Oracle Financial Services Revenue Management and Billing Cloud Service, Premium Edition

OR

Oracle Insurance Revenue Management and Billing Cloud Service, Premium Edition

Version 5.1.0.0.0

Implementation Guide

Revision 1.2

F82320-03

June 2023

ORACLE

Oracle Financial Services Revenue Management and Billing Cloud Service, Premium Edition/Oracle Insurance Revenue Management and Billing Cloud Service, Premium Edition Version 5.1.0.0.0 Implementation Guide

Note: To improve the content readability, the above two products are collectively referred to as Oracle Revenue Management and Billing Cloud Service, Premium Edition throughout this document.

F82320-03

Hazardous Applications Notice

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.

Trademark Notice

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.

Third-Party Content, Products, and Services Disclaimer

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.

Preface

About This Document

This document provides information about implementation of Oracle Revenue Management and Billing Cloud Service.

Intended Audience

This document is intended for the following audience:

- End-Users
- System Administrators
- Consulting Team
- Implementation Team

Organization of the Document

The information in this document is organized into the following sections:

Section No.	Section Name	Description
Section 1	Implementation Guidelines	Describes the global implementation guidelines that apply to Oracle Revenue Management and Billing Cloud Service running on Oracle Cloud Infrastructure (OCI).
Section 2	Data Conversion and Migration	Describes how the data conversion and migration can happen in case of Oracle Revenue Management and Billing Cloud Service.
Section 3	File-Based Integration	Describes how the implementations can integrate and exchange information from Oracle Revenue Management and Billing Cloud Service to other applications and vice versa through file-based integration.
Section 4	Oracle REST Data Services	Explains how to use the Oracle REST Data Services with Oracle Revenue Management and Billing Cloud Service.
Section 5	Web Services	Explains how to use web services with Oracle Revenue Management and Billing Cloud Service.

Conventions

The following conventions are used across this document:

Convention	Meaning
boldface	Boldface indicates graphical user interface elements associated with an action, or terms defined in the text.
italic	Italic indicates a document or book title.
`monospace`	Monospace indicates commands within a paragraph, URLs, code in examples, text that appears on the screen or entered in the application.

Acronyms

The following acronyms are used in this document:

Acronym	Meaning
CSF	Cloud Service Foundation
DevOps	Development Operations
ETL	Extract, Transform, and Load
FOP	Formatting Objects Processor
HTTP	Hypertext Transfer Protocol
IAM	Oracle Identity and Access Management
IWS	Inbound Web Services
JMS	Java Message Service
MO	Maintenance Object
OCI	Oracle Cloud Infrastructure
OCID	Oracle Cloud ID
ORDS	Oracle REST Data Services
ORMB	Oracle Revenue Management and Billing
ORMBCS	Oracle Revenue Management and Billing Cloud Service
OUAF	Oracle Utilities Application Framework
REST	Representational State Transfer
RTJSONSNDR	Sender for real-time JSON messages
SA	Service Agreement
SaaS	Software-as-a-Service

Acronym	Meaning
SOAP	Simple Object Access Protocol
SOAPSNDR	Sender for real-time HTTP/SOAP messages
SP	Service Point
US	Usage Subscription
WSDL	Web Service Definition Language
XML	Extensible Markup Language
XSL	Extensible Stylesheet Language

Related Documents

You can see the following documents for more information:

Document Name	Description
Oracle Revenue Management and Billing Cloud Service, Premium Edition Administration Guide	Explains how to manage the user accounts and their access for Oracle Revenue Management and Billing Cloud Services (ORMBCS) using Identity and Access Management with or without identity domains on Oracle Cloud Infrastructure (OCI).
Oracle Revenue Management and Billing Cloud Service, Premium Edition Frequently Asked Questions Guide	Lists various frequently asked questions (FAQs) regarding the implementation and operations of Oracle Revenue Management and Billing Cloud Services.
Oracle Revenue Management and Billing Cloud Service, Premium Edition Operations Guide	Provides information regarding different types of service requests (SRs) customers can submit to the Oracle Revenue Management and Billing Cloud Operations team during implementation and operations of the Oracle Revenue Management and Billing Cloud Services.
Oracle Revenue Management and Billing Cloud Service, Premium Edition Live Operations Guide	Provides guidelines regarding live operations of Oracle Revenue Management and Billing Cloud Services.
Oracle Revenue Management and Billing Chatbot Configuration Guide	Explains how to integrate Oracle Digital Assistant (ODA) with the ORMB Cloud Service.
Oracle Revenue Management and Billing Chatbot User Guide	Explains how to use the menu based Chatbot introduced in the ORMB Cloud Service.
Oracle Revenue Management and Billing ML Integration Guide	Explains how to integrate Machine Learning (ML) with the ORMB Cloud Service for anomaly detection.

Access to Oracle Support

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

Change Log

Revision	Last Update	Updated Section	Comments
1.1	01-Aug-2023	Cover Page	Updated Information
		Copyright Notice	Updated Information
		Preface	Added Information
		Section 1: Implementation Guidelines	Updated Information
		Section 1.4.1: Data Conversion Approach	Updated Information
		Section 2.3.2: Preparing Legacy Data Extract for Upload	Updated Information
1.2	O8-Oct-2024	Preface	Updated Information

Contents

1. Implementation Guidelines

This section describes global implementation guidelines that apply to Oracle Revenue Management and Billing Cloud Services running on Oracle Cloud Infrastructure (OCI).

Note that these cloud services are based on the Oracle Utilities Application Framework (OUAF), which supports many different configuration and extension methods, which is available for use in the cloud. This section provides recommendations for many aspects of set-up and operation of the service. Note that it assumes familiarity with OUAF concepts and tools.

In a nutshell, the top cloud service implementation rules to be aware of are the following:

- Use Groovy code in Scripts (not Java)
- Use existing data structures to extend the base model - such as Characteristics and the Fact table
- Use plug-in driven batch can be used in many scenarios for data fixes - this will ensure proper data validation

The guidelines are intended to help implementers to configure and run their cloud services efficiently. This section contains the following topics:

1.1 Post-Provisioning Setup

When a customer of Oracle Revenue Management and Billing Cloud Service receives notification that their cloud service was provisioned, there are number of tasks that must be performed before they can start with normal implementation activities.

This section provides implementation guidelines related to post-provisioning setup. It contains the following topics:

1.1.1 Initial Identity Management Setup

Initial set up related to Identity Management includes the following:

- Create a password for the cloud administrator and log in into each of the provisioned environments.
- Create a password for a special pre-defined user for the Process Automation Tool (user: K1IPROCESS).
- Define the initial set of user groups and assign appropriate access rights to each.
- Create Oracle Cloud Infrastructure Identity and Access Management (OCI IAM) Identity Domains user mapping configuration in each of the cloud service environments (optional).

See the **Identity and Access Management with Identity Domains** section in the *Oracle Revenue Management and Billing Cloud Service, Premium Edition Administration Guide* for more information.

1.1.2 Initial Object Storage Setup

Initial set up related to Object Storage includes the following:

- Create a password for the cloud administrator in the Oracle Cloud Infrastructure (OCI) account (that includes object storage) and login into the OCI console.
- Create the default structure in object storage for the cloud service:
    - Create default Users, Groups and Policies
    - Create default Compartments and Buckets
- Configure the cloud service connections to object storage:
    - Create key rings in each cloud service environment
    - Generate API keys in each cloud service environment
    - Create File Storage extendable lookup values in each environment for each object storage compartment (with necessary details for object storage)
    - Register the API keys in the appropriate OCI users for each of the cloud service environments

See the **Object Storage Setup** section in the *Oracle Revenue Management and Billing Cloud Service, Premium Edition Administration Guide* for more information.

1.1.3 Initial Cloud Service Setup

Initial set up for general cloud service includes the following:

- Perform the process automation tool setup in each of the provisioned environments (see details below).
- Setup the security definitions for process automation in each of the environments. This is done by executing the Process Automation Security Setup BPA script and providing the following input:
    - Login ID of the Process Automation User: K1IPROCESS
    - Password of that user in IAM Identity Domains
- Perform a Flush-All on the cloud service environments.

1.1.3.1 Process Automation Tool Setup (for a new cloud service)

To set up the Process Automation Tool for all your provisioned environments (for example, Test and Prod), run the following batch job in each environment:

- **Batch Code**: K1-IPAIS
- **User ID**: SYS_INT
- **Parameters**:
  - **SCRIPT01**: K1InvokePAS
  - **SCRIPT01_DATA**: INIT,<Internal Service Code>,<Current Env Code>,file-storage://OS-SHARED/CMA-Files,file-storage://OSSHARED/CMA-Files,<Env List>

    where:
    - <Internal Service Code> = RMB (See the **Short Code** in the Process Automation Product extendable lookup for more details)
    - <Current Env Code> = DEV01..10/TEST/TEST01..10/PROD
    - <Env List> = environment codes separated by comma without spaces, for example: TEST,PROD

For example, when the job runs in the PROD environment for Oracle Revenue Management and Billing Cloud Service (ORMBCS), the parameters would be as follows:

SCRIPT01_DATA: INIT,RMB,PROD,file-storage://OS-SHARED/CMA-Files,file-storage://OS-SHARED/CMA-Files,DEV,TEST,PROD

To finalize the process automation tools setup, run the process automation security setup in each environment. Users must flush their browser cache after this security set up.

1.1.3.2 Process Automation Tool Setup (after adding new environments to your cloud service)

If you add new environments to your cloud service, you must run a similar batch job in each environment.

For NEW environments, run the same job as after the initial provisioning (see above).

For EXISTING environments (that existed before the addition of the new environments), run the following:

- **Batch Code**: K1-IPAIS
- **User ID**: SYS_INT
- **Parameters**:
  - **SCRIPT01**: K1InvokePAS
  - **SCRIPT01_DATA**: ADD,,<Current Env Code>,,,<Added Env List - for the current environment>

    where:
    - <Current Env Code> = DEV01..10/TEST/TEST01..10/PROD

- `<Added Env List> =` environment codes separated by comma without spaces, for example: `TEST,PROD`

For example, if you added DEV01 and TEST01 environments, when the job runs in the PROD environment, the parameters should be as follows:

```
SCRIPT01_DATA: ADD,,PROD,,,DEV01,TEST01
```

To finalize the process automation tools setup, run the process automation security setup in each environment. Users must flush their browser cache after this security set up.

1.1.4 Language Pack Setup (Optional)

English (with the locale en-US) is provided as the default language by the system and all system metadata is delivered with English descriptions and labels. The system provides support for defining other languages and supports multiple languages in a single environment.

System users can use the system in their preferred language if a translation into that language has been provided. A user sees the system in the language defined on their user record. If enabled, users can use the **Switch Language** zone to switch to another supported language real time.

1.1.4.1 Available Languages with Cloud Services

The following table lists languages available with Oracle Revenue Management and Billing Cloud Service:

Product	Version	Language
Oracle Revenue Management and Billing Cloud Service	5.1.0.0.0	English

1.1.4.2 Setup Instructions

Use the following procedure to set up a language in Oracle Revenue Management and Billing Cloud Service:

1. Define the Language Code for the language to be added and indicate that it is enabled. For details on this procedure, see the **Defining Languages** section in the *Oracle Utilities Application Framework Administrative Guide*.

Note: Please use 'ARA' or 'ESA' as the Language Code for Arabic and for Latin American Spanish, respectively, when defining language codes for those languages.

2. Confirm that the desired language code is listed in the Available Languages with Cloud Services section above.
3. Run the "F1-LANG" batch control.
    - This process copies descriptions of all language-enabled tables from an existing translation (e.g., English). The copied values act as placeholders while the strings are translated into the new language. It is necessary to do this as a first step to create records using the new language code created in the previous step.
    - The batch process also updates the new language rows with the translated metadata descriptions from the language pack, if installed.

Note: The language pack updates all language entries for base owned system data. If your implementation updates base owned labels and descriptions after applying a language pack, they will be overwritten the next time an updated language pack is applied. Note that most user facing labels and messages support defining an Override Label or Override Description. This information is not updated by the base product and should be utilized if your implementation desires a specific label or description.

1.2 Security and Access

This section provides implementation guidelines related to security and access. It contains the following topics:

- Identity Management
- Server Access

1.2.1 Identity Management

This section contains the following topics:

- Use of Identity Domains in OCI Identity and Access Management
- User Provisioning with Identity and Access Management

1.2.1.1 Use of Identity Domains in OCI Identity and Access Management

In Oracle Cloud Infrastructure, cloud services are provisioned using Oracle Cloud Infrastructure Identity and Access Management (OCI IAM) Identity Domains to manage user creation, application access, passwords, etc. This service at the 'Oracle Apps' tier is included with the Oracle Revenue Management and Billing Cloud Service subscription. See Identity and Access Management with Identity Domains for more information on IAM.

By default, Oracle Cloud Infrastructure Identity and Access Management allow access to the application front-end from any IP address. There are capabilities in IAM Identity Domains to add sign-on policies that allow or deny IP addresses using allowlists (though some features may require higher tier licensing).

1.2.1.2 User Provisioning with Identity and Access Management

Application users are added through Oracle Cloud Infrastructure Identity and Access Management (IAM) which are used to manage the user lifecycle (i.e., you can disable a user, or reset a user's password in IAM). The access rights of the user within the application are controlled using the settings on the cloud service User record. Identity and Access Management uses Application Roles and Groups: a user must be linked to the Application Roles that they need access to. This linking can also be 'indirect' by linking a new user to a Group which has access. Creation of cloud service User records is done 'just-in-time' - upon the first login to the application, after authentication via Identity and Access Management, a call is made to verify access to the application and using the returned information including the user's IAM Groups, a template user in the cloud service can be found and used as the 'copy from' source.

Instructions: The security administrator should create an initial User record with full access to the cloud service (including administration functionality). This user should be used to configure "Template Users" and mappings to or IAM Groups. See the **Identity and Access Management with Identity Domains** section in the *Oracle Revenue Management and Billing Cloud Service, Premium Edition Administration Guide* for more information. Note that the Cloud Service Foundation also provides several Template Users that have necessary access for process automation.

1.2.2 Server Access

While server access is restricted exclusively to members of the Oracle Cloud Infrastructure, and Oracle Revenue Management and Billing Development Operations (DevOps) teams, logs are available to users.

1.3 Configuration Tools

This section describes specific implementation guidelines related to use of Oracle Utilities Application Framework Configuration Tools. It contains the following topics:

- Customization Tools Summary
- Algorithm Types and Algorithms
- Application Environments
- Creating Batch Processes

1.3.1 Customization Tools Summary

While most of each cloud service application's customization options are supported, some are not, and others may be limited in certain areas. The table below outlines configuration options and their availability when implementing Oracle Revenue Management and Billing Cloud Service.

Category	Option	Supported?	Comment
Business Entities	Add custom business objects for product maintenance objects.	Yes	Assuming the Maintenance Object supports Business Object functionality.
	Extend a product business object's structure and rules.	Yes	See the Algorithm Types and Algorithms section for more information.
	Add custom maintenance objects.	No	Creation of new tables is not supported. See the Database Access section for more information. Use of the SDK tool to generate Java artifacts is not supported. See the Algorithm Types and Algorithms section for more information.
User Interface	Add a custom portal.	Yes	Restricted to a single tab page in 20C and previous versions. Additional tab pages are supported as of release 21A.
	Extend a product portal with custom zones.	Yes	-

Category	Option	Supported?	Comment
	Extend a product multi-query search with custom query options.	Yes	-
	Customize a product menu. This includes adding new custom menu lines, hiding, and reordering lines.	Yes	-
	Add custom indexes to support custom queries	No	See the Database Access section for more information.
Batch processes	Add a custom batch process.	Yes	The program cannot be written in Java. See the Creating Batch Processes section for more information. The process may only access designated locations in Object Storage. See the File Access - Cloud Object Storage section for more information.
Web Services	Add custom inbound and outbound web services.	Yes	Use Outbound Messaging and Inbound Web Services. The services cannot rely on XSL transformations to occur in the cloud. See the File Access - Cloud Object Storage section for more information.
Reports	Add stored procedures to support custom reports.	No	See the Database Access section for more information.
	Run the high-volume extract reports like Bill Print via Analytics Publisher	No	See the *Oracle Revenue Management and Billing Cloud Service, Premium Edition Frequently Asked Questions Guide* for more information.

1.3.2 Algorithm Types and Algorithms

New algorithm types and algorithms can be created during implementation using Scripts. Custom Java-based algorithm types are NOT permitted.

Write custom algorithm types using either Groovy or Oracle Utilities Application Framework's XML-based scripting. See the **Defining Algorithms, Plug-In Scripts, and Using Groovy within Scripts** sections in the *Oracle Utilities Application Framework Administrative Guide* or online help for information about creating algorithms using Groovy.

Key Guidelines of Groovy scripting are:

- Review the third party groovy allowlist (available within the application)
- Be careful with goto statements - it is easy to create endless loops
- Review SQL Function Allowlist (See F1-SQLFunctionWhiteList Managed Content)
- Update/Delete SQL Statements are not allowed
- Explain Plan of the query needs to be examined for all SQLs written in custom code

When crafting custom SQL queries, you must consider performance. Run explain on all your SQLs using rule hint before delivering code range scans and nested loops only. Plans with 'table access' are not acceptable.

1.3.3 Application Environments

Each cloud service by default comes with two environments designated as Test, and Production. Test and Production are sized as full-sized environments based on the billable metric of the subscription. Customers can request additional non-production environments through the initial sales order or in a subsequent order for an additional subscription. When asking for more environments, the names of the base and additional environment are predefined and cannot be changed.

1.3.3.1 Environment Names and Codes

Cloud service environments have an environment code and name. The environment code is used to identify the environment and enable migration processes (such as configuration migrations) between the environments. The environment code is also used at installation/provisioning time. The table below lists all the possible environments that can be provisioned for each cloud service.

Environment Code	Name	Type	Default	Additional
TEST	Test	Test	Yes	No
PROD	Production	Production	Yes	No
DEV01..DEV10	Development 1 .. 10	Development	No	Yes
TEST01..TEST10	Test 1 .. 10	Test	No	Yes

1.3.3.2 Application/Environment Access and URL Tokens

When environment provisioning is complete, customers/implementers will receive a list of links to the various product environments included with their subscription. There are cases in which cloud service applications need to access other cloud service or on-premises applications or other environments, such as:

- Data/Configuration Migration
- Data Conversion
- Redirecting a user to another application as part of a business process transaction that is integrated across products
- Invoking web services of a different application (another cloud service or SOA for existing SOA-supported cloud integrations)
- Invoking a web service of the same application in a different environment. This type of communication is used to help automate inter-environment processes like configuration migrations.

The following URL Tokens are available for direct navigation or web service calls for each cloud service:

Token	Description
EXT_PUB	Prefix token that should be used to reference external addresses in Message Senders. For example, to reference paymentcorp.payusa.com endpoint URL you would need to use the following notation: @EXT_PUB@paymentcorp.payusa.com. **Note:** The target URL must be on the allowlist.
DEV_WS, DEV01_WS- DEV10_WS TEST_WS, TEST01_WSTEST10_WS PROD_WS	Web service addresses for all possible environments. For example, DEV_WS can point to Oracle Revenue Management and Billing Cloud Service in the DEV domain while PROD_WS will point to the same application but in the PROD domain.

Usage Examples:

- Internal-facing tokens such as DEV_WS can be used for inter-domain communications, such as the automation of configuration migration between product domains. These tokens are used by the Process Automation Tool within Cloud Service Foundation.
- External facing addresses will use the @EXT_PUB@ prefix. For example: @EXT_PUB@paymentcorp.payusa.com/api/int01/addPayment.
- The port is not required in the URL definition when using EXT_PUB as all outbound calls from cloud services are sent via Https and port 443 is implied.

1.3.4 Creating Batch Processes

Custom batch jobs can be written using the plug-in driven batch job functionality supported by the Oracle Utilities Application Framework. There are three broad categories of batch jobs that may be implemented using plug-in driven batch.

- **Ad-hoc Processing**: This covers any batch job that should select records in the system and perform some type of logic for each record.
    - The system provides a Select Records plug-in for retrieving the records in the system based on criteria. This plug-in requires the selection SQL (properly tuned) to be defined as a parameter. Logic in the plug-in script may be used to set filter criteria if needed. The plug-in script may be written using XPath scripting.
    - The system also provides a Process Record plug-in where each record may be reviewed, and some appropriate action may be performed. This plug-in may be written in either XPath or Groovy scripting.
- **Extract a Batch of Records**. This covers any batch job that produces an extract of records. The same plug-in spots described for Ad-hoc processing are applicable here. The Select Records plug-in is used for selecting the records eligible for extraction, for example from as staging table. The Process Record plug-in us responsible for returning the data to be written to the extract for each record. This plug-in may be written in either XPath or Groovy scripting. However, because the output of the plug-in is one or more schema objects to include in the extract, XPath scripting may be better suited.
- **Upload Records from a File**. This covers any batch job that needs to read a file and create records in the system based on the content. The system provides a File Upload plug-in spot. This plug-in is responsible for calling appropriate APIs to read the content of the file and store the data in appropriate tables, for example a staging table. This type of plug-in must use Groovy as the APIs are not accessible using the XPath scripting language.

1.4 Data Conversion Guidelines

This section provides general guidelines related to data conversion. Data conversion refers to the migration of data from a client's legacy system (on-premise or cloud) to the application database within Oracle Revenue Management and Billing Cloud Service. Since no direct access is permitted to the application database, data conversion support is provided to facilitate SQL Loader-based data upload via Cloud Service Foundation tools. Staging tables cleanup is also supported.

See the Data Conversion and Migration section for more detailed information about data conversion and migration.

1.4.1 Data Conversion Approach

The implementation project is expected to extract the legacy data into flat files and upload these files to a specific location on the cloud, and then to run a sequence of batch processes that moves the data into corresponding tables in the special Staging database schema. The subsequent processing of the staging data, along with its insertion into production tables, is specific for each cloud service. See the Data Conversion and Migration section for more information about specific data conversion and migration scenarios.

It is recommended to start with small set of data covering most of the unique / critical scenarios, perform the object / FK validations and try to resolve the data quality conversion issues by comprehensive testing (both online and batches). Gradually increase the data volume to avoid running full scale of converted data early resulting in application errors (invalid data will often lead to a lot of 'noisy errors' that can be avoided with this approach).

1.4.1.1 Data Conversion Tips

The following high-level tips are important for data conversion efforts:

1. **Data Upload Indexes and Constraints** - Data conversion is performed by processing legacy data extract files using SQL Loader. During the data upload, the indexes and constraints are disabled, and duplicate keys are not validated. See Oracle SQL Loader Documentation for details.

   Please ensure that you cleanse the data extract file and remove duplicates prior to the upload.

2. **Key Tables in Staging Area** - The Key tables in the staging area are not populated automatically. The Key Table data must be created with the corresponding Environment ID and then uploaded as a separate extract. See the Data Conversion and Migration section for more information.
3. **CLOB Data Upload with Secondary Files** - The CLOB data upload with secondary files is not supported when there are multiple CLOB columns in the table. Configure the conversion task type to include CLOB data in the main extract, amend Conversion Master Configuration, and regenerate Conversion Artifacts. See the Data Conversion and Migration section for more information.

1.5 Integration Guidelines

This section provides guidelines related to integration with Oracle Revenue Management and Billing Cloud Service. It contains the following topics:

- Integration Methods
- Allowlisting

1.5.1 Integration Methods

The primary integration methods supported with cloud services are (a) inbound and outbound files, and (b) inbound and outbound web services. Other protocols and methods (JMS, SQL Net, etc.) are not currently supported.

Besides standard Oracle Revenue Management and Billing integration modules, no additional extract, transform, and load (ETL) capabilities or middleware are provided with cloud service offerings. Oracle Cloud middleware solutions—such as SOA Cloud Service (available via Platform-as-a-Service) or Integration Cloud Service —need to be licensed to address advanced integration requirements such as complex ETL, orchestration, etc. Alternatively, an on-premise middleware solution could be used.

1.5.1.1 Integration Method: File-Based

Inbound File Processing: Files are uploaded to Object Storage and processed via scheduled batch jobs. Implementation-specific file parsing and processing logic can be introduced using browser-based Oracle Revenue Management and Billing tools. See the **Uploading Records** section of Plug-in Driven Background Processes in the *Oracle Utilities Application Framework Administrative Guide* or the online help for more information. Please also review the File Access - Cloud Object Storage section in Data Access and Analytics.

Outbound File Processing: File-based extracts can be generated and made available for download and further processing. Implementation-specific file processing and generation logic can be introduced using browser-based Oracle Revenue Management and Billing tools. See the **Processing System Records** section of Plug-in Driven Background Processes in the *Oracle Utilities Application Framework Administrative Guide* or the online help for more information.

Large-volume data conversion and loading is supported. See the Data Conversion Guidelines section for more information.

1.5.1.2 Integration Method: Web Services

Web services are supported through Inbound Web Services (IWS) and Outbound Messages. All inbound and outbound web services communication must be HTTPS. See the **Inbound Web Services** and **Outbound Messages** sections in the *Oracle Utilities Application Framework Administrative Guide* or the online help for more information.

Note that to call Inbound Web Services, you must provide a user/password for authentication and authorization (the user must be defined in Identity and Access Management Identity Domains with the 'AppWebServices' Application Role and as an application User). Inbound Web Services support both SOAP and REST. Outbound Messages may only reference public IP addresses, and those addresses must be on an 'allowlist' (which can be provided to Cloud Operations via a service request ticket).

For integrations that involve outbound synchronization to other systems driven by online activity, real-time synchronous outbound messages are not recommended. Rather use the business object batch monitor processing on a frequent basis to process queued messages. This involves using the deferred monitor batch set on the PENDING state of the Sync Request so that message processing occurs asynchronously. SSL certificates must be created using certification authority. Self-signed SSL certificates are not supported.

Upload and attachment of implementation-specific XSL files to process xml payloads is supported through the **Managed Content** portal for relevant product or cloud service. See the **Maintaining Managed Content** section in the *Oracle Utilities Application Framework Administrative Guide* or the online help for more information.

1.5.2 Allowlisting

Allowlisting is required to specify allowable access destinations on the public internet.

1.5.2.1 IP Allowlisting

IP Allowlists enable customers to control how data flows into or out of their SaaS environments.

Outbound Traffic

Outbound traffic is controlled via allowlist of IP addresses. Only HTTPS traffic is allowed to port 443.

Configuring IP Allowlists

To configure IP allowlists, customers must log a service request and follow the steps outlined in the **Cloud Operations** section of the *Oracle Revenue Management and Billing Cloud Service, Premium Edition Administration Guide* to provide configuration details.

1.6 Data Access and Analytics

This section provides guidelines related to data access and analytics. It contains the following topics:

- Analytics Publisher
- Database Access
- Reports and Queries
- File Access – Cloud Object Storage

1.6.1 Analytics Publisher

Data Visualization is included in the cloud service subscription, available via a separate URL for each environment. Analytics Publisher is available and included in the service as a reporting/query tool.

Note: Analytics Publisher and Data Visualization extract and display data from the production database.

SQL Developer Web is also available and included in the service for querying the database (see the **SQL Developer Web** section for more information).

Data extraction is supported via the Generalized Data Extract and Specialized Data Extract functionality, and/or DataConnect (ORMBCS) may be a starting point for extraction of data for a BI/reporting tool such as Cognos.

1.6.2 Database Access

No direct access is permitted to the application database either through Toad, SQL Developer, or command line utilities. This also means that you **cannot** create new tables or related data including new Maintenance Objects, custom audit tables, and database links.

Query access is supported both in Analytics Publisher and SQL Developer Web (see the **SQL Developer Web** section), which are available as part of the cloud deployment. For more information about Analytics Publisher, see https://www.oracle.com/middleware/technologies/analytics-publisher.html.

Analytics Publisher deployment includes a JDBC data source configured with credentials that allow access to read-only synonyms in the production schema. Note that in some cases it may be feasible to create a custom zone in the application to provide online display to view data.

1.6.3 Reports and Queries

Reports and queries can be run using Analytics Publisher, which is included with the cloud service deployment. Analytics Publisher deployment includes a JDBC data source configured with credentials that allow access to read-only synonyms in the production schema. Note that there are several output formats, and we have found that PDF performs best for larger reports.

To configure reports within cloud services via Analytics Publisher, implementations are required to configure the **Reporting Options** and relevant reporting Algorithms (see Application / Environment Access and URL Tokens for parameter values for base algorithms types). Reporting options with default configuration includes the following:

- Reporting Server from App Server = @BI_PUBLISHER_ADMIN_INT@
- Reporting Engine User ID = <use the user name having Analytics Publisher role assigned>
- Reporting Engine Password = <use the password for above mentioned user>
- Reporting folder = ormbcs <default reporting folder in Analytics Publisher but implementations can change it, if needed.>
- Reporting Server from Browser= @BI_PUBLISHER_ADMIN@

Oracle Revenue Management and Billing Cloud Service also offers the option of using SQL Developer Web and Data Visualization for reports and queries.

1.6.4 File Access – Cloud Object Storage

All inbound and outbound file-based data is staged in Oracle Cloud Object Storage (a separate service that the customer must license).

Cloud Object Storage involves creation of a set of compartments and buckets (each compartment can have many buckets, and have child compartments), and the compartments are represented within the application as values for the File Storage Configuration (F1-FileStorage) extendable lookup. If the customer creates a new compartment, a new value needs to be specified in the extendable lookup, with OCID references to the user/tenancy/compartment. Once that is set up, the application can reference buckets as needed.

The format for Object Storage paths is as follows:

```
file-storage://<Extendable Lookup value for Compartment>/<bucket>
- example: file-storage://OS-SHARED/CMA-Files
```

Typically, there are a few places where a 'path' to an Object Storage bucket can be specified, such as on batch job parameters, and some Master Configurations.

1.6.4.1 Uploading and Downloading File To and From Object Storage

There are two main options of exchanging files between Oracle Cloud Object Storage and the outside world:

- The Oracle Infrastructure Console User Interface which allows authorized users to upload or download files to and from object storage buckets.
- The object storage APIs which allow other applications to interact with object storage and exchange files as well as other actions.

When customers or implementers need to upload or download files in bulk, the option of the Oracle Infrastructure Console User Interface can be cumbersome.

2. Data Conversion and Migration

This section provides data conversion, migration, and implementation information relevant to the products included in Oracle Revenue Management and Billing Cloud Services. Most of the information is generic and applies to functionality that is available in each of the products as part of the Cloud Service Foundation. There are also some conversion tools that are documented with each specific product. (The specific products are referred to in this document as “products” or “applications”.)

This section contains the following topics:

2.1 Overview

This section provides guidelines for migrating and/or converting data between application environments, including moving data from existing applications into Oracle Revenue Management and Billing Cloud Service. Existing applications can include legacy applications as well as on-premise implementations of Oracle Revenue Management and Billing. See the **Conversion** section in *Oracle Utilities Application Framework Administrative Guide* for information about the general conversion process.

This overview section contains the following topics:

2.1.1 Data Conversion and Migration Overview

This section provides an overview of data conversion and migration. It contains the following topics:

2.1.1.1 Conversion Process Overview

The goal of the Conversion Process is to migrate data from a legacy application into a target environment, and to begin running the application in the cloud. Due to cloud-related technical restrictions, legacy data cannot be uploaded directly into the software-as-a-service (SaaS) database.

Legacy data must be extracted into files and compressed. The data files are uploaded to the cloud file storage location and then loaded into the target "staging" tables using Oracle SQL Loader. The data is validated, transformed, and finally inserted into "production" tables. Oracle Revenue Management and Billing Cloud Services include various tools supporting ad hoc SQL inquiries and reconciliation reports on both staging and production data.

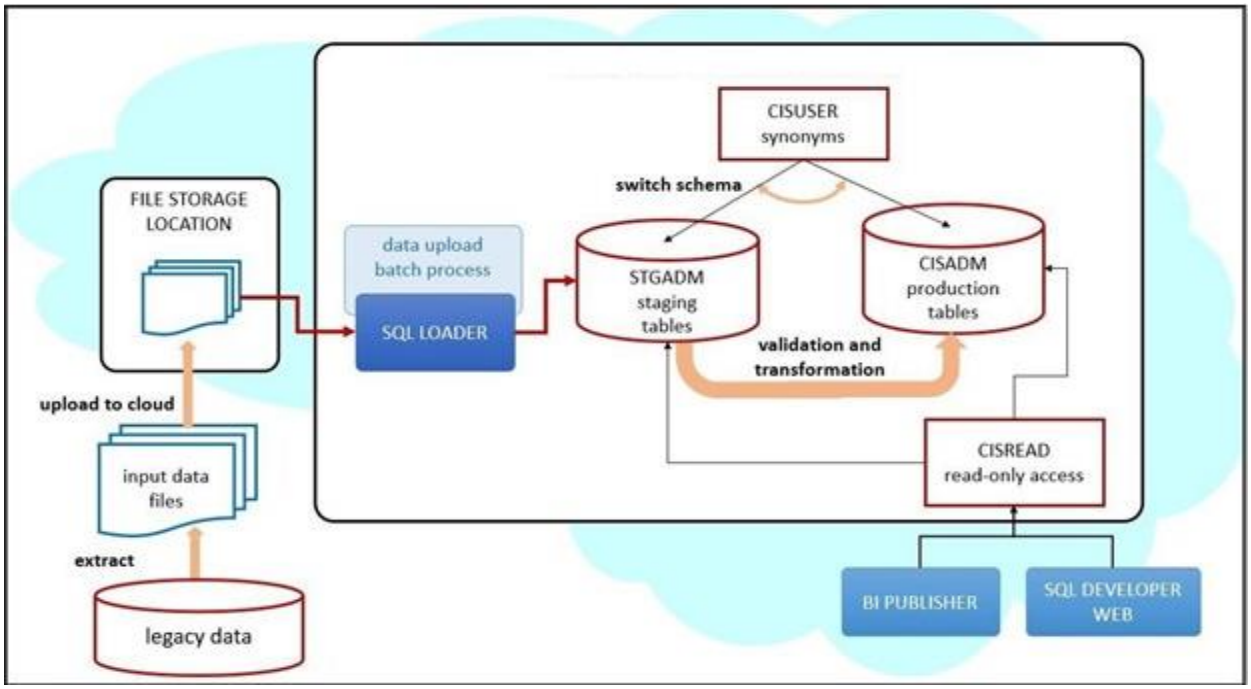

Figure 1: Conversion Process Overview

2.1.1.2 Implementation Effort

Implementers are expected to perform the following tasks for data conversion:

- Analyze the legacy data and decide what portion of it should be converted
- Map the legacy data to target Oracle Utilities Application Framework (OUAF)/Application data
- Develop legacy data extract process and produce input data files
- Adjust default data upload setup in OUAF/application, if needed
- Rehearse data upload and fine-tune configurations and/or legacy data extract, if needed
- Create reconciliation reports in BI Publisher
- Use uploaded data to try the subsequent conversion flows; bring the end-to-end conversion flow to perfection
- Execute the final conversion data upload run, a.k.a. cut-over
- Execute the application's data conversion processes.

- Disable conversion activities in the environment

2.1.1.3 What Is in the Newly Provisioned Environment?

The production instance is available for conversion.

Conversion activities do not co-exist well with the rest of the implementation. The massive data uploads, table truncation, and switching schema could disrupt business configurations development and testing. The production environment is the best candidate for conversion.

In the newly provisioned instance, the staging area in the database is created according to application specifications. The BI Publisher instance and SQL Developer Web/Oracle REST Data Services are connected to production and staging data.

The environment contains pre-configured conversion data upload setup.

The default configurations are suitable for typical table volumes and common data formats. If your implementation does not include extremely large data volumes, special data formats, or other idiosyncratic requirements, the default setup can be used "as is".

2.1.1.4 Data Conversion and Migration on Cloud

This section provides an overview of data upload support in Oracle Revenue Management and Billing Cloud Services. It contains the following topics:

- Provided by Cloud Service Foundation
- Provided by Applications

The highlighted portion of the flow shown below is supported by Oracle Revenue Management and Billing Cloud Service Foundation (CSF). The legacy data extract and the input file creation belong to the implementation. The business application provides conversion validation and transformation processes, as well as the definitions of the staging area.

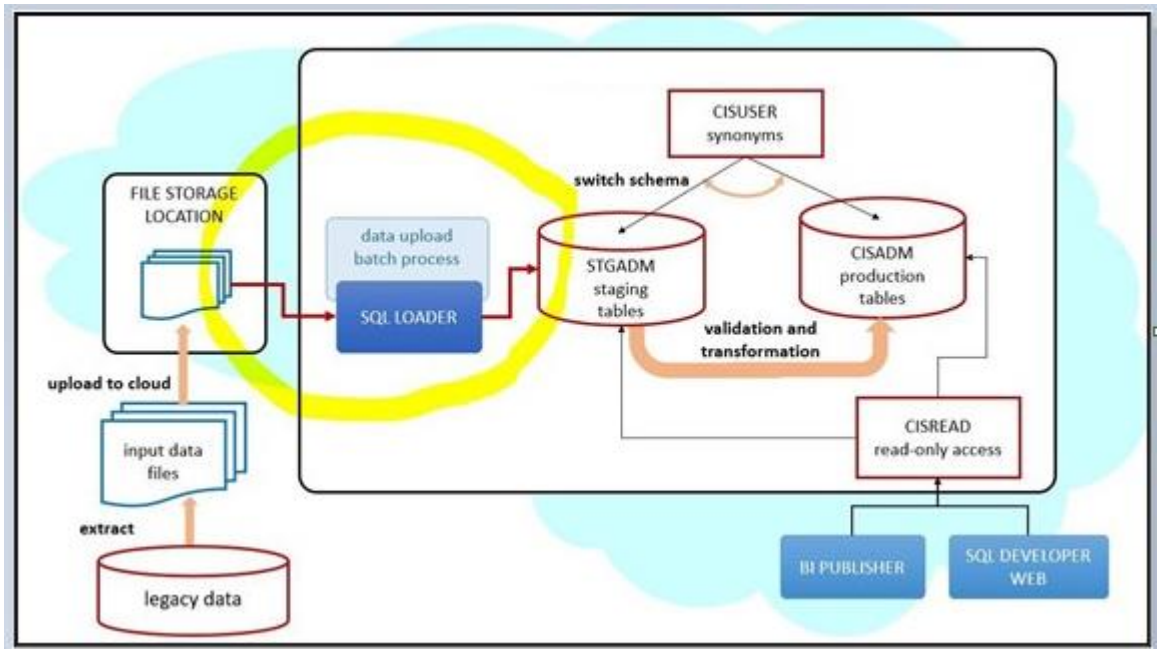

Figure 2: Data Conversion and Migration on Cloud

Provided by Cloud Service Foundation

Oracle Revenue Management and Billing Cloud Service Foundation (CSF) features metadata-driven configurable and customizable data upload with SQL Loader. It also provides support for basic database operations such as table clean-up (truncate), index enable/disable, and some others.

SQL Loader is an Oracle database utility that allows users to load data from external files into target DB tables. See Oracle DB SQL Loader Documentation for details.

The load of the Input Data File is performed according to the instructions recorded in a Control File. The Control File contains load options and parameters and a list of data fields with formatting and parsing instructions. For data upload on cloud, Control Files are pre-generated based on the metadata and conversion configurations and stored in the system.

Cloud Service Foundation allows users to generate control files, and it also provides a batch process that consumes the Input Data Files, reads the pre-generated Control File and calls SQL Loader.

Cloud Service Foundation delivers the following:

- Batch processes. CSF batch controls are "generic", with no default value for the parameter that specifies target table or maintenance object (MO). These batch controls are used mostly for development and testing purposes. Applications are likely to supply "specific" batch controls for each target table or MO.
    - Batch Controls: Load Data into Table or Maintenance Object, Truncate Table or MO Tables, Disable/Enable Indexes, Disable/Enable Triggers, Update Statistics, Populate Key Table, Cleanup Key Reference and XML resolution Tables, Generate Conversion Artifacts (bulk), and few others.
- Services accessible via online UI.
    - Switch Schema - Executes the stored procedure that is re-directing the CISUSER synonyms between staging and production.
    - Generate Conversion Artifacts - Creates input data file specifications and SQL Loader control files for specific converted objects. The artifacts are generated based on the metadata and according to the conversion data upload setup.
- The instance of BI Publisher that is connected to the database with read-only access to the staging and production schema tables.
- The instance of SQL Developer Web that is connected to the database with read-only access to the staging and production schema tables.
- Predefined Configurations:
    - Data delimiters and data format strings for date and date/time fields (Extendable Lookups)
    - Default Conversion Instructions (Conversion Task Types) for typical Table, MO, and Key Table
    - SQL Loader Control File fragments (Managed Content) for parallel and non-parallel load
    - Conversion Data Upload Master Configuration with default setup
    - User Groups and Template Users for conversion (suggested setup)

Provided by Applications

Each application comes with its own Conversion Accelerator that includes admin and system data for suggested upload configurations. Applications also provide a set of processes and tools to validate and transform the uploaded "staging" legacy data into real production form.

Application Conversion Tool

The Conversion Tool is usually comprised of processes, services, and configurations that support necessary legacy data validation and transformation.

The application connects to the @application_user schema in the database. This schema contains synonyms to the actual tables and views. Let's assume the legacy data has to be loaded into a table TABLEXXX. From the application perspective, the data upload process inserts the data into @application_user.TABLEXXX. The meaning of the synonym @application_user.TABLEXXX could be @production_schema.TABLEXXX or @staging_schema.TABLEXXX. Once the *switch_schema()* stored procedure is executed, the assumption is that the synonyms in @application_user schema are set to point to the "staging" tables, and the data upload may begin.

Possible approaches to the staging data upload into target tables are described below.

Approach: The legacy data is uploaded into a set of tables in the staging schema (STGADM).

Upon successful legacy data upload, the sequence of batch processes performs object level validation and FK validation of the data in the staging tables, generates new keys and finally inserts the data into production tables.

Approach: The legacy data is extracted and loaded into Initial Sync tables in the production schema (CISADM). This approach is relevant under special circumstances such as extremely large data volume. Another reason for loading data directly into production could be a migration of existing Oracle Revenue Management and Billing from on-premise to cloud, when the legacy data is a valid application data, conforming to target data formats and standards and there is no need for additional validations and key generation.

Note: The Switch Schema has to be performed anyway in order to set the internal system indicator to Staging and allow the data upload.

A custom control file should be created to insert the data into production tables. Upon successful legacy data upload, the data can be further processed using regular application processes.

Application Accelerators

The accelerator usually contains suggested configurations for data upload support. It may include:

- Conversion Instructions for tables/maintenance objects with special data requirements (Conversion Task Types)
- Alternative Control File fragments and custom Control Files (Managed Content)
- Conversion Master Configuration
- Specific Batch Controls for each converted Table or MO, suggested batch job/batch streams for suggested conversion activities orchestration
- Sample reconciliation reports
- Other system, admin, and/or configuration data

2.1.2 Terms and Definitions

The processes described in this section use the following terms:

Term	Definition
Upload	Data is uploaded directly into the database from a legacy application.
Bind	Direct relationships between records, such as between devices and assets, are created via batch processing.
Import	Data is imported from an existing Oracle Revenue Management and Billing application
ORMBCS	Oracle Revenue Management and Billing Cloud Service

2.1.3 Database Tables

The processes described in this section reference several different types of database tables. These are described below.

- **Production Tables**: Tables used by application when running in a production environment. Examples of production tables include CI_SP (Service Point – RMB). Production tables are accessible through the Table portal in the Oracle Revenue Management and Billing Cloud Service application.
- **Staging Tables**: Tables used to facilitate import and migration into the product database. Staging tables are not accessible through the Oracle Revenue Management and Billing Cloud Service application user interface.
- **Key Tables**: Tables used to facilitate generation of keys. Examples of key tables include CI_SP_K (Service Point Key), D1_SP_K W1_ASSET_K (Asset Key). Key tables are accessible through the Table portal in the Oracle Revenue Management and Billing Cloud Service application.

2.1.4 Scope and Assumptions

The processes described in this section are based on the following scope and assumptions:

- Legacy data has been loaded to the staging tables.

2.2 Data Conversion and Migration Design

There are several aspects implementation should consider when designing the legacy data extract processes and creating the Input Data Files. The data conversion process is very flexible and configurable and can be fine-tuned to address both application and client data specifics.

This section provides information about designing extract processes. It contains the following topics:

- Extract/Upload by Table or Maintenance Object
- CLOB Data in a Secondary File
- Multiple Data Files for Single Table or MO Upload

2.2.1 Extract/Upload by Table or Maintenance Object

The SQL Loader allows users to insert data into one or multiple tables from a single input file. Choose the more convenient option, depending on the structure of the legacy data (source), data volumes, and extract technique:

- **Table-level**. Extract file contains data for the single table. The data is loaded into a table in the OUAF/application database.
- **Maintenance Object-level**. Extract file contains data for the entire object. The data is loaded into a set of tables that represent the corresponding Maintenance Object in the OUAF/application database.

Both options are supported in Cloud Service Foundation. Generate the artifacts and review the differences in the specifications. The table below illustrates the difference between Table and Maintenance Object data file:

Target Object	Table: CI_PER Data file contains records for a single table.	Maintenance Object PERSON: Tables: CI_PER CI_PER_NAME CI_PER_ID ….etc Data file contains records for multiple tables within Maintenance Object. Table name serves as “record type” qualifier.
Input Data File Layout	1234, IND, Doe,… 5678, IND, Moon,… 9063, BUS, ABC Corp,..	CI_PER 1234, IND, Doe,… CI_PER 5678, IND, Moon,... CI_PER 9063, BUS, ABC Corp,.. CI_PER _ID 1234, SSN,72346781 CI_PER _ID 5678, SSN, 87635241 CI_PER _ID 9063, EIN, 09182835 CI_PER _ID 9063, TID, 82528555 CI_PER _NAME 1234, Doe, Mary CI_PER _NAME 5678, Moon, Barry

2.2.2 CLOB Data in a Secondary File

CLOB data can be supplied as part of the record in the "main" data file or as a secondary file. Once again, the decision should be made based on the source data volumes, extract techniques, and the availability of the CLOB data in most records.

- If most of the records have CLOB columns populated, and/or the CLOB field often contains large amount of data, it may make sense to use a secondary file.
- Otherwise, if the CLOB columns are rarely populated and/or the CLOB field rarely contains large amount of data, you may choose to include the CLOB data in the record.

Note: If supplied as secondary file, the CLOB data file must contain exactly as many records as the main file. This means that a line must be added even for empty CLOB fields.

Both options are supported. The definition is controlled by the Conversion Instruction (Conversion Task Type).

2.2.3 Multiple Data Files for Single Table or MO Upload

The Cloud Service Foundation data upload process supports the upload into single target (table or maintenance object) from multiple data files. For example, instead of extracting a large Payment table into a single *payment.csv* file, you can split the extract into *payment1.csv*, *payment2.csv*, *payment3.csv*, and so on. It is recommended to keep the file size under 2 gigabytes. The number of files is unlimited. Naming conventions apply. See the online help for more details.

2.3 Preparing for Conversion

This section describes how to prepare an environment and legacy data for conversion with Oracle Revenue Management and Billing Cloud Services. It contains the following topics:

- Preparing Environment for Conversion
- Preparing Legacy Data Extract for Upload

2.3.1 Preparing Environment for Conversion

Preparing an environment for conversion involves the following:

- Set Up Conversion Security
- Prepare Environment for Conversion

2.3.1.1 Set Up Conversion Security

Conversion activities comprise massive data manipulations and database operations such as disabling/enabling indexes, truncating tables, and other operations. Whoever works on the conversion project deals with the real client's data and may have access to sensitive customer information. Therefore, it is important to determine implementer's roles and responsibilities in advance, and to provide the user with the appropriate authorization level.

Use the pre-configured user groups *Conversion Administration*, *Conversion Development*, and *Conversion Operations*, along with the corresponding Template Users K1CNVADM, K1CNVDEV and K1CNVOPR. Alternatively, design and define your own conversion user authorization setup.

2.3.1.2 Prepare Environment for Conversion

To prepare an environment for conversion, you need to:

- Enable conversion activities in the environment.
    - Run K1-CNVEN batch.
- Import the Conversion Data Upload Accelerator if it was supplied by the application.
- Generate conversion artifacts.

- To generate artifacts for all eligible tables and/or maintenance objects, submit a batch job for the K1-CNVAG batch control and use batch parameters to specify the scope for the generation: everything, Tables only or Maintenance Objects only.
  - As a result, new Conversion Task is generated for each Table and each Maintenance Object eligible for conversion. The artifacts are linked to the Conversion Tasks as attachments.
- To generate artifacts for an individual table or maintenance object, select **Admin**, select **Conversion Support**, and select **Generate Conversion Artifacts**. Choose "Table" or "Maintenance Object" and run the generator.
  - As a result, a new Conversion Task is generated for the selected table or maintenance object. The artifacts are linked to the Conversion Tasks as attachments.
- Query Conversion Tasks that were created for various Tables and Maintenance Objects and explore the generated artifacts:
  - **Input Data File Specifications**. This file contains the detailed field by field formatting instructions and other notes about the expected contents of the input data file. Use these instructions when preparing the legacy data extract.
  - **Control File**. This file is used by SQL Loader during the data upload.
  - **File List**. This file lists the name of the input files that must be prepared for the data upload.
    - Multiple data files for a single object (Table or Maintenance Object) are expected if the data is being uploaded contains CLOB columns AND the upload is configured to load CLOB from secondary file.

- Switch schema to redirect the application to the staging data area.
  - Use the menu to navigate to the generator by selecting **Admin**, then selecting **Conversion Support**.
- Truncate tables in the staging data area to ensure that you will be uploading the data into clean empty tables.
- Disable indexes in the staging data area. This is required because SQL Loader is not capable of implicitly disabling partitioned indexes during the data upload.
- Disable triggers in the staging data area.

The environment is now ready for the legacy data upload:

- Conversion is enabled
- SQL Loader Control Files have been generated
- Synonyms in the database schema point to the staging data area tables

Notes:

- Conversion activities are possible if conversion is enabled in the environment. Once the legacy data is successfully migrated, you should disable conversion by running the K1-CNVDS batch. By doing this you set an internal indicator that is queried by conversion-related processes, such as switch schema, data upload, table cleanup, and index/statistics update. These processes will only run when conversion is enabled.

Important: The Disable Conversion process should be executed ONLY ONCE right before the system is ready for go live. It is one-time event and is irreversible. Once disabled, conversion activities cannot be fully re-enabled as the assumption is that the re-enabling is happening while the application is running live in production. Enabling conversion after it has been disabled will result in the application running in the Incremental Conversion mode with its limitations.

- Switching the schema sets an internal flag that indicates whether the synonyms are pointing to "staging" or "production" area. The data upload is only allowed when the application is running in a "staging" mode.
- It is recommended to perform truncate operations at the maintenance object level as it will prevent leaving orphan records in the database. When truncating tables one by one always truncate child tables first.

2.3.2 Preparing Legacy Data Extract for Upload

The legacy data mapping and extract will vary from one customer to another. The files created because of the extract process should conform to the specifications generated above. The resulting data extract files should be:

- Created according to the specifications
- Named according to the naming convention (see the online help and the specifications for more details)
- Optionally, the file might be compressed with gzip or zip (see the online help for details)

2.4 Data Conversion and Migration Steps

This section describes the steps involved in data conversion and migration. It contains the following fields:

- Upload Data into a Table or Maintenance Object
- Data Upload Orchestration

2.4.1 Upload Data into a Table or Maintenance Object

The data upload stage may begin only after conversion artifacts have been generated.

2.4.1.1 Review Input Data File Specifications

The following are the input data file specifications for review:

- Retrieve the Conversion Task associated with the Table XXX
    - Navigate to **Admin**, then **Conversion Support**, then **Conversion Task Query** and select the "Table/Maintenance Object" **Query Option**.
    - Use search to populate either Table or Maintenance Object search criteria.

- From the search results, pick the latest entry.

- Load Conversion Task and locate a collection of Attachments.
- Find an attachment that represents *Input File Specification*.
- Click on the context menu to launch **Attachment** view the attachment contents.

2.4.1.2 Create Input Data Files

The specification defines the expected input data record format. The data fields are listed in the order it expected to appear in each record. For each field, the specification contains the data type, size, and format. The specification also describes:

- Data delimiter
- Enclosing characters (to enclose a single blank that will represent empty nonnull able field)
- Date and date time formats
- CLOB data delimiter
- Expected names for the secondary data files

Extract the legacy data into a file according to the specification. Each line in the file should represent a row in the target table. In the maintenance object level extract, each record represents a row in one of the maintenance objects tables and the first 30 characters in each line contains the table name. If CLOB data is to be provided as secondary file, create CLOB data files.

Points to Note:

The SQL Loader treats invalid secondary file differently than a missing secondary file:

>> If the secondary file is missing, the process will report an error.

>> If the CLOB data in the secondary file is invalid, the CLOB field in the target table will be initiated into NULL or blank.

The input data file might be supplied uncompressed or compressed. Supported compressed formats include gzip and zip (See online help for details).

2.4.1.3 Switch Schema

Navigate to **Admin**, then **Conversion Support**, then **Switch Schema**, and select "Conversion" from the drop-down list and click **OK**.

2.4.1.4 Cleanup Target Table

Run the K1-SCLTB batch process, specifying the target table or maintenance object as a parameter.

2.4.1.5 Upload Data

Upload the input data file created above to the Object Storage location. Run the K1-CNVLD batch process, specifying the target table or maintenance object as a parameter. Detailed description of data upload parameters can be found in the online help.

2.4.1.6 Populate Key Tables

According to OUAF DB design standards, a corresponding Key Table exists for each table with system-generated or sequential primary key. Under normal circumstances, the key tables are populated when an application creates a "main" record. In a conversion situation, where the data is inserted directly into the database, there are two possibilities to populate the Key Table:

- Create an input data file for the Key Table and upload it using the same batch K1-CNVLD.
- Populate the key Table programmatically, by running K1-CPKTB after successful "main" table or MO data upload. This batch can be used for both Table and MO-level upload.

2.4.2 Data Upload Orchestration

The SQL Loader is running in multiple threads and therefore it is not performing table truncation before loading (command APPEND). Hence, the target tables should be truncated prior to the load. For better performance the indexes must be disabled before the load and re-enabled/statistics updated after the load. The batch jobs can be organized into various chain structures, as shown in the examples below.

Single Table Upload

Figure 3: Single Table Upload

Multiple Tables or MOs Upload

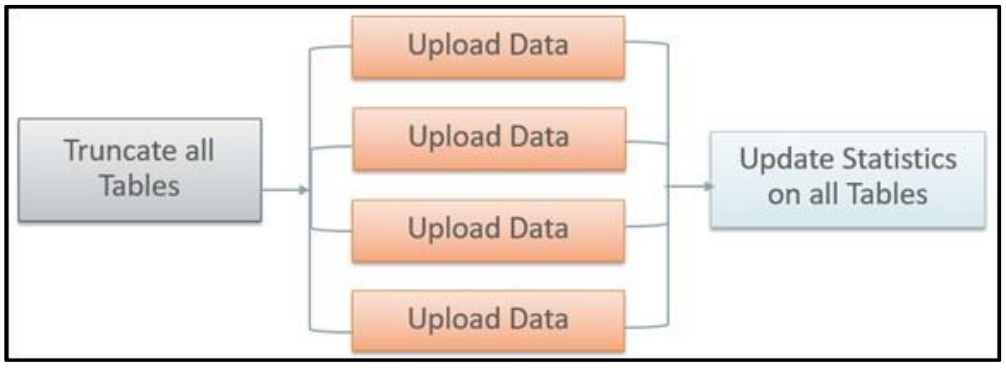

Figure 4: Multiple Tables or MOs Upload

There are multiple strategies to orchestrate the entire conversion run and to build the optimal sequence of the conversion processes. Below are some of the many possibilities:

- upload all legacy data extract files simultaneously, then run the subsequent validation and transformation processes for the converted object in a certain order of precedence, to preserve referential integrity
- begin the upload of very large tables in advance, so all upload is finished simultaneously, then validate & transform
- include legacy data upload batches in the batch job chain for the target object
- upload some of the data by maintenance object, some table by table

- process maintenance objects end-to-end simultaneously if there are no interdependencies

Full Conversion Chain per MO, Parallel Run

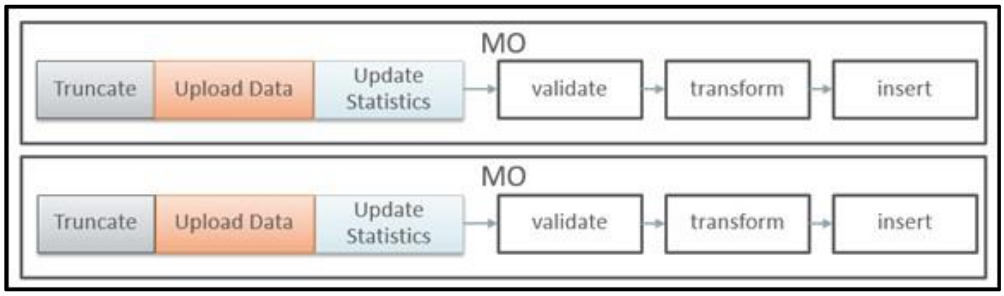

Figure 5: Parallel Run

Upload All + Subsequent Validate/Transform MOs

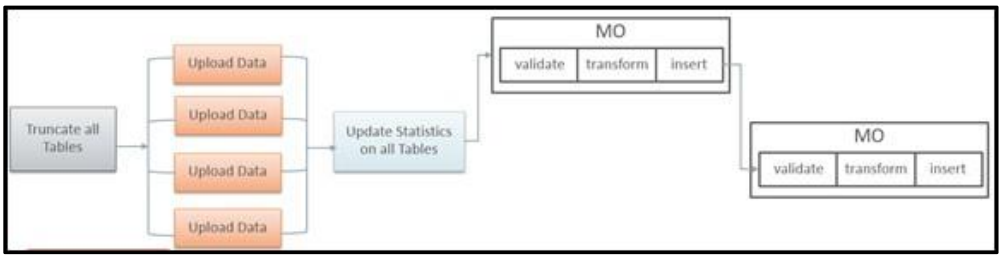

Figure 6: Upload All and Subsequent Validate/Transform Mos

2.5 Customizing Data Conversion and Migration

Conversion-related configurations define the expected extract file layout and the SQL Loader run-time upload options and parameters. SQL Loader's Control Files are generated based on these configurations. The Batch Job/Batch Job Chain setup defines the overall orchestration of the conversion process flows. This section describes customizations to the data upload process. It contains the following topics:

- Why Customize
- When to Customize
- What to Customize
- How to Customize
- Tips and Important Mistakes to Avoid
- Sample Artifacts and Data Files

2.5.1 Why Customize

There are several reasons for customizing conversion configurations, including:

- fine-tuning data upload performance
- handling unusual data volumes
- marking additional tables as eligible for conversion
- reducing creation of unnecessary input files

2.5.2 When to Customize

The layout of the legacy extract files should be finalized as soon as possible, to provide enough time for the extract process development.

The setup of the batch job chains is less critical at the beginning of the project. The initial suggested setup is likely to be included in the application Conversion Accelerator. Adjust the initial setup after you've performed the trial uploads of the actual data, assessed the performance, and figured the optimal flows.

2.5.3 What to Customize

This section lists the items that you can customize:

- Control File
- Additional Customization Items

2.5.3.1 Control File

Most of the customizations affect the contents of the generated Control File and the corresponding input data file specifications. The configurations are stored on the Conversion Task Types that represent Conversion Instructions.

- Customizing the Control File's load options and parameters may improve upload performance.
- Fully customized Control File allows you to use alternative record parsing and other advanced SQL Loader configuration techniques.
- When CLOB data is supplied as Secondary Files, the system is expecting the input data files to exist and be named following the specific naming convention.
- For example, if the table has multiple CLOB fields, for every CLOB field that was not excluded from conversion, the system is expecting the secondary file's name to be suffixed with _*<CLOB Field Name>*. See the online help for more details.

Data Delimiters and Enclosing Characters. Examine the default Conversion Instructions (Conversion Task Type) setup. Either select another delimiter from the existing list or add new value to the Extended Lookup.

CLOB as Secondary File? The indicator is defined on Conversion Instructions (Conversion Task Type).

Applicable when CLOB is supplied as Secondary File:

CLOB Columns Included in Conversion. By default, the control file is generated as if all CLOB fields are part of the converted data. The legacy data does not necessarily contain data for all CLOB fields, hence there is no reason to create empty files. The list of excluded CLOB columns is is defined on Conversion Instructions (Conversion Task Type). Create new Conversion Instructions (Conversion Task Type) for the Table or MO with multiple CLOB fields and specify the exclusion list.

Control File "Header" - Load Options. A text stored as Managed Content. Contains the control file's fragment with options and load parameters. You can amend the options according to *SQL Loader Documentation*. Examine the entries delivered with the product.

Points to Note:

The text contains several substitution parameters prefixed with %. The substitution happens at generation time or at run time. Preserve them while creating a custom Control File header.

If you wish to amend the load options and parameters only, create a new Managed Content entry. Modify default Conversion Instructions (Conversion Task Types) or create new ones and add Override Instructions to Conversion Master Configuration. Run Conversion Artifact Generator and create new customized Control File. See the online help for more details.

Custom Control File. A text stored as Managed Content and representing the entire Control File, including load options, parameters, and the field list.

Note: Preserve substitution parameters (see the note above). The input data file specifications are not generated when the Custom Control File is used. Make sure that the fields in the input data files correlates to the field's list in the custom Control File.

2.5.3.2 Additional Customization Items

Table's Conversion Eligibility. The table is considered eligible for conversion according to the indicator on the Metadata Table record. It is a system data and cannot be modified by the implementation. To make a non-converted Table eligible for Conversion, you should add an entry to the *Override Conversion Eligibility* list on the Conversion Master Configuration.

Conversion Orchestration. The suggested setup of the Batch Controls, Batch Jobs, and Chains is usually included in the application Conversion Accelerator. Adjust this setup by fine-tuning the number of threads, the chain structures and other batch job parameters.

2.5.4 How to Customize

Configurations can be amended on several levels:

- To modify the configuration globally, amend the default Conversion Instructions (Conversion Task Type) that is referenced on *Conversion Data Upload* Master Configuration.
- To modify the option globally for all tables, amend the default Conversion Instruction for Table (Conversion Task Type) that is referenced on *Conversion Data Upload* Master Configuration.
- To modify the option globally for all maintenance objects, amend the default Conversion Instruction for MO (Conversion Task Type) that is referenced on *Conversion Data Upload* Master Configuration.
- To modify the option for a specific tables or maintenance objects, create new Conversion Instruction (Conversion Task Type) and add the Override Instruction for Table or MO on *Conversion Data Upload* Master Configuration.
- To make a non-converted table eligible for conversion, add it to the Override Conversion Eligibility list on *Conversion Data Upload* Master Configuration.

Important: Regenerate Conversion Artifacts to apply the configuration changes. Download the updated input file specifications.

2.5.5 Tips and Important Mistakes to Avoid

Issue	Details
Run the process against the right target	The data upload only runs if the environment is pointing to the STAGING schema. Navigate to Conversion Support Switch Schema. On the popup screen the current schema is displayed. Make sure the current schema is **Staging**.
Provide data files according to the specifications. Regenerate the artifacts after modifying the data upload configurations.	SQL Loader loads the data according to the Control File. Input Data File Specifications describe what is expected from the input data file: • Names of the data files • Data format for all fields • Data delimiters to be used in the input data file Every time the configuration has changed the artifacts must be regenerated to keep the configurations and the input data specifications in sync.
Provide input data files with CLOB data if necessary	Conversion Instruction defines whether CLOB data is provided as part of the main file or as a separate file. The system expects the data files to be provided according to this definition. Open the Input Data Specifications and read carefully. If the specification mentions that CLOB is to be provided as a secondary file, this is what Control File would inspect. If you wish to include CLOB data in the main file, verify that the Conversion Instruction is set correctly. **If the configuration was modified, you must regenerate the artifacts.**
Avoid creating unnecessary data files for CLOB columns	By default, the system expects the data to be provided for all target table columns. If the table contains multiple CLOB columns AND the CLOB data is provided as a secondary file, it means one input data file per column. To exclude unnecessary CLOB columns for a table or maintenance object, configure Conversion Instructions using the *K1-ConvArtMultClobMOTaskType* or *K1-ConvArtMultClobTblTaskType* business object and specify the Override Conversion Instruction on the Master Configuration. Regenerate conversion artifacts and examine the input data specifications after changing the configuration.

Issue	Details
Avoid truncating the entire staging data unintentionally	The K1-SCLTB batch process allows you to truncate a specific table or maintenance object in the STAGING schema. The K1-CLNTB batch process allows you to truncate a specific table or maintenance object in the PRODUCTION schema. If submitted without input parameter specifying a table or maintenance object, these batches will process all tables eligible for conversion. This means that all your staging data will be wiped out at once.
Clean up duplicate PK values before the data upload	Indexes and constraints are disabled during data upload to boost performance. De-duplication during the data upload is not supported out-of-the-box: • SQL Loader direct path upload doesn't perform duplicate check • No direct database access means no possibility to modify data via direct SQL after the upload Keep track of the legacy data that has been already uploaded. If you re-upload the same data again, always clean up the target tables.
The business configurations and admin data must be finalized and populated in Production prior for legacy data upload. Populate the legacy data extract with valid FK references to the admin/control data.	**Once uploaded, the staging data cannot be "massaged"/modified thru direct SQL** (that because no database access is possible on cloud). Hence the overall conversion project steps are: • Design, test, and complete business configurations. During this stage, multiple trial data uploads with dummy data could be performed • Populate admin data in Production • Create legacy data extract with valid admin data FK References • Upload data into staging tables
Key Tables are not populated implicitly	The Key Tables in the staging schema tables are not populated automatically when the legacy data is uploaded into "main" tables. Upload the data into Key Tables separately or use the batch program provided by Cloud Service Foundation.

Issue	Details
Override Conversion Eligibility is supported on Table level only	The conversion eligibility is overridden for individual tables. Override the eligibility for all the tables that belong to the maintenance object if you decided to convert the entire maintenance object. **Note**: Overriding a table's conversion eligibility doesn't mean that the staging schema is automatically updated. It only means that the data upload processes will threat this table as a valid target table.

Loading Data Directly into the Production (CISADM) Schema	The following configuration steps are required to load data directly into the Production (CISADM) schema tables:
	• Create a custom Control File for the target table.
	  ○ Generate the control file with default conversion instructions and copy the contents.
	  ○ Modify the INTO... clause to add 'CISADM.' in front of a target table name
	  ○ Create a new Managed Content entry. Copy the entire control file text and save.
	• Create a new Conversion Task Type for the target Table
	• Specify the new Managed Content as an Override Control File.
	• In the Data Upload Support Master Configuration, create an Override Table Instruction entry. Specify the target table and the new Conversion Task Type.
	• Generate Conversion Artifacts for the table.

Issue	Details

Loading Very Large Data Volumes	Avoid SQL-based conditions in the control file when loading very large data volumes. The default values and SQL-based conditions will cause SQL Loader to switch to the conventional path load which performs row-by-row inserts.
	The best results are achieved with a direct path load.
	More threads doesn't necessarily mean better performance. The optimal overall data load performance is achieved when the threads (and their corresponding SQL Loader processes) are targeting different partitions.
	Additional guidelines:
	• Partitioning by month is required for best performance
	• Load multiple months in parallel for best performance & scalability
	  ○ Start with ONE thread per month
	  ○ Increment number of threads per month. If performance does not increase, try a smaller increment, or stay with your last best. For example: loaded 12 months data with 48 threads, 4 threads/month
	• Large data files are preferable
	  ○ Many small files have the overhead of spinning up new SQL*Loader process for each file
	  ○ Set longer SQL timeout on the data upload batch process
	• Disable indexes before loading
	• Rebuild indexes after direct path load
	• Reduce or stop the activities in the environment when performing the massive data upload

2.5.6 Sample Artifacts and Data Files

To assist implementers with the conversion and data upload process, multiple sample artifacts and data files are available. The sample files are provided with your cloud service documentation. The samples illustrate various data upload scenarios for table- and MO level upload. Within the master samples zip file, there are multiple zip archives, each of which contain the following:

- Control file, generated
- Input Data File Specification, generated
- Sample Data File, created according to the specification

The table below provides more details on each of the sample artifacts available.

Target Object	Sample Description
Interval Data Set (INT_DATA_SET)	Regular maintenance object, CLOB field as a secondary file. **Configuration:** Conversion Task Type K1-CNV-MO Multiple data files (3)
MO Customer Contact (CUST_CONTACT)	Regular maintenance object, CLOB fields in the main file. **Configuration:** same as Conversion Task Type K1-CNV-TABLE, but the *CLOB as Secondary File* indicator set to false.
Table Adjustment (CI_ADJ)	Regular table, CLOB field in the main file. **Configuration:** same as Conversion Task Type K1-CNV-TABLE, but the *CLOB as Secondary File* indicator set to false.
Table Initial Sync Request (F1_SYNC_REQ_IN)	Table with Multiple CLOBs as secondary files. **Configuration**: For table with multiple CLOBs, the special Conversion Task Type was created based on the K1-ConvArtMultClobTblTaskType business object. Override Control File (Managed Content) was created and used as a custom Control File. Review the sample and note that there is a conditional input data selection. Only records with BO = W1-CompositeSyncReqGISAsset would be uploaded. A custom Control File is necessary if you have a requirement to manipulate the data during upload. **Input Data File Specification**: Since the Control File is fully custom, including the field list, the generated specification is describing expected file names only. The data field formats, delimiters, sizes, and any other information related to the Input Data File layout should be determined based on the custom Control File.

3. File-Based Integration

The next three sections describe how implementations can integrate and exchange information from Oracle Revenue Management and Billing Cloud Service to other applications and vice versa through file-based integration. File upload and download processes are used by some implementation for data connect, payment upload, letters extract, financial extracts, other business processes.

Oracle Revenue Management and Billing Cloud Service can exchange data files from one application to another by:

- Directly accessing Oracle Cloud Object Storage
- Integrating with Oracle Integration Cloud

For more information about this approach, see the *Oracle Integration Cloud Documentation* at https://docs.oracle.com/en/cloud/paas/integration-cloud/.

These sections provide information on how Oracle Revenue Management and Billing Cloud Service access data files from Oracle Cloud Object Storage. It contains the following topics:

- Object Storage Connection Management
- File Export Sample Implementation
- File Import Sample Implementation
- Reporting Module FOP Changes
- File Upload Interface Changes

3.1 Object Storage Connection Management

This section outlines how to manage connections between Oracle Revenue Management and Billing Cloud Service and Oracle Object Storage. It contains the following topics:

- Oracle Object Storage Setup
- Oracle Revenue Management and Billing Cloud Service Configuration for Object Storage Connection
- Register API Key to Oracle Cloud Object Storage

3.1.1 Oracle Object Storage Setup

Before initiating a file transfer from an Oracle Revenue Management and Billing Cloud Service to Oracle Cloud Object storage or vice versa, you must first make sure that the basic administration tasks in Oracle Cloud Infrastructure related to Object Storage have been completed properly, and that the compartments and buckets where the import and export files are stored have been setup.

3.1.2 Oracle Revenue Management and Billing Cloud Service Configuration for Object Storage Connection

Authentication and connection between the Oracle Revenue Management and Billing Cloud Service and Object Storage enables batch processes to import and export files from and to Object Storage locations. Setting up this authentication requires the following in your Oracle Revenue Management and Billing Cloud Service:

- Creating API Keys
- Creating an Object Storage Connection

Note: You can use the same API Keys and Object Storage Connection setup for both import and export process.

3.1.2.1 Creating API Keys

Create a key ring for the cloud service environment. The key ring should be active and should have a set of private/public encryption key pairs. This key ring will be included in the Object Storage Connection Configuration.

Creating Key Rings and Pairs

Authentication between the Oracle Revenue Management and Billing Cloud Service and Oracle Object Storage requires an API signature key.

API key rings and key pairs are maintained in the **Key Ring** portal in the cloud service. This portal contains the following zones:

- **Key Ring**: Displays basic information about the key ring
- **Key Pairs**: Displays a list of key pairs for the current key ring

Key rings are defined by the following:

- **Key Ring**: A unique code for the key ring
- **Key Ring Class**: Signature (default)
- **Status**: Current status of the key ring

Note: Key pairs can only be generated for Active key rings. Once a key ring has been deactivated, you can no longer create key pairs for that ring.

- **Description**: A name for the key ring (this will be referenced in the File Location extendable lookup, see below)

Once the key ring is created, you need to generate and activate the key pair. Click **Generate Key** to generate a key pair for the key ring. Key Pairs are defined by the following:

- **Sequence**: The sequence of the key pair (the order in which the key pair was created)
- **Creation Date/Time**: The date/time when the key pair was created
- **Key Status**: The current status of the key pair. Key pairs are inactive when first created.
- **Public Key**: Click **View** to open a dialog box containing the public key.
- **Action**: Click **Activate** to activate an inactive key pair.

Click **Activate** in the **Actions** column in the **Key Pairs** zone. A dialog box opens displaying the following message: “Warnings: Activating a key assumes that you have already registered the public key with the appropriate third parties. Press **Cancel** to abort.” Click **OK** to activate the key. The **Key Status** column will change to “Active”.

Note: Be sure to register the API Key with Object Storage by copying the public key to Oracle Identification and Access Management. To copy the public key, click **View** in the **Actions** column in the **Key Pairs** zone, and select and copy the text in the **View Public Key** dialog box.

3.1.2.2 Creating An Object Storage Connection

Create an object storage connection via the File Storage Configuration extendable lookup (F1-FileStorage). This defines the Object Storage location where the files will be stored.

Creating File Storage Extendable Lookup Values

Apart from the authentication, the cloud service also needs information about the Object Storage locations to be used. Object Storage locations are defined by values in the File Storage Configuration (F1-FileStorage) extendable lookup. These file storage configurations will be referenced by the batch processes that will import or export records.

Values for the File Storage Configuration extendable lookup are defined by the following:

- **Value**: A unique code for the extendable lookup value. This value will be referenced as a batch control parameter value.
- **Description**: A description of the extendable lookup value
- **Status**: The current status of the value. Select “Active”.
- **File Storage Details**: This section defines details for the object storage location, including:
    - **File Adapter**: The type of file adapter for the location. Select “Oracle Cloud Object Storage”.
    - **User**: The user Oracle Cloud ID (OCID) for the object storage location
    - **Tenancy**: The tenancy Oracle Cloud ID (OCID) for the object storage location
    - **Compartment**: The compartment Oracle Cloud ID (OCID) for the object storage location
    - **Namespace**: The namespace for the object storage location
    - **Key Ring**: The Key Ring you created earlier
    - **Region**: The region of the object storage tenancy for the connection (Values for this field are defined in the F1_REGION_FLG lookup

See the **External File Storage** section in the *Oracle Utilities Application Framework Administrative Guide* for more information.

3.1.3 Register API Key to Oracle Cloud Object Storage

Once the key ring and key pair have been created in the Oracle Revenue Management and Billing Cloud Service, copy the public key from the key pair and add the public key to the **User API Key** in Oracle Identification and Access Management (IAM).

3.2 File Export Sample Implementation

This section provides an example of implementing a file export process. It contains the following topics:

- Creating a File Export Batch Process
- Configuring the Export Process

3.2.1 Creating a File Export Batch Process

Input/output file path soft parameter is required with extendable lookup and Bucket information while running batch.

Below batches have the impact of CFS environment and need the parameter accordingly.

Soft parameter value needs to be provided in Extendable Lookup/Bucket format.

Example: DWLDBILC:

Soft parameter Output directory path needs to be passed with extendable lookup and Bucket information.

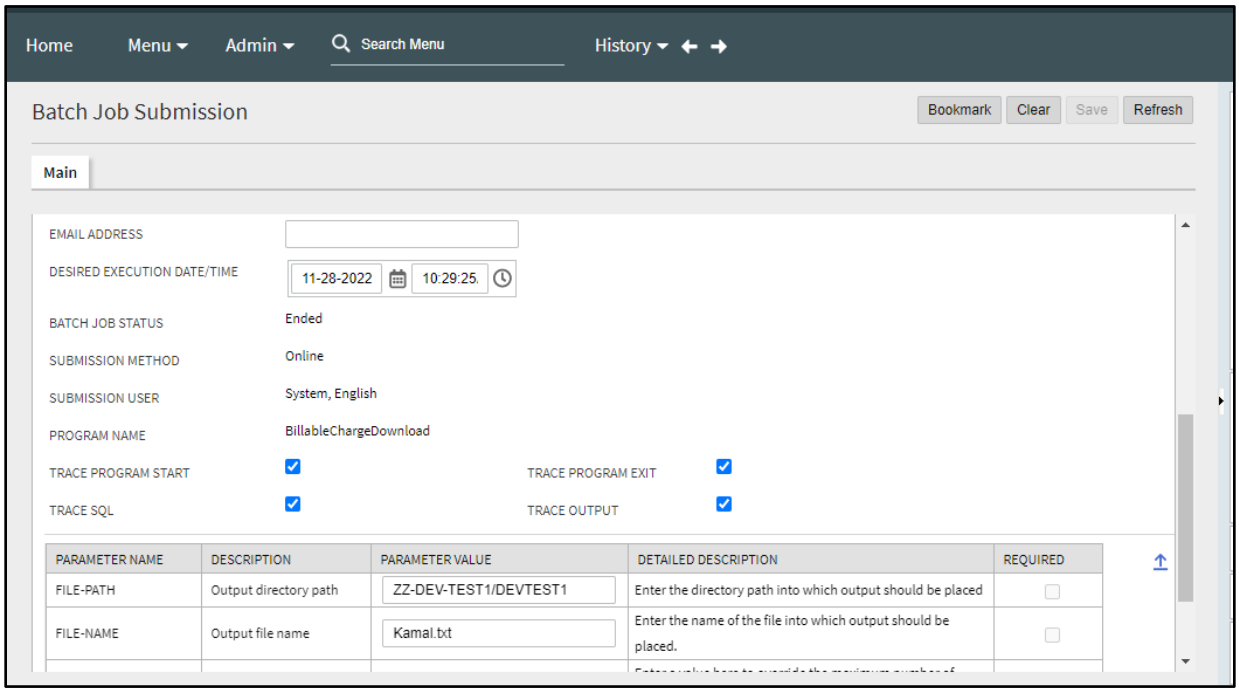

Figure 7: Batch Job Submission

After successful completion of batch, the file will get written in to the Bucket.

3.2.2 Configuring the Export Process

This section describes the setup needed to enable export processing, including establishing communication between the Oracle Revenue Management and Billing Cloud Service and Object Storage, configuring batch parameters, and testing the export process. This section contains the following topics:

- Setting Up Communication Between Cloud Service and Object Storage
- Configuring File Export Batch Parameters
- Testing the Export Process

3.2.2.1 Setting Up Communication Between Cloud Service and Object Storage

The export process requires authentication and communication between the Oracle Revenue Management and Billing Cloud Service and Object Storage. See Object Storage Connection Management for more information about setting up this communication and authentication.

3.2.2.2 Configuring File Export Batch Parameters

The next step is to configure the following parameters of the file export batch control created earlier:

- **File Name**: the name of the exported file
- **File Path**: the path to file location in Object Storage. The format for the path is as follows:

  file-storage://<File Location>/<Bucket>

  where:

  - **<File-Location>**: The File Storage Configuration extendable lookup value defined for that file. This will include the compartment identification.
  - **<Bucket>**: The object storage bucket in the compartment that is defined as part of the File Storage Configuration extendable lookup value.

    For example, the "File-Export" bucket in a compartment that is referenced in the "AB-Export" File Storage Configuration extendable lookup value can be referenced as:

    "file-storage://AB-Export/File-Export"

- **Other Parameters**: The batch control supports other optional parameters, including:
  - **XML Root Name**: used to declare the name of root-node in generated xml (when exporting files in xml format)
  - **File Delimiter**: used to define the delimiter used in delimited files See the batch control for information about other parameters.

3.2.2.3 Testing the Export Process

The last step is to test the export process. To test the export process:

1. Run the file export batch process.
2. Navigate to the Batch Run Tree portal to verify the job has run successfully.
3. Navigate to targeted bucket inside Object Storage and verify the exported file.

3.3 File Import Sample Implementation

The following steps summarize how to implement a new file import background process:

- Uploading File to Oracle Cloud Object Storage
- Creating a File Import Batch Process
- Configuring the Import Process

For more information on how to use Plug-in Driven background processing for import and upload, see the **Uploading Records** section of Plug-in Driven Background Processes in the *Oracle Utilities Application Framework Administrative Guide*.

3.3.1 Uploading File to Oracle Cloud Object Storage

Once you have a sample file in the correct format, you need to upload the file to the location in Object Storage from where you plan to upload and import your data.

3.3.2 Creating a File Import Batch Process

Input/output file path soft parameter is required with extendable lookup and Bucket information while running batch.

Below batches have the impact of CFS environment and need the parameter accordingly.

Soft parameter value needs to be provided in Extendable Lookup/Bucket format.

Batches:

1. C1-PUPSG
2. C1-APACK
3. C1-RECUP

Example: C1-PUPSG

Soft parameter File Directory and Archive Directory needs to be passed with extendable lookup and Bucket information.

Input file should be placed into the bucket before triggering the batch job submission.

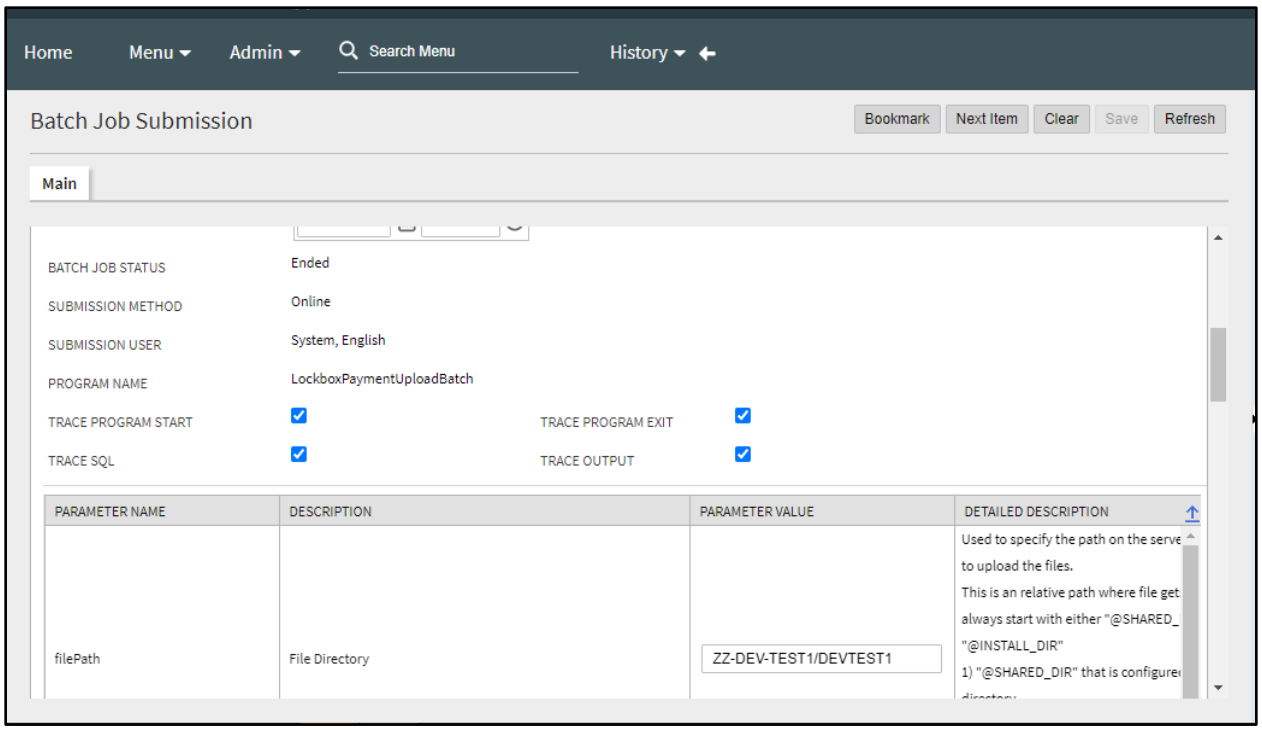

Figure 8: File Import Batch Process – Part 1

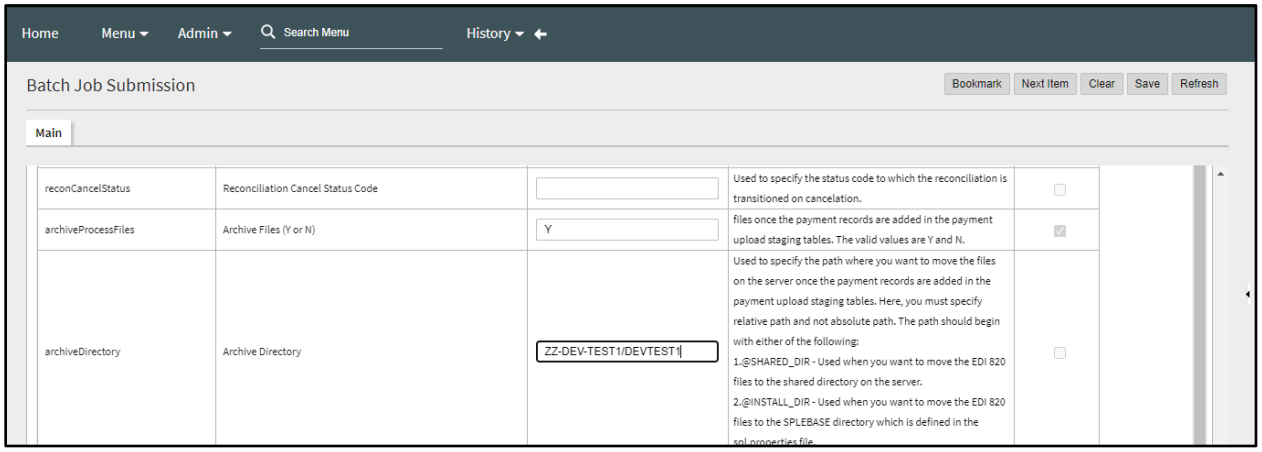

Figure 9: File Import Batch Process – Part 2

3.3.3 Configuring the Import Process

This section describes the setup needed to enable file import processing, including establishing communication between the Oracle Revenue Management and Billing Cloud Service and Object Storage, configuring batch parameters, and testing the process.

The following steps will enable file import batch processing to run and import the data from the import file to the appropriate application tables in the Oracle Revenue Management and Billing Cloud Service:

1. Setting Up Communication Between Cloud Service and Object Storage
2. Configuring File Import Batch Controls
3. Testing the Import Process

3.3.3.1 Setting Up Communication Between Cloud Service and Object Storage

The export process requires authentication and communication between the Oracle Revenue Management and Billing Cloud Service and Object Storage. See Object Storage Connection Management for more information about setting up this communication and authentication.

3.3.3.2 Configuring File Import Batch Controls

The next step is to configure the following parameters of the file import batch control created earlier:

- **File Name**: The name of the file to import.
- **File Path**: the path to the file location in Object Storage where import files will be located. The format for the path is as follows:

  file-storage://<File Location>/<Bucket>

  where:

  - **<File-Location>**: The File Storage Configuration extendable lookup value defined for that file location. This will include the compartment identification.
  - **<Bucket>**: The object storage bucket in the compartment that is defined as part of the File Storage Configuration extendable lookup value.

  For example, the "File-Import" bucket in a compartment that is referenced in the "INT-UPLOAD" File Storage Configuration extendable lookup value can be referenced as:

  file-storage://INT-UPLOAD/File-Import

- See the batch control for information about other parameters.

3.3.3.3 Testing the Import Process

The last step is to test the import process using the sample data.

To test the import process:

1. Run the file import batch process.
2. Navigate to the Batch Run Tree portal and make sure the batch process ran successfully.
3. Check that the records have been added to the FACT table.

3.4 Reporting Module FOP Changes

Referencing the below spl.properties related to FOP has been moved to Feature Configuration. Please refer the below changes related to Feature Configuration to fetch the required Properties.

ouaf.runtime.billView.directoryPath

ouaf.runtime.reportView.directoryPath

ouaf.application.reportingDir

Feature Config Name: C1-REPORTVW

Other values are samples – change it according to your server:

SPL.properties	Equivalent Feature Configuration
ouaf.runtime.billView.directoryPath	Report Generation (FOP) File Path
ouaf.runtime.reportView.directoryPath	Report Generation (FOP) File Path
ouaf.application.reportingDir	Report Generation XML Data File Path

Sample for CFS Environment

Report Download HTTP URL : https://ipaddress:port/ouaf/reportView

Report Generation (FOP) File Path :file-storage://Extendable_lookup/Container

Report Generation XSL Template File Path : file-storage://Extendable_lookup/Container

Report Generation XML Data File Path : file-storage://Extendable_lookup/Container

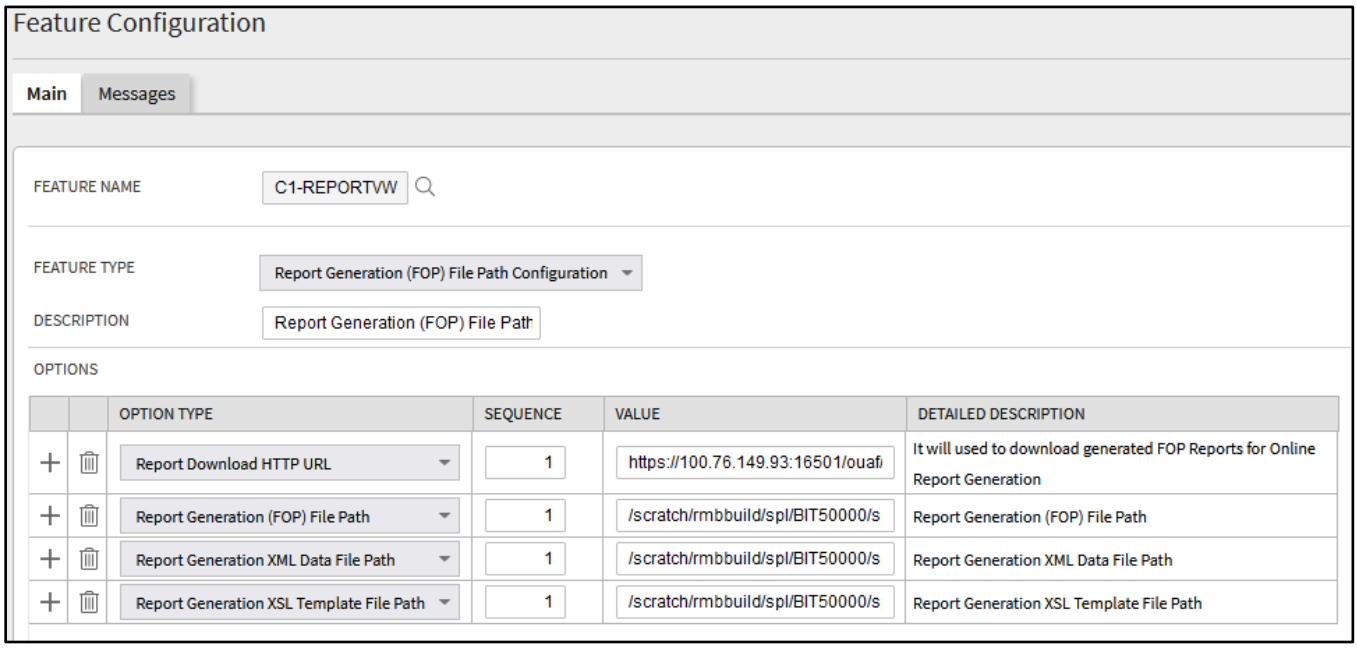

Figure 10: C1-REPORTVW Feature Configuration

3.5 File Upload Interface Changes

1. **spl.properties to Feature Configuration changes**

   Referencing the below spl.properties related to FRT has been moved to Feature Configuration. Please refer the below changes related to Feature Configuration to fetch the required Properties.

   com.oracle.ouaf.system.keystore.file

   com.oracle.ouaf.system.keystore.passwordFileName

   com.oracle.ouaf.system.keystore.type

   com.oracle.ouaf.system.keystore.padding

   com.oracle.ouaf.system.keystore.mode

com.oracle.ouaf.system.keystore.alias

ouaf.system.fileupload.pgp.pvt.key.filepath

Feature Config Name: C1-FLUPLD

SPL.properties	Equivalent Feature Configuration
com.oracle.ouaf.system.keystore.file	Key Store File Path
com.oracle.ouaf.system.keystore.passwordFileName	Key Store Password File Path
com.oracle.ouaf.system.keystore.type	Key Store Type
com.oracle.ouaf.system.keystore.padding	Key Store Padding
com.oracle.ouaf.system.keystore.mode	Key Store Mode
com.oracle.ouaf.system.keystore.alias	File Decryption Key Alias
ouaf.system.fileupload.pgp.pvt.key.filepath	Private Key File Path

2. **C1-FTRAN Batch Parameters to Feature Configuration changes**

   Referencing the below C1-FTRAN Batch Parameters has been moved to Feature Configuration. Please refer the below changes related to Feature Configuration to fetch the required Properties.

   filePath

   errorLogFilePath - For this Batch parameter the error log file path is derived from "filePath" parameter based on "File Object Storage Usage" feature config value. If the feature config value is 'N', Then the error log will be written in filePath/'error'/request_type_name path.

   If the feature config value is 'Y', Then the error log will be written in filePath loctaion itself with the file name as error_ 'request_type_name'_'file_name'.error.

   Feature Config Name: C1-FLUPLD

Batch Parameter Name	Equivalent Feature Configuration
filePath	File Upload Path
errorLogFilePath	Not Required (Auto derived)

3. **File Upload Interface Master Configuration to Feature Configuration changes**

   Referencing the below File Upload Interface Master Configuration has been moved to Feature Configuration. Please refer the below changes related to Feature Configuration to fetch the required Properties.

   Archive File Location

   For this Configuration the processed file path is derived based on "File Object Storage Usage" feature config value. If the feature config value is 'N', Then the processed file will be archived in 'Archive File Location'/'completed'/request_type_name path.

   If the feature config value is 'Y', Then the processed file will be archived in "Archive File Location" itself with the file name as completed_ 'request_type_name'_'file_name'.

Archive Error File Location

For this Configuration the error file path is derived from "Archive File Location" based on "File Object Storage Usage" feature config value.

If the feature config value is 'N', Then the error file will be archived in 'Archive Fileocation'/'error'/request_type_name path.

If the feature config value is 'Y', Then the error file will be archived in "Archive File Location" itself with the file name as error_ 'request_type_name'_'file_name'.

Feature Config Name: C1-FLUPLD

Master Configuration Name	Equivalent Feature Configuration
Archive File Location	File Upload Archive Path
Archive Error File Location	Not Required (Auto derived)

4. **In File Management System Screen 'File Path' field value to Feature Configuration changes**

   Referencing the below File Management System field has been moved to Feature Configuration. Please refer the below changes related to Feature Configuration to fetch the required Properties.

   File Path

   This field value is based on "File Object Storage Usage" feature config value.

   If the feature config value is 'N', Then It will work in the Existing behavior.

   If the feature config value is 'Y', Then the File Path value will be taken from Feature Configuration 'File Upload Path'.

   Feature Config Name: C1-FLUPLD

Batch Parameter Name	Equivalent Feature Configuration
File Path	File Upload Path (if 'File Object Storage Usage' is 'Y')

5. **File Management System Master Configuration 'Upload File Directory' to Feature Configuration changes**

   Referencing the below File Management System Master Configuration has been moved to Feature Configuration. Please refer the below changes related to Feature Configuration to fetch the required Properties.

   Upload File Directory

   This configuration value is based on "File Object Storage Usage" feature config value.

   If the feature config value is 'N', Then It will work in the Existing behavior.

   If the feature config value is 'Y', Then the Upload File Directory Path value will be taken from Feature Configuration 'File Management Path'.

Feature Config Name: C1-FLUPLD

Batch Parameter Name	Equivalent Feature Configuration
File Path	File Management Path (if 'File Object Storage Usage' is 'Y')

6. **Sample for CFS Environment (Values are samples – change it according to your server)**

Key Store File Path : file-storage://Extendable_lookup/Container

Key Store Password File Path : file-storage://Extendable_lookup/Container

Key Store Type : PKCS12

Key Store Padding : PKCS5Padding

Key Store Mode : CBC

File Decryption Key Alias : ouaf.system

Private Key File Path : file-storage://Extendable_lookup/Container

File Upload Path : file-storage://Extendable_lookup/Container

File Upload Archive Path : file-storage://Extendable_lookup/Container

File Management Path : file-storage://Extendable_lookup/Container

CFS Environment:

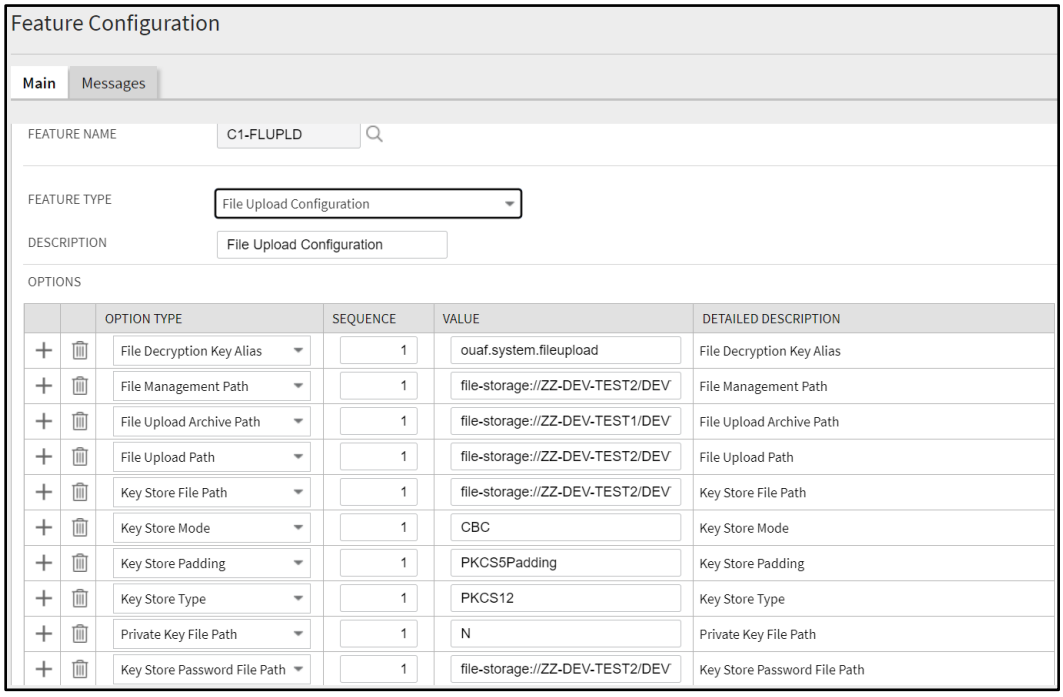

Figure 11: File Upload Configuration

4. Oracle REST Data Services

This section describes how to use Oracle REST Data Services with Oracle Revenue Management and Billing Cloud Service. It contains the following topics:

See the *Oracle REST Data Services Documentation* for more information about Oracle REST Data Services.

4.1 SQL Developer Web

Oracle SQL Developer Web is a part of Oracle REST Data Services (ORDS) and is the web-based version of Oracle SQL Developer that enables you to connect to an Oracle database and execute queries and scripts, create database objects, build data models, and monitor database activity.

Oracle Revenue Management and Billing Cloud Service use Oracle SQL Developer Web to connect to a cloud service database to execute read-only queries on various database schema objects.

Please refer the *Oracle REST Data Services Documentation* for more information about using Oracle SQL Developer Web.

Users must be assigned to the “SQL Developer Web Online User” application role to use SQL Developer Web with Oracle Revenue Management and Billing Cloud Service. See the **Pre-Defined Application Roles** section in the *Oracle Revenue Management and Billing Cloud Service, Premium Edition Administration Guide* for more information about application roles used with Oracle Revenue Management and Billing Cloud Services.

Access is provided to both CISREAD and STGADM database schemas to perform select/read-only queries.

- The CISREAD schema can be used to perform select and read-only queries of the production schema.
- The STGADM schema can be used to perform select and read-only queries of the staging schema (used with data migration and conversion).
- Oracle REST Data Services can access the production and staging schemas in both production and non-production environments.

4.2 REST APIs

Oracle REST Data Services also provides REST APIs that can be invoked via cURL to connect to an Oracle database and perform operations.

Users must be assigned to the "REST Enabled SQL" application role to use REST APIs with Oracle Revenue Management and Billing Cloud Service. See the **Pre-Defined Application Roles** section in the *Oracle Revenue Management and Billing Cloud Service, Premium Edition Administration Guide* for more information about application roles used with Oracle Revenue Management and Billing Cloud Services.

The following is an example syntax for CURL command.

```
curl -i -X POST --user <username>:<password> --data-binary "<SQL
statement>" -H "Content-Type: application/sql" -k <Oracle REST URL>
```

Contact your system administrator for Oracle SQL Developer Web and REST service URLs.

No additional configuration is required to use Oracle SQL Developer Web or REST services.

5. Web Services

This section describes how to use web services with Oracle Revenue Management and Billing Cloud Service. It contains the following topic:

- Web Services in Oracle Revenue Management and Billing Cloud Service

5.1 Web Services in Oracle Revenue Management and Billing Cloud Service

This section describes how to access SOAP and REST web services in the Oracle Revenue Management and Billing Cloud Service. It contains the following topics:

- Inbound Web Services
- Outbound Messages
- Web Service Catalog on Cloud Services
- Web Service Catalog on On-Premise Applications
- User Rights
- Debugging & Tracing Options

5.1.1 Inbound Web Services

In Oracle Revenue Management and Billing Cloud Service, inbound web services do not need to be deployed to be accessible. Once an inbound web service is set to active, it is ready to be used and accessed.

Oracle Revenue Management and Billing Cloud Service support both SOAP and REST services. Implementations can create custom inbound web services, but no XSLs can be referenced by the inbound web service.

5.1.1.1 SOAP Inbound Web Services

This section provides information related to SOAP-based inbound web services.

Accessing a Web Service WSDL on Cloud

You can access and view the WSDL (Web Service Definition Language) of a SOAP service through the cloud service application or by executing a curl command. Then WSDL file contains the structure, the schema and security specification for the desired web service.

All WSDLs are secured in Oracle Revenue Management and Billing Cloud Service. The WSDL URL must include some form of authentication to be accessed, such as a basic username and password.

Users cannot access the WSDL by providing the WSDL URL from a browser or from SOAPUI or any web service testing application. They must use the cloud service application or a curl command.

Cloud Service Application:

Use the following procedure to access a web service WSDL using an Oracle Revenue Management and Billing cloud service application:

1. Select **Admin**, then **Integration**, then **Inbound Web Service**, and then **Search** to access the **Inbound Web Service Query** portal.
2. Search for the SOAP service you wish to access (select "SOAP" from the **Web Service Class** drop-down list)
3. Once the SOAP web service is displayed in the **Inbound Web Service** portal, click the **WSDL** link in the **View WSDL** field.

Curl Command:

Use the following procedure to access a web service WSDL using a curl command:

The curl command format:

```
curl -k -X GET https://{host}:{port}/{tenant}/{domain}/{appName}/soap/api/iws/{IWSServiceName}?WSDL -u username:password
```

Where:

https://{host}:{port}/{tenant}/{domain}/{appName}	**Product Application URL** **Example of Oracle Revenue and Billing Management Application URL:** **https://cloudenv:port/ tenant/prod/rmb**
/soap/api/iws/	Fixed text part of the URL
IWSServiceName	SOAP Inbound Web Service Name
username	User name to login to the application
password	Password to login to the application

Getting the Endpoint URL on Cloud

You can obtain the endpoint URL for a SOAP service from the WSDL. Go to the service name part of the WSDL and you get the address location which is the endpoint URL.

The endpoint URL follows this format:

https://{host}:{port}/{tenant}/{domain}/{appName}/soap/api/iws/IWSServiceName

Testing and Using the SOAP Inbound Web Service

There are several resources to use when testing SOAP-based inbound web services.

This is an example of using SOAPUI to test an inbound web service.

1. Access the secured WSDL from the cloud service application or by executing the curl command as described above.
2. Save the WSDL file locally.
3. Create a new SOAP project using the saved local WSDL.
4. Provide the necessary information in the request message.
5. Provide basic authorization information.
    - Select "Basic" from the **Authorization** drop-down list.
    - Enter appropriate values in the **Username** and **Password** fields.

- Select the **Authenticate pre-emptively** option.

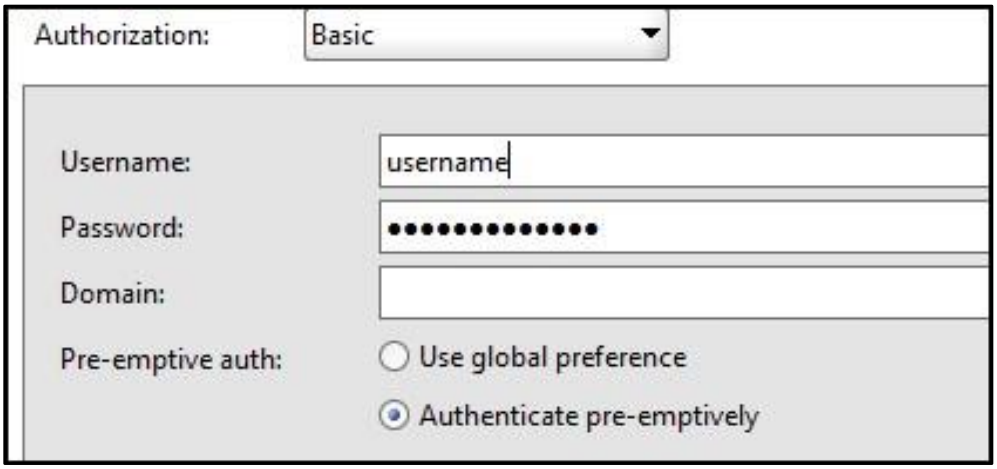

Figure 12: Basic Authorization

6. Test the service as appropriate to your requirements.

5.1.1.2 REST Inbound Web Services

This section provides information related to REST-based inbound web services.

Accessing REST API Specification

REST API specifications define or describe the reference endpoints in an API. You can access and view the API Specification of a REST inbound web service through the Oracle Revenue Management and Billing Cloud Service application or by making a REST call.

Cloud Service Application:

Use the following procedure to access a REST API specification using an Oracle Revenue Management and Billing Cloud Service application:

1. Select **Admin**, then **Integration**, then **Inbound Web Service**, and then **Search** to access the **Inbound Web Service Query** portal.
2. Search for the REST service you wish to access (select "REST" from the **Web Service Class** drop-down list)
3. Once the REST web service is displayed in the **Inbound Web Service** portal, click the **View Specification** link in the **API Specification** field.

REST Call - Cloud Service:

Use the following procedure to access a REST API specification using a REST call with a cloud service:

The endpoint URL to use to make the REST call uses the following format:

- https://{host}:{port}/{tenant}/{domain}/{appName}/rest/openapi/{IWSServiceName}

The endpoint URL to use to make the REST call to get a v2 Swagger Specification follows this format:

- https://{host}:{port}/{tenant}/{domain}/{appName}/rest/openapi/v2/{IWSServiceName}

When making the REST call, use the Get method. See the sample URL below.

Method: GET

URL: https://host:port/tenant/domain/appName/rest/ouaf/openapi/F1-GetIWSWSDL

REST Call - On-Premise Application:

Use the following procedure to access a REST API specification using a REST call with a on-premises application:

The endpoint URL to use to make the REST call to get an Open API Specification follows this format:

- https://{host}:{port}/{context}/rest/ouaf/openapi/{IWSServiceName}

The endpoint URL to use to make the REST call to get a v2 Swagger Specification follows this format:

- https://{host}:{port}/{context}/rest/ouaf/openapi/v2/{IWSServiceName}

When making the REST call, use the Get method. See the sample URL below.

Method: GET

URL: https://host:port/ouaf/rest/ouaf/openapi/v2/F1-GetIWSWSDL

Getting the Endpoint URL

A REST inbound web service can have multiple reference endpoints to access the resource and multiple methods for each endpoint. An endpoint URL points to a unique inbound web service name and an Operation name and shows the whole path to the resource.

The REST endpoint URL is obtained from the API Specification. It is a combination of the Computed URL and the URI Component.

The endpoint URL follows this format:

- {ComputedURL}/{iwsOperationURLComponent}

Where:

Parameter	Description
ComputedURL	Base URL referring to the common path for the API. The value is obtained from the REST API Specification of the REST inbound web service.
iwsOperationURLComponent	Reference a URI Component. The value is obtained from the REST API Specification - Method and URI collection or in the URI Component of the Inbound Web Service - Operations Collection.

Example:

The w1WorkActivity REST inbound web service API Specification has two resource URI Components:

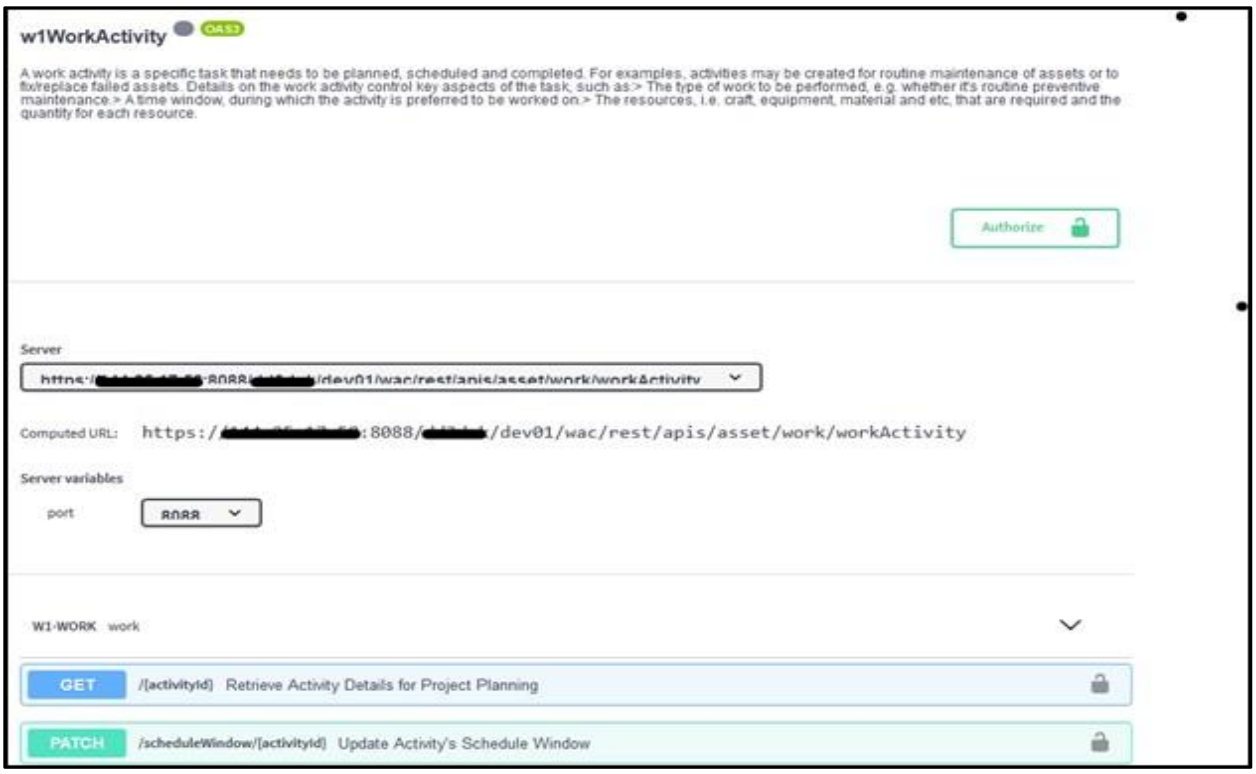

Figure 13: Work Activity

The endpoint URLs for this service are:

- https://{hostname}:{port}/{tenant}/dev01/rmb/rest/apis/asset/work/workActivity/{activityId}
- https://{hostname}:{port}/{tenant}/dev01/rmb/rest/apis/asset/work/workActivity/scheduleWindow/{activityId}

About the Computed URL

The Computed URL is the base URL used by the endpoints. A part of the URL is populated in a System Variable called F1_REST_BASE_URL. The F1_REST_BASE_URL system variable would only contain the configuration up to the 'apis' portion of the URL, and the remaining components of the URL would be derived by the application. The system variable supports the difference between formats used with on-premises applications and cloud services.

The Computed URL follows the format below:

For Cloud Applications:

- https://{host}:{port}/{tenant}/{domain}/{appName}/rest/apis/{ownerURLComponent}/{resourceCategoryURLComponent}/{iwsURLComponent}

For On-Premise Applications:

- https://{host}:{port}/{context}/rest/apis/{ownerURLComponent}/{resourceCategoryURLComponent}/{ iwsURLComponent}

Where:

String	Description
https://{host}:{port}/{tenant}/{domain}/{appName}	Cloud Product Application URL Example of Oracle Revenue Management and Billing Cloud Service Application URL: https://cloudenv:port/ tenant/prod/rmb
https://{host}:{port}/{context}	On-Premise Application URL Example of Oracle Revenue Management and Billing Cloud Service Application URL: https://hostname:port/ouaf
/rest/apis	Fixed text part of the URL
ownerURLComponent	The value is obtained from URI COMPONENT in Owner Configuration for REST services extendable lookup (F1-RESTOwnerURLComponent) where there is an entry for each owner flag. **Examples:** F1 (Framework): /common C1 (Customer): /customer
resourceCategoryURLComponent	The value is obtained from URI COMPONENT in Resource Category for REST services extendable lookup (F1-RESTResourceCategory) where there is an entry for each resource category. **Example:** F1-SYSTEM (System): /system

Note: The Computed URL for cloud applications may not always be the same as the application URL. Always get the value from the API Specification of the REST Service.

Methods and Parameters

REST inbound web services support the following HTTP Methods:

- Get
- Patch
- Post
- Put

The method is defined in the **HTTP Method** drop-down list in the **Operations** section of the **Inbound Web Service** portal.

Figure 14: Operations Section

For the Get, Put and Patch HTTP Methods, "Query" or "Path" parameters can be passed with the endpoint. This is specified using the **Parameter Type** drop-down list in the **Parameters** columns in the **Operations** section of the **Inbound Web Service** portal.

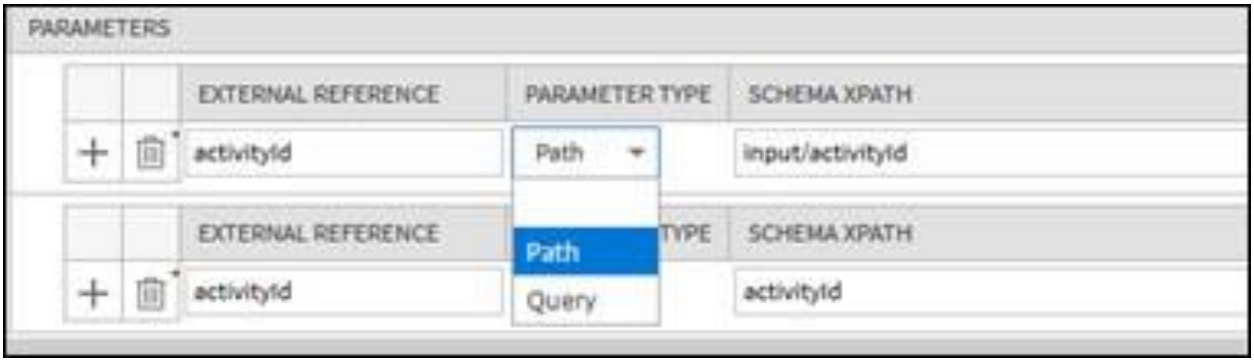

Figure 15: Parameters Section

- **Path parameters**: These are parameters that are part of the endpoint and are required. Usually, they are represented in the endpoint with curly braces.
- **Query parameters**: These parameters are often optional. They are not part of the endpoint but rather are included in the endpoint URL after a question mark, followed by name value pairs.

Example:

'../getAccountBills/{accountId}?startDate=20190101&endDate=20190630'.

Testing and Using the REST Inbound Web Service

There are several resources to use when testing REST inbound web services.

This is an example of using POSTMAN to test an inbound web service.

1. Go to the REST API Specification to get the endpoint URL and the HTTP Method for the web service you wish to test.
2. Open POSTMAN and populate the necessary information.
    a. Provide the HTTP Methods and the Request URL
    b. On the **Authorization** tab, choose "Basic Auth Type" and provide the user name and password
    c. On the **Header** tab, provide the following key value pair to send and accept json messages.

Key	Value	Description
Accept	application/json	Indicate the response media type that is acceptable
Content-Type	application/json	Indicate the media type of the resource. To send and accept xml messages change the value to "application/ xml".

d. On the **Body** tab, provide the request message. Sample request and response messages can also be found on the REST API Specification by expanding each resource URI.

5.1.2 Outbound Messages

This section describes the setup of components used to send outbound messages by invoking the SOAP or REST service of the target application. The information below assumes outbound message types have been created for each web service.

Make sure the target environment you are accessing is allowlisted in the Oracle Revenue Management and Billing Entitlement. This is requested by opening a service request for the Oracle Revenue Management and Billing Cloud Service Operations team.

5.1.2.1 Using SOAP Services

This section contains the following topics:

- Setup Message Senders
- Setup the External System

Setup Message Senders

Invoking a SOAP-based service involves creating one or more new real-time Message Senders.

Use the following procedure to create a Message Sender:

1. From the **Admin** menu, select **Integration**, then **Message Sender**, then **Add**.
2. Enter a unique code and **Description** for the Message Sender.
3. Populate the following values:
    - **Invocation Type:** Real-time
    - **Message Class:** SOAPSNDR (Sender for real-time HTTP/SOAP messages)
    - **Active:** Select the check box
    - **MSG Encoding:** UTF-8 message encoding

4. Select the **Context** tab and set the values for the following context types:

Context Type	Context Value (Sample Values)	Description
HTTP Header	soapAction="process"	Get the value from the soap:operation in the WSDL **Example:** `<wsdl:operation name="Get_APPT_SVC"> <soap:operation soapAction="process"/>`
HTTP Login User	Username	User ID to access the service
HTTP Login Password	Password	Password to access the service
HTTP Method	POST	-
HTTP Timeout	60	-
HTTP Transport Method	SendReceive	-
Message Namespace URI	http://oracle.com/GetApptMessage	Get the value from the schema namespace in the WSDL. This entry should be defined when the External System - Outbound Message Type - Namespace Option is set to Configured on the Message Sender.

HTTP URL 1	@EXT_PUB@ server:port/servicename	The value should follow the format below:
		`@EXT_PUB@server:port/servicename`
		where:
		@EXT_PUB@ refers to the outbound proxy and append the endpoint url without the https:// protocol.
		Get the endpoint url value from the soap:address location in the WSDL.
		Example:
		`<wsdl:service name="receiveApptReq_PortType"> <wsdl:port name="receiveApptReq_PortType_pt" binding="receiveApptReq_PortType_binding"> <soap:address`

Context Type	Context Value (Sample Values)	Description
		`location="https://server:port/servicename"/>`
SOAP Insert Timestamp (Y/N)	Y	This is only needed when the wsdl policy of the service being invoked is wss_username_token_service policy or any policy that require a timestamp.

5. Click **Save**.

Setup the External System

Associate the outbound message types to their corresponding message senders in an external system.

Define the following details for each outbound message type:

- **Outbound Message Type**: The outbound message type created for the outbound message
- **Processing Method**: Real-time
- **Message Sender**: The Message Sender to invoke the SOAP web service
- **Date/Time Format**: XSD
- **Namespace Option**: Configured on Sender

Note: For Message Senders using the SOAPSNDR message class, message XSL do not need to be defined just to add the SOAP Envelope. The SOAPSNDR message class will add the SOAP Envelope before sending the message out.

5.1.2.2 Using REST Services

This section contains the following topics:

Setup Message Senders

Invoking a REST-based service involves creating one or more new real-time Message Senders.

Use the following procedure to create a Message Sender:

1. From the **Admin** menu, select **Integration**, then **Message Sender**, then **Add**.
2. Enter a unique code and **Description** for the Message Sender.
3. Populate the following values:
    - Invocation Type: Real-time
    - Message Class: RTJSONSNDR (Sender for real-time JSON messages)
    - Active: Select the check box
    - MSG Encoding: UTF-8 message encoding

4. Select the **Context** tab and set the values for the following context types:

Context Type	Context Value (Sample Values)	Description
HTTP Login User	Username	User ID to access the service
HTTP Login Password	Password	Password to access the service
HTTP Method	POST	-
HTTP Timeout	60	-
HTTP URL 1	@EXT_PUB@RESTEndpointURL	The value should follow the format below: @EXT_PUB@RESTEndpointURL Where: @EXT_PUB@ refers to the outbound proxy and append the REST endpoint url without the https:// protocol.

5. Click **Save**.

Setup the External System

Associate the outbound message types to their corresponding message senders in an external system.

Define the following details for each outbound message type:

- **Outbound Message Type**: The outbound message type created for the outbound message.
- **Processing Method**: Real-time
- **Message Sender**: The Message Sender to invoke the REST web service
- **Date/Time Format**: XSD
- **JSON Conversion Method**: Base JSON Conversion

Note: For Message Senders using the SOAPSNDR message class, message XSL do not need to be defined. SOAPSNDR message class will add the SOAP Envelope before sending the message out.

5.1.3 Web Service Catalog on Cloud Services

For Oracle Integration Cloud to retrieve the REST catalog or SOAP catalog from Oracle Revenue Management and Billing cloud service applications, use the following endpoint URLs.

The endpoint URL for the REST Catalog Service can be obtained by following this format:

- https://{host}:{port}/{tenant}/{domain}/{appName}/rest/openapi/ iws/catalog

The endpoint URL for the SOAP Catalog Service can be obtained by following this format:

- https://{host}:{port}/{tenant}/{domain}/{appName}/soap/api/iws/ ServiceCatalog

5.1.4 Web Service Catalog on On-Premise Applications

For Oracle Integration Cloud to retrieve the REST catalog or SOAP catalog from Oracle Revenue Management and Billing on-premise applications, use the following endpoint URLs. The endpoint URL for the REST Catalog Service can be obtained by following this format:

- https://{host}:{port}/{context}/rest/ouaf/openapi/iws/catalog

The endpoint URL for the SOAP Catalog Service can be obtained by following this format:

- https://{host}:{port}/{context}/webservices/builtin/ServiceCatalog?WSDL

5.1.5 User Rights

Web service calls must be authorized for the calling user. In other words, the user must exist as an OUAF User with adequate application services for the underlying services called by the inbound web service, and the debug services. The debug Application Services are F1DEBUG to enable a URL with the debug=true setting, and F1USERLOG to view user logs in the online system.

You will probably not want to grant any more access to the inbound web service calling user than they absolutely need, so Oracle recommends creating a separate User Group as needed to support very specific access. Check the User Group and Application Services settings for the user.

For example: if the inbound web service is reading an Account, then the user will need read rights on the Account service (CILCACCP).

5.1.6 Debugging & Tracing Options

This section outlines options for tracing and debugging issues when accessing inbound web services.

5.1.6.1 REST Inbound Web Services

The following debugging and tracing options apply when using REST services:

- REST calls can be made within the application using the **View Specification** link, which will show the curl format of the call, response, etc.
- REST services can be invoked with a debug parameter (http:<cloud url>//restapi?debug=true) to show all the debug logs when checking via the application.
- Using the debug parameter will provide additional information in the kibana logs (tech log information) about execution of each step of the scripts.
- To see the user log, the 'calling user' must log into the online application and view User Logs.
- If the **Trace** flag checked on inbound web service, requests and responses are written to the user log.

5.1.6.2 SOAP Inbound Web Services

The following debugging and tracing options apply when using SOAP services:

- SOAP requests with trace enabled on the inbound web service shows the request message and response message in the user logs, but not in Kibana.
- Using the Debug flag in the soap envelope enables debug mode but does not enable tracing. The debug flag in soap header can be passed as <debug> or <Debug> and values can be true or yes. For example: <debug>true</debug>.
- If the **Trace** flag checked on the inbound web service, requests and responses are written to user log.

