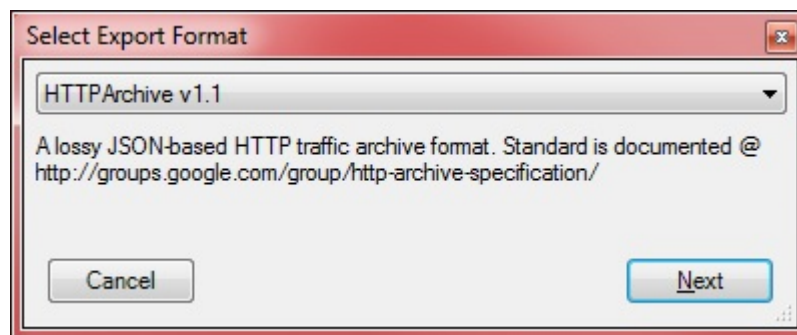
3. Smart View を開始し、トレースをキャプチャする環境にアクセスします。
4. Smart View、REST API または EPM 自動化を使用して、環境に高い処理負荷を与えるアクティビティを実行します。たとえば、Fiddler がアクティビティを記録できるように Smart View でフォームを開きます。

Fiddler は、開始されたプロセスを記録します。



5. Fiddler で次のステップを実行します。

- a. 「File」、「Export Sessions」の順に選択し、「All Sessions」または「Selected Sessions」を選択します。Fiddler の実行中に他の Web サイトに接続していた場合は、「Selected Sessions」を選択して、環境に関連するセッションを選択します。
- b. 「Select Export Format」で、エクスポート形式として「HTTPArchive v1.1」を選択します。
- c. 「Next」をクリックします。



- d. 「Export As HTTPArchive v1.1」で、ファイルを格納するディレクトリを選択し、ファイル名を指定します。

- e. 「Save」をクリックします。

リプレイ・ファイルの作成

リプレイ・ファイルは、資格証明(ユーザー名とパスワード)と HAR ファイルの名前を含む CSV ファイルです。指定された HAR ファイルが EPM 自動化の `replay` コマンドを使用して実行され、システムに負荷を与えます。

指定するユーザー名とパスワードには、HAR ファイルに含まれるアクティビティを実行する権限があることを確認します。

`replay` コマンドを実行すると、EPM 自動化はリプレイ・ファイルの各行を並列で実行して、サービスに負荷を与えます。たとえば、リプレイ・ファイルの内容が 10 行ある場合は、EPM 自動化によって 10 セッションがリプレイされます。これにより、指定の負荷がサービスにかかっている状況でもユーザー・エクスペリエンスが許容できることを確認するテストを実行できます。HAR ファイルに含まれる各アクティビティは逐次実行されます。

`replay` コマンドの実行の詳細は、[replay](#) を参照してください。

リプレイ・ファイルを作成するには:

1. Microsoft Office Excel を開いて、新しいワークシートを開始します。
2. ユーザー名、パスワード、HAR ファイルの場所を、行 1 の列 A、B、C それぞれに入力します。

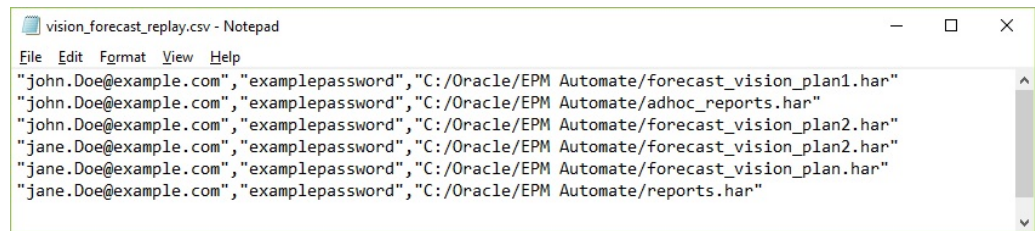
このステップを繰り返して、追加の行を作成します。

ノート:

HAR ファイルの場所として絶対パスを指定する必要があります。ファイル・パスにはディレクトリの区切りとしてスラッシュ(/)を使用します。バックスラッシュ(\)は使用しないでください。

3. ファイルを保存します
4. 「名前を付けて保存」で次のステップを実行します。
 - a. リプレイ・ファイルを格納するディレクトリを選択します。
 - b. 「ファイル名」に名前を指定し、「ファイルの種類」で「csv (カンマ区切り) (*.csv)」を選択します。
 - c. 「Save」をクリックします。

サンプルのリプレイ・ファイルを次に示します。



```
vision_forecast_replay.csv - Notepad
File Edit Format View Help
"john.Doe@example.com","examplepassword","C:/Oracle/EPM Automate/forecast_vision_plan1.har"
"john.Doe@example.com","examplepassword","C:/Oracle/EPM Automate/adhoc_reports.har"
"john.Doe@example.com","examplepassword","C:/Oracle/EPM Automate/forecast_vision_plan2.har"
"jane.Doe@example.com","examplepassword","C:/Oracle/EPM Automate/forecast_vision_plan2.har"
"jane.Doe@example.com","examplepassword","C:/Oracle/EPM Automate/forecast_vision_plan.har"
"jane.Doe@example.com","examplepassword","C:/Oracle/EPM Automate/reports.har"
```

トレース・ファイルの生成

`replay` コマンドの実行中に、Oracle サポートと共有して問題をトラブルシューティングするためのトレース・ファイルを生成できます。Oracle サポートはトレース・ファイルを使用して、環境が Oracle Smart View for Office アクティビティを処理した方法を理解します。

オプションの `trace=true` パラメータを `replay` コマンドで使用して、XML 形式のトレース・ファイルを生成します。HAR ファイルの各アクティビティでこのパラメータを使用すると、EPM 自動化は、アクティビティに対する Smart View のレスポンスを含むトレース・ファイルを作成します。

トレース・ファイルの名前は `trace-N.xml` のように付けられます。たとえば、`trace-1.xml` で、`N` は 1 から始まるカウンタです。同じ名前の複数の HAR ファイルがリプレイ・ファイルに指定された場合、EPM 自動化によってトレース・ファイルが 1 つのフォルダにまとめられます。

1 つの HAR ファイルに関連する複数のトレース・ファイルは、EPM 自動化を実行したディレクトリ内のフォルダに格納されます。EPM 自動化によって、リプレイ・ファイルに指定した HAR ファイルごとに 1 つのフォルダが作成されます。EPM 自動化では、サーバーの現在のシステム時刻と HAR ファイル名を `YYYY_MM_DD_HH_MM_SS_HAR_FILE_NAME` の形式で組み合わせて、フォルダの名前が付けられます。たとえば、HAR ファイル名が `forecast1.har` の場合、フォルダ名は `2016_06_08_10_21_42_forecast1` となることがあります。

サンプル・リプレイ・セッション

複数の HAR ファイルを使用して `replay` コマンドを実行する方法について説明します。

この項では次を前提としています。

- 次の HAR ファイルを作成しました。各 HAR ファイルには同じセットのアクティビティを含めることができます。詳細は、[HAR ファイルの作成](#)を参照してください。
 - `C:\Oracle\EPM Automate\forecast_vision_plan1.har`
 - `C:\Oracle\EPM Automate\forecast_vision_plan2.har`
 - `C:\Oracle\EPM Automate\forecast_plan2.har`
- 次の内容を含むリプレイ・ファイル `C:/Oracle/EPM Automate/vision_forecast_replay.csv` を作成しました(詳細は、[リプレイ・ファイルの作成](#)を参照)。

ノート:

`replay` ファイル内のファイル・パスにはディレクトリの区切りとしてスラッシュ (`/`)を使用します。バックスラッシュ (`\`)は使用しないでください。

```
john.doe@example.com,examplePwd,C:/Oracle/EPM Automate/  
forecast_vision_plan1.har  
john.doe@example.com,examplePwd,C:/Oracle/EPM Automate/
```

```
forecast_vision_plan2.har  
john.doe@example.com,examplePwd,C:/Oracle/EPM Automate/forecast_plan2.har
```

replay コマンドを実行するには:

1. コマンド・プロンプト・ウィンドウで、EPM 自動化がインストールされたディレクトリ、たとえば C:\Oracle\EPM Automate\bin にナビゲートします。
2. サービス管理者として環境にサインインし、replay コマンドを実行します。

```
epmautomate login john.doe@example.com examplePassword https://test-cloud-pln.pbcs.us1.oraclecloud.com myIdentityDomain
```

```
epmautomate replay "c:/Oracle/EPM Automate/vision_forecast_replay.csv"  
duration=12 lagTime=5.5 trace=true
```

EPM 自動化によって、リプレイ情報がコンソールに表示され、指定時間(この例では 12 分)後に処理が停止されます。直前のコマンドには trace=true パラメータが含まれるため、トレース・フォルダおよびトレース・ファイルも作成されます。

コマンドは C:\Oracle\EPM Automate\bin から実行されたため、EPM 自動化によってトレース・ファイルは次のフォルダに格納されました。これらのフォルダの名前は HAR ファイルの名前に基づいて決まることに注意してください。

- C:\Oracle\EPM Automate\bin\2017_01_08-12_52_37-forecast_plan2-jdoe@example.com
- C:\Oracle\EPM Automate\bin\2017_01_08-12_52_37-forecast_vision_plan1-jdoe@example.com
- C:\Oracle\EPM Automate\bin\2017_01_08-12_52_37-forecast_vision_plan2-jdoe@example.com

3. 環境からサインアウトします。

```
epmautomate logout
```

C

特殊文字の処理

Oracle Fusion Cloud Enterprise Performance Management のパスワード、プロキシ・パスワードおよびコマンド・パラメータ値には、特殊文字を使用できます。EPM 自動化でこのような文字を取り扱うには、特別な処理が必要です。

この項の例では、サンプルのパスワードを使用して、特殊文字の使用方法を説明します。

パラメータと値のペアを二重引用符で囲むことをお勧めします。

Windows

次の特殊文字は、特殊文字または特殊文字を含むパラメータ値の周囲を二重引用符(")を使用してエスケープする必要があります。

ノート:

名前に&が含まれるフォルダ(C:\Oracle\A&B など)から EPM 自動化を実行することはできません。

表 C-1 特殊文字の処理: Windows

文字	説明	エスケープの例
)	右カッコ	<ul style="list-style-type: none">Example")"pwd1 または"Example)pwd1"
<	より小さい	<ul style="list-style-type: none">Example"<"pwd1 または"Example<pwd1"
>	より大きい	<ul style="list-style-type: none">Example">"pwd1 または"Example>pwd1"
&	アンパサンド	<ul style="list-style-type: none">Example"&"pwd1 または"Example&pwd1"
	パイプ	<ul style="list-style-type: none">Example" "pwd1 または"Example pwd1"
"	引用符	<ul style="list-style-type: none">Example""pwd1 または"Example"pwd1"

Windows バッチ・ファイルのプレーン・テキスト・パスワードでの感嘆符の使用

EPM 自動化で使われる Windows バッチ・ファイルのプレーン・テキスト・パスワードでの感嘆符(!)の使用は、次のように処理する必要があります。

1. エスケープ文字として、感嘆符の前に 2 つのキャレット記号(^)を使用します。たとえば、パスワードが Welc0me! の場合、Welc0me^^! とエンコードします。

2. 次の宣言を含めることで、ファイルの冒頭に `DisableDelayedExpansion` を設定して、バッチ・ファイルを更新します。
`setlocal DisableDelayedExpansion`
3. スクリプト内に `setlocal EnableExtensions EnableDelayedExpansion` 宣言が存在する場合は、それを削除します。

UNIX/Linux

UNIX および Linux オペレーティング・システムでは、バックスラッシュ(\)を使用して特殊文字をエスケープする必要があります。

ノート:

- !(感嘆符)をエスケープするには、パスワードを一重引用符で囲むか、エスケープ文字としてバックスラッシュ(\)を使用します。
- \、\$、'、"をエスケープするには、パスワードを二重引用符で囲むか、エスケープ文字として円記号()を使用します。

表 C-2 特殊文字の処理: UNIX/Linux

文字	説明	エスケープの例
(左カッコ	Example\(pwd1
)	右カッコ	Example\)pwd1
<	より小さい	Example\ <pwd1< td=""> </pwd1<>
>	より大きい	Example\>pwd1
`	アポストロフィ	Example\'pwd1
!	感嘆符	<ul style="list-style-type: none"> • 'Example!pwd1' または • Example\!pwd1
#	ハッシュ	Example\#pwd1
&	アンパサンド	Example\&pwd1
	パイプ	Example\ pwd1
;	セミコロン	Example\;pwd1
.	ピリオド	Example\.pwd1
"	引用符	<ul style="list-style-type: none"> • Example\"pwd1 または • "Example\"pwd1"
'	一重引用符	<ul style="list-style-type: none"> • Example\'pwd1 または • "Example\'pwd1"
\$	ドル記号	<ul style="list-style-type: none"> • Example\\$pwd1 または • "Example\\$pwd1"
\	バックスラッシュ	<ul style="list-style-type: none"> • Example\\pwd1 または • "Example\\pwd1"

UNIX または Linux スクリプトのプレーン・テキスト・パスワードでの感嘆符の使用

UNIX/Linux スクリプトでは、シェル変数に格納されている EPM 自動化パスワードに特殊文字が含まれている場合は、エスケープ・シーケンスとして 3 つのバックスラッシュを使用して、文字列を二重引用符で囲みます。たとえば、lzi[AC0(e*7Qd)jE というパスワードがシェル変数 password に含まれている場合、スクリプト内では次のように記述します。

```
password="lzi[AC0\\(e*7Qd\\)jE"
```

D

各 Cloud EPM サービスに固有のコマンド

- [Account Reconciliation](#) のコマンド
- [Financial Consolidation and Close](#) のコマンド
- [Narrative Reporting](#) のコマンド
- [Oracle Enterprise Data Management Cloud](#) のコマンド
- [Planning](#)、[Planning](#) モジュール、[フリーフォーム](#)、[Strategic Workforce Planning](#) および [Sales Planning](#) のコマンド
- [Profitability and Cost Management](#) のコマンド
- [Enterprise Profitability and Cost Management](#) のコマンド
- [Tax Reporting](#) のコマンド

Account Reconciliation のコマンド

Account Reconciliation 用の EPM 自動化のコマンド

addUsers	groupAssignmentAuditReport	renameSnapshot
addUsersToGroup	help	replay
addUsersToTeam	importARApplicationProperties	resetService
addUserToGroups	importBackgroundImage	restoreBackup
archiveTmTransactions	importLogoImage	roleAssignmentAuditReport
assignRole	importBalances	roleAssignmentReport
cloneEnvironment	importDataManagement	runAutomatch
copyFileFromInstance	importMapping	runBatch
copyFromObjectStorage	importPreMappedBalances	runComplianceReport
copyFromSFTP	importPreMappedTransactions	runDailyMaintenance
copySnapshotFromInstance	importProfiles	runDataRule
copyToObjectStorage	importRates	runDMReport
copyToSFTP	importRCAttributeValues	runIntegration
createGroups	importReconciliationAttributes	runMatchingReport
createReconciliations	importSnapshot	sendMail
deleteFile	importTMAAttributeValues	setApplicationAdminMode
deleteGroups	importTmPremappedTransactions	setDailyMaintenanceStartTime
downloadFile	invalidLoginReport	setDemoDates
encrypt	listBackups	setEncryptionKey
exportAccessControl	listFiles	setIdleSessionTimeout
exportARApplicationProperties	login	setIPAllowlist
exportBackgroundImage	logout	setRestrictedDataAccess
exportDataManagement	provisionReport	setManualDataAccess
exportLogoImage	purgeArchivedTmTransactions	setPeriodStatus
exportMapping	purgeTmTransactions	setVirusScanOnFileUploads
exportSnapshot	recreate	skipUpdate
feedback	refreshCube	unassignRole
getApplicationAdminMode	removeUserFromGroups	updateUsers
getDailyMaintenanceStartTime	removeUsers	upgrade
getIdleSessionTimeout	removeUsersFromGroup	uploadFile
getIPAllowlist	removeUsersFromTeam	userAuditReport
getRestrictedDataAccess		userGroupReport
getVirusScanOnFileUploads		

Financial Consolidation and Close のコマンド

Financial Consolidation and Close 用の EPM 自動化のコマンド

addUsers	exportTaskManagerAccessControl	renameSnapshot
addUsersToGroup	exportValidIntersections	replay
addUsersToTeam	feedback	resetService
addUserToGroups	getApplicationAdminMode	restoreBackup
applicationAdminMode	getDailyMaintenanceStartTime	restructureCube
assignRole	getEssbaseQryGovExecTime	roleAssignmentAuditReport
clearDataByProfile	getIdleSessionTimeout	roleAssignmentReport
cloneEnvironment	getIPAllowlist	runBatch
copyDataByProfile	getRestrictedDataAccess	runBusinessRule
copyFileFromInstance	getSubstVar	runDailyMaintenance
copyFromObjectStorage	getVirusScanOnFileUploads	runDataRule
copyFromSFTP	groupAssignmentAuditReport	runDMReport
copyOwnershipDataToNextYear	help	runIntegration
copySnapshotFromInstance	importAppSecurity	runRuleSet
copyToObjectStorage	importConsolidationJournals	runIntercompanyMatchingReport
copyToSFTP	importData	runPipeline
createGroups	importDataManagement	runSupplementalDataReport
deleteFile	importJobConsole	runTaskManagerReport
deleteGroups	importMapping	sendMail
deployEJTemplates	importMetadata	setApplicationAdminMode
deployFormTemplates	importOwnershipData	setDailyMaintenanceStartTime
deployTaskManagerTemplate	importSnapshot	setDemoDates
downloadFile	importSupplementalCollectionData	setEJJournalStatus
essbaseBlockAnalysisReport	importSupplementalData	setEncryptionKey
executeReportBurstingDefinition	importValidIntersections	setEssbaseQryGovExecTime
exportDataManagement	invalidLoginReport	setIdleSessionTimeout
exportEssbaseData	listBackups	setIPAllowlist
encrypt	listFiles	setRestrictedDataAccess
exportAppAudit	login	setVirusScanOnFileUploads
exportAppSecurity	logout	setManualDataAccess
exportConsolidationJournals	maskData	setSubstVars
exportData	provisionReport	simulateConcurrentUsage
exportEJournals	recomputeOwnershipData	skipUpdate
exportJobConsole	recreate	snapshotCompareReport
exportLibraryDocument	refreshCube	unassignRole
exportMapping	removeUserFromGroups	updateGuidedLearningSettings
exportMetadata	removeUsers	updateUsers
exportOwnershipData	removeUsersFromGroup	upgrade
exportSnapshot	removeUsersFromTeam	uploadFile
exportTaskManagerAccessControl		userAuditReport
exportValidIntersections		

Financial Consolidation and Close 用の EPM 自動化のコマンド

exportSnapshot	userGroupReport
	validateConsolidationMetadata

Narrative Reporting のコマンド

Narrative Reporting 用の EPM 自動化のコマンド

addUsers	getIdleSessionTimeout	restoreBackup
addUsersToGroup	getIPAllowlist	roleAssignmentAuditReport
addUserToGroups	getRestrictedDataAccess	roleAssignmentReport
assignRole	getVirusScanOnFileUploads	runDailyMaintenance
cloneEnvironment	groupAssignmentAuditReport	sendMail
copyFileFromInstance	help	setDailyMaintenanceStartTime
copyFromObjectStorage	importLibraryArtifact	setEncryptionKey
copyFromSFTP	invalidLoginReport	setIdleSessionTimeout
copyToObjectStorage	listBackups	setIPAllowlist
copyToSFTP	listFiles	setManualDataAccess
createGroups	login	setRestrictedDataAccess
createNRSnapshot	logout	setVirusScanOnFileUploads
deleteFile	provisionReport	skipUpdate
deleteGroups	recreate	unassignRole
downloadFile	removeUserFromGroups	updateUsers
encrypt	removeUsers	upgrade
executeBurstDefinition	removeUsersFromGroup	uploadFile
exportLibraryArtifact	replay	userAuditReport
feedback	resetService	userGroupReport
getDailyMaintenanceStartTime		

Oracle Enterprise Data Management Cloud のコマンド

Oracle Fusion Cloud Enterprise Data Management の EPM 自動化コマンド

addUsers	getIdleSessionTimeout	replay
addUsersToGroup	getIPAllowlist	resetService
addUserToGroups	getRestrictedDataAccess	restoreBackup
assignRole	getVirusScanOnFileUploads	roleAssignmentAuditReport
cloneEnvironment	groupAssignmentAuditReport	roleAssignmentReport
copyFileFromInstance	help	runDailyMaintenance
copyFromObjectStorage	importDimension	sendMail
copyFromSFTP	importSnapshot	setDailyMaintenanceStartTime
copySnapshotFromInstance	invalidLoginReport	setEncryptionKey
copyToObjectStorage	listBackups	setIdleSessionTimeout
copyToSFTP	listFiles	setIPAllowlist
createGroups	loadDimensionViewpoint	setRestrictedDataAccess
deleteFile	loadViewpoint	setManualDataAccess
deleteGroups	login	setVirusScanOnFileUploads
downloadFile	logout	skipUpdate
encrypt	provisionReport	unassignRole
exportDimension	recreate	updateUsers
exportDimensionMapping	removeUserFromGroups	upgrade
exportSnapshot	removeUsers	uploadFile
extractDimension	removeUsersFromGroup	userAuditReport
extractPackage	renameSnapshot	userGroupReport
feedback		
getDailyMaintenanceStartTime		

Planning、Planning モジュール、フリーフォーム、Strategic Workforce Planning および Sales Planning のコマンド

Planning、Planning モジュール、フリーフォーム、Strategic Workforce Planning および Sales Planning の EPM 自動化コマンド

addUsers	getApplicationAdminMode	replay
addUsersToGroup	getDailyMaintenanceStartTime	resetService
addUserToGroups	getEssbaseQryGovExecTime	restoreBackup
applicationAdminMode	getIdleSessionTimeout	restructureCube
assignRole	getIPAllowlist	roleAssignmentAuditReport
autoPredict * 脚注を参照	getRestrictedDataAccess	roleAssignmentReport
clearCube	getSubstVar	runBatch
cloneEnvironment	getVirusScanOnFileUploads	runBusinessRule
compactCube	groupAssignmentAuditReport	runDailyMaintenance
copyFileFromInstance	help	runDataRule
copyFromObjectStorage	importAppAudit	runDMReport
copyFromSFTP	importAppSecurity	runIntegration
copySnapshotFromInstance	importCellLevelSecurity	runPipeline
copyToObjectStorage	importData	runPlanTypeMap
copyToSFTP	importDataManagement	runRuleSet
createGroups	importJobConsole	sendMail
deleteFile	importMapping	setApplicationAdminMode
deleteGroups	importMetadata	setDailyMaintenanceStartTime
dismissIPMInsights**	importSnapshot	setDemoDates
downloadFile	importValidIntersections	setEncryptionKey
enableQueryTracking	invalidLoginReport	setEssbaseQryGovExecTime
encrypt	listBackups	setIdleSessionTimeout
essbaseBlockAnalysisReport	listFiles	setIPAllowlist
executeAggregationProcess	login	setManualDataAccess
executeReportBurstingDefinition	logout	setRestrictedDataAccess
exportAppAudit	maskData	setSubstVars
exportAppSecurity	mergeDataSlices	setVirusScanOnFileUploads
exportCellLevelSecurity	provisionReport	simulateConcurrentUsage
exportData	recreate	skipUpdate
exportDataManagement	refreshCube	snapshotCompareReport
exportEssbaseData	removeUserFromGroups	sortMember
exportJobConsole	removeUsers	unassignRole
exportLibraryDocument	removeUsersFromGroup	updateGuidedLearningSettings
exportMapping	renameSnapshot	updateUsers
exportMetadata		upgrade
exportSnapshot		uploadFile
exportValidIntersections		userAuditReport
feedback		userGroupReport

Planning、Planning モジュール、フリーフォーム、Strategic Workforce Planning および Sales Planning の EPM 自動化コマンド

*このコマンドは、フリーフォーム、Strategic Workforce Planning および Sales Planning ではサポートされていません。

**このコマンドは、フリーフォームではサポートされていません。

Profitability and Cost Management のコマンド

Profitability and Cost Management 用の EPM 自動化のコマンド

addUsers	getEssbaseQryGovExecTime	renameSnapshot
addUsersToGroup	getIdleSessionTimeout	replay
addUserToGroups	getIPAllowlist	resetService
applyDataGrants	getRestrictedDataAccess	restoreBackup
assignRole	getVirusScanOnFileUploads	roleAssignmentAuditReport
clearPOV	groupAssignmentAuditReport	roleAssignmentReport
cloneEnvironment	help	runBatch
copyFileFromInstance	importDataManagement	runCalc
copyFromObjectStorage	importMapping	runDailyMaintenance
copyFromSFTP	importSnapshot	runDataRule
copyPOV	importTemplate	runDMReport
copySnapshotFromInstance	invalidLoginReport	runIntegration
copyToObjectStorage	listBackups	sendMail
copyToSFTP	listFiles	setDailyMaintenanceStartTime
createGroups	loadData	setEncryptionKey
deleteFile	loadDimData	setEssbaseQryGovExecTime
deleteGroups	login	setIdleSessionTimeout
deletePOV	logout	setIPAllowlist
deployCube	mergeSlices	setRestrictedDataAccess
downloadFile	optimizeASOCube	setManualDataAccess
enableApp	programDocumentationReport	skipUpdate
encrypt	provisionReport	setVirusScanOnFileUploads
exportDataManagement	recreate	unassignRole
exportMapping	removeUserFromGroups	updateUsers
exportQueryResults	removeUsers	upgrade
exportSnapshot	removeUsersFromGroup	uploadFile
exportTemplate		userAuditReport
feedback		userGroupReport
getDailyMaintenanceStartTime		

Enterprise Profitability and Cost Management のコマンド

Enterprise Profitability and Cost Management の EPM 自動化コマンド

addUsers	exportTaskManagerAccessControl	removeUsers
addUsersToGroup	exportValidIntersections	removeUsersFromGroup
addUserToGroups	feedback	renameSnapshot
applicationAdminMode	getApplicationAdminMode	replay
assignRole	getDailyMaintenanceStartTime	resetService
calculateModel	getEssbaseQryGovExecTime	restoreBackup
clearCube	getIdleSessionTimeout	roleAssignmentAuditReport
compactCube	getIPAllowlist	roleAssignmentReport
copyDataByPointOfView	getRestrictedDataAccess	runBatch
cloneEnvironment	getSubstVar	runBusinessRule
copyFileFromInstance	getVirusScanOnFileUploads	runDailyMaintenance
copyFromObjectStorage	groupAssignmentAuditReport	runDataRule
copyFromSFTP	help	runDMReport
clearDataByPointOfView	importAppAudit	runIntegration
copySnapshotFromInstance	importAppSecurity	runPipeline
copyToObjectStorage	importCellLevelSecurity	sendMail
copyToSFTP	importData	runRuleSet
createGroups	importDataManagement	runTaskManagerReport
deleteFile	importJobConsole	setApplicationAdminMode
deleteGroups	importMapping	setDailyMaintenanceStartTime
downloadFile	importMetadata	setDemoDates
deletePointOfView	importSnapshot	setEncryptionKey
deployTaskManagerTemplate	importValidIntersections	setEssbaseQryGovExecTime
enableQueryTracking	invalidLoginReport	setIdleSessionTimeout
encrypt	listBackups	setIPAllowlist
executeAggregationProcess	listFiles	setManualDataAccess
executeReportBurstingDefinition	login	setRestrictedDataAccess
exportAppAudit	logout	setSubstVars
exportAppSecurity	maskData	setVirusScanOnFileUploads
exportCellLevelSecurity	mergeDataSlices	skipUpdate
exportData	provisionReport	snapshotCompareReport
exportDataManagement	recreate	sortMember
exportEssbaseData	refreshCube	unassignRole
exportJobConsole	removeUserFromGroups	updateGuidedLearningSettings
exportLibraryDocument		updateUsers
exportMapping		upgrade
exportMetadata		uploadFile
exportMetadata		userAuditReport
exportSnapshot		userGroupReport
		validateModel

Tax Reporting のコマンド

Tax Reporting 用の EPM 自動化のコマンド

addUsers	getDailyMaintenanceStartTime	replay
addUsersToGroup	getEssbaseQryGovExecTime	resetService
addUsersToTeam	getIdleSessionTimeout	restoreBackup
addUserToGroups	getIPAllowlist	restructureCube
applicationAdminMode	getRestrictedDataAccess	roleAssignmentAuditReport
assignRole	getSubstVar	roleAssignmentReport
clearDataByProfile	getVirusScanOnFileUploads	runBatch
copyDataByProfile	groupAssignmentAuditReport	runBusinessRule
copyFileFromInstance	help	runDailyMaintenance
copyFromObjectStorage	importAppSecurity	runDataRule
copyFromSFTP	importCellLevelSecurity	runDMReport
copyOwnershipDataToNextYear	importData	runIntegration
copySnapshotFromInstance	importDataManagement	runPipeline
copyToObjectStorage	importJobConsole	runRuleSet
copyToSFTP	importMapping	runSupplementalDataReport
createGroups	importMetadata	runTaskManagerReport
deleteFile	importOwnershipData	sendMail
deleteGroups	importSnapshot	setApplicationAdminMode
deployFormTemplates	importSupplementalCollectionData	setDailyMaintenanceStartTime
deployTaskManagerTemplate	importSupplementalData	setDemoDates
downloadFile	importValidIntersections	setEncryptionKey
encrypt	invalidLoginReport	setEssbaseQryGovExecTime
essbaseBlockAnalysisReport	listBackups	setIdleSessionTimeout
executeReportBurstingDefinition	listFiles	setIPAllowlist
exportAppAudit	login	setManualDataAccess
exportCellLevelSecurity	logout	setRestrictedDataAccess
exportData	maskData	setSubstVars
exportDataManagement	provisionReport	setVirusScanOnFileUploads
exportEssbaseData	recomputeOwnershipData	simulateConcurrentUsage
exportJobConsole	recreate	skipUpdate
exportLibraryDocument	refreshCube	snapshotCompareReport
exportMapping	removeUserFromGroups	unassignRole
exportMetadata	removeUsers	updateGuidedLearningSettings
exportOwnershipData	removeUsersFromGroup	upgrade
exportSnapshot	removeUsersFromTeam	updateUsers
exportTaskManagerAccessControl	renameSnapshot	uploadFile
exportValidIntersections		userAuditReport
feedback		userGroupReport
getApplicationAdminMode		