Oracle® Financial Services

Behavior Detection Administration Guide

Release 8.1.2.9.0
G27520-04
February 2025

Oracle Financial Services Behavior Detection Administration Guide, Release 8.1.2.9.0

G27520-04

Copyright © 1994, 2025, Oracle and/or its affiliates.

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

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

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

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

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

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

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

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

7 Managing Batch Processing Utilities

8 Managing Administrative Utilities

E Processing Derived Tables and Fields

F BD Datamap Details

Document Control

This topic lists the changes that have been made to this guide in each release.

Table Revision History

Date	Edition	Description
February 2025	First edition of 8.1.2.9.0	There are no content changes to this guide in this release. The look and feel of the document has been updated.
August 2024	First edition of 8.1.2.8.0	Updated document to capture the OFSBD 8.1.2.8.0 release. In Appendix F, BD Datamap Details, added AccountCustomerRole and AccountToCustomer to the following tables: • AML Brokerage - Pre-Watch List Datamaps • AML Banking - Pre-Watch List Datamaps • Fraud Detection - Pre-Watch List Datamaps • Insurance - Pre-Watch List Datamaps In Appendix G, Datamaps Matrix, added AccountCustomerRole and AccountToCustomer to the BD Datamaps table.
February 2024	First edition of 8.1.2.7.0	Updated document to capture OFSBD 8.1.2.7.0 Release.
October 2023	First edition of 8.1.2.6.0	Updated document to capture OFSBD 8.1.2.6.0 Release. In Appendix F, BD Datamap Details, updated the following datamap tables: • Updated the ML Brokerage - Pre-Watch List Datamaps by changing the execution order for Loan-DailyActivity_RepCurrencyUpd and LoanProfile_LoanProfileStage. • Updated the AML Brokerage - Summary Datamaps to include details for the ExternalEntityDailyProfile and ExternalEntityProfile datamaps.
June 2023	First edition of 8.1.2.5.0	Created document to capture OFSBD 8.1.2.5.0 Release.
March 2023	First edition of 8.1.2.4.0	Created document to capture OFSBD 8.1.2.4.0 Release.

Table (Cont.) Revision History

Date	Edition	Description
December 2022	First edition of 8.1.2.3.0	Created document to capture OFSBD 8.1.2.3.0 Release.
September 2022	First edition of 8.1.2.2.0	Created document to capture OFSBD 8.1.2.2.0 Release.
June 2022	First edition of 8.1.2.1.0	Created document to capture OFSBD 8.1.2.1.0 Release.
March 2022	First edition of 8.1.2.0.0	Created document to capture OFSBD 8.1.2.0.0 Release.

List of Figures

List of Tables

1

About This Guide

This guide explains the concepts behind the Oracle Financial Services Behavior Detection (OFSBD), and provides comprehensive instructions for proper system administration, as well as daily operations and maintenance.

Audience

This Administration Guide is designed for use by the Installers and System Administrators. Their roles and responsibilities, as they operate within OFSBD, include the following:

- Installer: Installs and configures OFSBD at a specific deployment site. The Installer also installs and upgrades any additional Oracle Financial Services solution sets and requires access to deployment-specific configuration information, such as machine names and port numbers).
- System Administrator: Configures, maintains, and adjusts the system, and is usually an employee of a specific Oracle customer. The System Administrator maintains user accounts and roles, monitors data management and Alert Viewer, archives data, loads data feeds, and performs post-processing tasks. In addition, the System Administrator can reload cache.

> **Note:**
>
> Administrators who have access to any of the Financial Crime and Compliance Management (FCCM) modules such as Anti-Money Laundering, Fraud, and so on, will get unrestricted access to the administration utilities that are required to administer the module.

Administrators must have knowledge of UNIX and LINUX.

Scope of this Guide

This guide describes the physical and logical architecture of the OFSBD. It also provides instructions for installing and configuring OFSBD, its subsystem components, and any third-party software required for operation.

OFSBD is powered by advanced data mining algorithms and sophisticated pattern recognition technologies. It provides an open and scalable infrastructure that supports rich, end-to-end functionality across all Oracle Financial Services solution sets. OFSBD's extensible, modular architecture enables a customer to deploy new solution sets readily as the need arises.

This guide provides information about how to administer the following products:

- Anti-Money Laundering (AML)
- Fraud

Your implementation may not include all of these products.

Documentation Accessibility

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

Access to Oracle Support

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

Related Resources

For more information about Oracle Financial Services, refer to the following Behavior Detection application documents:

- Scenario Manager User Guide
- Administration Tools User Guide
- Services Guide
- Data Interface Specification (DIS)
- BD Configuration Guide
- BD Installation Guide
- KYC Administration Guide

Additionally, you may find pertinent information in the OFSAAI documentation:

- Oracle Financial Services Analytical Applications Infrastructure User Guide
- Oracle Financial Services Analytical Applications Infrastructure Installation and Configuration

For installation and configuration information about Sun Java System, BEA, and Apache software, refer to the appropriate documentation that is available on the associated websites.

Conventions

The following text conventions are used in this document.

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

Abbreviations Used in this Guide

This table lists the abbreviations used in this guide and their associated descriptions.

Table 1-1 Abbreviations Used in this Guide

Abbreviation	Description
OFSBD	Oracle Financial Services Behavior Detection
AML	Anti-Money Laundering
AAI	Analytical Applications Infrastructure
CSA	Common Staging Area
FSDM	Financial Services Data Model
BD	Behavior Detection
OFS	Oracle Financial Services
KYC	Know Your Customer
FATCA	Foreign Account Tax Compliance Act
DQ	Data Quality
DT	Data Transformation

2

About Oracle Financial Services Behavior Detection (OFSBD)

This chapter provides a brief overview of the Oracle Financial Services Behavior Detection (OFSBD) in terms of its architecture and operations.

This chapter focuses on the following topics:

- Behavior Detection Architecture
- Operations
- Utilities

2.1 Behavior Detection Architecture

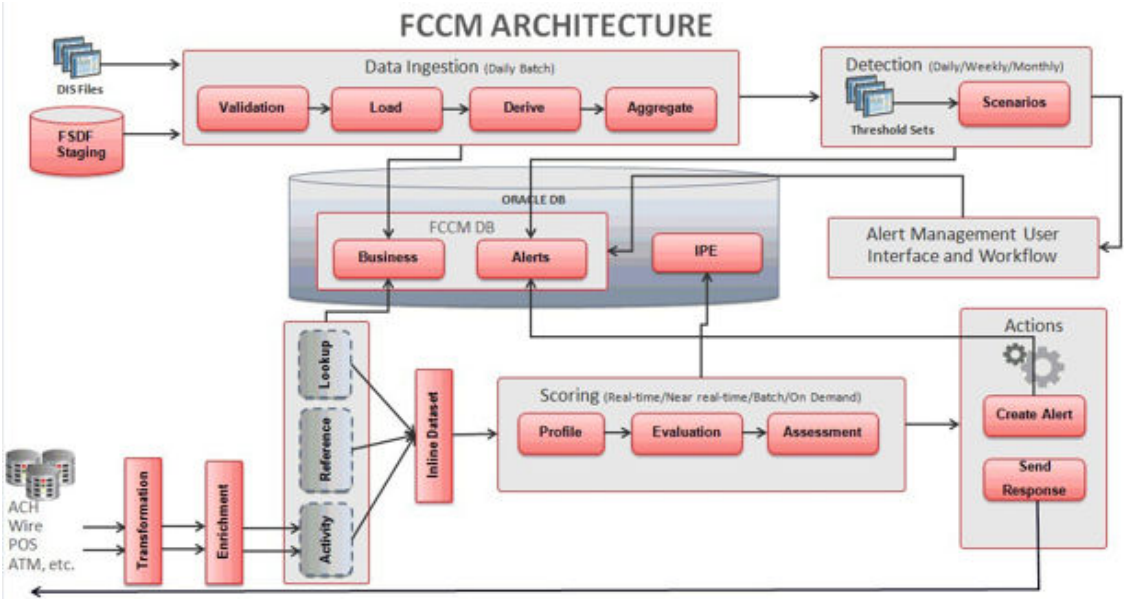
An architecture is a blueprint of all the parts that together define the system: its structure, interfaces, and communication mechanisms. A set of functional views can describe an architecture.

The following views illustrate the implementation details of the architecture:

- **Tiers:** Illustrates system components and their dependencies.
- **Deployment View:** Illustrates the deployment of components to processing nodes.
- **Security View:** Emphasizes the security options between processing nodes through a specialized deployment view.

The following sections describe these views.

Figure 2-1 OFSBD Architecture

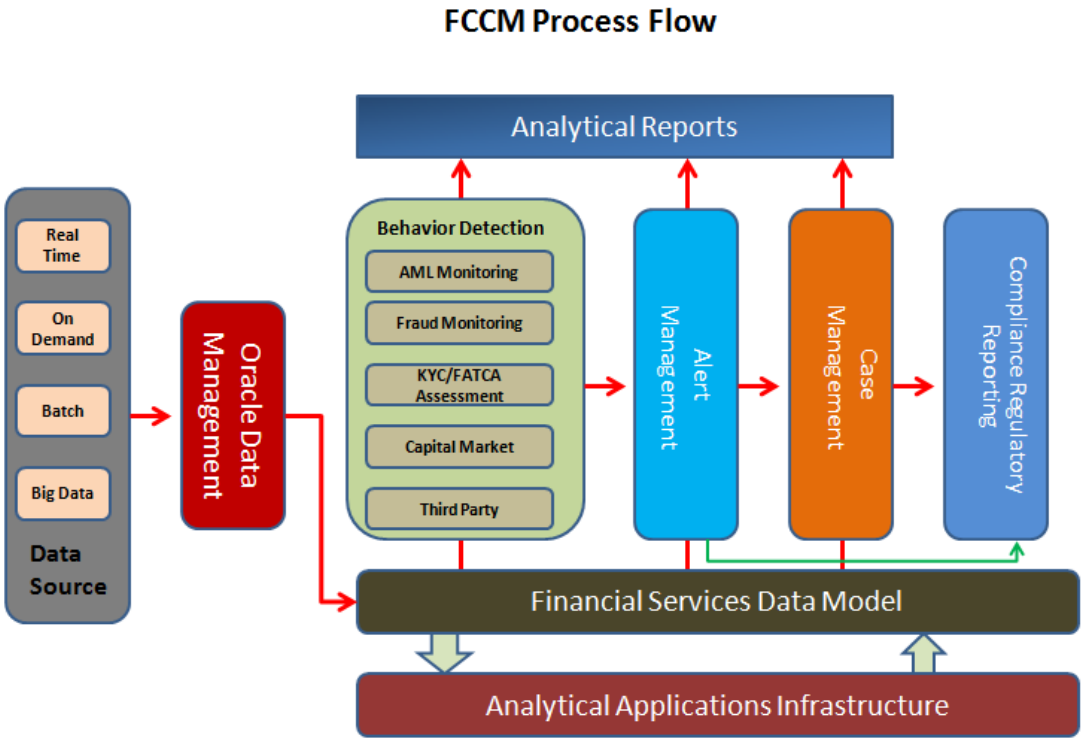


The architecture is composed of a series of tiers and components. Each tier can include one or more components that are divided into small installable units. A solution set requires installation of the associated components.

Tiers

Tiers represent a product or logical grouping of products under which there may be common components and subsystems. The following image is a graphical representation of the tiers:

Figure 2-2 Tiers



The following are the tiers:

- **Oracle Financial Services Analytical Applications Infrastructure (OFSAAI):** Oracle Financial Services Analytical Applications Infrastructure is the complete end-to-end Business Intelligence solution that allows you to tap your organization's vast store of operational data to track and respond to business trends. It also facilitates analysis of the processed data. Using OFSAAI, you can query and analyze data that is complete, correct, and consistently stored at a single place. It can filter data that you are viewing and using for analysis.
- **Oracle Financial Services Behavior Detection Applications:** Oracle solutions, such as Anti-Money Laundering, Fraud Detection, Alert Purge, Currency Transaction Reporting, and FATCA Management, extend the Oracle Financial Services Behavior Detection Applications pack. Each adds domain-specific content to provide the required services for addressing a specific business problem. It includes reusable domain artifacts such as scenarios, input data transformation code, and profiling scripts. A solution set also provides the required presentation packages and custom application objects for supporting user-interface functionality specific to the business domain.

- **Oracle Financial Services Behavior Detection (OFSBD):** Oracle Financial Services Behavior Detection (OFSBD) contains the following subsystems:
  - **Data Management:** Provides data preparation logical functions, which include adapters for files and messages. The functions also include datamap XML for data derivations and aggregations.
    The Oracle Financial Services Ingestion Manager receives, transforms, and loads Market data, Business data (such as, Transactions or Orders and Trades), and Reference data (such as Account and Customer and Employee information) that alert detection processing requires. The template for receiving this information is defined in the Data Interface Specification (DIS). The Ingestion Manager typically receives Market data from a real-time Market data feed or file adapter interface, and both Business and Reference data through the file adapter interface. The Data Management subsystem transforms Market, Business, and Reference data to create derived attributes that the detection algorithms require (much of the loaded data is as is). The system extracts and transforms data and subsequently loads the data into the database. After loading the base tables, the Oracle client's job scheduling system invokes processing datamaps to derive and aggregate data. The Data Management component also uses the Fuzzy Name Matcher Utility to compare names found in source data with names in the Watch List.

    The Oracle client implements Ingestion Manager by setting up a batch process that conforms to the general flow that this chapter describes. Typically, the system uses a job scheduling tool such as AAI Batch Scheduler to control batch processing of Ingestion Manager.
  - **Behavior Detection:** Provides data access, behavior detection, and job services, which include Oracle Financial Services Behavior Detection (OFSBD), Financial Services Data Model (FSDM), and scenarios specific to a particular solution set. OFSBD uses sophisticated pattern recognition techniques to identify behaviors of interest, or scenarios, that are indicative of potentially interesting behavior. A pattern is a specific set of detection logic and match generation criteria for a particular type of behavior. These behaviors can take multiple representations in a firm’s data.

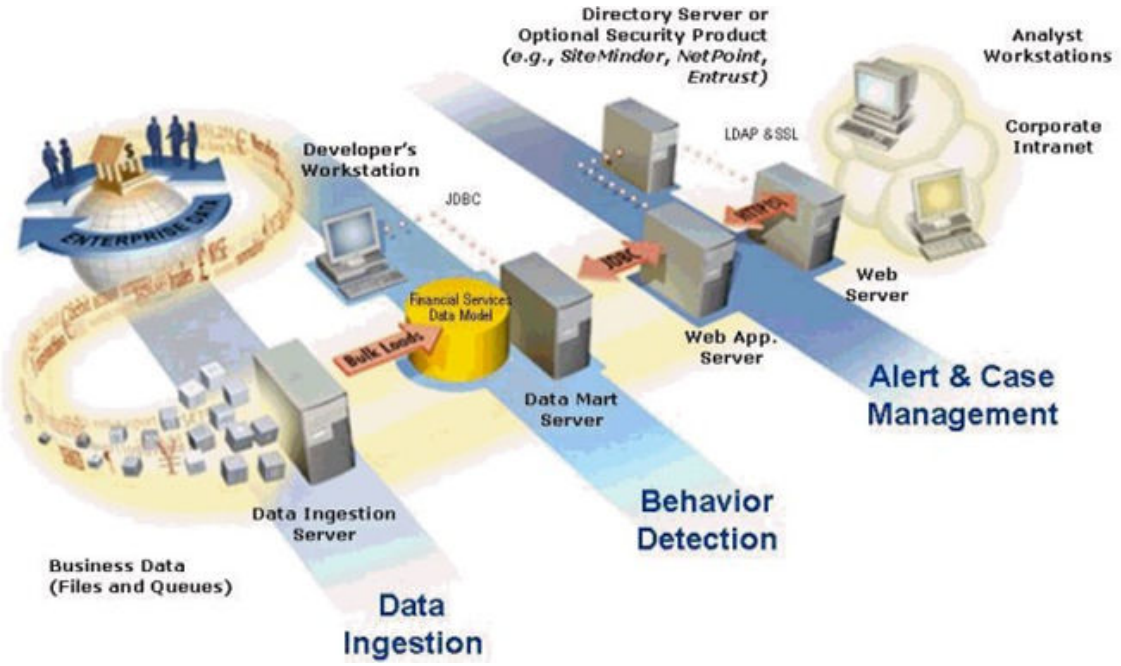
    OFSBD detection modules are divided into scenarios that typify specific types of business problems or activities of interest. The scenarios are grouped into scenario classes that represent categories of behaviors or situations that have common underlying characteristics. The scenario class dictates the action choices available and the data that is displayed when an alert is created.
  - **Alert Viewer:** Provides a user interface and workflow for managing alerts, reporting, and searching business data.
    An alert represents a unit of work that is the result of the detection of potentially suspicious behavior by Oracle Scenarios. OFSBD routinely generates alerts as determined by the configuration of the application in your environment, typically nightly, weekly, monthly, and quarterly. Alerts can be automatically assigned to an individual or group of users and can be reassigned by a user. Alert Viewer contains the Alert Viewer to support triage of an alert, Correlations, and Watch List Management.

  A set of components further divides each OFSBD subsystem. Components are units of a tier that can be installed separately onto a different server. Table 3 outlines the tiers and components. When installed, contents and files related to these components can be located in the folder listed in the Directory Name column. The location and paths to these folders may vary depending on your specific implementation. In some cases, individual deployments can add subsystems to meet a client’s custom requirements.

Deployment View

The OFSBD architecture from the perspective of its deployment illustrates deployment of the major subsystems across servers. Additionally, the deployment view shows the primary communications links and protocols between the processing nodes.

Figure 2-3 OFSBD Architecture - Deployment View



The complex interactions between the components of the Alert Viewer and Enterprise Case Management tiers becomes apparent in the deployment view. The Alert Viewer and Enterprise Case Management tiers require the following:

- Web browser
- Web server
- Web application server

Oracle Financial Services Alert Viewer and Enterprise Case Management tiers use OFSAAI for handling both authentication and authorization. The Alert & Case Management subsystem also supports the use of an External Authentication Management (EAM) tool to perform user authentication at the web server, if a customer requires it. OFSBD components can operate when deployed on a single computer or when distributed across multiple computers. In addition to being horizontally scalable, OFSBD is vertically scalable in that replication of each of the components can occur across multiple servers.

Security View

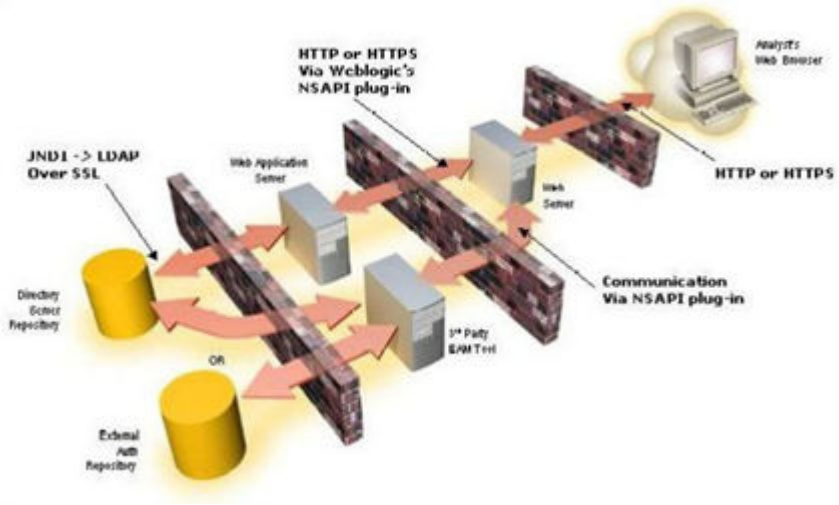
The security view describes the architecture and use of security features of the network in a Behavior Detection architecture deployment. Behavior Detection uses an inbuilt Security Management System (SMS) for its authentication and authorization. The SMS has a set of database tables which store information about user authentication.

Installation of 128-bit encryption support from Microsoft can secure the web browser. Oracle encourages using the Secure Socket Layer (SSL) between the web browser and web server

for login transaction, while the web Application server uses a browser cookie to track a user's session. This cookie is temporary and resides only in browser memory. When the user closes the browser, the system deletes the cookie automatically.

Behavior Detection uses Advanced Encryption Standard (AES) security to encrypt passwords that reside in database tables in the ATOMIC schema on the database server and also encrypts the passwords that reside in configuration files on the server.

Figure 2-4 Security View



The EAM tool is an optional third-party pluggable component of the security view. The tool's integration boundaries provide an Authorization header, form field with principal, or embedded principal to the web Application server through a web server plug-in. The tool also passes the same user IDs that the OFSBD directory server uses.

The following tables outline the tiers and components.

Table 2-1 Data Management Components

Component	Directory Name	Contents
Ingestion Manager	ingestion_manager	Java components, scripts, and stored procedures
Financial Services Data Model	database	Database utilities and database creation scripts
BDF Datamaps	bdf	Datamap XML and configuration parameters.

Table 2-2 Behavior Detection Components

Component	Directory Name	Contents
Behavior Detection	behavior_detection	(Subsystem)
Behavior Detection	bdf	Datamap XML and configuration parameters.
Detection Algorithms	algorithms	C++ behavior detection algorithms

Table 2-2 (Cont.) Behavior Detection Components

Component	Directory Name	Contents
Scenario Manager	toolkit	Job and scenario editors

Table 2-3 Alert Viewer Components

Component	Directory Name	Contents
Alert Viewer Web	solution\am	JSPs used in Alert Viewer
Alert Viewer UI	ftpshare\< alert infodom>\erwin\forms	XMLs for rendering the UI
Web Services	services	Web services for watch list scanning and for the Alert Viewer supervisor (used when posting alerts to Behavior Detection)
Correlation	-	-
Administration Tools	admin_tools	Web-enabled Administration Tools
Watch List Management	-	-

2.2 Utilities

OFSBD database utilities enable you to configure and perform pre-processing and post-processing activities. The following sections describe these utilities.

Batch Utilities

Behavior Detection database utilities enable you to configure and perform batch-related system pre-processing and post-processing activities.

- **Alert Purge Utility**:Provides the capability to remove erroneously generated matches, alerts, and activities.
- **Batch Control Utility**: Manages the start and termination of a batch process (from Data Management to alert post-processing) and enables access to the currently running batch.
- **Calendar Manager Utility**: Updates calendars in the system based on pre-defined business days, holidays, and *days off*, or non-business days.
- **Data Retention Manager:** Provides the capability to manage the processing of partitioned tables in Behavior Detection. This utility purges data from the system based on configurable retention period defined in database.
- **Database Statistics Management**: Manages statistics in the database.
- **Flag Duplicate Alerts Utility**: Enables you to run a script daily after the generation of alerts to identify pairs of alerts that are possible duplicates and adds a system comment to each alert.
- **Refreshing Temporary Tables**: Refreshes temporary tables that the Behavior Detection process uses and estimates statistics for the newly populated tables.
- **Truncate Manager**: Truncates tables that require complete replacement of their data.

Administrative Utilities

Several Behavior Detection database utilities that configure and perform system pre-processing and post-processing activities are not tied to the batch process cycle:

- **Data Analysis Tool:** Assists a Data Miner or Data Analyst in determining how well a customer has populated the Production Data Model.
- **Get Dataset Query with Thresholds Utility:** Enables you to extract dataset SQL complete with substituted thresholds for analysis of the SQL outside of the Behavior Detection application.
- **Scenario Migration Utility:** Extracts scenarios, datasets, networks, and associated metadata from a database to flat files and loads them into another environment.
- **Alert Correlation Rule Migration Utility**:Enables you to move correlation rules and their audit trails from a source environment to a target environment.
- **Investigation Management Configuration Migration Utility**:Enables you to load data related to alerts into the OFSBD.
- **Watch List Services**:Enables you to query the BD watch lists to find a specific or a partial match.
- **Alert Processing Web Services**: Enables you to execute additional processing steps in an existing service operation.
- **Password Manager Utility**:Enables you to change a password for a specific user in a subsystem apart from Alert Viewer and administration tools.
- **Oracle Sequences**: Enables you to update and maintain the Oracle sequences used in OFSBD.

For more information on Administrative Utilities, see Managing Administrative Utilities.

2.3 Operations

As the OFSBD administrator, you coordinate the overall operations of OFSBD: Data Management, Behavior Detection, and Post-Processing.

In a production environment, an Oracle client typically establishes a processing cycle to identify occurrences of behaviors of interest (that is, scenarios) at a specific frequency. Each cycle of OFSBD process begins with Data Management, Behavior Detection, and Post-Processing, which prepares the detection results for presentation for the users. Several factors determine specific scheduling of these processing cycles, including availability of data and the nature of the behavior that the system is to detect. The following sections describe each of the major steps in a typical production processing cycle:

- Start Batch
- Managing Data
- Behavior Detection
- Post-Processing
- End Batch

Start Batch

Using the Batch Control Utility, you can manage the beginning of the OFSBD batch process (see Managing Batch Processing Utilities for more information).

Managing Data

The OFSBD Ingestion Manager controls the Data Management process. The Data Interface Specification (DIS) contains specific definition of the types and format of business data that can be accepted for ingestion. The Ingestion Manager supports files and messages for the ingestion of data. Data Management involves receiving source data from an external data source in one of these forms. The Ingestion Manager validates this data against the DIS, applies required derivations and aggregations, and populates the OFSBD database with the results (see Managing Data for more information).

Behavior Detection

During Behavior Detection, OFSBD Algorithms control the scenario detection process. The Detection Algorithms search for events and behaviors of interest in the ingested data in the FSDM. Upon identification of an event or behavior of interest, the algorithms record a match in the database.

OFSBD executes the following processes in this order to find and record scenario matches:

1. The system populates temporary tables in the database; some scenarios depend on these tables for performance reasons.
2. A network creation process generates and characterizes networks, filtering the links that the system evaluates in the construction of these networks. This is only relevant for certain scenarios.
3. A match is created by executing scenarios. These scenarios are used to detect the behaviors of interest that correspond to patterns or the occurrences of prespecified conditions in business data. The process also records additional data that the analysis of each match may require.

Post-Processing

During post-processing of detection results, Behavior Detection prepares the detection results for presentation to users. Preparation of the results depends upon the following processes:

- Match Scoring: Computes a ranking for scenario matches indicating a degree of risk associated with the detected event or behavior.
- Alert Creation: Packages the scenario matches as units of work (that is, alerts), potentially grouping similar matches together, for disposition by end users. This is applicable when multiple matches with distinct scores are grouped into a single alert.
- Update Alert Financial Data: Records additional data for alerts such as the related Investment Advisor or Security involved in the alert which may be useful for display and analysis.
- Alert Assignment: Determines the user or group of users responsible for handling each alert.
- Auto-Close: Based on configurable rules, closes alerts which are considered to be of lower priority based on attributes of the alert or the alert focus.
- Highlight Generation: Generates highlights for alerts that appear in the alert list in the Alert Viewer subsystem and stores them in the database.
- Historical Data Copy: Identifies the records against which the current batch's scenario runs generated alerts and copies them to archive tables. This allows for the display of a snapshot of information as of the time the alert behavior was detected.
- Alert Notification: Sends e-mail to assignees about the alerts that are assigned to them.

End Batch

The system ends batch processing when processing of data from the Oracle client is complete (see Ending a Batch Process, for more information). The Alert & Case Management subsystem then controls the alert and case management processes. See *Alert Viewer User Guide* for more information.

3

Managing User Administration and Security Configuration

This chapter provides instructions for setting up and configuring the Security Management System (SMS) to support Behavior Detection (BD) applications, user authentication, and authorization.

This chapter focuses on the following topics:

Administrator User Privileges

User administration involves creating and managing users and providing access rights based on their roles. This section discusses the following:

- Administrator permissions
- Creating and mapping users and user groups
- Loading and mapping security attributes

The following lists the access permissions of the Alert Viewer Administrator under BD:

- User Security Administration
- Alert Assigner Editor
- Alert Creator Editor
- Alert Scoring Editor
- Common Web Service
- User Administration
- Security Management System
- Security Attribute Administration
- Manage Common Parameters
- Unified Metadata Manager

> **Note:**
>
> If KYC/FATCA is deployed with BD, the respective Administrator must be mapped with the KYC/FATCA Administrator group, as well for other BD-related access.

User Provisioning Process Flow

The following table lists the various actions and associated descriptions of the user administration process flow:

Table 3-1 User Provisioning Process Flow

Action	Description
Managing User Administration	Create users and map users to user groups. This allows Administrators to provide access, monitor, and administer users.
Adding Security Attributes	Load security attributes. Security attributes are loaded using either Excel or SQL scripts.
Mapping Security Attributes to Organizations and Users	Map security attributes to users. This is done to determine which security attributes control the user's access rights.

Requirements to Access BD Applications

A user gains access to BD applications based on the authentication of a unique user ID and password. To access the BD applications, you must fulfill the following conditions:

Table 3-2 Requirements

Applications	Conditions
Alert Viewer	• Set of privileges that associate functional role with access to specific system functions. • One or more associated organizational affiliations that control the user's access to alerts. • Relationship to one or more scenario groups. • Access to one or more jurisdictions. • Access to one or more business domains.
Watch List Management	• Set of policies that associate functional roles with access to specific system functions. • Access to one or more jurisdictions. • Access to one or more business domains.
Administration Tools	Set of policies that associate the admin functional role with access to specific system functions.

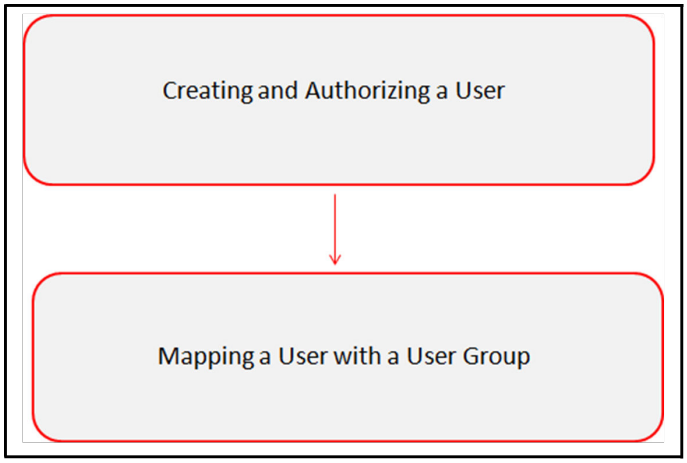
3.1 Managing User Administration

This section allows you to create, map, and authorize users defining a security framework which has the ability to restrict access to the respective BD applications.

Managing Identity and Authorization

This section explains how to create a user and provide access to BD applications. The following figure shows the process flow of identity management and authorization:

Figure 3-1 Managing Identity and Authorization Process Flow



The following table lists the various actions and associated descriptions of the user administration process flow:

Table 3-3 Administration Process Flow

Action	Description
Creating and Authorizing Users and User Groups	Create a user. This involves providing a user name, user designation, and the dates between which the user is active in the system.
Mapping Users with User Groups	Map a user to a user group. This enables the user to have certain privileges that the mapped user group has.

Creating and Authorizing Users and User Groups

The SYSADMN and SYSAUTH roles can be provided to users in the BD application. User and role associations are established using Security Management System (SMS) and are stored in the config schema. User security attribute associations are defined using Security Attribute Administration.

For more information on creating and authorizing a user, see the *Oracle Financial Services Analytical Applications Infrastructure User Guide*.

Mapping Users with User Groups

This section explains how to map Users and User Groups. With this, the user will have access to the privileges as per the role. The SYSADMN user maps a user to a user group in the BD application. The following table describes the predefined Alert Viewer User Roles and corresponding User Groups.

Table 3-4 Alert Viewer (AM) Roles and User Groups

Role	Group Name	User Group Code
Alert Viewer	Alert Viewer User Group	ALERTVIEWERGRP
AM Scenario Group	AM Scenario Group User Group	AMDATAMNRGRP
Alert Viewer Administrator	Mantas Administrator User Group	AMMANADMNGR

The following table describes the KYC and FATCA Case Management User Roles and corresponding User Groups.

Table 3-5 FATCA Case Management Roles and User Groups

Role	Group Name	User Group Code
KYC Relationship Manager	KYC Relationship Manager User Group	CMKYCRMUG
KYC Supervisor	KYC Investigator User Group	CMKYCINVSTGTRUG
KYC Analyst	KYC Analyst User Group	CMKYCANALYSTUG
KYC Administrator	KYC Administrator User Group	KYCADMNGRP
FATCA Supervisor	FATCA Supervisor User Group	FTCASUPERVISRUG
FATCA Analyst	FATCA Analyst User Group	FTCAANALYSTUG
FATCA Auditor	FATCA Auditor User Group	FTCAAUDITORUG
FATCA Administrator	FATCA Admin User Group	FTCAADMINUG

The following table describes the Watch List User Roles and corresponding User Groups.

Table 3-6 Watch List Roles and User Groups

Role	Group Name	User Group Code
Watch List Supervisor	Watchlist Supervisor Group	WLSUPERVISORUG

If you want to change the user group mapping for users who are already mapped to one or more groups, you must deselect the preferences for the Home page if it has been set. To change the preferences, follow these steps:

1. In the Home page, click the user name. A drop-down list appears.
2. Click **Preferences**. The Preferences page appears.
3. Select the appropriate Property Value.
4. Click**Save**.

Users should not be mapped to both the CR Supervisor/Analyst role and IP Manager/Manager Supervisor role. The only acceptable role combinations for a user are the Employee role and one of the following four roles:

- CR Supervisor
- CR Analyst
- IP Manager
- Manager Supervisor

The maximum role combinations should be limited to two. For more information on mapping User with User Groups, see *Oracle Financial Services Analytical Applications Infrastructure User Guide*.

For any customized user group creation and user group-role mapping, see Appendix C, User Administration.

3.2 Adding Security Attributes

This section explains about security attributes, the process of uploading security attributes, and mapping security attributes to users in the BD application.

This section covers the following topics:

- About Security Attributes
- Loading Security Attributes

About Security Attributes

Security Attributes help an organization classify their users based on their geography, jurisdiction, and business domain, in order to restrict access to the data that they can view. You need to map the roles with access privileges, and since these roles are associated with user groups, the users associated with the user groups can perform activities throughout various functional areas in the BD application.

The following sections describe the security attributes:

- **Jurisdiction:** OFSFCCM solutions use Jurisdictions to limit user access to data in the database. Records from the Oracle client that the Ingestion Manager loads must be identified with a jurisdiction and users of the system must be associated with one or more jurisdictions. In the Alert Viewer system, users can view only data or alerts associated with jurisdictions to which they have access. You can use a jurisdiction to divide data in the database. For example:
  - Geographical: Division of data based on geographical boundaries, such as countries, states, and so on.
  - Organizational: Division of data based on different legal entities that compose the client's business.
  - Other: Combination of geographic and organizational definitions. In addition, it is client driven and can be customized.

  In most scenarios, a jurisdiction also implies a threshold that enables use of this data attribute to define separate threshold sets based on jurisdictions. The list of jurisdictions in the system reside in the KDD_JRSDCN table.

  > **Note:**
  >
  > BD application supports up to 1000 jurisdictions.

- **Business Domain:** Business domains are used for data access controls similar to jurisdiction but have a different objective. The business domain can be used to identify records of different business types such as Private Client verses Retail customer, or to provide more granular restrictions to data such as employee data. The list of business domains in the system resides in the KDD_BUS_DMN table. The system tags each data record provided through the Ingestion Manager to one or more business domains. It also associates users with one or more business domains in a similar fashion. If a user has access to any of the business domains that are on a business record, the user can view that record. The business domain field for users and data records is a multi-value field. For example, you define two business domains: Private Client and Retail Banking.
  A record for an account that is considered both has BUS_DMN_SET=ab. If a user can view business domain a or b, the user can view the record. You can use this concept to

protect special classes of data, such as data about executives of the firm. For example, you can define a business domain as e: Executives. You can assign this business domain to the employee, account and customer records that belong to executives. Thus, only specific users of the system have access to these records. If the executive's account is identified in the Private Client business domain as well, any user who can view Private Client data can view the executive's record. Hence, it is important not to apply too many domains to one record.

The system also stores business domains in the KDD_CENTRICITY table to control access to Research against different types of entities. Derived External Entities and Addresses inherit the business domain set that is configured in KDD_CENTRICITY for those focus types.

- **Scenario Group:** Scenario groups are used for data access controls. A scenario group refers to a group of scenarios in the BD applications that identify a set of scenario permissions and to which a user has access rights. The list of scenario groups in the system resides in the KDD_SCNRO_GRP table.
- **Organization:** Organizations are used for data access controls. Organizations are user group to which a user belongs. The list of Organizations in the system resides in the KDD_ORG table.

Loading Security Attributes

This section covers the following topics:

3.2.1 Loading Jurisdictions

(Required) <Enter a short description here.>

To load jurisdictions in the database, follow these steps:

1. Add the appropriate record to the KDD_JRSDCN database table as mentioned in the following table.

Table 3-7 Loading Jurisdictions

Column Name	Description
JRSDCN_CD	Code (one to four characters) that represents a jurisdiction such as N for North, or S for South.
JRSDCN_NM	Name of the jurisdiction such as North or South.
JRSDCN_DSPLY_NM	Display name of the jurisdiction such as North or South.
JRSDCN_DESC_TX	Description of the jurisdiction such as Northern US or Southern US.

The data in the KDD_JRSDCN database table is loaded through the ATOMIC schema.

2. Add records to the table using an SQL script similar to the following sample script.

```
INSERT INTO KDD_JRSDCN (JRSDCN_CD,
JRSDCN_NM,JRSDCN_DSPLY_NM,JRSDCN_DESC_TX)
VALUES ('E', 'East', 'East', 'Eastern')
```

The KDD_JRSDCN table is empty after system initialization and needs to be populated before the system can operate.

3.2.2 Loading Business Domains

To load a business domain, follow these steps:

1. Add the appropriate user record to the KDD_BUS_DMN database table as mentioned in the following table.

Table 3-8 KDD_BUS_DMN Table Attributes

Column Name	Description
BUS_DMN_CD	Single-character code that represents a business domain such as a, b, or c.
BUS_DMN_DESC_TX	Description of the business domain such as Institutional Broker Dealer or Retail Banking.
BUS_DMN_DSPLY_NM	Display name of the business domain , such as INST or RET.
MANTAS_DMN_FL	Flag that indicates whether Oracle Financial Services Behavior Detection specified the business domain (Y). If a BD client specified the business domain, you should set the flag to N.

The KDD_BUS_DMN table already contains predefined business domains for the Oracle client.

2. Add records to the table using an SQL script similar to the following sample script.

```
INSERT INTO KDD_BUS_DMN (BUS_DMN_CD, BUS_DMN_DESC_TX, BUS_DMN_DSPLY_NM,
MANTAS_DMN_FL) VALUES ('a', 'Compliance Employees', 'COMP', 'N');
INSERT INTO KDD_BUS_DMN (BUS_DMN_CD, BUS_DMN_DESC_TX, BUS_DMN_DSPLY_NM,
MANTAS_DMN_FL) VALUES ('b', 'Executives'
'EXEC', 'N');
COMMIT;
```

3. Update the KDD_CENTRICITY table to reflect access to all focuses within the business domain with the following command:

```
update KDD_CENTRICITY set bus_dmn_st = 'a'
where KDD_CENTRICITY. CNTRY_TYPE_CD = 'SC'
```

3.2.3 Loading Scenario Groups

To load a Scenario Group, follow these steps:

1. Add the appropriate value in the KDD_SCNRO_GRP database table as mentioned in the following table.

Table 3-9 KDD_SCNRO_GRP Table Attributes

Column Name	Description
SCNRO_GRP_ID	Scenario group identifier
SCNRO_GRP_NM	Scenario Group Name.

2. Add records to the table using an SQL script similar to the following sample script.

```
INSERT INTO KDD_SCNRO_GRP(SCNRO_GRP_ID,SCNRO_GRP_NM) VALUES (66,'BEX');
INSERT INTO KDD_SCNRO_GRP(SCNRO_GRP_ID,SCNRO_GRP_NM) VALUES (77,'CST');
COMMIT;
```

3.2.4 Loading Scenario Group Memberships

To load a Scenario Group Membership, follow these steps:

1. Add the appropriate value in the KDD_SCNRO_GRP_MEMBERSHIP database table as mentioned in the following table.

Table 3-10 KDD_SCNRO_GRP_MEMBERSHIP Table Attributes

Column Name	Description
SCNRO_ID	Scenario Identifier
SCNRO_GRP_ID	Scenario group identifier
SCNRO_GRP_NM	Scenario Group Name.

2. Add records to the table using an SQL script similar to the following sample script.

```
INSERT INTO KDD_SCNRO_GRP_MEMBERSHIP
(SCNRO_ID,SCNRO_GRP_ID,SCNRO_GRP_NM) VALUES (113000016,66,'BEX') ;
INSERT INTO KDD_SCNRO_GRP_MEMBERSHIP
(SCNRO_ID,SCNRO_GRP_ID,SCNRO_GRP_NM) VALUES (113000016,77,'CST') ;
```

3.2.5 Loading Organizations

To load an organization in the database, follow these steps:

1. Add the appropriate record to the KDD_ORG database table as mentioned in the following table.

Table 3-11 KDD_ORG Table Attributes

Column Name	Description
ORG_CD	Unique identifier for this organization.
ORG_NM	Short name for this organization that is used for display purposes.
ORG_DESC_TX	Description of this organization.
PRNT_ORG_CD	Parent organization of which this organization is considered to be a child. NOTE: This should reference an ORG_CD in the KDD_ORG table..

Table 3-11 (Cont.) KDD_ORG Table Attributes

Column Name	Description
MODFY_DT	Last modified date and time for this organization record.
MODFY_ID	User ID of the user who last modified this organization data. NOTE: This should reference a user in the Investigation Owner table (KDD_REVIEW_OWNER.OWNER_SEQ_ID). You can also set the value to owner_seq_id 1, which is SYSTEM, if another suitable ID is not available.
COMMENT_TX	Additional remarks added by the user.

2. Add records to the table using an SQL script similar to the following sample script.

```
INSERT INTO KDD_ORG
(ORG_CD,ORG_NM,ORG_DESC_TX,PRNT_ORG_CD,MODFY_DT,MODFY_ID,COMMENT_TX)
VALUES ('ORG1','COMPLIANCE ORG','DEPARTMENT FOR INVESTIGATION','ORG1
PARENT ORG','01-JUN-2014',1234,'ADDING KDD_ORG ENTRIES')
```

3.3 Mapping Security Attributes to Organizations and Users

The Mapping Security Attributes to Users functionality/section enables you to determine which security attribute controls a user's access. Using this UI, an Administrator can map both Organizations and Users to different Security attributes.

To map a Security Attribute, follow these steps:

1. Log in as the Alert Viewer Administrator. The OFSAAI Applications page is displayed.
2. Click **Financial Services Money Laundering**.
3. In the Navigation List, select **Behavior Detection**, then select **Administration**. The Anti Money Laundering page is displayed.
4. Mouse over the **Administration** menu, select the **User Administration** sub-menu, and click **Security Attribute Administration**. The Security Attribute Administration page is displayed.
5. Select user type from **Choose User Type** drop-down list. The following options are available:
   - Organization
   - User

   Before proceeding with providing a user access through this UI, ensure that you have created a user and all necessary data is available in the appropriate database tables.
6. Depending on the User Type you have selected, the available options in the **Choose User** drop down list is updated. Select the user from **Choose User** drop-down list. The relevant Security Attribute Administration page is displayed.

> **Note:**
>
> In order to update the user profiles before proceeding with mapping any security attributes, select User from the Choose User Type drop-down list. When selected, all the updates made to all the user profiles through User Maintenance UI are imported from the CSSMS_USR_PROFILE table of the ATOMIC schema to the KDD_REVIEW_OWNER table of the ATOMIC schema. If you delete a user through the Security Management System screen, you should come back to the Security Attribute Administration screen and select the value User from the Choose User Type drop-down list. Then the deleted user will be updated in the KDD_REVIEW_OWNER table against the column actv_flg as N, and that user is inactive.

Table 3-12 Security Attributes

Fields	Description
Organization	Select an organization from the drop-down list. A User or Organization's access to other Organizations depends on the selection(s) made for this organization parameter, such as, if a user is mapped to Org1 and Org2, it implies that this user can access alerts which belong to these two organizations, provided other security attributes are also matching.
Own Alert Flag	Select whether this user type will own a alert flag from the drop-down list. The Own Alert flag is required for taking ownership of the alerts.
Business Organization	The default Business Organization is displayed, but you can select the business organization from the drop-down list.
Jurisdictions	Select the jurisdictions from the drop-down list. Mapping of one or more jurisdictions to a user or organization allows this user or organization to access alerts, watch lists, and watch list members that belong to the mapped jurisdiction. The selected jurisdictions are displayed in Jurisdictions section after you save your selection.
Business Domain	Select the business domains from the drop-down list. Mapping of one or more business domains to a user or organization allows this user or organization to access alerts, watch lists, and watch list members that belong to the mapped business domains. The selected jurisdictions are displayed in Jurisdictions section after you save your selection.
Scenario Group	Select the scenario group from the drop-down list. Mapping of one or more Scenario Groups to a user or organization allows this user or organization to access alerts that belong to the mapped scenario Group. The selected jurisdictions are displayed in Jurisdictions section after you save your selection.

7. Click **Save**. The following confirmation message displays: *Would you like to save this action?*
8. Click **OK**. The following confirmation message displays: *The update operation successful*.
9. Click **OK**. The updated Security Attribute page is displayed.

3.3.1 Removing Security Attributes

This section allows you to delete the mapped security with Users.

To remove security attributes, follow these steps:

1. Navigate to the **Security Attributes** page.
2. Select one or more check boxes in the respective security attributes such as Business Domain, Jurisdictions, and so on.
3. Click **Remove**. The following confirmation message displays: *Are you sure you want to delete this records?*
4. Click **OK**. The selected record is deleted from the list.
5. Click **Save**. The changes are updated.

4

Managing Data

This chapter explains how your raw business data can be loaded into the Oracle Financial Crimes Data Model (FCDM) in various ways.

The following approaches are available either through the OFSDF Common Staging Area Model (CSA) or converting the raw data into Data Interface Specification (DIS) flat files.

This chapter focuses on the following topics:

About Data Management

Data Management consists of Data Processing. Data loaded into the FSDM is processed for data derivation and data aggregation using the BDF processing datamaps. The processing refers to the wide range of activities to include data enrichment and data transformation.

Data Loading and Processing Flow Overview

In BD applications, data is loaded into the FSDM from the following data sources:

- BD Flat File Interface
- Common Staging Area (CSA) in FSDF

Data stored in the FCDM is then processed using BD processing datamaps where additional data derivations and aggregations are stored in the FCDM.

- CSA: The CSA provides a single repository for data storage for multiple functional areas and applications having the Common Staging Area Model and Reporting Data Model. The Common Staging Area Model provides a simplified, unified data sourcing area for inputs required by FCCM using BD.
- Flat Files: The flat files contain data provided by the client. This data is loaded into the Financial Crimes Data Model (FCDM).
- FCDM: The FCDM is a database which consists of well organized business data for analysis. It determines the structured data which stores persistent information in a relational database and is specified in a data modeling language.
- BD Datamaps: The BD datamaps load Business, Market and Reference data required for alert processing. It does the data derivation and aggregation after the BD Ingestion Manager loads the base tables.

4.1 Managing Data Loading

Your raw business data can be loaded into the Oracle Financial Services Data Model (FSDM) in various ways. The following approaches are available either through the OFSDF Common

Staging Area Model (CSA) or converting the raw data into Data Interface Specification (DIS) files.

The following approaches are used to load the data:

- FSDF CSA Data Load
- BD Ingestion Flat File Data Load
- Managing Data Processing

> **Note:**
>
> BDF datamaps should be used for data loading, as the Ingestion Manager is only used for specific datamaps. See Appendix D, Managing Data, for detailed information about which datamaps must be executed using BDF and Ingestion Manager.

FSDF CSA Data Load

This section covers the following topics:

- Using Behavior Detection Datamaps
- AccountProfitAndLoss Datamap
- Multiple Batch Processing

4.1.1 Using Behavior Detection Datamaps

The Behavior Detection (BD) datamap takes the data from the CSA, enhances it, and then loads it into a target database table (FSDM). The Data Interface Specification (DIS) datamaps are used to load client-provided data, either through DIS files as specified in the DIS or through CSA tables.

> **Note:**
>
> All the DIS datamaps in the Behavior Detection Flat File Interface for which staging representation is marked as **Yes** are applicable for CSA loading. For more information, see Behavior Detection Flat File Interface.

To load data in the FSDM using BD, follow these steps:

1. Configure the *DIS.source* parameter to FSDW. For more information on configuring other parameters, see Behavior Detection Flat File Interface.
2. Execute the **Account** datamap which loads data into the Account (ACCT) table using the following sample script: `<OFSAAI Installed Directory>/bdf/scripts/execute.sh Account`

   This step can be repeated for all datamaps for which staging representation is marked as **Yes**.If there are any errors or rejections in loading data, refer to the `<OFSAAI Installed Directory>/bdf/logs` path to know about the errors in the log file.
3. <Enter the next step.>

> **Note:**
>
> If BDF jobs fail intermittently with the following error: *java.sql.SQLException: Stream has already been closed.* You can resolve this issue by following these steps:

a. Navigate to $FIC_HOME/bdf/scripts.

b. Take a backup of the existing execute.sh file.

c. Add the following parameter `Doracle.jdbc.useFetchSizeWithLongColumn=true` after the `$X_ARGS_GEN` in the execute.sh file.

For Example:

```
 $JRE_EXE -server -Xms${MINHEAP}
 - Xmx${MAXHEAP} -classpath
$CLASSPATH $X_ARGS_GEN -
Doracle.jdbc.useFetchSizeWithLongColumn=true
com.ofss.bdf.common.BDFProcessLauncher
$BDF_ROOT $*
```

d. Provide proper permission and execute the datamap.

4.1.2 AccountProfitAndLoss Datamap

Follow these steps to run the AccountProfitAndLoss Datamaps.

1. Stg_Account_Balances cannot have Multiple Legal Entity entries and not supported by the current version of the product.
2. You must populate the data for the following metadata tables:
   - DIM_BALANCE_TYPE
   - STG_BALANCE_TYPE_MASTER
3. Insert Scripts for metadata tables:

```
insert into Stg_Balance_Type_Master (FIC_MIS_DATE, V_BALANCE_TYPE,
V_BALANCE_TYPE_DESC, V_BALANCE_TYPE_NAME, V_DATA_ORIGIN)
values (<to be provided by client>, 'TOTAL_PL_BASE_AM', 'Change in the
total market value of all security positions held by the account expressed
in base currency.', 'Total Profit Loss - Base', <client to populate source
system name>)
/
insert into Stg_Balance_Type_Master (FIC_MIS_DATE, V_BALANCE_TYPE,
V_BALANCE_TYPE_DESC, V_BALANCE_TYPE_NAME, V_DATA_ORIGIN)
values (<to be provided by client>, 'OPTION_PL_BASE_AM', 'Change in the
total market value of all option security positions held by the account
expressed in base currency.', 'Option Profit Loss - Base', <client to
populate source system name>)
/
insert into Dim_Balance_Type (N_BALANCE_TYPE_CD, V_BALANCE_TYPE,
V_DESCRIPTION, V_CREATED_BY, V_LAST_MODIFIED_BY, D_LAST_MODIFIED_DATE,
D_CREATED_DATE, FIC_MIS_DATE, F_LATEST_RECORD_INDICATOR,
D_RECORD_START_DATE, D_RECORD_END_DATE, V_BALANCE_TYPE_NAME, V_DATA_ORIGIN)
```

```
values (1, 'TOTAL_PL_BASE_AM', 'Change in the total market value of all
security positions held by the account expressed in base currency.', null,
null, null, null, null, null, null, null, 'Total Profit Loss - Base',
<client to populate source system name>)
/
insert into Dim_Balance_Type (N_BALANCE_TYPE_CD, V_BALANCE_TYPE,
V_DESCRIPTION, V_CREATED_BY, V_LAST_MODIFIED_BY, D_LAST_MODIFIED_DATE,
D_CREATED_DATE, FIC_MIS_DATE, F_LATEST_RECORD_INDICATOR,
D_RECORD_START_DATE, D_RECORD_END_DATE, V_BALANCE_TYPE_NAME, V_DATA_ORIGIN)
values (2, 'OPTION_PL_BASE_AM', 'Change in the total market value of all
option security positions held by the account expressed in base
currency.', null, null, null, null, null, null, null, null, 'Option Profit
Loss - Base', <client to populate source system name>)
/
```

4.1.3 Multiple Batch Processing

You can segregate WatchLists (WL) so that the multiple countries can be processed in a single instance in parallel. The risk and WatchList information associated to each country's WatchList and WatchList members are maintained.

This section covers the following topics:

- Run BDF Ingestion in Parallel for Multiple Batches
- Configure WatchList Management UI to support multiple batches
- BD Ingestion Flat File Data Load

4.1.3.1 Run BDF Ingestion in Parallel for Multiple Batches

This supports the segregation of WatchList (WL) such that the multiple Countries can be processed in a single instance.

To run the BDF Ingestion in parallel for multiple batches, follow these steps.

1. Start **Mantas Batch**. This can be a default batch (DLY).
2. Set **Mantas Date**
3. Create a copy of *KDD_PRCSNG_BATCH_CONTROL* for each batch you plan to run in parallel. The KDD_PRCSNG_BATCH_CONTROL table will need to be suffixed with `<_first two characters of the batch name>`.

   This has to be done only once, unless a new batch is being added.

   Example: To create copy of KDD_PRCSNG_BATCH_CONTROL for the France (FR) batch, run

   ```
   CREATE TABLE KDD_PRCSNG_BATCH_CONTROL_FR AS SELECT * FROM
   KDD_PRCSNG_BATCH_CONTROL WHERE 1=2;
   ```

4. Insert data for running multiple batches into `"KDD_PRCSNG_BATCH"`, `"KDD_PRCSNG_BATCH_CONTROL_<first two characters of the batch name>"` and `"KDD_PRCSNG_BATCH_SRC"` tables.

Example: Insert for France (FR) batch.

```
 INSERT INTO KDD_PRCSNG_BATCH (PRCSNG_BATCH_NM,PRCSNG_BATCH_DSPLY_NM,
PRCSNG_ORDER, EOD_BATCH_NM, PRCSNG_BATCH_DESC)
VALUES ('FR','France Batch',11,'FR', NULL);
INSERT INTO KDD_PRCSNG_BATCH_CONTROL_FR (PRCSNG_BATCH_ID, DATA_DUMP_DT,
PRCSNG_BATCH_NM, EOD_PRCSNG_BATCH_FL)
VALUES (100, '10-DEC-15', 'FR', 'Y');
INSERT INTO KDD_PRCSNG_BATCH_SRC (PRCSNG_BATCH_NM, SRC_ORIGIN, SRC_DESC)
VALUES ('FR', 'FR', 'France');
 COMMIT;
```

> **Note:**
>
> The KDD_PRCSNG_BATCH_CONTROL_<first two characters of the batch name> table must be truncated as per **Step 12**. For each subsequent batch run, insert the data for the next day's run into `KDD_PRCSNG_BATCH_CONTROL_<first two characters of the batch name>` table.

5. The KDD_PRCSNG_BATCH_JRSDCN_MAP table is specifically used by the Watchlist Management UI. It includes mapping between *PRCSNG_BATCH_NM* and *JRSDCN_CD*. Insert relevant data into the KDD_PRCSNG_BATCH_JRSDCN_MAP table

   Example:

```
 INSERT INTO KDD_PRCSNG_BATCH_JRSDCN_MAP (PRCSNG_BATCH_NM, JRSDCN_CD)
 VALUES ('FR', 'EMEA');
INSERT INTO KDD_PRCSNG_BATCH_JRSDCN_MAP (PRCSNG_BATCH_NM, JRSDCN_CD)
VALUES ('SG', 'APAC');
 COMMIT;
```

6. Create the following directories under `$FIC_HOME/bdf/config`:
   - datamaps_org
   - queries_org
7. Grant 755 permission to these directories.
8. Copy the original BDF datamaps from `$FIC_HOME/bdf/config/datamaps` to `$FIC_HOME/bdf/config/datamaps_org` directory and CSA queries from `$FIC_HOME/bdf/config/queries` to `$FIC_HOME/bdf/config/queries_org` directory (this is a onetime activity).

> **Note:**
>
> If a patch is released for BDF datamap or queries then the patch will place the updated files in the datamaps or queries directory. Once the patch is applied, you must copy the datamap xml file from the datamaps folder to datamaps_org folder and SQL query from queries folder to queries_org folder manually.

9. There are six sub-directories under the `$FIC_HOME/bdf/config/derivations` directory. For each of these sub-directories, create a copy (along with contents of the sub-directory) of and suffix `<_first two characters of the batch name>` to it.

Example: For France (FR) batch the sub-directory account will be copied (along with its contents) as **Account_FR**. If your batch is for Singapore (SG) the sub-directory will be copied as **Account_SG**.

10. If your implementation uses a custom BDF datamap parameter file under `$FIC_HOME/bdf/config/custom` directory, copy the custom file and suffix `<_first two characters of the batch name>.xml`.

    Example: The custom parameter file for an account should be copied as **Account_FR** for France (FR) batch.

11. In the BDF.xml file, make full refresh **FALSE** by changing the following parameters:

    ```
    <Parameter name="Load.FullRefresh" type ="BOOLEAN" value="true"/> to
    <Parameter name="Load.FullRefresh" type ="BOOLEAN" value="false"/>
    ```

    Change the value of the following parameter from "**FILE**" to "**FSDW**". This will only work for CSA ingestion.

    ```
    <Parameter name="DIS.Source" type ="STRING" value="FILE"/>
    change to
    <Parameter name="DIS.Source" type ="STRING" value="FSDW"/>
    ```

12. Before the daily batch run, truncate the following tables:

    - FO_TRXN_STAGE
    - FO_TRXN_PARTY_STAGE
    - FO_TRXN_PARTY_STAGE_RISK
    - CLIENT_BANK_SMRY_MNTH
    - ACCT_PEER_TRXN_SMRY_MNTH
    - HH_SMRY_MNTH
    - HH_BAL_POSN_SMRY

    If the tables are not truncated, an *ORA-00001: unique constraint violation error* will generate for these tables for the next day run. This is only applicable when running multiple batches in parallel using parallel_bdf.sh script.

    The *parallel_bdf.sh"* file is located at `$FIC_HOME/bdf/script`s. Grant **755** permission to this folder.

    The parallel_bdf.sh takes two input parameters; the BDF datamap name and the `<first two characters of the batch name>`.

    Command: `parallel_bdf.sh <BDF datamap name> <first two characters of the batch name>`

    To run this command, navigate to $FIC_HOME/bdf/scripts folder and run "parallel_bdf.sh" by giving it the required two parameters

    Example command for the France (FR) batch:

    ```
    parallel_bdf.sh Account FR
    parallel_bdf.sh AccountAddress FR
    parallel_bdf.sh AccountPhone FR
    parallel_bdf.sh Account_AccountCustRiskUpd FR
    parallel_bdf.sh PreviousWatchList_WatchList FR
    parallel_bdf.sh NameMatchStaging FR
    ```

```
parallel_bdf.sh WireTransaction_FrontOfficeTransaction FR
parallel_bdf.sh WireTransaction_FrontOfficeTransactionRevAdj FR
```

Example command for the Singapore (SG) batch::

```
parallel_bdf.sh Account SG
parallel_bdf.sh AccountAddress SG
parallel_bdf.sh AccountPhone SG
parallel_bdf.sh Account_AccountCustRiskUpd SG
parallel_bdf.sh PreviousWatchList_WatchList SG
parallel_bdf.sh NameMatchStaging SG
parallel_bdf.sh WireTransaction_FrontOfficeTransaction SG
parallel_bdf.sh WireTransaction_FrontOfficeTransactionRevAdj SG
```

13. When the BDF ingestion ends for a batch, truncate the respective `KDD_PRCSNG_BATCH_CONTROL_<first two characters of batch name>` tables. This is required for WLM UI functionality for the particular batch to be unlocked.
14. Once BDF ingestion completes for all the batches, end the default batch started at Step 1 by executing End Mantas Batch.

    Detection (which includes Scenarios) and Post Processing jobs will continue to run in sequence.

    In order to proceed with detection and post processing jobs you must run the start mantas batch and set the mantas date for a particular batch. Once all jobs for the batch are completed, run the end mantas batch. Repeat this process for all the batches in your system in sequence.

    Example: To run detection and post processing jobs for the France (FR) batch, start the mantas batch for FR (France), set the mantas date and run the jobs for FR batch. Once all jobs are over for FR batch, run the end mantas batch for FR. Do the same for other batches such as US (United States), SG (Singapore), and so on.

4.1.3.2 Configure WatchList Management UI to Support Multiple Batches

WatchList Management UI supports both Single Batch and Multiple Batch Mode.

To configure running ingestion in parallel for multiple batches, follow these steps (one time activity).

1. Navigate to folder `$FIC_HOME/ficweb/webroot/WEB-INF/classes`
2. Edit the WLM.properties file.

   Default entry in file "WLM.properties" should follow the convention below:

   ```
   # Configure the BatchMode-Allowed values are Y for Single Batch and N for
   Multiple Batch
   Mode. Default Value is Y
   SingleBatch=Y
   ```

   For supporting multiple batches this needs to be set to '**N**' `SingleBatch=N`
3. Once configured, regenerate the war and deploy the war file.

4.2 BD Ingestion Flat File Data Load

The loading process receives, transforms, and loads Market, Business, and Reference data that alert detection and assessment investigation processing requires. After loading the base tables, the Oracle client's job scheduling system invokes BD datamaps to derive and aggregate data.

Overview

All DIS datamaps in the Behavior Detection Flat File Interface for which staging representation is marked as Yes are applicable for Flat File loading. For more information, see Behavior Detection Flat File Interface.

Using Behavior Detection Datamaps

The Behavior Detection (BD) datamap takes the data from the flat files, enhances it, and then loads it into a target database table (FSDM).

To load data in the FSDM using Flat Files, follow these steps:

1. Place the ASCII.dat flat files in the `<OFSAAI Installed Directory>/bdf/inbox` directory.
2. Configure the DIS.source parameter to **FILE**. For more information on configuring other parameters, see Appendix D - Managing Data.
   Configure the *DIS.Source* parameter to **FILE-EXT** for loading flat files through the external table. In order to load the flat files using the external table, the *ext_tab_dir_path* variable must also be set to the inbox directory and the database UNIX account must have read and write privileges to it.
3. Execute the Account datamap which loads into the Account (ACCT) table: `<OFSAAI Installed Directory>/bdf/scripts/execute.sh Account`

> **Note:**
>
> If there are any errors in loading, refer to the `<OFSAAI Installed Directory>/bdf/logs path.`

Using Pre-processing and Loading

The pre-processor component (runDP) use XML configuration files in the /config/datamaps directory to verify that the format of the incoming Oracle client data is correct and validate its content, specifically:

- Error-checking of input data
- Assigning sequence IDs to records
- Resolving cross-references to reference data
- Checking for missing records
- Flagging data for insertion or update

The loader component (runDL) receive pre-processed Reference data and business data. The components then load this data into the database.

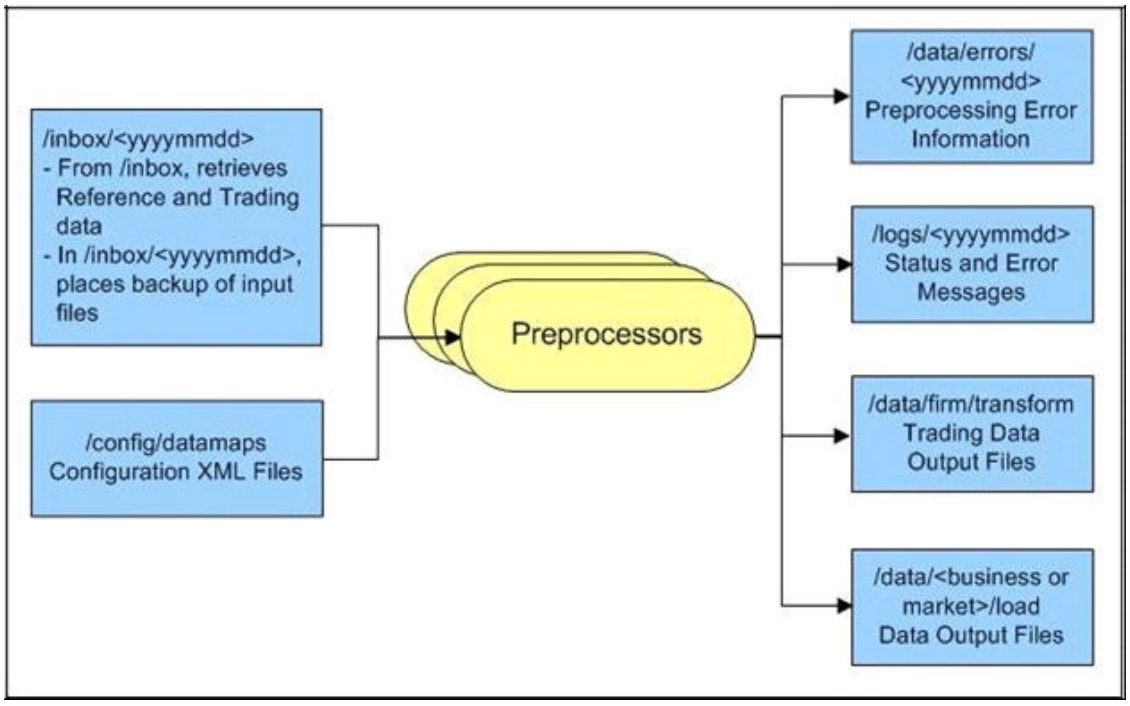
> **Note:**
>
> The Pre-processor addresses only those files that match naming conventions that the DIS describes, and which have the date and batch name portions of the file names that match the current data processing date and batch. Oracle clients must only supply file types required by the solution sets on their implementation.

To load data in the FSDM using Pre-processing and Loading, follow these steps:

1. Place the ASCII.dat flat files in the `<OFSAAI Installed Directory>/ ingestion_manager/inbox` directory. The component then performs data validation and prepares the data for further processing.
2. Execute runDP and runDL using the following sample scripts:
   - For runDP: `<OFSAAI InstalledDirectory>/ingestion_manager/scripts/runDP.sh AccessEvents`
   - For runDL:`<OFSAAI InstalledDirectory>/ingestion_manager/scripts/runDL.sh AccessEvents`

Pre-processors place output files in the directories. The following figure summarizes Pre-processing input and output directories.

Figure 4-1 Input and Output Directories



For more information on the directory structure, see Appendix D - Managing Data.

Configuring RunDP/RunDL

For flat files, Behavior Detection receives firm data in ASCII.dat flat files, which an Oracle client's data extraction process places in the /inbox directory.

Ways of Data Loading

The following ways of data loading is applicable only for DIS files defined with load operation as Overwrite.

- Full Refresh Data Loading: For full refresh data loading, first data is truncated and then new data is inserted. For example, suppose five records are loaded on Day 1. If new data is required on Day 2 based on the business keys defined on the DIS files, a full refresh data load can be done.
  To do a full refresh data load, set load.fullrefresh to true in the <OFSAAI Installed Directory>/bdf/config/BDF.xml path. For more information, see BDF.xml Configuration Parameters.

  The time taken to do a full refresh data load is less than for an incremental load, although complete data must be provided every time.
- Incremental (Delta) Data Loading: For incremental data loading, the following can be done:
  - Data can be merged
  - Existing data can be updated
  - New data can be inserted

  For example, suppose five records are loaded on Day 1. If four new records need to be inserted and one existing record needs to be updated based on the business keys defined on the DIS files, an incremental data load can be done. To do an incremental data load, set load.fullrefresh to false in the `<OFSAAI Installed Directory>/bdf/config/BDF.xml` path. For more information, see BDF.xml Configuration Parameters.

> **Note:**
>
> The time taken to do an incremental data load is more than for a full refresh data load, although there is no need to give complete data every time. Only updated or new data is required.

4.3 Encrypting Data Files

To minimize exposure of data or personal information to users with access to the server, Oracle clients can encrypt ingestion files using a simple encryption technique which requires a generic 16 digit encryption key, combination of numericals and alphabets, such as: AmritaP123456789

Standard "AES" key spec and transformation "AES/ECB/PKCS5Padding" are used for encryption and decryption. Client can encrypt files using these on their own.

To run data ingestion on encrypted files, follow these steps:

1. Encrypt the ingestion files by running encryptFileUtil.sh, as shown below:
   ```
   encryptFileUtil.sh <ALG_FILE_PWD> false
   <absolute_path_to_the_ingestion_files_you_want_to_encrypt>
   ```

   For example:

   ```
   encryptFileUtil.sh AmritaP123456789 false /scratch/ofsaaweb/BD806A/bdf/
   inbox/Account_20151209_DLY_01.dat
   ```

2. Update the *BDF.Encryption.Password* parameter in the bdf.xml file in `<FIC_HOME>/config/install` path with the encryption key as shown below:

```
<Parameter name="BDF.Encryption.Password" type="STRING"
value="<Encryption Key>"/>
```

3. Update the BDF.Encryption.Enable parameter in the bdf.xml file in <FIC_HOME>/config/install path with the encryption key as shown below:

```
<Parameter name="BDF.Encryption.Enable" type="STRING"
value="true"/>
```

4. Run execute.sh to invoke file ingestion.

4.4 Managing Data Processing

This section explains the concept of data processing and various methods of data processing.

This section covers the following topics:

- Generating Change Logs with BD
- Processing Data Using BD
- Processing Data Using FDT and MDT

The following datamaps are currently supported for change log functionality:

- Account
- AccountAddress
- AccountPhone
- AccountEmailAddress
- AccountToCustomer
- Customer
- CustomerAddress
- CustomerPhone
- CustomerEmailAddress
- AccountRestriction
- InsurancePolicyToCustomer
- EmployeeAddress
- SettlementInstruction

4.4.1 Generating Change Logs with BD

Change log and Change log summary records with BD will be generated through BD.

When loading referential DIS files that are defined as Overwrite, it is possible for BD to generate Change Log records which signify when certain fields associated with a reference data entity have changed. This is done by comparing the contents of the DIS file with the current contents of the associated database table. For performance reasons, this change log processing can be done when external tables are used to load the DIS files, so it is a

requirement that **DIS.Source=FILE-EXT**. This requires an external directory, which is created during installation. In order to give access to an Oracle user, place the .dat files in the external directory.

The change log records can also be derived with **DIS.Source = 'FSDW'** (CSA Ingestion). While FILE_EXT derives the change log based on comparison of reference data with newly ingested modified data (through the DAT FILE) on the next day, with the *DIS.Source=FSDW*, the change log is derived on comparing the reference data which is loaded to FCDM tables from staging table data.

> **Note:**
>
> To derive the change log records the change log parameters in `<OFSAAI Installed Directory>/BDF/config/BDF.xml` should be uncommented. The change log can only be derived from the second day onwards. Since change log functionality derives changes by comparing the data of two days, the first day data acts as a reference against which the second day data is compared and changes are derived.

Change log records can be generated in the following ways:

- Compare fields on a single reference data record that can be identified by a primary key. For example, an Account record can be identified by an Account Identifier. When an Account file is ingested, the Primary Customer Identifier on Account XYZ is compared to the Primary Customer Identifier currently in the database for Account XYZ. If they are different, then a Change Log record is created. This process only accounts for updates to already existing records. Change Log records are not created for new reference data records or deleted reference data records.
- Compare the set of values for a given field on several reference data records that map to a given key.
  For example, an Account Address record is identified with a combination of Account Identifier and Address Record Number. However, the information required is whether an Account Address record for a given Account has a field value that is different than any other Account Address record for that Account. For example, every Account Address record has a Country field. If there are two Account Address records for Account XYZ in the database with values for Country of US and CN, respectively. On the next day, an Account Address file is processed and there is an Account Address for Account XYZ with a value for Country of IR. A Change Log record is generated for the Country field of this Account Address record. Furthermore, in the case of Account Address, it is not just the Account Identifier of an Account Address record that is of interest. The Address Purpose is also of interest. So when we look in the database for Account Address records that match a given Account Address record in a DIS file, we look to match both the Account Identifier field and the Address Purpose field.

This processing is controlled by parameters in `<OFSAAI Installed Directory>/bdf/config/BDF.xml`. All of these parameters have been commented out, which means change log processing is turned off by default. To derive the change log records if **DIS.Source = 'FILE-EXT',** the relevant parameters for the DIS files of interest should be copied to `<OFSAAI Installed Directory>/bdf/config/custom/BDF.xml` and uncommented.

Table 4-1 Change Log Parameters

Parameter	Description
ChangeLog.<DIS File Type>.Fields	The fields of this particular DIS file type which will be monitored for changes.
ChangeLog.<DIS File Type>.IsSet	Whether change log records are generated based on mechanism 1 above (false) or mechanism 2 (true). The default is false.
ChangeLog.<DIS File Type>.QueryKey	This is only relevant when IsSet=true. This defines the key that is used to query for reference data records matching the given one. In the Account Address example given above, the value would be AccountIdentifier,AddressPurpose. If this parameter is not present, then the business key located in the given DIS file type's data map (for example bdf/datamaps/ AccountAddress.xml) is used.
ChangeLog.<DIS File Type>.OutputKey	This is only relevant when IsSet=true. This defines the set of fields that are mappedto the Key1, Key2, Key3, and Key4 fields of a Change Log record. This can be different from the QueryKey and business key in order to match what is expected in Change Log DIS file records, and also to support the Change Log Summary data maps. If this parameter is not present, then the business key located in the given DIS file type's data map (for example, bdf/ datamaps/ AccountAddress.xml) is used.

To turn on Change Log processing for a given DIS file type, all the parameters for that file type must be uncommented. The values of the *ChangeLog.<DIS File Type>.Fields* parameter are preset based on the needs of the KYC application. If different fields are required, then this parameter should be changed. It is not necessary to change any of the other parameters.

For Example: If Address Street line fields are to be considered for change log generation, then the *ChangeLog.<DIS File Type>. Fields* parameter should be changed for that particular table as shown below.

```
<Parameter name="ChangeLog.AccountAddress.Fields" type ="STRING"
value="Country,Region,State,City,PostalCode,MailHandlingInstruction"
list="true"/>
```

should be changed to

```
<Parameter name="ChangeLog.AccountAddress.Fields"
type ="STRING"
value="Country,Region,State,City,PostalCode,MailHandlingInstruction,
StreetLine1,StreetLine2,StreetLine3,StreetLine4,StreetLine5,StreetLine6"
list="true"/>
```

As in the example above, StreetLine1,StreetLine2,StreetLine3,StreetLine4,StreetLine5 and StreetLine6 will also be considered for change log generation. Similar steps can be followed for other change log related tables well.

Change Log records are written to the CHG_LOG table as the DIS file is being loaded. There are no additional scripts to be run. As soon as the parameters are uncommented, Change Log records are generated the next time DIS files are loaded.

4.4.2 Processing Data Using BD

The BD datamap component is responsible for taking data from one or more source files or staging tables, transforming and enhancing it, and then loading it into a target database table.

This section covers the following topics:

- BD Derived Datamap Types
- Datamap Categories
- Processing Datamaps
- Configuring Risk Zones
- Customizing Review Reason Text
- Datamaps

The following types of datamaps are available:

- **DIS datamaps**: DIS datamaps are used to ingest client provided data, either through DIS files as specified in the DIS or through tables in the FSDF.
- **Derived datamaps**: Derived datamaps are used to transform the client provided data and populate other tables for use by scenarios and/or UI functionality.

BD datamaps can perform the following activities:

- Update summaries of trading, transaction, and instructionactivity
- Assign transaction and entity risk through watch listprocessing
- Update various Balances and Positions derivedattributes

For a complete list of the BD datamaps used in OFSAAI and a brief explanation of the each datamap, see Appendix F - BD Datamap Details

4.4.2.1 BD Derived Datamap Types

The Oracle solution implemented determines the required BD datamaps, or a subset thereof.

- AML Brokerage Datamaps
- AML Banking Datamaps
- Fraud Detection Datamaps
- Insurance Datamaps

> **Note:**
>
> Caution: If you are running multiple solutions, you must perform table comparisons to avoid running duplicate datamaps.

The following table describes the columns in the datamap tables that each section provides.

Table 4-2 Datamap Table Descriptions

Column	Description
Datamap Number	Unique, five-digit number that represents a particular datamap.
Datamap Name	Unique name of each datamap.
Predecessor	Indicator that processing of datamaps cannot begin until completion of predecessor datamaps.

4.4.2.1.1 <Enter Topic Title Here>

The following sections describe the Datamaps that are required for deriving and aggregating data for the AML Brokerage solution:

- AML Brokerage - Pre-Watch List Datamaps
- AML Brokerage - Watch List Datamaps
- AML Brokerage - Post-Watch List Datamaps
- AML Brokerage - Summary Datamaps
- AML Brokerage - Balances and Positions Datamaps

Each section provides a table that illustrates the datamaps and order of each datamap. This table describes the process by datamap number, datamap name, and internal or external predecessors, if any.

Optional Datamaps are used to perform processing to support other datamaps in multiple functional areas. These datamaps may or may not be completely relevant to a particular solution set. Execute the datamap if a scenario in your implementation requires this information.

4.4.2.1.2 Trusted Pair

Trusted pair Tables can be ingested using two ingestion processes, either DIS File or CSA tables, to populate the KDD_TRUSTED_PAIR and KDD_TRUSTED_PAIR_MBR business tables.

> **Note:**
>
> BD supports only one method of managing trusted pairs per installation. Clients may elect to create and manage trusted pairs through the loading of trusted pairs via a DIS file or utilize the Trusted Pairs API for creation and management of trusted pairs. However, both the methods should not be utilized concurrently.

> **Note:**
>
> The KDD_TRUSTED_PAIR and KDD_TRUSTED_PAIR_MBR tables use full refresh data loading. The data is first truncated and then new data is inserted. Complete data must be provided every time these commands are executed.

DIS File Ingestion

The Trusted Pair DIS file is different from typical DIS file. In this, the same DIS file is used to populate two separate tables; KDD_TRUSTED_PAIR and KDD_TRUSTED_PAIR_MBR. These tables can be populated by executing the following commands:

- runDP.sh TrustedPair
- runDL.sh TrustedPair
- runDL.sh TrustedPairMember

CSA Ingestion

The Trusted Pair CSA ingestion is different from typical CSA ingestion. Two separate tables KDD_TRUSTED_PAIR and KDD_TRUSTED_PAIR_MBR are populated from the same CSA table. These tables can be populated by executing the following commands:

- execute.sh TrustedPair
- execute.sh TrustedPairMember

> **Note:**
>
> Currently, only Full Load of both Trusted Pair (KDD_TRUSTED_PAIR) and Trusted Pair Member (KDD_TRUSTED_PAIR_MBR) tables is supported. You must always truncate and full load.

4.4.2.2 Datamap Categories

Each datamap can include one or more of the following categories:

- Optional
- Pre-Watch List
- WatchList
- Post-Watch List
- Summary
- Balances and Positions

The Datamap categories may or may not be required for all solutions.

4.4.2.3 Processing Datamaps

This section provides the required datamaps for deriving and aggregating data based on the solution. Discussions of the datamaps appear in the order that processing must execute them during data loading, and include tables that describe each datamap. Datamap numbers that the accompanying tables provide also reflect this order.

Where predecessors exist, processing of datamaps cannot begin until completion of predecessor datamaps. These dependencies, or predecessors, may be internal to the datamap type, or external to the datamap type such as Summary datamaps dependent on watch list datamaps.

> **Note:**
>
> If there is any performance issue with the running sequence of datamaps, it can be re-arranged. However. the predecessor for the datamap must be completed before running the datamap.

```
FrontOfficeTransactionParty_InstnSeqID
FrontOfficeTransactionParty_HoldingInstnSeqID
```

If there is any performance issue with the datamap *FrontOfficeTransactionParty_HoldingInstnSeqID*, the datamap position can be rearranged in the batch script. Since there is the possibility that the previous process (*FrontOfficeTransactionParty_InstnSeqID*) is still running, the current datamap is waiting for the resources to be released.

Example for Internal Dependency

For example, processing can run the FrontOfficeTransactionParty_InstnSeqID datamap immediately after completion of FinancialInstitution_FOTPSPopulation and AccountToClientBank_FOTPSInstitutionInsert.

Example for External Dependency

Processing cannot run the AccountProfile_Trade datamap until and unless the FrontOfficeTransactionPartyRiskStage_EntityActivityRiskInsert datamap is run.

4.4.2.4 Configuring Risk Zones

Risk Zones are the threshold value by which an increase in a party's effective risk will trigger a review of the trusted pair is configurable.

However, if the party's risk has not increased by enough points to move it to a higher risk zone, then no risk review action is initiated on the trusted pair. In any case, the party's risk will be updated on the applicable Trusted Pair member record. The default risk zones are configured as:

- RiskZone1Lower=1
- RiskZone1Upper=3
- RiskZone2Lower=4
- RiskZone2Upper=5
- RiskZone3Lower=6
- RiskZone3Upper=7
- RiskZone4Lower=8
- RiskZone4Upper=10

The ranges of risk values within each zone are configurable but the number of risk zones shall remain at 4. If an implementation chooses not to use all Risk Zones then they can disable them by setting the risk ranges out of bounds. For example, Risk Zone 1 and Risk Zone 2 may have a lower and upper value of 0.

> **Note:**
>
> Ensure that the trusted pair file is run before the risk zones.

4.4.2.5 Customizing Review Reason Text

Where the party's effective risk has increased by enough points to move it to a higher risk zone, the system also records the reason for marking the record for review. This is done using the TrustedPairReviewReasonText1 and TrustedPairReviewReasonText2 parameters.

Sample strings currently used for review reason text are as follows:

```
TrustedPairReviewReasonText1=Recommend Cancel - risk of <Party1> increased
from <A> to <B>
TrustedPairReviewReasonText2= and risk of <Party2> increased from <C> to <D>
```

The string for Review Reason Text parameters is translatable. You can change these strings except the values in angular brackets like <Party1>, <A>, <B>, <Party2>, <C>, and <D>.

If the system determines that the Trusted Pair record that has experienced a threshold triggering risk increase is still in a Risk Escalated Recommend Cancel (RRC) state (that is, a Supervisor has not reviewed the recommendation), the system appends the new review reason text to the existing reason text on the current Recommend Cancel version of the Trusted Pair record. A semi-colon (;) and a single space is used as the method of appending.

> **Note:**
>
> While appending a new review reason text to the existing text, the system finds that appending text will result in the field exceeding 2500 characters. In this case, the system will overwrite the existing review reason text on the current Rec Cancel version of the Trusted Pair record with the current review reason text.

The above mentioned parameters for configuring risk zones and customizing review reason text are located in the `<OFSAAI Installed Directory>/bdf/config/BDF.xml` file. Risk review only happens if *managing_tp_from_ui* is set to **Y** in the *installMantas.properties.sample* properties file.

> **Note:**
>
> Datamaps 10970,10980,10990, 11000,11010,11020 can be run in parallel.

4.4.2.6 Datamaps

This section displays the different BD datamap types.

This topic covers the following topics:

- AML Banking Datamaps
- Fraud Detection Datamaps

- Insurance Datamaps
- Configuring Risk Zones

AML Banking Datamaps

The following sections describe the required datamaps for deriving and aggregating data for the AML Banking solution:

- AML Banking - Pre-Watch List Datamaps
- AML Banking - Watch List Datamaps
- AML Banking - Post-Watch List Datamaps
- AML Banking - Summary Datamaps

Fraud Detection Datamaps

The following sections describe the datamaps that are required for deriving and aggregating data for Fraud Detection:

- Fraud Detection - Pre-Watch List Datamaps
- Fraud Detection - Watch List Datamaps
- Fraud Detection - Post-Watch List Datamaps
- Fraud Detection - Summary Datamaps Detection

Insurance Datamaps

The following sections describe the datamaps that are required for deriving and aggregating data for the Insurance Solution:

- Insurance - Pre-Watch List Datamaps
- Insurance - Watch List Datamaps
- Insurance - Post-Watch List Datamaps
- Insurance - Summary Datamaps

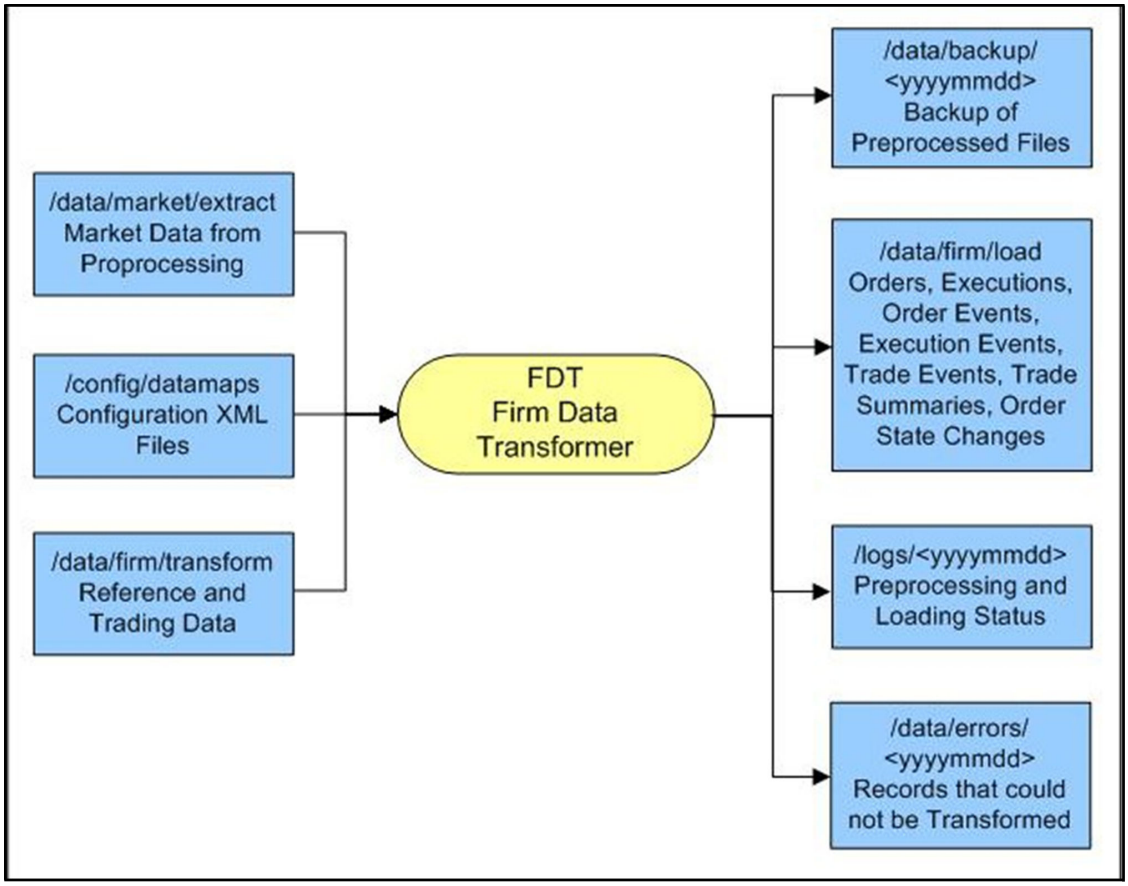
4.4.3 Processing Data Using FDT and MDT

The following sections describe how Ingestion Manager processes trade-related data, orders and executions, and trades through the Firm Data Transformer (FDT).

FDT Process Flow

The following figure illustrates the FDT process flow:

Figure 4-2 Firm Data Transformer (FDT) Processing



The FDT performs the following actions:

- Processes all files that reside in the /data/firm/transform directory for the current date and batch.
- Terminates automatically after processing files that it found at startup.
- Ignores files that the system adds after processing begins; the system may process these files by starting FDT again, after exiting from the previous invocation.

Order and Trade Execution files are processed through the Firm Data Transformer (FDT). Before running runFDT.sh, Pre-processor has to be executed, using the following commands:

```
<OFSAAI Installed Directory>/ingestion_manager/scripts/runDP.sh TradeExecution
<OFSAAI Installed Directory>/ingestion_manager/scripts/runDP.sh Order
<OFSAAI Installed Directory>/ingestion_manager/scripts/runDP.sh OpenOrder
```

During execution of the runFDT.sh script, the FDT performs the following actions:

- Enriches data.
- Produces summary records for orders and trades.
- Calculates derived values to support detection needs.
- Derives state chains (that is, order life cycle states, marketability states, and displayability states).

- Provides data for loading into FSDM.

The system executes the FDT with the runFDT.sh script; the following provides a sample command: `<OFSAAI Installed Directory>/ingestion_manager/scripts/runFDT.sh`

When Ingestion Manager executes runFDT.sh, it places output files in the directories described in the following table.

Table 4-3 runFDT.sh Output Directories

Directory	Description
/data/firm/transform	Rollover data that processing saves for the next run of the FDT. Includes open and closed orders, old executions, old trades, old derived trades, lost order events, and lost trade execution events.
/logs/<yyyymmdd>	Status and error messages.
/data/errors/<yyyymmdd>	Records that the system was unable to transform.
/data/backup/<yyyymmdd>	Backup of Pre-processed input files.
/data/firm/load	Transformed output files for loading into the database.

After running run FDT, the system executes data loaders using the runDL.sh script; the following provides a sample command:

```
<OFSAAI Installed Directory>/ingestion_manager/scripts/runDL.sh Order
<OFSAAI Installed Directory>/ingestion_manager/scripts/runDL.sh OrderSummary
<OFSAAI Installed Directory>/ingestion_manager/scripts/runDL.sh TradeExecution
<OFSAAI Installed Directory>/ingestion_manager/scripts/runDL.sh Execution
<OFSAAI Installed Directory>/ingestion_manager/scripts/runDL.sh Trade
<OFSAAI Installed Directory>/ingestion_manager/scripts/runDL.sh DerivedTrade
```

FDT processes are also available with BDF. To perform this action, you must execute the following datamaps in the order given:

1. OpenOrderStage
2. OrderStage
3. TradeExecutionEventStage
4. Scrty_TradeExecutionStageInsert
5. Scrty_OrderStageInsert
6. MktCntr_OrderStageInsert
7. OrderStage_DQupdate
8. TradeExecutionEventStage_DQupdate
9. OrderStage_FDTupdate
10. OrderStage_RmngQtupdate
11. OrderSummary
12. OrderSummary_OpenOrdrInsrt
13. OrderSummary_QtyUpdate
14. OrderStage_OpenOderUpd

15. OrderSummary_Update
16. OrderStage_OrdrSeqUpd
17. OrderEvent_OrderStage
18. Execution_NewEvents
19. Execution_CancelAndReplace
20. Execution_CancelEvents
21. Execution_CorrectionEvents
22. Trade_NewEvents
23. Trade_CancelAndReplace
24. Trade_CorrectionEvents
25. Trade_CancelEvents
26. Trade_DerivedTrade
27. Trade_OrigSeqIDUpd
28. Trade_ParentSeqIDUpd
29. Trade_RplcngSeqIDUpd
30. TradeExecutionEvent_Trade
31. TradeExecutionEvent_Execution
32. TradeExecutionEvent_CancelReplaceTrade
33. TradeExecutionEvent_FirmRefTrade
34. TradeExecutionEvent_MktRefTrade
35. Trade_RefData
36. Execution_Update

Trade related Data maps cannot run using Multi Batch Functionality.

Populating Summary Information for Market Data

As part of end of day processing, Market and Trade data summary information gets updated in the following path of the Java Utility: `<OFSAAI Installed Directory>/ingestion_manager/scripts/ process_firm_summary.sh`

4.5 Managing Data For BD Applications

This section explains different methods used to load and process data in various BD applications. Data Loading For AML/Fraud/KYC/FATCA/CTR Applications.

To load the AML, Fraud, KYC, FATCA, or CTR applications, follow these steps:

1. Process the loaded data using BD datamaps in FSDM. For more information, see Managing Data Processing.
2. Interface files in the same group loaded through different loading method can be executed in parallel.
3. Run AML BD transformation. For more information on the AML datamaps, see AML Brokerage Datamaps and AML Banking Datamaps.
4. For network scenarios, refresh the temporary tables.

These steps use Group 1 Group 2 Group 3 Group 4 Group 5.

5

Behavior Detection Jobs

This chapter provides an overview of the OFSBD Job Protocol and explains how the System Administrator monitors jobs, and starts and stops jobs when necessary. In addition, it describes the necessary scripts that you use for OFSBD jobs.

> **Note:**
>
> If you are using a job script that allows for multiple parameters, the values for the parameters must be separated by spaces ( ) and not commas (,).

This chapter focuses on the following topics:

- About the OFSBD Job Protocol
- Performing Dispatcher Tasks
- Performing Job Tasks
- Clearing Out the System Logs
- Recovering Jobs from a System Crash
- Executing Batches Through the OFSAAI User Interface

5.1 About the OFSBD Job Protocol

The system initiates all OFSBD jobs by using a standard operational protocol that utilizes each job's metadata, which resides in a standard set of database tables.

OFSBD Job Protocol processes include the following:

- Dispatcher: Polls the job metadata for new jobs that are ready for execution. This daemon process starts a MANTAS process for each new job.
- Mantas: Creates a new job entry based on a template for the job that has the specific parameters for this execution of the job (that is, it clones a new job).

The OFSBD administrator invokes the dispatcher and MANTAS processes by running the shell scripts that are mentioned in the following table.

Table 5-1 OFSBD Job Protocol Shell Scripts

OFSBD Job Protocol Process Shell Script	Description
start_mantas.sh	Starts all OFSBD jobs. This script invokes the cloner and MANTAS processes. This is the integration point for a third-party scheduling tool such as Maestro or AutoSys.
start_chkdisp.sh	Calls on the *check_dispatch.sh* script to ensure that the dispatcher runs.
stop_chkdisp.sh	Stops the dispatcher process.

Table 5-1 (Cont.) OFSBD Job Protocol Shell Scripts

OFSBD Job Protocol Process Shell Script	Description
restart_mantas.sh	Changes job status codes from the ERR status to the RES status so that the dispatcher can pick up the jobs with the RES status.
recover_mantas.sh	Changes job status codes for jobs that were running at the time of a system crash to the ERR status. After running this script, the restart_mantas.sh script must be run to change the ERR status code to RES in order for the dispatcher to be able to pick up these jobs.

In the OFSBD Job Protocol, the processes use a variety of metadata that the OFSBD database provides. Some of this metadata specifies the jobs and their parameters that are associated with the regular operations of an OFSBD installation. Some of this metadata captures the status of job execution and is useful for monitoring the progress of an OFSBD operational cycle.

Understanding the OFSBD Job Protocol

OFSBD Jobs are created through the Scenario Manager. Jobs are grouped together to run in parallel through Job Template Groups in the KDD_JOB_TEMPLATE table. These templates associate an algorithm to run with parameters that the algorithm requires. Template groups enable you to identify what jobs to run.

The following table provides an example of a job template group with two job templates.

Table 5-2 KDD_JOB_TEMPLATE with Sample Job Template Group

JOB_ID	TEMPLATE_GROUP_ID
37	1
41	1

Understanding the Dispatcher Process

The dispatcher process polls the job metadata waiting for jobs that must be run. To control system load, the dispatcher also controls the number of jobs that run in parallel.

Generally, the dispatcher process should be running continuously, although it is possible to run jobs without a dispatcher. For each job in the template group, the dispatcher runs a MANTAS process.

The dispatcher tracks jobs for status and completion, and reports any failure to the dispatch log.

> **Note:**
>
> If you observe job failures when running on the AIX operating system, it may be due to resource constraints of the AIX system. In this case, you must try reducing the number of jobs you are attempting to run in parallel or try running the jobs sequentially.
> Refer to Starting the Dispatcher and Stopping the Dispatcher for more information.

Understanding the MANTAS Process

The dispatcher runs jobs using the MANTAS process. This process runs the appropriate algorithm, tracks status in the KDD_JOB and KDD_RUN tables. One MANTAS process can result in multiple KDD_RUN records.

The MANTAS process also logs job progress and final status.

Applying a Dataset Override

The dataset override feature permits dataset customizations specific to your site, which can be retained outside of the scenario metadata. The override to a dataset definition is stored in a file accessible by the Behavior Detection engine. The dataset override feature allows improved performance tuning and the ability to add filters that are applicable only to your site's dataset.

When the system runs a job, it retrieves the dataset definition from the database. The Behavior Detection engine looks in the configured directory to locate the defined dataset override. The engine uses the override copy of the dataset instead of the copy stored in the scenario definition in the database, if a dataset override is specified.

The following constraints apply to overriding a dataset:

- The columns returned by the dataset override must be identical to those returned by the product dataset. Therefore, the dataset override does not support returning different columns for a pattern customization to use.
- The dataset override can use fewer thresholds than the product dataset, but cannot have more thresholds than the product dataset. Only thresholds applied in the dataset from the scenario are applied.

If a dataset override is present for a particular dataset, the override applies to all jobs that use the dataset.

5.1.1 Configuring the Dataset Override Feature

To configure a dataset override, follow these steps:

1. Modify the *install.cfg* file for algorithms to identify the directory where override datasets are stored.

   The file resides in the following directory: `<OFSAAI Installed Directory>/behavior_detection/algorithms/MTS/ mantas_cfg/ install.cfg`

   The dataset override is specified with this property: *kdd.custom.dataset.dir*

   > **Note:**
   >
   > Specify the directory for the above given property using a full directory path, not a relative path. If you do not (or this property is not in the install.cfg file), the system disables the dataset override automatically.

2. Create the dataset override file in the specified directory with the following naming convention: `dataset<DATASET_ID>.txt`

   The contents of the file should start with the SQL definition in KDD_DATASET.SQL_TX. This SQL must contain all of the thresholds still represented such as *@Min_Indiv_Trxn_Am.*

5.2 Performing Dispatcher Tasks

The dispatcher service runs on the server on which OFSBD is installed. Once the dispatcher starts, it runs continuously unless a reason warrants shutting it down or it fails due to a problem in OFSBD.

This section covers the following topics:

Setting Environment Variables

Environment variables are set up during the OFSBD installation process. These generally do not require modification thereafter. All behavior detection scripts and processes use the system.env file to establish their environment.

About the System.env File

The following table describes environment variables in the system.env file. This file can be found at `<OFSAAI Installed Directory>/behavior_detection/algorithms/MTS/share`

Table 5-3 OFSBD Environment Variables in system.env File

Variable	Description
KDD_HOME	Install path of the OFSBD software.
KDD_PRODUCT_HOME	Install path of the solution set. This is a directory under KDD_HOME.

The following table describes database environment variables in the system.env file.

Table 5-4 Database Environment Variables in system.env File

Variable	Environment	Description
ORACLE_HOME	Oracle	Identifies the base directory for the Oracle binaries. You must include: • $ORACLE_HOMEand$ORACLE_HOME/bininthe PATH environment variable value. • $ORACLE_HOME/libinthe LD_LIBRARY_PATHenvironmentvariable value.
ORACLE_SID	Oracle	Identifies the default Oracle database ID/name to which the application connects.

Table 5-4 (Cont.) Database Environment Variables in system.env File

Variable	Environment	Description
TNS_ADMIN	Oracle	Identifies the directory for the Oracle network connectivity, typically specifying the connection information (SID, Host, Port) for accessing Oracle databases through SQL*NET.

The following table shows operating system variables in the system.env file.

Table 5-5 Operating System Environment Variables in system.env File

Variable	Description
PATH	Augmented to include `<OFSAAI Installed Directory>/behavior_detection/algorithms/MTS/bin` and the `$ORACLE_HOME, $ORACLE_HOME/bin` pair (for Oracle).
LD_LIBRARY_PATH, LIBPATH, SHLIB_PATH (based on operating system)	Augmented to include `<OFSAAI Installed Directory>/behavior_detection/algorithms/MTS/lib` and `$ORACLE_HOME/lib` (for Oracle)

5.2.1 Starting the Dispatcher

Oracle provides a script to check the status of the dispatcher automatically and restart it, if necessary. Oracle recommends this method of running the dispatcher.

Although multiple jobs and MANTAS instances can run concurrently in OFSBD, only one dispatcher service per database per installation should run at one time.

1. Verify that the dispatcher is not already running by typing `ps -ef | grep dispatch` and pressing **Enter** at the system prompt.

   If the dispatcher is running, an instance of the dispatcher appears on the screen for the server. If the dispatcher is not running, proceed to Step 2.

2. Type `start_chkdisp.sh <sleep time>` and press **Enter** at the system prompt to start the dispatcher.

   The dispatcher queries the database to check for any new jobs that must be run. In between these checks, the dispatcher sleeps for the time that you specify through the <*sleep time*> parameter (in minutes).

   Optional parameters include the following:

   - dispatch name: Provides a unique name for each dispatcher when running multiple dispatchers on one machine.
   - JVM size: Indicates the amount of memory to allocate to Java processing. The script executes and ends quickly. The dispatcher starts and continues to run in the background.

5.2.2 Stopping the Dispatcher

To view active jobs and then shut down the dispatcher, follow these steps:

You do not normally shut down the dispatcher except for reasons such as the following:

- Problems while executing scenarios, make it necessary to stop processing.
- The dispatcher and job processes are reporting errors.
- The dispatcher is not performing as expected.
- You must shut down the system for scheduled maintenance.
- You want to run the start_mantas.sh, restart_mantas.sh, or recover_mantas.sh script without the dispatcher already running. You can then save your log files to the server on which you are working rather than the server running the dispatcher.

> **Note:**
>
> The dispatcher which started from the Behavior Detection jobs in the UI should be stopped before restarting servers. **Caution:** If you shut down the dispatcher, all active jobs shut down with errors.

When you are ready to restart the dispatcher and you want to see which jobs had real errors and which jobs generated errors only because they were shut down during processing, review the error messages in the job logs.

For those jobs that shut down and generate errors because the dispatcher shut down, a message similar to the following appears: *Received message from dispatcher to abort job.* If the job generates a real error, a message in the job log file indicates the nature of the problem.

1. Type `ps -efw | grep mantas` and press **Enter** at the system prompt.

   All instances of the MANTAS process that are running appear on the screen. Only one instance of MANTAS should run for each active job.

2. Type `stop_chkdisp.sh <dispatcher name>` and press **Enter** at the system prompt. This script shuts down the dispatcher.

5.2.3 Monitoring the Dispatcher

The install.cfg file that was set up during server installation contains the *kdd.dispatch.joblogdir* property that points to a log file directory. The log directory is a repository that holds a time-stamped record of dispatcher and job processing events.

Each time the dispatcher starts or completes a job, it writes a status message to a file called `dispatch.log` in the log directory. This log also records any failed jobs and internal dispatcher errors. The dispatch.log file holds a time-stamped history of events for all jobs in the chronological sequence that each event occurred. To monitor the dispatch.log file as it receives entries, follow these steps:

1. Change directories to the log directory.
2. Type `tail -f dispatch.log` and press **Enter** at the system prompt.

   The log file scrolls down the screen.

3. Press **Ctrl+C** to stop viewing the log file.

4. Type `lpr dispatch.log` and press **Enter** at the system prompt to print the dispatch.log file.

> **Note:**
>
> The dispatch.log file can be a lengthy printout.

5.3 Performing Job Tasks

At the system level, the OFSBD administrator can start, restart, copy, stop, monitor, and diagnose jobs.

This section cover the following topics:

- Understanding the Job Status Codes
- Starting Behavior Detection Jobs
- Starting Jobs Without the Dispatcher
- Restarting a Job
- Restarting Jobs Without the Dispatcher
- Stopping Jobs
- Monitoring and Diagnosing Jobs

Understanding the Job Status Codes

The following status codes are applicable to job processing and the dispatcher. The OFSBD administrator sets these codes through an OFSBD Job Editor:

- NEW (start): Indicates a new job that is ready to be processed.
- RES (restart): Indicates that restarting the existing job is necessary.
- IGN (ignore): Indicates that the dispatcher should ignore the job and not process it. This status identifies Job Templates.

The following status codes appear in the KDD_JOB table when a job is processing:

- RUN (running): Implies that the job is running.
- FIN (finished): Indicates that the job finished without errors.
- ERR (error): Implies that the job terminated due to an error.

5.3.1 Starting Behavior Detection Jobs

The OFSBD administrator starts jobs by running the *start_mantas.sh* script.

To start a new job in OFSBD, follow these steps:

1. Create the new job and job description through an OFSBD Job Editor in the Scenario Manager.

   OFSBD automatically assigns a unique ID to the job when it is created.

2. Associate the new job to a Job Template Group using the KDD_JOB_TEMPLATE table. Refer to Understanding the OFSBD Job Protocol.

3. Execute the `start_mantas.sh` script as follows: `start_mantas.sh <template id>`

The following events occur automatically:

a. The job goes into the job queue.

b. The dispatcher starts the job in turn, invoking the MANTAS process and passing the job ID and the thread count to the MANTAS process.

c. The MANTAS process creates the run entries in the OFSBD metadata tables. Each job consists of one or more runs.

d. The MANTAS process handles the job runs.

After a job runs successfully in OFSBD, you can no longer copy, edit, or delete the job. The start_mantas.sh script waits for all jobs in the template group to complete.

5.3.2 Starting Jobs Without the Dispatcher

Clients who use multiple services to run jobs for one OFSBD database must run the jobs without dispatcher processes. If the client does use dispatchers on each machine, each dispatcher may run each job, which causes duplicate detection results.

To run a job template without a dispatcher, add the parameter **-nd** to the command line after the template ID, as follows: `start_mantas.sh <template id> -nd`

Doing so causes the start_mantas.sh script to execute all jobs in the template, rather than depending on the dispatcher to run them. The jobs in the template group run in parallel.

The dispatcher can ensure that it is only running a set number of max jobs at any given time (so if the max is set to 10 and a template has 20 jobs associated to it, only 10 run simultaneously). When running without the dispatcher, you must ensure that the number of jobs running do not overload the system. In the event a job run dies unexpectedly (that is, not through a caught exception but rather a fatal signal), you must manually verify whether any jobs are in the RUN state but do not have a MANTAS process still running, which would mean that the job threw a signal. You must update the status code to **ERR** to restart the job.

To start a new job in Behavior Detection Framework without the dispatcher, follow these steps:

1. Create the new job and job description through an OFSBD Job Editor.

   OFSBD automatically assigns a unique ID to the job when it is created.

2. Associate the job to a Job Template Group using the *KDD_JOB_TEMPLATE* table.

3. Execute the start_mantas.sh script with the following parameters:

```
start_mantas.sh <template id> [-sd DD-MON-YYYY]
[-ed DD-MON-YYYY] [-nd]
```

   where the optional job parameters **-sd** and **-ed** (start date and end date, respectively) are used to constrain the data that an algorithm job pulls back.

   For example, if these parameters are passed into an Alert Creator job, the Alert Creator considers only matches for a grouping that has a creation date within the range that the parameters specify.

> **Note:**
>
> After a job runs successfully in OFSBD, you can no longer copy, edit, or delete the job.

5.3.3 Restarting a Job

If the dispatcher stops, all jobs stop. You must restart the dispatcher and restart all jobs, including the job that generated real errors.

Restarting a job is necessary when one or both of the following occurs:

- The dispatcher generates errors and stops during MANTAS processing. When the dispatcher is running, the OFSBD administrator can restart a job (or jobs) by changing each job's status code from ERR to RES.
- A job generates errors and stops during MANTAS processing. If a job stops processing due to errors, correct the problems that caused the errors in the job run and restart the job.

> **Note:**
>
> If the dispatcher has stopped, you must restart it.

To restart a job, follow these steps:

1. Type `restart_mantas.sh <template group id>` at the system prompt.
2. Press **Enter**.

   When the dispatcher picks up a job from the job queue that has a code of RES, it automatically restarts the job (Refer to Starting Behavior Detection Jobs). By default, the *restart_mantas.sh* script looks for jobs run on the current day.

   To restart a job that was run on a specific date, you must provide the optional date parameter such as `restart_mantas.sh <template group id> <DD-MON-YYYY>`.

5.3.4 Restarting Jobs Without the Dispatcher

Restarting a job without the dispatcher is necessary when a job generates errors and stops during MANTAS processing. If a job stops processing due to errors, correct the problems that caused the errors in the job run and restart the job.

To start a new job in OFSBD, execute the *restart_mantas.sh* script with the following parameters:

```
restart_mantas.sh <template id> [-sd DD-MON-YYYY] [-ed DD-MON-YYYY] [-nd]
```

where the optional job parameters **-sd** and **-ed** (start date and end date, respectively) are used to constrain the data that an algorithm job pulls back.

5.3.5 Stopping Jobs

It may be necessary to stop one or more job processes when dispatcher errors, job errors, or some other event make it impossible or impractical to continue processing. In addition to stopping the processes, administrative intervention may be necessary to resolve the cause of the errors.

To stop a job, you must stop its associated MANTAS process. To obtain the process IDs of active jobs and mantas processes, follow these steps:

1. Type `ps -efw | grep mantas` and press **Enter** at the system prompt.

   The MANTAS processes that are running appear on the computer screen as shown in the following example:

   ```
   00000306 7800 1843 0 Jul 16 ttyiQ/iAQM 0:00
   /kdd_data1/kdd/server/bin/mantas -j 123
   ```

   The MANTAS process ID number appears in the first display line in the second column from the left (**7800**). The job ID number appears in the second display line in the last column (**-j 123**).
2. Find the job and MANTAS process ID that you want to stop.
3. Type `kill <mantas process ID>` at the system prompt and press **Enter**.

   This command stops the MANTAS process ID, which also stops its associated job.

5.3.6 Monitoring and Diagnosing Jobs

In addition to the dispatch.log file that records events for all jobs, the system creates a job log for each job.

A job log records only the events that are applicable to that specific job. By default, a job log resides in the `$KDD_PRODUCT_HOME/logs` directory. You can configure the location of this log in the `<OFSAAI Installed Directory>/behavior_detection/algorithms/MTS/mantas_cfg/install.cfg` file.

$KDD_PRODUCT_HOME is the path of `<OFSAAI Installed Directory>/behavior_detection/ algorithms/MTS`

If you do not know the location of the log directory, check the install.cfg file. The `log.mantaslog.location` property indicates the log location. The default is `$KDD_PRODUCT_HOME/logs`, but this location is configurable.

When troubleshooting a job processing problem, first look at the file dispatch.log for the sequence of events that occurred before and after errors resulted from a job. Then, look at the job log to diagnose the cause of the errors. The job log provides detailed error information and clues that can help you determine why the job failed or generated errors.

The log file name for a job appears in the following format in the log directory:

```
job<job_id>-<date>-<time>.log
```

where **<job_id>** is the job ID and **<date>** and **<time>** represent the job's starting timestamp.

If the job errors occurred due to a problem at the system level, you may must resolve it. If you believe that the job errors were generated due to incorrect setups in OFSBD, you should notify the System Administrator, who can correct the problem setups.

> **Note:**
>
> - The dispatch.log may contain a JVM core dump. This does not indicate the actual cause of an error. In order to find the underlying error, you must refer to the job log.
> - If the scenario execution is successful but the log shows ORA-00001: unique constraint violation error for table *KDD_PTTRN_ROLE* (where DPLY_PSTN_CT = 999), this error can be ignored.
>
> This error indicates a timing issue of the jobs running in parallel. One job is performing the insert during the check and insert of another job.
>
> If a record was already inserted by another parallel job then the algorithm writes this error to log, re-attempts selecting the existing variable and successfully continues the work.

To monitor a specific job or to look at the job log history for diagnostic purposes, follow these steps:

1. Type `tail -f <log>` at the system prompt and press **Enter**, where **<log>** is the name of the job log file. The job log scrolls down the screen.
2. Press **Ctrl+C** to stop the display.
3. Type `lpr job<job_id>-<date>-<time>` at the system prompt and press **Enter** to print the job log.

> **Note:**
>
> Caution: This job log file may be a lengthy printout.

5.4 Clearing Out the System Logs

Periodically, you must clear out the dispatch and job log files. Otherwise, the files become so large that they are difficult to use as diagnostic tools and their size can impact the performance of the system.

Oracle recommends that the Oracle client establish a policy as to the frequency for clearing the logs and whether to archive them before clearing.

> **Note:**
>
> Caution: Before you shut down the dispatcher to clear the system logs, verify that no jobs are active.

Clearing the Dispatch Log

To clear the dispatch.log file, follow these steps:

1. Shut down the dispatcher by following the procedure for Stopping the Dispatcher.
2. Type `cd <$KDD_PRODUCT_HOME>/logs` at the system prompt, where **<$KDD_PRODUCT_HOME>** is your product server installation directory.

3. Type `rm dispatch.log` to clear the dispatcher log.
4. Type `start_chkdisp.sh <sleep time>` and press **Enter** to restart the dispatcher. Refer to Starting the Dispatcher for more information.

Clearing the Job Logs

To clear the job logs, follow these steps:

1. Stop the dispatcher. (Refer to Stopping the Dispatcher for more information).
2. Type `cd <directory>` at the system prompt, where **<directory>** is your log directory. By default, a job log resides in the directory *$KDD_PRODUCT_HOME/logs*. You can configure the location of this log in the `<OFSAAI Installed Directory>/behavior_detection/algorithms/MTS/mantas_cfg/install.cfg file`.
   If you do not know the location of the log directory, check the install.cfg file. The *log.mantaslog.location* property indicates the log location; the default is *$KDD_PRODUCT_HOME/logs* but this location is configurable.
3. Do either of the following:
   - Type `rm job<job_id>-<date>-<time>.log` at the log directory prompt to clear one job log, where **<job_id>-<date>-<time>** is the name of a specific job log.
   - Type `rm job*` to clear all job logs.

5.5 Recovering Jobs from a System Crash

If the system crashes, all active jobs (**status_cd = RUN**) fail. You can recover the jobs by running the script recover_mantas.sh.

This script changes the status_cd to **RES** so that these jobs can restart and finish running. The recover_mantas.sh script has an optional parameter—the date on which the system ran the start_mantas.sh script. This parameter has a `DD-MM-YYYY` format. The default value is the current date.

Running the recover_mantas.sh script with this parameter ensures the script recovers only the jobs started that day. The dispatcher must be running to pick up the restarted jobs. This results in either a successful completion (**status_cd = FIN**) or failure (**status_cd = ERR**).

You can restart jobs that ended in failure by running the restart_mantas.sh script. The `restart_mantas.sh <template group id>` script changes the status_cd from **ERR** to **RES** for any jobs passed in the template group that have a *status_cd* of **ERR** for the dispatcher to pickup.

5.6 Executing Batches Through the OFSAAI User Interface

System Administrators can run Behavior Detection jobs and Post Processing jobs from the OFSAAI UI. Activities can be performed through a batch process that can be executed once a year or periodically such as Daily, Weekly, Monthly, Quarterly, and Half-yearly depending on a firm's requirement.

> **Note:**
>
> For the batches to start, iccserver, router, AM and message server must be started in the same sequence as mentioned. For more information on starting servers, refer to the *Oracle Financial Services Advanced Analytical Applications Infrastructure (OFS AAAI) Applications Pack Installation and Configuration Guide.*

This section includes the following topics:

- Adding Behavior Detection Batches
- Adding Tasks to a BD Batch
- Setting Task Precedence
- Running a Single Task Using a Batch
- Scheduling a Batch Once
- Scheduling a Daily Batch
- Scheduling a Weekly Batch
- Configuring a Monthly Batch
- Monitoring a Batch After Execution
- Canceling a Batch After Execution
- Re-starting a Batch
- Re-running a Batch

> **Note:**
>
> Available cursors in database should be set to a minimum of 1000. Before restarting the Webserver, dispatcher should be ended.

5.6.1 Adding Behavior Detection Batches

To add a batch, follow these steps:

1. Log in as the Alert Viewer Administrator. The OFSAAI Applications page is displayed.
2. Click **Financial Services Money Laundering**.
3. In the Navigation List, select **Common Tasks**, then select **Operations**, then **Batch Maintenance**. The Batch Maintenance page is displayed.
4. In the Batch Name section, click **Add**. The Add Batch Definition page is displayed.
5. Enter the batch details as described in the following table:

Table 5-6 New Batch Details

Field	Description
Batch Name	Enter the name for the new batch.
Batch Description	Enter a description for this batch.

Table 5-6 (Cont.) New Batch Details

Field	Description
Duplicate Batch	Select this check box if the batch is a duplicate batch.
Sequential Batch	Select this check box if the batch must be run sequentially to another batch.
Batch ID	The Batch ID will be auto-populated.

6. Click **Save**. The added batch appears in the Batch Name section of the Batch Maintenance page.

5.6.2 Setting up Ingestion through AAI

Ingestion through AAI can be achieved by calling the customized shell scripts from the OFSAA Framework Batch Operations Module.

The following scripts can be customized through OFSAAI:

- set_mantas_date.sh
- start_mantas_batch.sh
- runDP.sh
- runDL.sh
- execute.sh
- runFDT.sh
- end_mantas_batch.sh
- process_firm_summary.sh
- process_market_summary.sh

The custom shell script must be kept under `<FIC_HOME>/ficdb/bin` and associated to an OFSAAI Data Transformation (DT).

The following Custom shell scripts are present in `<FIC_HOME>ficdb/bin`,which can be used directly in OFSAAI Data Transformation (DT).

- SetMantasDate.sh
- StartMantasBatch.sh
- AlertAssignment.sh
- EndMantasBatch.sh

For more information about OFSAAI Data Transformation (DT), refer to *Post Load Changes* in the *Oracle Financial Services Analytical Applications Infrastructure User Guide*.

Similarly, you must create custom shell scripts for the following and associate them to an OFSAAI Data Transformation (DT).

- runDP.sh
- runDL.sh
- execute.sh
- runFDT.sh

- process_firm_summary.sh
- process_market_summary.sh

5.6.3 Setting Task Precedence

After you have created a task, you must indicate which tasks must be executed prior to the newly created task in a batch.

1. Log in as the Alert Viewer Administrator. The OFSAAI Applications page is displayed.
2. Click **Financial Services Money Laundering**.
3. In the Navigation List, select **Common Tasks**, then select **Operations**, then **Batch Maintenance**. The Batch Maintenance page is displayed.
4. In the **Batch Name** section, select the batch that you want to set task precedence for.
5. In the **Task Details** section, click  . The Task Precedence Mapping window is displayed.
6. Move the tasks which must be executed prior to this task from the **Available Tasks** pane to the **Selected Task**s pane.
7. Click **OK** after you have selected all tasks which must precede the task. The selected tasks are listed in the Precedence column of the Task Details section.

5.6.4 Running a Single Task Using a Batch

From the Batch Execution page, you can also run a single task from a batch.

> **Note:**
>
> Running a single task using a batch is not a recommended approach and should be done only for debugging a particular task.

To run a single task using a batch, follow these steps:

1. Login as the Alert Viewer Administrator. The OFSAAI Applications page is displayed.
2. Click **Financial Services Money Laundering**.
3. In the Navigation List, select **Common Tasks**, then select **Operations**, then **Batch Execution**. The Batch Execution page is displayed.
4. In the Batch Details section, select the particular batch that you want to execute.
5. In the Task Details section, click **Exclude/Include**. The Task Mapping window is displayed.
6. Retain the tasks that you want to execute under **Available Tasks** section and move the rest to the **Set Tasks** section.
7. Click **OK**. The following warning message is displayed: *If you exclude a task, it will be skipped when executing the batch but, the precedence will not be altered. Do you want to exclude the selected tasks)?*
8. Click **OK**.
9. Click **Execute Batch**.

5.6.5 Scheduling a Batch Once

To schedule a batch that you want to run only once, follow these steps:

1. Log in as the Alert Viewer Administrator. The OFSAAI Applications page is displayed.
2. Click **Financial Services Money Laundering**.
3. In the Navigation List, select **Common Tasks**, then select **Operations**, then **Batch Scheduler**. The Batch Scheduler page is displayed.
4. Select a batch that you want to schedule from the list of available batches. The Batch Scheduler section is expanded and displays additional options.
5. Click **New Schedule**.
6. Set the frequency of the new schedule as **Once**.
7. Enter the schedule time of the batch by specifying the **Start Date** and the **Run Time**.
8. Click **Save**. The batch will run at the specified date and time.

5.6.6 Scheduling a Daily Batch

To schedule a batch that you want to run daily, follow these steps:

1. Log in as the Alert Viewer Administrator. The OFSAAI Applications page is displayed.
2. Click **Financial Services Money Laundering**.
3. In the Navigation List, select **Common Tasks**, then select **Operations**, then **Batch Scheduler**. The Batch Scheduler page is displayed.
4. Select a batch that you want to schedule from the list of available batches. The Batch Scheduler section is expanded and displays additional options.
5. Click **New Schedule**.
6. Set the frequency of the new schedule as **Daily**.
7. Enter the schedule time of the batch by specifying the **Dates**, **Run Time**, and **Every** information.
8. Click **Save**. The batch will run at the specified date and time.

5.6.7 Scheduling a Weekly Batch

To schedule a batch that you want to run weekly, follow these steps:

1. Log in as the Alert Viewer Administrator. The OFSAAI Applications page is displayed.
2. Click **Financial Services Money Laundering**.
3. In the Navigation List, select **Common Tasks**, then select **Operations**, then **Batch Scheduler**. The Batch Scheduler page is displayed.
4. Select a batch that you want to schedule from the list of available batches. The Batch Scheduler section is expanded and displays additional options.
5. Click **New Schedule**.
6. Set the frequency of the new schedule as **Weekly**.
7. Enter the schedule time of the batch by specifying the **Dates**, **Run Time**, **Every** , and **Working days of the Week** information.

8. Click **Save**. The batch will run at the specified date and time.

5.6.8 Configuring a Monthly Batch

To schedule a batch that you want to run monthly, follow these steps:

1. Log in as the Alert Viewer Administrator. The OFSAAI Applications page is displayed.
2. Click **Financial Services Money Laundering**.
3. In the Navigation List, select **Common Tasks**, then select **Operations**, then **Batch Scheduler**. The Batch Scheduler page is displayed.
4. Select a batch that you want to schedule from the list of available batches. The Batch Scheduler section is expanded and displays additional options.
5. Click **New Schedule**.
6. Set the frequency of the new schedule as **Monthly**.
7. Enter the schedule time of the batch by specifying the **Dates** and **Run Time** information.
8. Click **Save**. The batch will run at the specified date and time.

5.6.9 Monitoring a Batch After Execution

Monitoring a batch helps you track the status of execution of an individual task that was included in the batch. Through monitoring, you can also track the batch status which in turn helps you in debugging.

To monitor a batch after it is executed, follow these steps:

1. Log in as the Alert Viewer Administrator. The OFSAAI Applications page is displayed.
2. Click **Financial Services Money Laundering**.
3. In the Navigation List, select **Common Tasks**, then select **Operations**, then **Batch Monitor**. The Batch Monitor page is displayed.
4. Select a batch from the **Batch Details** lists that you want to monitor.
5. From **Batch Run Details** section, select an **Information Date** and the **Batch Run ID** from the drop-down list.
6. Click **Start Monitoring** to start the monitoring. The Batch Status, Task Details, and Event Log sections are populated with information about this batch's execution.

5.6.10 Canceling a Batch After Execution

Cancellation of a batch cancels a current batch execution.

> **Note:**
>
> This is not recommended and should be done only when the batch was fired accidentally or when a particular is taking too long to execute.

To cancel a batch after it is executed, follow these steps:

1. Log in as the Alert Viewer Administrator. The OFSAAI Applications page is displayed.
2. Click **Financial Services Money Laundering**.

3. In the Navigation List, select **Common Tasks**, then select **Operations**, then **Batch Cancellation**. The Batch Cancellation page is displayed.
4. Under the **Batch Details** section, select the batch whose execution you want to cancel.
5. Click **Cancel Batch**.

5.6.11 Re-starting a Batch

You can restart a batch execution when they have fail in their execution. When you restart a batch, it starts from the task at which it had failed. This happens when the failed task issue is debugged and resolved.

> **Note:**
>
> It is recommended that you debug and resolve a failed task before restarting the batch execution.

To restart a batch execution, follow these steps:

1. Log in as the Alert Viewer Administrator. The OFSAAI Applications page is displayed.
2. Click **Financial Services Money Laundering**.
3. In the Navigation List, select **Common Tasks**, then select **Operations**, then **Batch Execution**.The Batch Execution page is displayed.
4. Select **Restart** from the Batch Mode section.
5. Select the batch from the Batch Details section that you want to restart.
6. Select the **Information Date** and **Batch Run ID** for the selected batch from the drop-down list.
7. Click **Execute Batch**.

5.6.12 Re-running a Batch

You can rerun a batch execution when you want all the tasks from a successful batch execution to be executed again from the beginning. When a successfully executed batch is rerun, a different Batch Run ID is created for each instance for the same Information Date.

> **Note:**
>
> Creation of different Batch Run ID for each rerun of a batch is optional depending upon a firm's requirement.

To rerun a batch, follow these steps:

1. Log in as the Alert Viewer Administrator. The OFSAAI Applications page is displayed.
2. Click **Financial Services Money Laundering**.
3. In the Navigation List, select **Common Tasks**, then select **Operations**, then **Batch Execution**.The Batch Execution page is displayed.
4. Select **Rerun** from the Batch Mode section.

5. Select the batch from the Batch Details section that you want to rerun.
6. Select the **Information Date** and **Batch Run ID** for the selected batch from the drop-down list.
7. Click **Execute Batch**.

6
Post-Processing Tasks

During post-processing of ingested data, Behavior Detection prepares the detection results for presentation to users.

Preparation of the results depends upon the following processes:

- Augmentation: Collects information for pattern detection, which enables proper display or analysis of these results may be required. This process is automatically executed at the end of each scenario run.
- Match Scoring: Computes a ranking for scenario matches indicating a degree of risk associated with the detected event or behavior.
- Alert Creation: Packages the scenario matches as units of work (that is, alerts), potentially grouping similar matches together, for disposition by end users.
- Update Alert Financial Data: Records additional data for alerts such as the related Investment Advisor or Security involved in the alert.
- Alert Assignment: Determines the user or group of users responsible for handling each alert.
- Auto-Close (optional): Closes alerts that are of a lower priority to the business.
- Automatic Alert Suppression (optional): Suppresses alerts that share specific scenario and focal entity attributes for a particular time frame.
- Highlight Generation: Generates highlights for alerts that appear in the alert list in the Alert Viewer subsystem and stores them in the database.
- Historical Data Copy: Identifies the records against which the current batch's scenario runs generated alerts and copies them to archive tables.

> **Note:**
>
> You can re-run any failed post-processing job.

Order of Running Post-Processing Administrative Tasks

Run the post-processing administrative tasks in this order:

1. Match Scoring(501)
2. Multi Match Alert Creation (502)
3. Single Match Alert Creation(503)
4. Update Alert Financial Data
5. Alert Scoring(504)
6. Alert Assignment
7. Auto-Close(506)
8. Highlight Generation

9. Historical Data Copy

> **Note:**
>
> For all the post processing jobs MANTAS batch should be up and running.

6.1 Match Scoring

Behavior Detection provides a mechanism to compute a score for matches to provide an initial prioritization.

Match Scoring rules are created using the Scoring Editor from the Administration Tools. Refer to the *Administration Tools User Guide* for more information.

Running the Match Scoring Job

The Match Scoring job is part of the Behavior Detection subsystem. Behavior Detection delivers job template group **501** to run the Match Scoring job.

To run the Match Scoring job, follow these steps:

1. Verify that the dispatcher is running.
2. Run the `start_mantas.sh <template id>` script as follows: `start_mantas.sh 501`

All new matches in the system are scored.

6.2 Alert Creation

Matches are converted into alerts with the Alert Creator processes. These processes are part of the Behavior Detection subsystem.

The system uses two types of Alert Creator jobs:

- Multi-match Alert Creator: Generates alerts for matches that share a common focus, are from scenarios in the same scenario group, and possibly share other common attributes. Each focus type has a separate job template.
- Single-match Alert Creator: Generates one alert per match.

> **Note:**
>
> The *KDD_JRSDCN* table is empty after system initialization and requires populating before the system can operate. If a new jurisdiction is to be added, it should be added to KDD_JRSDCN table.

Running the Alert Creation Job

The Alert Creator is part of the Behavior Detection subsystem. Behavior Detection provides default job templates and job template groups for running Alert Creator. These jobs can be modified using Administration Tools. Refer to the *Administration Tools User Guide*for more information.

The following sections describe running each type of Alert Creator.

Run Multi-match Alert Creator

To run the multi-match Alert Creator, follow these steps:

1. Verify that the dispatcher is running.
2. Run the start_mantas.sh script as follows: `start_mantas.sh 502` where **502** is the job template that Behavior Detection provides to run the Alert Creator algorithm.

Run Single Match Alert Creator

To run the single match Alert Creator, follow these steps:

1. Verify that the dispatcher is running.
2. Run the start_mantas.sh script as follows: `start_mantas.sh 503` where **503** is the job template that Behavior Detection provides to run the Alert Creator algorithm.

6.2.1 Understanding Advanced Alert Creator Configuration

The Alert Creator algorithm can support grouping strategies that the Administration Tools do not support. To use these advanced strategies, you must enter Alert Creator rules directly into the database.

The executable retrieves new, unowned single matches generated from specified types of scenarios. It then groups them based on one of four implemented algorithms and a specified list of bindings for grouping. It requires parameter settings to designate the following:

- Choice of grouping algorithm to use.
- Scenario types associated with the set of matches to consider for grouping.
- Bindings on which to base break group compatibility

Grouping Algorithms

When grouping algorithms, choose from the following:

- BIND_MATCH: The Alert Creation module creates alerts based on matches with matching bindings/ values based on a provided list of bindings to use when determining groupability.
- BIND_BEHAVIOR_SCENARIO_CLASS: The Alert Creation module creates alerts based on matches with matching scenario group code and with matching bindings/values based on a provided list of bindings to use when determining groupability.
- BIND_BEHAVIOR_SCENARIO: The Alert Creation module creates alerts based on matches with matching scenario ID and with matching bindings/values based on a provided list of bindings to use when determining groupability.
- BIND_BEHAVIOR_PATTERN: The Alert Creation module creates alerts based on matches with matching pattern ID and with matching bindings/values based on a provided list of bindings to use when determining groupability.
- SINGLE_ALERT_MATCH: The Alert Creation module creates alerts for all remaining matches. A alert is created for each of the remaining matches, as long as they bind one of the centricity names in the bindings string. This is the catch all algorithm that ensures that all matches that have a bound centricity value and a corresponding alert is created.

For a BIND_MATCH grouping rule, the system compares bindings (KDD_BREAK_BINDING) values for matches to determine whether it can group matches together into an alert.

For example, the grouping algorithm interprets `!TRADER ?ASSOC_SCRTY` to create an alert; each break set to be grouped must have a TRADER binding in which the values for that binding must match and each must either have an ASSOC_SCRTY binding in which the values match OR each must be missing the ASSOC_SCRTY binding. Alerts that mentioned ASSOC_SCRTY could only be grouped with other alerts that mentioned ASSOC_SCRTY. Similarly, alerts that did not mention ASSOC_SCRTY could only be grouped with other alerts that did not mention ASSOC_SCRTY.

This list is order-dependent and at least one binding should be marked as required using an exclamation point (!) to prevent grouping of all miscellaneous matches into one big break. The order helps determine the centricity in the first binding name in the binding string. The centricity name is used to determine the alert's centricity ID.

6.3 Update Alert Financial Data

OFSBD provides some enhanced data on alerts to support searching by alerts based on business data. For example, Trader-focused alerts may be searched based on the security involved in the activity. Update Alert Financial Data is the process that populates this information.

To update alert financial data, run the following command from the `<OFSAAI Installed Directory>/database/db_tools/bin` directory:

```
upd_kdd_review_fin.sh <batch_id> <YYYYMMDD>
```

If <*batch_id*> and the batch date <*YYYYMMDD*> are not provided, the system derives this data for matches created in the current batch.

The log for this process is under the <*OFSAAI Installed Directory*>*/database/db_tools/logs* directory. The name of the file is *run_stored_procedure.log*.

6.4 Alert Assignment

OFSBD provides a mechanism to assign alerts to a predefined owner (either an individual user or a pool of users). When performing alert assignment, the module fetches new, unowned alerts for a given product and assigns them to an owner using a rule-based strategy.

You can configure assignment rules by using the Administration Tools. Refer to the *Administration Tools User Guide*, for more information.

The assignment framework allows customers to write their own Java code to replace the product functionality with their own customized functionality. The modules that can be replaced include the assignment-eligible objects, the assignment rule processing logic, and the manner in which the assignment results are output (currently results are written out to the database for batch assignment, or passed back in a SOAP XML response for the assignment web services call). For more information on how to take advantage of this feature, please contact Oracle Support.

Running the Alert Assignment Job

The Alert Assignment Job is part of the OFSBD subsystem.

To run an Alert Assignment job, run the execute.sh script as follows: `<OFSAAI Installed Directory>/bdf/scripts/execute.sh AlertAssignment`

By default, Behavior Detection writes log messages for this script in the `<OFSAAI Installed Directory>/bdf/logs/<Processing Date>/AlertAssignment.log` file.

6.5 Auto-Close

OFSBD provides a mechanism to close alerts automatically that do not warrant investigation.

The system can close alerts based on their age, status, score, focus type, generating scenario, or any combination of these attributes. The system regularly evaluates all candidate alerts and closes each alert that satisfies the criteria. The system maintains closed alerts for audit purposes and they are still available for display such as from the Relationship tab in the OFSBD UI) and processing , such as by reopening an alert.

Defining the Auto-Close Alert Algorithm

The *KDD_AUTO_CLOSE_ALERT* table provides all operation sets, and their respective operations, that the system uses to determine whether it should close an alert. The table includes the following:

- Operations are logical expressions that can be used to close alerts such as *alert score > 50*, *age > 30*. A set of operations based on the same attribute, such as score, form an operation set.
- The *OPRTN_SET_ID* column is a grouping of mutually exclusive operations. Each operation specifies the next step that is applied to alerts that satisfy the operation. This next step is either to close the alert or execute the Next operation Set (NEXT_OPRTN_SET_ID column), or branch to further evaluate the alerts.
- The *XPRSN_ORDER_ID* column sets up an order of precedence by which the system attempts to satisfy the operations. Enter **NULL** if the entry is linked from another entry that has a value in the *XPRSN_ORDER_ID* column.
- The *ALERT_ATTR_ID* column identifies the attribute of the alert for evaluation.
- The *OPRTR_CD* column specifies the type of operation to be performed. Allowed values are `=, !=, >, <, >=, <=, contains, or IN`. While using the **IN** operator, the right-hand side variables should be separated by**|** such as `NW|OP`.
- The value in the *VALUE_TX* column provides the right-hand side of the operation being evaluated.
- If the current operation is satisfied, and it is not the final operation in the operation set (indicated by a **NULL** value in the *NEXT_OPRTN_SET_ID* column), the process jumps to the *NEXT_OPRTN_ SET_ID*. If the NEXT_OPRTN_SET_ID is **NULL**, and the operation is **true**, the system closes the alert.
- The *DMN_CD* column is the OFSBD product code.
- The *CLS_ACTIVITY_TYPE_CD* column specifies the activity type code of the closing action to associate with an alert that is closed by this rule. This column is optional. If the column is **NULL**, the system uses the default auto-close activity type code.
- The *CMMNT_TX* column specifies an optional text comment to associate with an alert that is closed by this rule.

The Auto-Close Alert algorithm does not close a locked alert. The system locks an alert when an analyst investigates it, and then unlocks it when the analyst releases it. All locked alerts are skipped until the next time the Auto-Close Alert algorithm is run. The OFSBD administrator must fill in rows in the KDD_AUTO_CLOSE_ALERT table with the criteria for auto-closing the alerts. The system uses the KDD_REVIEW table to provide available attributes for use in the Auto-Close algorithm.

Set Up Auto-Close Rules

To set up auto-close rules, formulate the criteria for auto-closing alerts using the attributes in the Alert Closing Attributes (KDD_AUTO_CLOSE_ALERT) table. The Alert Identifier (ALERT_ATTR_ID) column is needed later in this set of instructions.

The following table describes commonly used Alert Closing Attributes.

Table 6-1 Commonly Used Alert Closing Attributes

Alert Attribute	Alert Identifier (ALERT_ATTR_ID)
Alert Age	113000057
Due Date	113000024
Focus Type	113000010
Last Action	113000038
Owner's Organization	113000056
Previous Match Count All	113000054
Previous Match Count Same Scenario	113000053
Scenario	113000013
Score	113000022
Status	113000008
Status Name	113000055
Processing Batch Name	113000068
Jurisdiction	113000067
Previous Match Count Same Scenario Group	113000064
Scenario Group	113000014

View All Alert Closing Attributes

To view a full set of Alert Closing Attributes, follow these steps:

1. Run the following query:

```
Select A.ATTR_ID, A.ATTR_NM
From KDD_ATTR A, KDD_DATASET_ATTR B
where A.ATTR_ID=B.ATTR_ID and B.DATASET_ID=113000002
```

> **Note:**
>
> If the alert attribute that corresponds with a particular alert identifier contains a NULL value, the Auto-Close algorithm does not interpret these values and returns a fatal Behavior Detection error.

2. Formulate operations for the auto-closing criteria. Operations contain only one mathematical operator such as `>`, `<`, `or =`. Operation sets include one or more operations chained together by the *NEXT_OPRTN_SET* column.
3. Determine an order of precedence for the operations (that is, what to test first, second, and so forth). Each operation's precedence must be unique within the KDD_AUTO_CLOSE_ALERT table. An error occurs if two operations have the same precedence. All operations must have precedence or the system does not test them.

4. Assign an operation ID to each operation. This ID must be unique within KDD_AUTO_CLOSE_ALERT.
5. Assign an operation ID to each operation within each operation set. Use IDs close together for operations within the same operation set. The system uses this ID to link together operations within the same operation set by placing the next ID for testing in the Next Operation ID (*NEXT_OPRTN_SET_ID*) column.
6. Determine the rows to insert into the KDD_AUTO_CLOSE_ALERT table from the following columns:
   - OPRTN_SET_ID is the operation set ID.
   - XPRSN_ORDER_ID, the operation ID, the precedence must be unique for each operation across the table. This column can contain a NULL value.

     > **Note:**
     >
     > When an operation set is reached by linking from another operation set, you can leave the *XPRSN_ORDER_ID* at **NULL**. For operations sets that are not reached through another operation set, the XPRSN_ORDER_ID is required.

   - ALERT_ATTR_ID (Refer to Step 1).
   - OPRTR_CD is the mathematical operator for the operation.
   - VALUE_TX is the right-hand side of the operation.
   - NEXT_OPRTN_SET_ID is the ID that identifies the next operation in the operation set, or **NULL** if no operations exist. Inserting an ID into the *NEXT_OPRTN_SET* column previously called creates a loop and results in an error.
   - DMN_CD is the OFSBD product code.
   - The CLS_ACTIVITY_TYPE_CD column specifies the activity type code of the closing action. The activity type code that this column specifies must exist in the KDD_ACTIVITY_TYPE_CD table and the KDD_ACTIVITY_TYPE_CD. Verify that the *AUTO_CLOSE_FL* is set to **'Y**' for this code to be valid.
   - The CMMNT_TX column specifies an optional text comment.

Running the Auto-Close Alert

Auto-Close Alert is part of the Behavior Detection subsystem. OFSBD provides default job templates and job template groups for running Auto-Close Alert. You can modify these jobs using the Administration Tools. Refer to the *Administration Tools User Guide* for more information.

To run Auto-Close Alert, follow these steps:

1. Verify that the dispatcher is running.
2. Run the *start_mantas.sh* script as follows: `start_mantas.sh 506` where, **506** is the job template that OFSBD provides to run the Auto-Close algorithm.

6.5.1 Sample Auto-Closing Alert Rule

You may want to close an alert when the match score is less than 75 and the status code is equal to NW (New), or the review is more than 30 days old. If so, follow these steps:

1. Determine the ATTR_ID for the columns to reference in the KDD_REVIEW table.

- SCORE has ATTR_ID 113000022.
- STATUS has ATTR_ID 113000008.
- AGE has ATTR_ID 113000057.

2. Formulate the operations.

   The match score is less than 75 and the status code is equal to `NW = (SCORE < 75) AND (STATUS = NW)`

   Reviews more than thirty days old `= (AGE > 30)`

3. Determine an order of precedence for the criteria.

   For example, to determine whether reviews are more than thirty days old, assign `(AGE > 30)` a precedence of **1**, and `(SCORE < 75) AND (STATUS = NW)` a precedence of **2**.

4. Assign an operation ID to each operation within the operation set.

   The operation ID must be unique within the database. The numbers may be any number not already in the table.

   ```
    OPRTN_SET_ID 100 -> (SCORE < 75) AND (STATUS = NW)
   OPRTN_SET_ID 200 -> (AGE > 30)
   ```

5. Assign an ID to each operation within the already divided operations:

   ```
   OPRTN_SET_ID 100 -> (SCORE < 75)
   OPRTN_SET_ID 101 -> (STATUS = NW)
   OPRTN_SET_ID 200 -> (AGE > 30)
   ```

6. Assign the next operation set to chain the operations together.
   - Optionally: assign or close an activity type code and/or comment to the operation.

7. Insert the rows into the KDD_AUTO_CLOSE_ALERT table.

   The following table resembles the entries into the KDD_AUTO_CLOSE_ALERT table for the (AGE > 30) auto-close alert.

Table 6-2 KDD_AUTO_CLOSE_ALERT (AGE > 30)

OPRT-N_SET_ID	XPRSN_ORDER_ID	ALERT_ATTR_ID	OPRTR_CD	VALUE_TX	NEXT_OPRTN_SET_ID	DMN_CD	CLS_ACTIVITY_-TYPE_CD	CMMNT_TX
200	1	1130000057	>	30	NULL	MTS	MTS 203	Close if age greater than 30

> **Note:**
>
> The NEXT_OPRTN_SET_ID is NULL because this operation set contains only one operation. The following table shows how to set it to the next operation's ID within the operation set.

The following table resembles entries into the KDD_AUTO_CLOSE_ALERT table for the (SCORE < 75) and (STATUS = NW) auto-close alert.

Table 6-3 KDD_AUTO_CLOSE_ALERT (SCORE < 75) and (STATUS = "NW")

OPRT-N_SET_ID	XPRSN_OR DER_ID	ALERT_ ATTR_ID	OPRTR_ CD	VALUE_ TX	NEXT_O PRTN_ SET_ID	DMN _CD	CLS_AC TIVITY_-TYPE_C D	CMMNT_ TX
100	2	1130000 2 2	<	75	101	MTS	NULL	NULL
101	NULL	1130000 08	=	NW	NULL	MTS	NULL	NULL

6.6 Automatic Alert Suppression

Behavior Detection provides actions that enable an analyst to specify that the system close a particular entity's alerts on a specific scenario automatically. This is called Alert Suppression. The system runs the Alert Suppression algorithm to close newly-generated alerts that match an active suppression rule.

The system can suppress alerts with the status of NEW based on their creation date, generating scenario, and focal entity. The algorithm evaluates all candidate alerts and suppresses each alert that satisfies the criteria. The suppressed alerts, to which the system assigns a status of Closed, remain for audit purposes and are still available for display, such as through the Relationship tab, and processing, such as reopening an alert.

> **Note:**
>
> Alert Suppression tables use full refresh data loading. The data is first truncated and then new data is inserted. Complete data must be provided every time these commands are executed.

Defining the Suppress Alert Algorithm

The Suppress Alert algorithm does not suppress locked alerts. The system locks an alerts while an analyst takes an action on it, and then unlocks the alert when the analyst releases it. The system skips all locked alerts until the next time it runs the Suppress Alert component. When a user takes an action on an existing alert to suppress future alerts, the suppression rule populates the KDD_AUTO_SUPPR_ALERT table with the criteria for automatically suppressing and canceling suppression of the alerts.

Running the Suppression Job

The suppression job is part of the Behavior Detection subsystem. OFSBD provides default job templates and job template groups for running Auto-Close Alert. You can modify these jobs using the Administration Tools. Refer to the *Administration Tools User Guide* for more information.

To run the suppression job, follow these steps:

1. Verify that the dispatcher is running.
2. Run the start_mantas.sh script as follows: `start_mantas.sh 507` where, **507** is the job template that OFSBD provides to run the suppression job algorithm.

6.7 Highlight Generation

The Alert Viewer subsystem displays alert and match highlights in the Alert List and Alert Context sections of the OFSBD UI.

The system calculates and stores these highlights in the database as part of the batch cycle using the following shell script: `run_highlights.ksh`

This script is part of the Database Tools that resides in the `<OFSAAI Installed Directory>/database/db_tools/bin` directory. This script attaches to the database using the user that the *utils.database.username* property identifies in the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` file. You run highlight generation after the creation of alerts and before the system ends the batch with the *end_mantas_batch.sh* script. By default, Behavior Detection writes log messages for this script in the `<OFSAAI Installed Directory>/database/db_tools/logs/highlights.log` file.

Highlight Generation Limits

The limit of highlight generation in post processing is 50,000. This limit can be increased based on your alerting data. If a scenario generates more than 50K alerts, then only the highlights of the first 50K alerts are generated. All the other highlights will be null.

Increasing Highlight Generation Limits

If your alert count exceeds 50k, follow these steps to increase the limit:

1. End the current batch, if running.
2. Change the *maxCount* value in the file found at `<FIC_HOME>/database/db_tools/mantas_cfg/ etc/xml/DB_AlertContext.xml`
   For example, for an alert count of 100: change `DataBag name="FindAlerts" table="KDD_REVIEW" maxCount="50000">` to `<DataBag name="FindAlerts" table="KDD_REVIEW" maxCount="100000">`
3. Set the batch date.
4. Start the batch.

Displaying Missing Highlights

If you are experiencing an existing highlight issue for backdated data, follow these steps to display the missing highlights:

1. End the current batch, if running.
2. Change the *maxCount* value in the file found at `<FIC_HOME>/database/db_tools/mantas_cfg/etc/xml/DB_AlertContext.xml`
   For example, for an alert count of 100: change `DataBag name="FindAlerts" table="KDD_REVIEW" maxCount="50000">` to `<DataBag name="FindAlerts" table="KDD_REVIEW" maxCount="100000">`
3. Set the batch date to the date for which the highlight issue exists.
4. Start the batch.
5. Get the previous *PRCSNG_BATCH_ID* from the KDD_PRCSNG_BATCH_HIST for the data dump date of the batch for which the highlights were **NULL**.
6. Copy the current *PRCSNG_BATCH_ID* in notepad.

7. Take the *PRCSNG_BATCH_ID* from Step 5 and manually update it in the KDD_PRCSNG_BATCH_CONTROL table.
8. Rerun the `run_highlights.ksh` and `run_hdc.ksh`.
9. Take the *PRCSNG_BATCH_ID* from Step 6 and manually update it in the KDD_PRCSNG_BATCH_CONTROL table.
10. Verify that the highlights display in the UI.

6.8 Historical Data Copy

Behavior Detection maintains records that are directly involved with detected behaviors in a set of archive, or ARC, tables. The Historical Data Copy (HDC) process identifies the records against which the current batch's scenario runs generated alerts and copies them to the ARC tables.

The run_hdc.ksh and upd_kdd_review_fin.sh must run upon completion of all detection and other alert post-processing , such as scoring and assignment, but before the system ends the batch with the following shell script: `end_mantas_batch.sh`

> **Note:**
>
> This script is part of the Database Tools that reside in the `<OFSAAI Installed Directory>/database/db_tools/bin` directory.

The `run_hdc.ksh` shell script manages the HDC process. This process connects to the database as the user that the *truncate.database.username* property identifies in the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` file. This property should identify the Atomic Schema user, a user in the database with write access to tables in Behavior Detection Atomic schema.

To improve performance, you can adjust two configurable parameters in the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` file.

Table 6-4 HDC Configurable Parameters

Parameter	Recommended Value	Descriptions
hdc.batchsize	10000	Number of break match key IDs are included in each batch thread for data retrieval.
hdc.maxthreads	2x (Number of CPUs)	Maximum number of concurrent threads that HDC uses for retrieving data to tune performance.

To run the Historical Data Copy (HDC) process, follow these steps.

1. Navigate to `<OFSAA installed directory>/database/db_tools/bin/execute run_hdcBD.ksh` By default, log messages for this script are written in the `<OFSAAI Installed Directory>/ database/db_tools/logs/hdc.log` file.
2. Verify the *ARC* tables to check the HDC data copy.

7

Managing Batch Processing Utilities

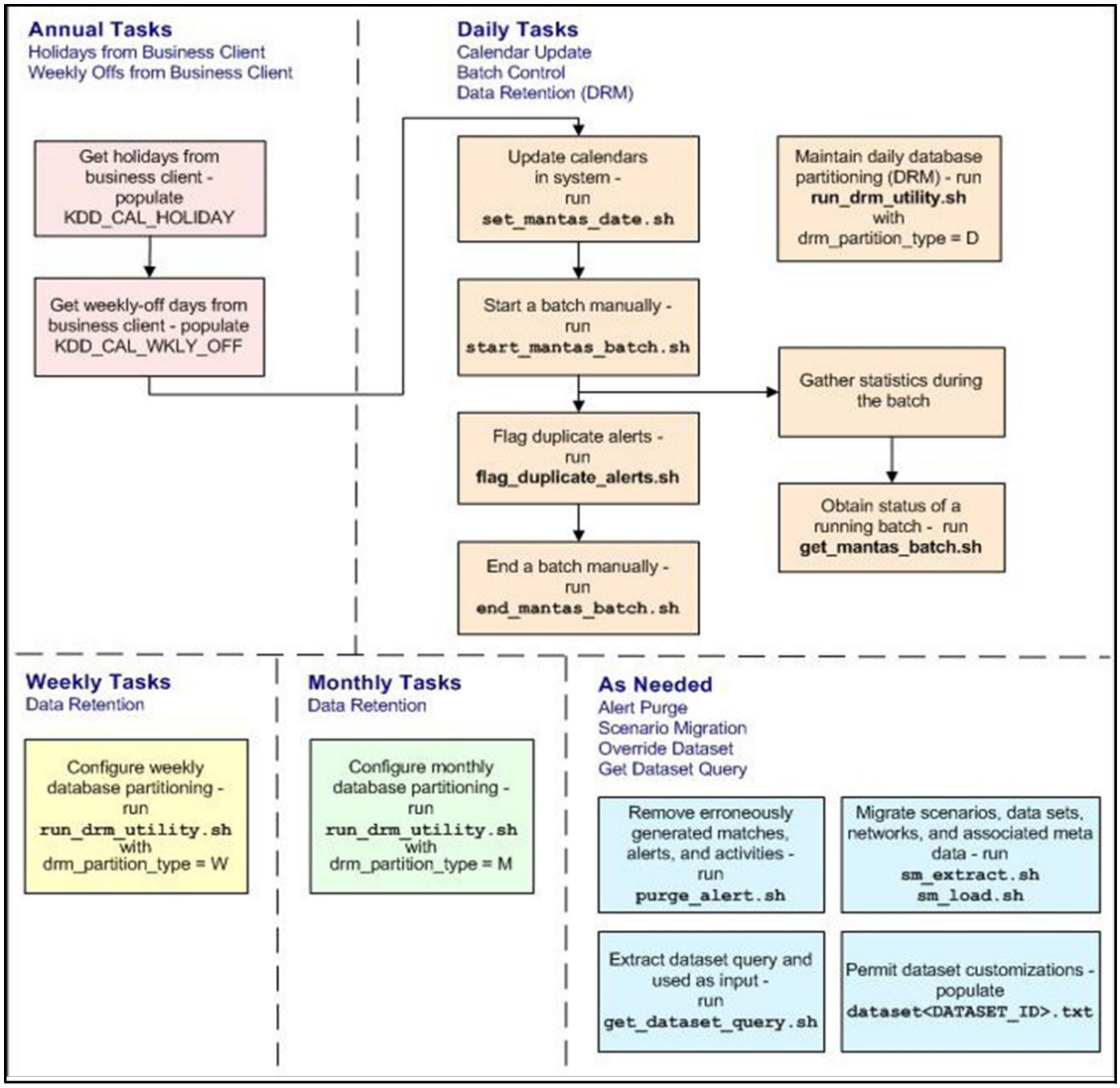
OFSBD provides utilities that enable you to set up and modify a selection of batch-related database processes.

Behavior Detection database utilities enable you to configure and perform batch-related system pre-processing and post-processing activities. These are described in the following sections:

- Managing Common Resources for Batch Processing Utilities: Configuration files enable the utilities to share common resources such as database configuration, directing output files, and setting up logging activities.
- Managing Annual Activities: Calendar management tasks must be performed at least annually, such as loading holidays and weekly off-days, to ensure accurate population of business calendars.
- Managing Alert Purge Utility: Provides the capability to remove alerts (along with their matches and activities) generated erroneously or which have exceeded the retention policies of the organization.
- Managing Batch Control Utility: Manages the start and termination of a batch process (from data management to alert post-processing) and enables access to the currently running batch.
- Managing Calendar Manager Utility: Updates calendars in the OFSBD system based on predefined business days, holidays, and days off or non-business days.
- Managing Data Retention Manager: Provides the capability to manage the processing of partitioned tables in Behavior Detection. This utility purges data from the system based on configurable retention period defined in database.
- Database Statistics Management: The system uses a script to manage Oracle database statistics. These statistics determine the appropriate execution path for each database query.
- Managing Flag Duplicate Alerts Utility: Enables you to run a script daily after the generation of alerts to identify pairs of alerts that are possible duplicates and adds a system comment to each alert.
- Managing Notifications: Enables you to configure users of the Alert Viewer subsystem to receive e-mail when alerts are assigned to them.
- Refreshing Temporary Tables: Enables you to manage temporary tables created as part of the detection process.
- Managing Truncate Manager: Calls the script to truncate tables that require complete replacement of their data.
- Managing ETL Process for Scenario Tuning: Calls the scripts to insert and update records in DATA tables.

The following image illustrates the frequency with which you use these batch-related database utilities when managing activities: daily, weekly, monthly, annually, or as needed.

Figure 7-1 Managing Database Activities with Utilities



This image illustrates the following:

- Daily tasks are initially dependent on the annual tasks that you perform, such as obtaining holiday and weekly off-days from an Oracle client.
- Daily tasks can include updating Behavior Detection calendars and managing batch processes. You may must configure data partitioning on a daily, weekly, or monthly basis.

Tasks that you perform when needed can include deleting extraneous or invalid matches and alerts, or migrating scenarios and other information from one environment to another , such as from test to production.

> **Note:**
>
> Either the Sber Bank IRS Batch or Product IRS Batch should be executed. Do not execute these batches together or in sequence.

7.1 Managing Common Resources for Batch Processing Utilities

Configuration files enable the utilities to share common resources such as database configuration, directing output files, and setting up logging activities.

Install Configuration

Configuration information resides in the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` configuration file. The configuration file contains modifiable instructions for Oracle database drivers and provides information that each utility requires. It also provides the user name and password that you must connect to the database. In this file, you can modify values of specific utility parameters, change the locations of output files, and specify database details for extraction and data loading.

The `install.cfg` file contains information unique to each utility and common configuration parameters; headings in the file clearly identify a utility's parameters. You can also modify the current logging configuration , such as activate or deactivate particular logging levels and specify locations for logging entries. This section provides a sample install.cfg file with common and utility-specific information. Logging information appears at the end of the file. You should ensure that the ATOMIC schema name is in uppercase.

```
# @(#)Copyright (c) 2018 Oracle Finanacial Services Software Inc. All Rights
Reserved.
# @(#) $Id: install.cfg $
#
# This configuration file supports the following database utilities:
#  Calendar Mangager
#  Batch Control
#  Truncate Manager
#  Scenario Migration
#  Alert Purge
#  Data Retention Manager
#  Email Notification
#  Data Analysis Tool
# The file contains some properties that are common and specific properties
for each
# of the tools.

################ COMMON CONFIGURATION ENTRIES ######################

NLS_LENGTH_SEMANTICS=CHAR
database.driverName=oracle.jdbc.driver.OracleDriver
utils.database.urlName=jdbc:oracle:thin:@ofss2221324.in.oracle.com:1521:Ti5O12
L64
utils.database.username=f802_fccm
utils.database.password=NzBXdzslR43hh0nWkaqYvA==
schema.algorithms.owner=f802_fccm
schema.algorithms.password=NzBXdzslR43hh0nWkaqYvA==
schema.web.owner=f802_fccm
schema.web.password=NzBXdzslR43hh0nWkaqYvA==
schema.report.owner=f802_fccm
schema.report.password=NzBXdzslR43hh0nWkaqYvA==

schema.mantas.owner=f802_fccm
schema.mantas.password=NzBXdzslR43hh0nWkaqYvA==
```

```
utils.miner.user=f802_fccm
utils.miner.password=NzBXdzslR43hh0nWkaqYvA==
schema.business.owner=f802_fccm
schema.business.password=NzBXdzslR43hh0nWkaqYvA==
schema.market.owner=f802_fccm
schema.market.password=NzBXdzslR43hh0nWkaqYvA==
utils.data.directory=/scratch/ofsaadb/BD802_Final/BD802FL/database/db_tools/
data
ingest.user=f802_fccm
ingest.password=NzBXdzslR43hh0nWkaqYvA==

schema.kdd.owner=f802_fccm
schema.kdd.password=NzBXdzslR43hh0nWkaqYvA==
casemng.schema.owner=f802_fccm
casemng.schema.password=NzBXdzslR43hh0nWkaqYvA==

################ CALENDAR MANAGER CONFIGURATION ##################

# The look back and look forward days of the provided date.
# These values are required to update the KDD_CAL table. The maximum look
back or forward
# is 999 days.
calendar.lookBack=400
calendar.lookForward=14

############### BATCH CONTROL CONFIGURATION ####################

# When ending the batch, age alerts in calendar or business days
age.alerts.useBusinessDays=Y

############### TRUNCATE MANAGER ################################

# Specify the database username and password for truncation manager
truncate.database.username=${ingest.user}
truncate.database.password=${ingest.password}

################ SCENARIO MIGRATION CONFIGURATION ######################

#### GENERAL SCENARIO MIGRATION SETTINGS

#Specify the flags for whether scoring rules and wrapper datasets need to be
extracted or loaded
score.include=N
wrapper.include=N

#Specify the Use Code for the scenario. Possible values are 'BRK' or 'EXP'
load.scnro.use=BRK

#If custom patterns exist for a product scenario, set to 'Y' when loading a
scenario hotfix.
#This should normally be set to 'N'.
load.ignore.custom.patterns=N

#Specify the full path of depfile and name of fixfile used for extraction and
loading
#Note : fixfile need not be specified in case of loading
```

```
sm.depfile=/scratch/ofsaadb/BD802_Final/BD802FL/database/db_tools/mantas_cfg/
dep.cfg

sm.release=5.7.1

#### EXTRACT

# Specify the database details for extraction
extract.database.password=${utils.database.password}

# Specify the case schema name for both extraction and load .
caseschema.schema.owner=f802_fccm

# Specify the jdbc driver details for connecting to the source database
extract.conn.driver=${database.driverName}
extract.conn.url=jdbc:oracle:thin:@ofss2221324.in.oracle.com:1521/Ti5O12L64

#Source System Id
extract.system.id=

# Specify the schema names for Extract
extract.schema.mantas=${schema.mantas.owner}
extract.schema.case=f802_fccm
extract.schema.business=${schema.business.owner}
extract.schema.market=${schema.market.owner}
extract.user.miner=${load.user.miner}
extract.miner.password=${utils.miner.password}

# File Paths for Extract

#Specify the full path in which to place extracted scenarios
extract.dirname=/scratch/ofsaadb/BD802_Final/BD802FL/database/db_tools/data

#Specify the full path of the directory where the backups for the extracted
scripts would be maintained
extract.backup.dir=/scratch/ofsaadb/BD802_Final/BD802FL/database/db_tools/
data/temp

#Controls whether jobs and thresholds are constrained to IDs in the product
range (product.id.range.min
# through product.id.range.max). Values are Y and N. If the range is not
restriced, you can use range.check

# to fail the extract if there are values outside the product range.

extract.product.range.only=N
extract.product.range.check=N

#### LOAD

# Specify the jdbc driver details for connecting to the target database
load.conn.driver=${database.driverName}
load.conn.url=${utils.database.urlName}

#Target System ID
load.system.id=Ti5O12L64
```

```
# Specify the schema names for Load
load.schema.mantas=${schema.mantas.owner}
load.schema.case=f802_fccm
load.schema.business=${schema.business.owner}
load.schema.market=${schema.market.owner}
load.user.miner=${utils.miner.user}
load.miner.password=${utils.miner.password}.
#Directory where scenario migration files reside for loading
load.dirname=/scratch/ofsaadb/BD802_Final/BD802FL/database/db_tools/data
# Specify whether threshold can be updated
load.threshold.update=Y
# Specify whether score can be updated
load.score.update=Y

# Specify whether or not to verify the target environment on load
verify.target.system=N

################ ALERT PURGE CONFIGURATION ##########################
# Set the Alert Purge input variables here.
# (use the word "null" as the value of any parameters that are not
#  to be used)
#

# Specify whether or not to consider Matches
limit_matches=N

# Specify whether or not to purge the data
purge=Y

# Specify batch size for which commit should perform
batch_size=5000
job=null
scenario=null
# enter dates, with quotes in the following format:
#   'DD-MON-YYYY HH24:MI:SS'
start_date=null
end_date=null
alert_status=NW

# Specify purge db user
purge.database.user=f802_fccm

# Specify purge db user password.
purge.database.password=

# Specify whether alerts has to be purged or not.
purge_alert_flag=Y

# Specify whether fatca cases/assessments has to be purged or not.
purge_fatca_flag=Y

# Specify whether case has to be purged or not.
purge_case_flag=Y

# Specify defualt rule set.
purge_default_rule_set=
```

```
# Specify total number of threads should be used for the process.
purge_threads_no=10

# Specify report directory for report on process performed.
purge_report_directory=

# Specify product version
purge_product_version=
#Base Working Directory required to put the temporary log from Database Server
ap.storedproc.logdir=/tmp

#The common Path required to put the SQL files to execute
commonSQLFilePath=/scratch/ofsaadb/BD802_Final/BD802FL/database/db_tools/data
######### DATA RETENTION MANAGER CONFIGURATION ########################
#
# Set the Data Retention Manager input variables here.
##
drm_operation=P
drm_partition_type=D
drm_owner=${schema.business.owner}
drm_object_name=A
drm_weekly_proc_fl=N
######### Email Notification ##########################################
#
# The following sections contain information on configuring email
# notification information. If you wish to use Exchange, you must purchase
# Java Exchange Connector, obtain a license and the jec.jar file. The license
# file must be placed in the mantas_cfg file, and the jec.jar file must be
# copied to the db_tools/lib directory. Then, edit the file
# db_tools/bin/run_push_email.ksh, uncomment the JEC_JARS= line.
#
#######################################################################
# Currently only smtp, smtps, or exchange
email.type=smtp

# Number of notifications that can run in parallel
notification.threads=4

# Max number of active db connections
utils.database.max_connections=4
# From address for sent mails. This is ignored in Exchange mode. If omitted
in SMTP mode, the mail account associated
# with the Unix/Linux account is used.
email.from=
# SMTP settings
email.smtp.host=mailhost.us.oracle.com
# smtp port is usually 25 for smtp, 465 for smtps
email.smtp.port=25
email.smtp.auth=false
email.smtp.user=
email.smtp.password=
email.smtp.useHTML=true
# Exchange settings *** See above for instructions to enable this ***
#  Your Exchange administrator should help identify these settings
#
```

```
email.exchange.server=
email.exchange.domain=
email.exchange.user=
email.exchange.password=
email.exchange.prefix=Exchange
email.exchange.mailbox=
email.exchange.useSSL=true
email.exchange.useFBA=true
email.exchange.useNTLM=false
email.exchange.draftsfoldername=drafts
email.exchange.useHTML=true

#HTML email styles
email.style.header=font-family:Arial, Helvetica, sans-serif;font-size:10pt;
color:black;
email.style.hr=color: #555; background-color: #f00; height: 1px;
email.style.title=font-family:Arial, Helvetica, sans-serif;font-style:
bold;font-size:12pt;
email.style.message=font-family:Arial, Helvetica, sans-serif;font-size:11pt;
email.style.table=font-family:Arial, Helvetica, sans-serif;border:1px solid
#000; border-collapse:collapse;
email.style.th=font-style: bold;border:1px solid #000; border-
collapse:collapse; padding: 4px; background:#C7DAED
email.style.tr=font-size:10pt
email.style.td=border:1px solid #000; border-collapse:collapse; padding: 4px
email.style.footer=font-family:Arial, Helvetica, sans-serif;font-size:10pt;
color:black;
email.style.disclaimer=font-style: italic;

######### PDF ARCHIVE CONFIGURATION ##########################
# Set the maximum number of pdf export threads.
pdf.archival.maxthreads=3
# Number of alerts/cases per export web service call.
pdf.archival.service.batchsize=5
# URL of the Alert Viewer service
alertviewer.service.url=@ALERT_VIEWER_SERVICE_URL@
######### HIGHLIGHTS GENERATION CONFIGURATION ##########################
#
# Set the default currency code.
#
# See /mantas_cfg/etc/xml/CUR_Currencies.xml for supported currency
# codes.
#
currency.default=USD

######### HDC CONFIGURATION ##########################
#
# Set the maximum number of hdc threads.
#
hdc.maxthreads=1
hdc.batchsize=10000

######### Data Analysis Tool CONFIGURATION ##########################
#

# Username and password for connecting to the database
```

```
dat.database.username=${ingest.user}
dat.database.password=${ingest.password}

# Input file for analysis
dat.analysis.input=/scratch/ofsaadb/BD802_Final/BD802FL/database/db_tools/
mantas_cfg/analysis_aml.xml

# Output file and file format control
dat.analysis.output=/scratch/ofsaadb/BD802_Final/BD802FL/database/db_tools/
data/analysis.html

# Valid values for dat.output.format are HTML and TEXT
dat.output.format=HTML
# Delimiter only applies to TEXT output format
dat.output.delimiter=,
######### Execute Query Tool CONFIGURATION ########################
#
# Username and password for connecting to the database

eqt.database.username=${ingest.user}
eqt.database.password=${ingest.password}
############# Database Builder Utility Configuration ###############
#
# File containing tokens and their value
db_tools.tokenfile=/scratch/ofsaadb/BD802_Final/BD802FL/database/db_tools/
mantas_cfg/db_variables.cfg
Oracle.DuplicateRow=1
Oracle.ObjectExists=955,2260,2275,1430,1442,1451,957,1408,2261,1543
Oracle.ObjectDoesNotExist=942,1418,1434,2441,904,4043,1927,2443

dbscript.execution.users=(system|business|mantas|market|miner|ingest|report|
kdd|algorithms|case|config|fatca|ctr|kyc|fsdf|dbutil|web)

############# Correlation Migration Utility Configuration ###############
#
corrRuleMig.CorrRuleFileNm=
corrRuleMig.loadHistory=Y
aps.service.url=http://:8070/mantas/services/AlertProcessingService
aps.service.user=test
aps.service.user.password=

############# Config Migration Utility Configuration ###############
config.filenm.prefix=Config

#################### LOG CONFIGURATION ################################
#
# Trace SQL exception.  Set to "true" for SQL tracing,
# "verbose" to trace low-level JDBC calls
#
com.sra.kdd.tools.database.debug=true
# Specify which priorities are enabled in a hierarchical fashion, i.e., if
# DIAGNOSTIC priority is enabled, NOTICE, WARN, and FATAL are  also enabled,
# but TRACE is not.
# Uncomment the desired log level to turn on appropriate level(s).
# Note, DIAGNOSTIC logging is used to log database statements and will slow
```

```
# down performance.  Only turn on if you need to see the SQL statements being
# executed.
# TRACE logging is used for debugging during development.  Also only turn on
# TRACE if needed.
log.fatal=true
log.warning=true
log.notice=true
log.diagnostic=true
log.trace=true
log.time.zone=US/Eastern

# Specify whether logging for a particular level should be performed
# synchronously or asynchronously.
log.fatal.synchronous=true
log.warning.synchronous=true
log.notice.synchronous=true
log.diagnostic.synchronous=true
log.trace.synchronous=true

# Specify the format of the log output. Can be modified according to the
format
# specifications at:
# http://logging.apache.org/log4j/docs/api/org/apache/log4j/PatternLayout.html
# NOTE: Because of the nature of asynchronous logging, detailed information
# (class name, line number, etc.) cannot be obtained when logging
# asynchronously.  Therefore, if this information is desired (i.e. specified
# below), the above synchronous properties must be set accordingly (for the
# levels for which this detailed information is desired). Also note that this
# type of detailed information can only be obtained for Java code.
log.format=%d [%t] %p %m%n
# Specify the full path and filename of the message library.
log.message.library=/scratch/ofsaadb/BD802_Final/BD802FL/database/db_tools/
mantas_cfg/etc/mantas_database_message_lib_en.dat
# Specify the full path to the categories.cfg file
log.categories.file.path=/scratch/ofsaadb/BD802_Final/BD802FL/database/
db_tools/mantas_cfg/

# Specify where a message should get logged for a category for which there is
# no location property listed above.
# This is also the logging location of the default MANTAS category unless
# otherwise specified above.
# Note that if this property is not specified, logging will go to the console.
log.default.location=/scratch/ofsaadb/BD802_Final/BD802FL/database/db_tools/
logs/Utilities.log
# Specify the location (directory path) of the mantaslog, if the mantaslog
# was chosen as the log output location anywhere above.
# Logging will go to the console if mantaslog was selected and this property
is
# not given a value.
log.mantaslog.location=/scratch/ofsaadb/BD802_Final/BD802FL/database/db_tools/
logs/mantaslog.log

# Specify the hostname of syslog if syslog was chosen as the log output
location
# anywhere above.
# Logging will go to the console if syslog was selected and this property is
```

```
# not given a value.
log.syslog.hostname=

# Specify the hostname of the SMTP server if an e-mail address was chosen as
# the log output location anywhere above.
# Logging will go to the console if an e-mail address was selected and this
# property is not given a value.
log.smtp.hostname=

# Specify the maxfile size of a logfile before the log messages get rolled to
# a new file (measured in MBs).
# If this property is not specified, the default of 10 MB will be used.
log.max.size=

#NOTE: The values for the following variables need not be changed
# Specify the ID range for wrapper datasets
dataset.wrapper.range.min=113000001
dataset.wrapper.range.max=114000000
product.id.range.min=113000000
product.id.range.max=200000000
```

7.1.1 Log4j2.xml Configuration

In the <OFSAAI Installed Directory>/database/db_tools/log4j2.xml files file, you can modify the default location to where you want to direct logging output for each utility. The entries that you make require a specific format; the file contains instructions and examples of correct formatting.

This topic provides a sample of the logging information in the Log4j2.xml file.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">

<Appenders>


    <RollingFile name="CALENDAR_MANAGER" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/calendar_manager.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/calendar_manager.log</FileName>
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [CALENDER_MANAGER] [%5p] - %m%n</
Pattern>
      </PatternLayout>
      <Policies>
        <SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

    <RollingFile name="PURGE_UTIL" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/purge.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/purge.log</FileName>
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [PURGE_UTIL] [%5p] - %m%n</
```

```
Pattern>
      </PatternLayout>
      <Policies>
        <SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

        <RollingFile name="BATCH_CONTROL" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/batch_control.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/batch_control.log</FileName>
      <PatternLayout>
<Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [BATCH_CONTROL] [%5p] - %m%n</Pattern>
      </PatternLayout>
      <Policies>
<SizeBasedTriggeringPolicy size="10000kb"/>
</Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

    <RollingFile name="DATA_RETENTION_MANAGER" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/DRM_Utility.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/DRM_Utility.log</FileName>
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [DATA_RETENTION_MANAGER] [%5p] -
%m%n</Pattern>
      </PatternLayout>
      <Policies>
        <SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

    <RollingFile name="TRUNCATE_MANAGER" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/truncate_manager.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/truncate_manager.log</FileName>
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [TRUNCATE_MANAGER] [%5p] - %m%n</
Pattern>
      </PatternLayout>
      <Policies>
        <SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

    <RollingFile name="COMMON_UTILITIES" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/common_utilities.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/common_utilities.log</FileName>
      <PatternLayout>
<Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [COMMON_UTILITIES] [%5p] - %m%n</Pattern>
      </PatternLayout>
      <Policies>
<SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
```

```
    </RollingFile>

    <RollingFile name="EXTRACT" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/extract.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/extract.log</FileName>
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [EXTRACT] [%5p] - %m%n</Pattern>
      </PatternLayout>
      <Policies>
        <SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

    <RollingFile name="LOAD" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/load.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/load.log</FileName>
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [LOAD] [%5p] - %m%n</Pattern>
      </PatternLayout>
      <Policies>
        <SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

    <RollingFile name="REFRESH_TEMP_TABLE" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/refresh_temp_table.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/refresh_temp_table.log</FileName>
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [REFRESH_TEMP_TABLE] [%5p] -
%m%n</Pattern>
      </PatternLayout>
      <Policies>
<SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

    <RollingFile name="RUN_STORED_PROCEDURE" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/run_stored_procedure.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/run_stored_procedure.log</
FileName>
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [RUN_STORED_PROCEDURE] [%5p] -
%m%n</Pattern>
      </PatternLayout>
      <Policies>
        <SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

    <RollingFile name="GET_DATASET_QUERY" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/get_dataset_query.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/get_dataset_query.log</FileName>
```

```
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [GET_DATASET_QUERY] [%5p] -
%m%n</Pattern>
      </PatternLayout>
      <Policies>
        <SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

    <RollingFile name="DATA_ANALYSIS_TOOL" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/data_analysis_tool.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/data_analysis_tool.log</FileName>
      <PatternLayout>
<Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [DATA_ANALYSIS_TOOL] [%5p] - %m%n</
Pattern>
      </PatternLayout>
      <Policies>
<SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

    <RollingFile name="DB_BUILDER" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/db_builder.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/db_builder.log</FileName>
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [DB_BUILDER] [%5p] - %m%n</
Pattern>
      </PatternLayout>
      <Policies>
        <SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>
    <RollingFile name="ARCHIVE_PDF" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/pdf_archive.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/pdf_archive.log</FileName>
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [ARCHIVE_PDF] [%5p] - %m%n</
Pattern>
      </PatternLayout>
      <Policies>
        <SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>


    <RollingFile name="HIGHLIGHT_GENERATOR" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/highlight_generator.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/highlight_generator.log</
FileName>
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [HIGHLIGHT_GENERATOR] [%5p] -
%m%n</Pattern>
```

```
      </PatternLayout>
      <Policies>
<SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

    <RollingFile name="HDC" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/hdc.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/hdc.log</FileName>
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [HDC] [%5p] - %m%n</Pattern>
      </PatternLayout>
      <Policies>
        <SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

    <RollingFile name="REPORT" append="true"
filePattern="@ORION_DB_DBTOOLS_PATH@/logs/report.log">
      <FileName>@ORION_DB_DBTOOLS_PATH@/logs/report.log</FileName>
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [REPORT] [%5p] - %m%n</Pattern>
      </PatternLayout>
      <Policies>
        <SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

            <Console name="stdout" target="SYSTEM_OUT">
            <PatternLayout>
                <pattern>
                    [%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{1} - %msg%n
                </pattern>>
</PatternLayout>
        </Console>
    </Appenders>

     <Loggers>
          <Logger name="CALENDAR_MANAGER" level="info" additivity="false">
              <AppenderRef ref="CALENDAR_MANAGER" level="trace"/>
              <AppenderRef ref="stdout" level="error"/>
          </Logger>

           <Logger name="PURGE_UTIL" level="info" additivity="false">
              <AppenderRef ref="PURGE_UTIL" level="trace"/>
              <AppenderRef ref="stdout" level="error"/>
          </Logger>

          <Logger name="BATCH_CONTROL" level="info" additivity="false">
              <AppenderRef ref="BATCH_CONTROL" level="trace"/>
              <AppenderRef ref="stdout" level="error"/>
          </Logger>
```

```
            <Logger name="HDC" level="info" additivity="false">
                <AppenderRef ref="HDC" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>


            <Logger name="HIGHLIGHT_GENERATOR" level="info" additivity="false">
                <AppenderRef ref="HIGHLIGHT_GENERATOR" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>


             <Logger name="DATA_RETENTION_MANAGER" level="info"
additivity="false">
                <AppenderRef ref="DATA_RETENTION_MANAGER" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>

            <Logger name="DB_BUILDER" level="info" additivity="false">
                <AppenderRef ref="DB_BUILDER" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
</Logger>

             <Logger name="DB_BUILDER_SQL" level="info" additivity="false">
                <AppenderRef ref="DB_BUILDER" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>

              <Logger name="EXTRACT" level="info" additivity="false">
                <AppenderRef ref="EXTRACT" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>

              <Logger name="CORRRULEMIGRATIONUTIL_EXTRACT" level="info"
additivity="false">
                <AppenderRef ref="EXTRACT" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>

             <Logger name="CONFIGURATIONMIGRATIONUTIL_EXTRACT" level="info"
additivity="false">
                <AppenderRef ref="EXTRACT" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>

             <Logger name="LOAD" level="info" additivity="false">
                <AppenderRef ref="LOAD" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>

            <Logger name="CORRRULEMIGRATIONUTIL_LOAD" level="info"
additivity="false">
                <AppenderRef ref="LOAD" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>
```

```
            <Logger name="CONFIGURATIONMIGRATIONUTIL_LOAD" level="info"
additivity="false">
                <AppenderRef ref="LOAD" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>

             <Logger name="REFRESH_TEMP_TABLE" level="info" additivity="false">
                <AppenderRef ref="REFRESH_TEMP_TABLE" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>

             <Logger name="RUN_STORED_PROCEDURE" level="info"
additivity="false">
                <AppenderRef ref="RUN_STORED_PROCEDURE" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>

             <Logger name="GET_DATASET_QUERY" level="info" additivity="false">
                <AppenderRef ref="GET_DATASET_QUERY" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>

             <Logger name="REPORT" level="info" additivity="false">
                <AppenderRef ref="REPORT" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>

            <Logger name="DATA_ANALYSIS_TOOL" level="info" additivity="false">
                <AppenderRef ref="DATA_ANALYSIS_TOOL" level="trace"/>
                <AppenderRef ref="stdout" level="error"/>
            </Logger>

        <Root level="error">
            <AppenderRef ref="stdout"/>
        </Root>
    </Loggers>
<!--     <root>
        <priority value="##PRIORITY##"></priority>
    </root> -->
</log4j:configuration>
```

7.2 Managing Annual Activities

OFSBD requires that you perform certain calendar management tasks at least annually: loading holidays and weekly off-days from an Oracle client. This ensures that OFSBD has the necessary information for populating its own business calendars.

Loading Holidays

On an annual basis, you must populate holidays for the upcoming calendar year into the Behavior Detection KDD_CAL_HOLIDAY database table. This ensures that the table contains holidays for at least the next year.

This section provides an example of a SQL script for loading the table.

```
INSERT INTO KDD_CAL_HOLIDAY ( CLNDR_NM, CLNDR_DT, HLDY_NM,
HLDY_TYPE_CD ) VALUES ( 'SYSCAL', TO_DATE( '01/01/2017',
'MM/DD/YYYY'), 'New Year''s Day - 2017', 'C');

INSERT INTO KDD_CAL_HOLIDAY ( CLNDR_NM, CLNDR_DT, HLDY_NM,
HLDY_TYPE_CD ) VALUES ( 'SYSCAL', TO_DATE( '01/16/2017',
'MM/DD/YYYY'), 'Martin Luther King Jr.''s Birthday - 2017', 'C');

INSERT INTO KDD_CAL_HOLIDAY ( CLNDR_NM, CLNDR_DT, HLDY_NM,
HLDY_TYPE_CD ) VALUES ( 'SYSCAL', TO_DATE( '02/20/2017',
'MM/DD/YYYY'), 'President''s Day - 2017', 'C');

INSERT INTO KDD_CAL_HOLIDAY ( CLNDR_NM, CLNDR_DT, HLDY_NM,
HLDY_TYPE_CD ) VALUES ( 'SYSCAL', TO_DATE( '04/14/2017',
'MM/DD/YYYY'), 'Good Friday - 2017', 'C');

INSERT INTO KDD_CAL_HOLIDAY ( CLNDR_NM, CLNDR_DT, HLDY_NM,
HLDY_TYPE_CD ) VALUES ( 'SYSCAL', TO_DATE( '05/29/2017',
'MM/DD/YYYY'), 'Memorial Day - 2017', 'C');

INSERT INTO KDD_CAL_HOLIDAY ( CLNDR_NM, CLNDR_DT, HLDY_NM,
HLDY_TYPE_CD ) VALUES ( 'SYSCAL', TO_DATE( '07/04/2017',
'MM/DD/YYYY'), 'Independence Day - 2017', 'C');

INSERT INTO KDD_CAL_HOLIDAY ( CLNDR_NM, CLNDR_DT, HLDY_NM,
HLDY_TYPE_CD ) VALUES ( 'SYSCAL', TO_DATE( '09/04/2017',
'MM/DD/YYYY'), 'Labor Day - 2017', 'C');

INSERT INTO KDD_CAL_HOLIDAY ( CLNDR_NM, CLNDR_DT, HLDY_NM,
HLDY_TYPE_CD ) VALUES ( 'SYSCAL', TO_DATE( '11/22/2017',
'MM/DD/YYYY'), 'Thanksgiving Day - 2017', 'C');

INSERT INTO KDD_CAL_HOLIDAY ( CLNDR_NM, CLNDR_DT, HLDY_NM,
HLDY_TYPE_CD ) VALUES ( 'SYSCAL', TO_DATE( '12/25/2017',
'MM/DD/YYYY'), 'Christmas Day - 2017', 'C');

COMMIT;
```

The following table describes the contents of the KDD_CAL_HOLIDAY table.

Table 7-1 KDD_CAL_HOLIDAY Table

Column Name	Description
CLNDR_NM	Specific calendar name.
CLNDR_DT	Date that is a holiday.
HLDY_NM	Holiday name , such as Thanksgiving or Christmas.
HLDY_TYPE_CD	Indicates whether the business is Closed (C) or Shortened (S).
SESSN_OPN_TM	Indicates the opening time of the trading session for a shortened day. The format is HHMM.
SESSN_CLS_TM	Indicates the closing time of the trading session for a shortened day. The format is HHMM.

Table 7-1 (Cont.) KDD_CAL_HOLIDAY Table

Column Name	Description
SESSN_TM_OFFSET_TX	Indicates the timezone offset for SESSN_OPN_TMand SESSN_CLS_TM.

When the system runs the `set_mantas_date.sh script`, it queries the KDD_CAL_HOLIDAY table for the maximum date for each calendar in the table.

> **Note:**
>
> If the maximum date is less than 90 days ahead of the provided date, the process logs a warning message that the specific calendar's future holidays need updating. If any calendars have no holiday records, the system logs a Warning message that the specific calendar has no recorded holidays for the appropriate date range.

Loading Non-business Days

After obtaining non-business days (or weekly off-days; typically Saturday and Sunday) from an Oracle client, load this information for the upcoming calendar year into the KDD_CAL_WKLY_OFF table. This section provides an example of an SQL script for loading the table.

> **Note:**
>
> By default, the system identifies Saturdays and Sundays as non-business days in the system calendar (SYSCAL).

The following table describes the contents of the KDD_CAL_WKLY_OFF table.

Table 7-2 KDD_CAL_WKLY_OFF

Column Name	Description
CLNDR_NM	Specific calendar name.
DAY_OF_WK	Value that represents the day of the week: Sunday=1, Monday=2, Tuesday=3,Wednesday=4, Thursday=5, Friday=6, Saturday=7.

> **Note:**
>
> If the table does not contain records for any calendar in the list, the system logs a Warning message that the specific calendar contains no weekly off-days.

7.3 Managing Alert Purge Utility

The ingestion of certain data can result in the creation of false matches, alerts, and activities. While correction and data re-ingestion is possible, the system does not remove these erroneously generated matches, alerts, and activities automatically.

There may also be cases when the alerts have been residing in the database due to the retention policies imposed by the regulatory bodies, or the internal policies of the respective organization.

The Alert Purge Utility enables you to identify and remove such matches, alerts, and activities selectively, based on a number of parameters (like the Behavior Detection Job ID, Behavior Detection Scenario ID, Behavior Detection Scenario Class, or a date range with optional alert status codes). Additional parameters enable you to simulate a purge run to determine all found matches, alerts, and activities using the input parameters. You can also limit the alerts in the purge process only to those that contain false matches.

The utility consists of a UNIX shell script, Java executables, a XML File and a configuration file in which you define the process parameters to use in the purge processing. The system directs output to a configurable log file; processing appends this log with information about subsequent executions of the scripts.

This section covers the following topics:

- Directory Structure
- Logs
- Precautions
- Using the Alert Purge Utility
- Sample Alert Purge Processes

Directory Structure

The following table describes the directory structure for the Alert Purge Utility.

Table 7-3 Alert Purge Utility Directory Structure

Directory	Description
bin/	Contains executable files, including the run_alert_purge.shshell script.
lib/	Contains required class files in .jarformat.
mantas_cfg/	Contains configuration files , such as install.cfgand categories.cfg, in which you can configure properties and logging attributes.
logs/	Keeps the <OFSAAI Installed Directory>/database/db_tools/logs/purge.log file that the utility generates during execution.
data/	Keeps .sqlfiles for execution.
.xml	Contains the Purge Rules Configuration File (PurgeRules.xml), which is used for configuring the Alert Purge rules.

Logs

As the Alert Purge Utility performs alert detection activities, it generates a log that it enters in the `<OFSAAI Installed Directory>/database/db_tools/logs/purge.log` file (the logging process time-stamps all entries). The log file contains relevant information such as status of the purge processing, log-relevant information, and error records.

You can modify the current logging configuration for the Alert Purge Utility in the `<OFSAAI Installed Directory>/database/db_tools/log4j2.xml` files. For more information about logging in these configuration files, refer to Managing Common Resources for Batch Processing Utilities and APPENDIX A - Logging.

Precautions

You use the utility to rid the system of falsely-generated matches and alerts. Other than recorded information in the `<OFSAAI Installed Directory>/database/db_tools/logs/purge.log` file, the system does not capture audit information for this process. The utility does not update other alerts' prior counts as a result of purging alerts.

> **Note:**
>
> - The utility also purges any alert or case which is used to trigger Auto Suppression or establish Trusted Parties. However, this would not affect the Suppression Rule or the Trusted Pair except that the *kdd_auto_suppr_alert.trgr_alert_id,kdd_trus ted_pair.trgr_alert_id,or kdd_trusted_pair.trgr_case_id* columns are set to a null value.
> - Run the Alert Purge Utility one process at a time. Multiple, simultaneous executions of the utility may lead to unexpected results and compromise the relational integrity of match, alert, and action data. When no users are editing or viewing any of the alerts, actions, or associated information (including matches derived from the alerts and actions specified, alerts derived from the specified actions, and actions derived from the specified alerts). However, you can run the utility during editing or viewing of other alerts and related information. You can also run the utility during alert post-processing, subject to time constraints.
> - The recommended numbers of alerts that can be purged in a batch is 10,000 alerts. This may take a few hours to complete. As this is not a daily activity, Oracle clients should plan this accordingly.

7.3.1 Using the Alert Purge Utility>

The Alert Purge Utility is not part of an automated batch process. You run this manual process only when necessary.

The following sections describe configuring and executing the utility, as well as the utility's process flow.

Configuring the Alert Purge Utility

To configure the Alert Purge Utility, follow these steps:

1. Navigate to the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg`.

2. Edit the parameters in the `install.cfg` file to the desired settings. This file contains common configuration information that the Alert Purge Utility and other utilities require for processing. The following is a sample section from the install.cfg file for configuration information specific to this utility:

```
################ ALERT PURGE CONFIGURATION ##########################
# Set the Alert Purge input variables here.
# (use the word "null" as the value of any parameters that are not
#  to be used)
#

# Specify whether or not to consider Matches
limit_matches=N

# Specify whether or not to purge the data
purge=Y

# Specify batch size for which commit should perform
batch_size=5000

job=null
scenario=null
# enter dates, with quotes in the following format:
#   'DD-MON-YYYY HH24:MI:SS'
start_date=null
end_date=null
alert_status=NW

# Specify purge db user
purge.database.user=f802_fccm

# Specify purge db user password.
purge.database.password=

# Specify whether alerts has to be purged or not.
purge_alert_flag=Y

# Specify whether fatca cases/assessments has to be purged or not.
purge_fatca_flag=Y

# Specify whether case has to be purged or not.
purge_case_flag=Y

# Specify defualt rule set.
purge_default_rule_set=

# Specify total number of threads should be used for the process.
purge_threads_no=10

# Specify report directory for report on process performed.
purge_report_directory=

# Specify product version
purge_product_version=
```

```
#Base Working Directory required to put the temporary log from Database
Server
ap.storedproc.logdir=/tmp

#The common Path required to put the SQL files to execute
commonSQLFilePath=/scratch/ofsaadb/BD804_Final/BD804FL/database/db_tools/
data
```

> **Note:**
>
> Not specifying a value of **null** , such as leaving a value blank, in this section of the install.cfg file causes undesirable results.

The following table describes required and optional parameters for this utility.

Table 7-4 Alert Purge Utility Parameters

Parameter	Description
purge	Determines how the utility performs processing, depending on the specified value: • N(default): Performs all processing up to the point of the purge. The utility identifies resulting matches, alerts, and actions, but performs no purging. • Y: Performs the above in addition to purging matches, alerts, and actions.
limit_matches	Identifies restrictions on the matches to delete: • Y (default): If a match that you want to delete is part of an alert that contains matches that you do not want to delete, do not delete this match either (applies to multi-match alerts). • N: Deletes all selected matches for purging based on the input criteria. The utility deletes only alerts and associated actions that exclusively contain matches to be purged. **Note:** The system purges matches that do not relate to alerts, regardless of the value of limit_matches.
batch_size	*Optional:* Sets the batch size of purge actions to minimize log space use. Specifying a non- positive value or specifying no value uses the default of 5,000 rows.
purge_alert_flag	Determines whether or not the utility would purge alerts, depending on the specified value: • N: Does not purge the alerts irrespective of whether or not they identified according to the purge rule being used. This may be used when purging only the cases. • Y(default): Purges the alerts as identified by the purge rule used to perform the purge operation.

Table 7-4 (Cont.) Alert Purge Utility Parameters

Parameter	Description
purge_case_flag	Determines whether or not the utility would purge cases, depending on the specified value: • N: Does not purge the cases irrespective of whether or not they identified accordingto the purge rule being used. This may be used when purging only the cases. • Y(default): Purges the cases as identified by the purge rule used to perform the purge operation.
purge_default_rule_set	*(Optional)* Indicates the default set of rules to be used for purging alerts. You may either specify the purge rules to be used against this parameter, or pass the name of the specific purge rules) as command line parameters. You may specify a single purge rule, or a comma separated list of purge rules to be used as default when no other purge rule is provided from the command line.
purge_threads_no	*(Optional)* Identifies the number of concurrent threads to create for purging the alerts to optimize the performance. Specifying a non-positive value or specifying no value uses the default of 10 threads.
purge_report_directory	Identifies the absolute path to the directory where the purge activity report should be generated. The report file name has a name similar to Purge_<YYYYMMDD.HH.MM.SS>.txt. Here <YYYYMMDD.HH.MM.SS>represents current timestamp when the utility was executed.
purge_product_version	Identifies the OFSBD Product Version installed by the client.

The `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/etc/xml/PurgeRules.xml` file contains purge rules configuration information that the Alert Purge Utility requires for processing. The following sample section from the `PurgeRules.xml` file provides configuration information for this utility.

```
<?xml version="1.0" encoding="utf-8"?>
<xs:RuleSet xmlns:xs="http://namespaces.mantas.com/RuleSet">
  <Alert>
        <Rule id="1">
            <IdentifierList>286,4565,4537</IdentifierList>
     <ScenarioIdList>114697002</ScenarioIdList>
              <ScenarioClassList>CR</ScenarioClassList>
              <CreateDate>
                    <StartDate>2011-05-25</StartDate>
                    <EndDate>2011-05-25</EndDate>
              </CreateDate>
              <DomainCode>MTS</DomainCode>
              <BatchId>2</BatchId>
              <ThresholdSetIds>118745206,118710066</ThresholdSetIds>
              <LastActionDate>
                    <StartDate>2016-05-25</StartDate>
```

```
                  <EndDate>2016-05-25</EndDate>
            </LastActionDate>
            <Status>CL</Status>
            <JobIds>102202</JobIds>
      </Rule>
   </Alert>
  <Case>
      <Rule id="2">
            <IdentifierList>CA51300004,CA3773,CA3757,CA3766</IdentifierList>
            <CaseTypeList>FR_EE,FR_ON</CaseTypeList>
            <CreateDate>
                  <Age>1Y</Age>
            </CreateDate>
            <LastActionDate>
                  <StartDate>2016-06-22</StartDate>
                  <EndDate>2016-06-22</EndDate>
</LastActionDate>
      </Rule>
    </Case>
</xs:RuleSet>
```

The following table describes the Purge Rules Configuration Parameters.

Table 7-5 Purge Rules Configuration Parameters

Parameter	Description
Alert/Case	Identifies and encapsulates the purge rules for Alerts. You may define any number of purge rules for alerts.
Rule	Identifies a set of rules to be used for purging Alert Information. All Alert Purge rules defined in this file must be provided a unique positive integer ID (as specified against the ID attribute). The value provided against the ID attribute is used by the utility to identify the rules to be used for carrying out the purge operations. Not specifying a unique value for the ID attribute may lead to undesirable results.
IdentifierList	Identifies a list of Alert IDs to be purged. You may specify more than one alert or case ID by separating them by comma.
ScenarioIdList	Identifies a list of Scenario IDs for which the alerts are to be purged. You may specify more than one Scenario ID by separating them by comma. This property is specific to alerts only. This should not be specified for cases
ScenarioClassList	Identifies a list of Scenario Class for which the alerts are to be purged. You may specify more than one Scenario Class by separating them by comma. This property is specific to alerts only. This should not be specified for cases

Table 7-5 (Cont.) Purge Rules Configuration Parameters

Parameter	Description
CreateDate	Identifies the dates to be considered for purging the alerts by their creation date. The date range may be provided in terms of Start Date or End Date, or the Age of the Alert or Case calculated from the current day/month/year. • **StartDate**: Identifies the date from when the alerts are to be consideredfor purging. The date should be provided in the format YYYY-MM-DD. • **EndDate**:Identifies the date up to which the alerts are to be purged. The date should be provided in the format YYYY-MM-DD • **Age**: Identifies the age of the Alert/Case to be purged relative to the current date/month/year. Acceptable values for this parameter constitutes a non- negative number followed by D (Days), M (Months) or Y (Years). If we specify age of a record is 1 Day means it should complete 1 day in the database. That is from current day to yesterday. The example below gives more details: (Assume Current date: 21 NOV 2012) • Case1: – if age = 1Y: Date range would be considered: 21 NOV 2012 to 21 NOV 2011 (includes both days) – if age = 5Y: Date range would be considered: 21 NOV 2012 to 21 NOV 2007 (includes both days) • Case2: – if age = 1M: Date range would be considered: 21 NOV 2012 to 21 OCT2012 (includes both days) – if age = 5M: Date range would be considered: 21 NOV 2012 to 21 JUN2012 (includes both days) • Case3: – if age = 1D: Date range would be considered: 21 NOV 2012 to 20 NOV 2012 (includes both days) – if age = 5D: Date range would be considered: 21 NOV 2012 to 16 NOV 2012 (includes both days) – if age = 0D: Date range would be considered: 21 NOV 2012 to 21 NOV 2012 (that is, current date only) If only EndDate is specified, utility would consider it as on or before that date, in case of only StartDate being provided, utility would consider it as on or after that date. In-case both dates are specified utility would consider both the dates and the dates in between them.

Table 7-5 (Cont.) Purge Rules Configuration Parameters

Parameter	Description
BatchId	Identifies the list of Batch IDs for which the alerts should be purged. This property is specific to alerts only.
DomainCode	Identifies the list of domains for which the alerts should be purged. Acceptable values include: • MTS • TST • PFM • NVZ This property is specific to alerts only.

Table 7-5 (Cont.) Purge Rules Configuration Parameters

Parameter	Description
LastActionDate	Identifies the dates to be considered for purging the alerts by he date on which last action was taken on them. The date range may be provided in terms of Start Date or End Date, or the Age of the Alert or Case calculated from the current day/month/year. • **StartDate**: Identifies the date from when the alerts/cases are to beconsidered for purging.The date should be provided in the format YYYY-MM-DD • **EndDate**:Identifies the date up to which the alerts are to be purged. The date should be provided in the format YYYY-MM-DD • **Age**: Identifies the age of the Alert or Case to be purged relative to the current date/month/ year. Acceptable values for this parameter constitutes a non- negative number followed by D (Days), M (Months) or Y (Years). If we specify age of a record is 1 Day means it should complete 1 day in the database. That is from current day to yesterday. The example below gives more details: (Assume Current date: 21 NOV 2012) • Case1: – if age = 1Y: Date range would be considered: 21 NOV 2012 to 21 NOV 2011 (includes both days) – if age = 5Y: Date range would be considered: 21 NOV 2012 to 21 NOV 2007 (includes both days) • Case2: – if age = 1M: Date range would be considered: 21 NOV 2012 to 21 OCT2012 (includes both days) – if age = 5M: Date range would be considered: 21 NOV 2012 to 21 JUN2012 (includes both days) • Case3 – if age = 1D: Date range would be considered: 21 NOV 2012 to 20 NOV 2012 (includes both days) – (ii)if age = 5D: Date range would be considered: 21 NOV 2012 to 16 NOV 2012 (includes both days) – if age = 0D: Date range would be considered: 21 NOV 2012 to 21 NOV 2012 (that is, current date only) If only EndDate is specified, utility would consider it as on or before that date, in case of only StartDate being provided, utility would consider it as on or after that date. If both dates are specified utility would consider both the dates and the dates in between them.

Table 7-5 (Cont.) Purge Rules Configuration Parameters

Parameter	Description
Status	Identifies a list of Status Codes against which the Alert or Case should be purged. You may specify more than one Status Code by separating them by comma.
JobIds	Identifies the list of Job IDs for which the alerts should be purged. You may specify more than one Job ID by separating them by comma. This property is specific to alerts only.
ThresholdSetIds	Identifies the list of Threshold Set IDs for which the alerts should be purged. You may specify more than one Threshold Set ID by separating them by comma. This property is specific to alerts only.

7.3.2 Executing the Alert Purge Utility

To execute the Alert Purge Utility, follow these steps:

1. Verify that the Behavior Detection database is operational: `tnsping <database instance name>`
2. Verify that the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` configuration file contains the correct source database connection and logging information.
3. Access the directory where the shell script resides: `cd <OFSAAI Installed Directory>/database/db_tools/bin`
4. Start the Alert Purge shell script: `run_alert_purge.sh -purge`

   Executing this command sets the environment classpath and starts the utility. You may also pass command line arguments to the utility, and execute the utility in any of the following ways:

   - You may pass a list of purge rules (as configured in PurgeRules.xml file) separated by a comma (,) following the convention of `alert_rule_<i0>` for alert-related rules and `case_rule_<i0>` for case-related rules; here *i0* is an integer representing the corresponding rule number in the purgeRules.xml file. `./run_alert_purge.sh -purge alert_rule_<i0>,alert_rule_<i1>,case_rule_<i2>….`
   - You may instruct the utility not to purge any alerts, but only cases, and vice-versa. If the value passed is '`alert=N`' the utility considers this as *no* to purge alerts `./run_alert_purge.sh -purge alert=N If the value passed is 'case=N'` the utility considers this as no to purge cases `./run_alert_purge.sh -purge case=N`
   - You may instruct the utility only to simulate the purge process and not purge the alerts by passing a command line parameter '`test=Y`'. In this case, the utility considers this as running in test mode and generates the report of alerts that would have purged. `./run_alert_purge.sh -purge test=Y`
   - You can provide all these parameters or a combination of these parameters irrespective of order, once at a time, to the utility as shown in the example below: `./run_alert_purge.sh -purge case=N alert_rule_<i0>,alert_rule<i1> test=Y`

Note:

a. If the utility is executed without any command line arguments, then utility considers the install.cfg parameter "purge_default_rule_set" value for purging the alert rules defined in PurgeRules.xml.

b. The following install.cfg parameters are no longer considered for purging the alerts in this version:

- job=null
- scenario=null
- start_date=null
- end_date=null
- alert_status=NW

7.3.3 Processing for Purging

The process for purging is as follows:

1. Once you execute the `run_alert_purge.sh` script, the Alert Purge Utility generates a listing of actions, matches, and alerts or cases that it must purge according to the rules specified at the command line, or the default rule set configured in the install.cfgfile.
2. After the script is executed, the actions, alerts, and cases are recorded in the `<OFSAAI Installed Directory>/database/db_tools/logs/purge.log` file.

Note:

- The utility presumes that you have determined the input parameters to specify what matches, alerts, and actions to purge. The utility does not check against the data to verify what it should purge.
- To capture the SQL statements naming, set `log.diagnostic=true` in the install.cfg.

3. The utility then purges actions, then matches, then alerts, according to the contents of the *KDD_AP_ACTION, KDD_AP_MATCH*, and *KDD_AP_ALERT* tables.
4. The utility captures purging results and any errors in the purge.log and a report (having the naming convention `Purge_<YYYYMMDD.HH.MM.SS>.txt`) files.

Note:

The Alert Purge Utility purges data from archive tables for erroneous alerts. Also, the system does not update score and previous match count values associated with generated matches and alerts since creation of the erroneous matches.

7.3.3.1 Automatic Restart Capability

The Alert Purge Utility has an automatic restart capability in that any interruption in the purge processing resumes at that point, regardless of the input parameters.

The system documents log information about the interruption in the `<OFSAAI Installed Directory>/database/db_tools/logs/purge.log` file. Otherwise, any restart that has not progressed to the purge component behaves as a new processing run. The restart capability allows interrupted purges to resume at a convenient point, but is unable to execute all desired input parameters.

7.3.4 Sample Alert Purge Processes

This section includes examples of the Purge Alerts process based on input parameters. These example patterns are also applicable for filtering cases.

Example 1

If user specifies only one rule 'xyz' for purging alerts and assume it as follows:

```
<Alert>
……..
            <Rule id="xyz">
      <IdentifierList>3775,3731,3669,3663</IdentifierList>
    <Status>CL</Status>
</Rule>
……..
    </Alert>
```

The utility filters in the existing alerts for IDs 3775,3731,3669,3663 and* status having Closed (CL).

Here and* specifies the logical and operation specified by sql.

In this case, the alert has closed status among the existing alert IDs of (3775, 3731, 3669, and 3663).

```
<Alert>
……..
<Rule id="xyz">
<IdentifierList>3775,3731,3669,3663</IdentifierList>
    <Status>CL</Status>
    <ScenarioIdList>114697002, 114690106</ScenarioIdList>
    <JobIds>456789</JobIds>
</Rule>
……..
</Alert>
```

The utility filters in the existing alerts for IDs 3775,3731,3669,3663 and* having status Closed (CL) and* having Scenario IDs 114697002,114690106 and having Job Id 456789.

Example 2

If user specifies multiple rules for purging:

```
<Alert>
……..
<Rule id="pqr">
<IdentifierList>3775, 3731,3669,3663</IdentifierList>
    <Status>CL</Status>
    <JobIds>456789</JobIds>
</Rule>
<Rule id="xyz">
    <ScenarioIdList>114697002,114690106</ScenarioIdList>
    <CreateDate>
<StartDate>2011-05-25</StartDate>
<EndDate>2011-05-29</EndDate>
</CreateDate>
</Rule>
……..
</Alert>
```

The utility prepares a query to filter alerts so that rule 'pqr' (fetches alerts as per the single rule de-scribed above) or* rule 'xyz' (fetches alerts as per the single rule described above) or*... That is, union of the alerts from all the rules would be filtered.

Here or* specifies the logical or operation specified by sql.

7.4 Managing Batch Control Utility

The Batch Control Utility enables you to manage and record the beginning and ending of a Behavior Detection batch process. It also enables you to access the currently running batch.

You control the process through a job scheduling tool such as Maestro or Unicenter Autosys. This utility consists of a Java file that resides in the directory `<OFSAAI Installed Directory>/database/db_tools/lib` and UNIX script files that reside in `<OFSAAI Installed Directory>/database/db_tools/bin`:

- start_mantas_batch.sh starts the batch process.
- end_mantas_batch.sh ends the batch process.
- get_mantas_batch.sh obtains the name of the currently running batch.

The utility also uses common parameters in the configuration file `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg`.

This section covers the following topics:

> **Note:**
>
> To calculate the age in business days versus calendar days, verify that the `age.alerts.useBusinessDays` setting in the `<OFSAAI Installed Directory>/database/db_tools/ mantas_cfg/install.cfg` file has a value of Y (yes).

Batches in Behavior Detection

Except for the Alert Viewer subsystem, batches govern all other activity in the Behavior Detection system. A batch provides a method of identifying a set of processing. This includes all activities associated with data management and Behavior Detection.

Deployment of a system can be with a single batch or with multiple batches. You can use multiple batches to permit intra-day processing to generate results several times per day, or to separate processing based on servicing multiple time zones.

Behavior Detection provides two types of batches:

- End-of-day: Represent processing at the completion of a business day for a set of data. Some processes are only appropriate for end-of-day batches. For example, daily activity summary derivations and calculating alert ages are activities that occur only in end-of-day batches. Multiple end-of-day batches per day can run if the Behavior Detection installation supports multiple time zones , such as New York and Singapore.
- Intra-day: Used when loading data between end-of-day batches to obtain more frequent detection results. For example, running a batch of trading-compliance scenarios at 10:00 A.M. can identify behaviors relevant to the opening of the market without waiting for the end of the day to be able to act.

Directory Structure

The following table provides the directory structure for the Batch Control Utility, in `<OFSAAI Installed Directory>/database/db_tools/`:

Table 7-6 Batch Control Utility Directory Structure

Directory	Contents
lib/	Required class files in .jar format.
mantas_cfg/	Configuration files , such as install.cfg and categories.cfg, in which you can configure properties and logging attributes.
logs/	File batch_control.logthat the utility generates during execution.
bin/	Executable files, including the *start_mantas_batch.sh, end_mantas_batch.sh, and get_mantas_batch.shshell* scripts.

Logs

As the Batch Control Utility manages batch processing, it generates a date-stamped log in the `<OFSAAI Installed Directory>/database/db_tools/logs/batch_control.log` file. The log file contains relevant information such as status of various batch control processes, results, and error records.

You can modify the current logging configuration for the Alert Purge Utility in the `<OFSAAI Installed Directory>/database/db_tools/log4j2.xml` files. For more information about

logging in these configuration files, refer to Managing Common Resources for Batch Processing Utilities, and Appendix A - Logging.

7.4.1 Using the Batch Control Utility

The Batch Control Utility typically runs as part of automated processing that a job scheduling tool such as Maestro or Unicenter AutoSys controls. The utility starts and terminates through a shell script, using values in parameters that particular configuration files contain.

You can use the Batch Control Utility to run the following types of batches:

- End-of-day: Represent processing at the completion of a business day for a set of data. Some processes are only appropriate for end-of-day batches. For example, daily activity summary derivations and calculating alert ages are activities that occur only in end-of-day batches. Multiple end-of-day batches per day can run if the Behavior Detection installation supports multiple time zones , such as New York and Singapore.
- Intra-day: Used when loading data between end-of-day batches to obtain more frequent detection results. For example, running a batch of trading-compliance scenarios at 10:00 A.M. can identify behaviors relevant to the opening of the market without waiting for the end of the day to be able to act.

The following sections describe this process, including tasks that you can perform when configuring the utility or running it manually (that is, starting, stopping, or obtaining a batch name).

- Configuring the Batch Control Utility
- Setting Up Batches
- Starting a Batch Process Manually
- Processing for Batch Start
- Ending a Batch Process
- Processing for End Batch
- Identifying a Running Batch Process
- Obtaining a Batch Name

7.4.1.1 Configuring the Batch Control Utility

To configure the batch control utility, follow these steps:

1. Navigate to the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg file`.This file contains common configuration information that Batch Control and other utilities require for processing.
2. Use the following sample section from the install.cfg file to input configuration information specific to this utility, including the single parameter that batch control requires.

```
############### BATCH CONTROL CONFIGURATION ####################

# When ending the batch, age alerts in calendar or business days.
age.alerts.useBusinessDays=Y
```

The value of the `age.alerts.useBusinessDays` parameter indicates that at completion of an end-of-day batch process, the Behavior Detection application calculates the age of

active alerts by number of calendar days (N) or business days (Y). The value of this parameter resides in the KDD_CAL table.

The utility connects to the database employing the user that the `utils.database.username` property specifies in the install.cfg file.

7.4.1.2 Setting Up Batches

OFSBD delivers with a default batch called DLY. The KDD_PRCSNG_BATCH table includes this batch and must contain all batches in the system.

When a batch starts as part of an automated process, it uses the batch names and other start-up information in this table. The DLY processing batch with ALL as the source origin is reserved for instances where one batch load is required, ignoring source systems. If you wish to associate specific source systems to DLY, then the DLY/ALL record must be deleted from the KDD_PRCSNG_BATCH_SRC table.

The following table provides the contents of the KDD_PRCSNG_BATCH table.

Table 7-7 KDD_PRCSNG_BATCH Table Contents

Column Name	Description
PRCSNG_BATCH_NM	Name of the batch , such as DLY.
PRCSNG_BATCH_DSPLY_ NM	Readable name for the batch, such as Daily.
PRCSNG_ORDER	Relative order of a batch run within processing.
EOD_BATCH_NM	Name of the batch that is this batch's end-of-day. This name is the same as the name for PRCSNG_BATCH_NM if the row represents an end-of-day batch.
PRCSNG_BATCH_NM	Description of this processing batch.

Each row in the KDD_PRCSNG_BATCH table represents a batch. Each batch identifies the batch that is the corresponding end-of day batch. The following examples illustrate this concept.

Single Batch

In this example, the KDD_PRCSNG_BATCH table contains a single batch per day. This is typical of deployment of a single geography for which a solution set does not require detection more than once daily. The KDD_PRCSNG_BATCH table may look similar to the example below.

Table 7-8 Sample KDD_PRCSNG_BATCH Table with Single Batch

PRCSNG_BATCH_NM	PRCSNG_BATCH_DSPLY_NM	PRCSNG_ORDER	EOD_BATCH_NM
DLY	Daily Batch	1	DLY

Single Site Intra-day Processing

In this intra-day batch example, the system is servicing a single time zone but runs an additional batch during the day to identify behaviors related to overnight trading, as shown in the following example.

Table 7-9 Sample KDD_PRCSNG_BATCH Table with Intra-day Processing

PRCSNG_BATCH_NM	PRCSNG_BATCH_DSPLY_NM	PRCSNG_ORDER	EOD_BATCH_NM
MAIN	Main Evening Batch	2	MAIN
MORN	Morning Batch	1	MORN

In this configuration, run the Calendar Manager Utility only during the MORN batch. Refer to Managing Calendar Manager Utility for more information. You can run the Data Retention Manager either in the MORN or MAIN batch. If you run it in the MAIN batch, define at least one buffer partition so that the MORN batch does not fail due to inadequate partitions. Refer to Managing Data Retention Manager, for more information.

Multiple Countries

As an Oracle client loading data through CSA, the system groups various source systems into one processing batch, so that it can call upon a specific batch and load data from specific source systems within that batch. This allows the handling of different batch loads from different countries running on the same staging instance. The association of the source systems to processing batch are captured in the KDD_PRCSNG_BATCH_SRC FSDM table. The following columns are available in this table:

Table 7-10 KDD_PRCSNG_BATCH_SRC FSDM Columns

Column	Data Type	Null	Primary Key	Default Value
PRCSNG_BATCH_NM	VARCHAR2(20)	Not Null	Yes	DLY To load only the US source for a batch, for example, Batch1, another record, Batch1, needs to be added.
SRC_ORIGIN	VARCHAR2(3)	Not Null	Yes	ALL To load only the US source for a batch, for example, Batch1, another record, US, needs to be added.
SRC_DESC	VARCHAR2(25 5)	Null	No	Productized Daily Processing Batch for all Source Systems

If you want to load only the US source for a batch, for example, Batch1, then another record, US Source System Load, needs to be added.

A single deployment supports detection against data from New York, London, and Hong Kong. In this case, three batches are all end-of-day batches, as the following table describes.

Table 7-11 Sample KDD_PRCSNG_BATCH Table with Multiple Country Processing

PRCSNG_BATCH_NM	PRCSNG_BATCH_DSPLY_NM	PRCSNG_ORDER	EOD_BATCH_NM
HK	Hong Kong	1	HK
LND	London	2	LND
NY	New York	3	NY

Since Hong Kong's markets open first, this is the first batch. You should run the Calendar Manager and Data Retention Manager at the start of the HK batch.

Upon setup of the batches, Behavior Detection processing begins with the `start_mantas_batch.sh` shell script. The final step in a batch is calling the `end_mantas_batch.sh` shell script.

7.4.1.3 Starting a Batch Process Manually

To start a batch manually, follow these steps:

1. Verify that the Behavior Detection database is operational: `tnsping <database instance name>`
2. Verify that the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` configuration file contains the correct source database connection information.
3. Access the directory where the shell script resides: `cd <OFSAAI Installed Directory>/database/db_tools/bin`
4. Run the batch control shell script: `start_mantas_batch.sh <batch name>` where *<batch name>* is the name of the batch. This parameter is case-sensitive.

> **Note:**
>
> If you enter an invalid batch name, the utility terminates and logs a message that describes the error. The error message appears on the console only if you have output to the console enabled in the `<OFSAAI Installed Directory>/database/db_tools/ mantas_cfg/categories.cfg` file. Refer to "Configuring Console Output, for more information.

7.4.1.4 Processing for Batch Start

After establishing the required Java environment and initiating various Java processing activities, the Batch Control Utility does the following:

1. The utility verifies that the provided batch name contains only the characters A-Z, a-z, and 0-9 by querying the KDD_PRCSNG_BATCH table.
2. The utility determines whether a batch is running by querying the KDD_PRCSNG_BATCH_CONTROL table. The following table describes the KDD_PRCSNG_BATCH_CONTROL table.

Table 7-12 KDD_PRCSNG_BATCH_CONTROL Table Contents

Column Name	Description
PRCSNG_BATCH_ID	Current batch process ID.
PRCSNG_BATCH_NM	Name of the current batch process.
DATA_DUMP_DT	Current business day. The Calendar Manager Utility places this information in the table.
EOD_PRCSNG_BATCH_FL	Flag that indicates whether the batch is an end-of-day process (Y or N).

3. The utility records information about the batch in the KDD_PRCSNG_BATCH_HIST table. This table contains a history of all batches that appear by start date and end date.

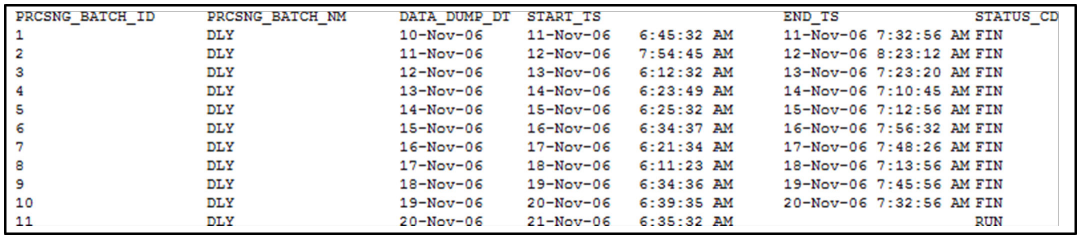
Table 7-13 KDD_PRCSNG_BATCH_HIST Table Contents

Column Name	Description
PRCSNG_BATCH_ID	Current batch process ID.
PRCSNG_BATCH_NM	Name of the current batch process.
DATA_DUMP_DT	Business day on which the batch ran.
START_TS	Time that the batch started.
END_TS	Time that the batch ended (if applicable).
STATUS_CD	Status code that indicates whether the batch is currently running (RUN) or has finished (FIN).

4. The Batch Control Utility logs a message in the `<OFSAAI Installed Directory>/database/ db_tools/logs/batch_control.log` file, stating that the batch process has begun.

   Querying the KDD_PRCSNG_BATCH_HIST table for confirmation that the batch has started displays information similar to that shown below. In the last entry, note the appearance of RUN for STATUS_CD and lack of end time in END_TS.

Figure 7-2 Sample KDD_PRCSNG_BATCH_HIST Table—Batch Start Status



PRCSNG_BATCH_ID	PRCSNG_BATCH_NM	DATA_DUMP_DT	START_TS	END_TS	STATUS_CD
1	DLY	10-Nov-06	11-Nov-06 6:45:32 AM	11-Nov-06 7:32:56 AM	FIN
2	DLY	11-Nov-06	12-Nov-06 7:54:45 AM	12-Nov-06 8:23:12 AM	FIN
3	DLY	12-Nov-06	13-Nov-06 6:12:32 AM	13-Nov-06 7:23:20 AM	FIN
4	DLY	13-Nov-06	14-Nov-06 6:23:49 AM	14-Nov-06 7:10:45 AM	FIN
5	DLY	14-Nov-06	15-Nov-06 6:25:32 AM	15-Nov-06 7:12:56 AM	FIN
6	DLY	15-Nov-06	16-Nov-06 6:34:37 AM	16-Nov-06 7:56:32 AM	FIN
7	DLY	16-Nov-06	17-Nov-06 6:21:34 AM	17-Nov-06 7:48:26 AM	FIN
8	DLY	17-Nov-06	18-Nov-06 6:11:23 AM	18-Nov-06 7:13:56 AM	FIN
9	DLY	18-Nov-06	19-Nov-06 6:34:36 AM	19-Nov-06 7:45:56 AM	FIN
10	DLY	19-Nov-06	20-Nov-06 6:39:35 AM	20-Nov-06 7:32:56 AM	FIN
11	DLY	20-Nov-06	21-Nov-06 6:35:32 AM		RUN

7.4.1.5 Ending a Batch Process

When a batch ends as part of an automated process, the utility retrieves the batch name and other information from the KDD_PRCSNG_BATCH table.

To stop a batch process manually, follow these steps:

1. Verify that the Behavior Detection database is operational. `tnsping <database instance name>`

2. Verify that the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` configuration file contains the correct source database connection information.
3. Access the directory where the shell script resides: `cd <OFSAAI Installed Directory>/database/db_tools/bin`
4. Start the batch shell script: `end_mantas_batch.sh`

   If you enter an invalid batch name, the utility terminates and logs a message that describes the error. The error message appears on the console only if you have output to the console enabled in the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/categories.cfg` configuration file.

7.4.1.6 Processing for End Batch

After establishing the required Java environment and initiating various Java processing activities, the Batch Control Utility does the following:

1. Determines whether a batch is running by querying the KDD_PRCSNG_BATCH_CONTROL table.
2. Records information about the batch in the KDD_PRCSNG_BATCH_ HIST table. This table contains a history of all batches that appear by start date and end date. The following sample illustrates a table query; an end time-stamp in **END_TS** and status of **FIN** in STATUS_CD for the bolded entry indicates that the batch has ended.

Figure 7-3 Sample KDD_PRCSNG_BATCH_HIST Table—Batch End Status

PRCSNG_BATCH_ID	PRCSNG_BATCH_NM	DATA_DUMP_DT	START_TS		END_TS		STATUS_CD
1	DLY	10-Nov-06	11-Nov-06	6:45:32 AM	11-Nov-06	7:32:56 AM	FIN
2	DLY	11-Nov-06	12-Nov-06	7:54:45 AM	12-Nov-06	8:23:12 AM	FIN
3	DLY	12-Nov-06	13-Nov-06	6:12:32 AM	13-Nov-06	7:23:20 AM	FIN
4	DLY	13-Nov-06	14-Nov-06	6:23:49 AM	14-Nov-06	7:10:45 AM	FIN
5	DLY	14-Nov-06	15-Nov-06	6:25:32 AM	15-Nov-06	7:12:56 AM	FIN
6	DLY	15-Nov-06	16-Nov-06	6:34:37 AM	16-Nov-06	7:56:32 AM	FIN
7	DLY	16-Nov-06	17-Nov-06	6:21:34 AM	17-Nov-06	7:48:26 AM	FIN
8	DLY	17-Nov-06	18-Nov-06	6:11:23 AM	18-Nov-06	7:13:56 AM	FIN
9	DLY	18-Nov-06	19-Nov-06	6:34:36 AM	19-Nov-06	7:45:56 AM	FIN
10	DLY	19-Nov-06	20-Nov-06	6:39:35 AM	20-Nov-06	7:32:56 AM	FIN
11	DLY	20-Nov-06	21-Nov-06	6:35:32 AM	21-Nov-06	7:39:32 AM	FIN

3. Calculates the age of all open alerts and writes it to `KDD_REVIEW.AGE` if the *EOD_BATCH_FL* is **Y** in the KDD_PRCSNG_BATCH_CONTROL table.
4. Updates the KDD_REVIEW table for all alerts from the current batch to set the Processing Complete flag to Y. This makes the alerts available for alert viewer.
5. Deletes any records in the KDD_DOC table that the system marks as temporary and are older than 24 hours.
6. Logs a message in the `<OFSAAI Installed Directory>/database/db_tools/logs/batch_control.log` file, stating that the end batch process has begun.

7.4.1.7 Identifying a Running Batch Process

At times, you may must know the name of a currently running batch, or verify that a batch is active. For example, during intra-day detection processing, many batches may be running simultaneously and you must identify one or more by name. If you set the batch control logging to display at the console, be aware that log messages are mixed with the output of the shell script; the output can be difficult to read.

To identify a running batch process, follow these steps:

1. Access the directory where the shell script resides: cd <OFSAAI Installed Directory>/database/db_tools/bin
2. Start the batch shell script: get_mantas_batch.sh The name of the currently running batch is written to standard output (refer to Configuring Console Output for more information).

7.4.1.8 Obtaining a Batch Name

After establishing the required Java environment and initiating various Java processing activities, the Batch Control Utility retrieves the name of the currently running batch from the KDD_PRCSNG_BATCH_CONTROL table.

The utility retrieves the name of the currently running batch from the KDD_PRCSNG_BATCH_CONTROL table. The utility returns the batch name to standard output.

7.5 Managing Calendar Manager Utility

After loading holidays into the KDD_CAL_HOLIDAY table and weekly off-days into the KDD_CAL_WKLY_OFF table, you can use the Calendar Manager Utility to update and manage OFSBD system calendars.

The `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` configuration file contains modifiable inputs that you use to run the utility (refer to Install Configuration for more information).

This section contains the following topics:

- Directory Structure
- Logs
- Calendar Information
- Using the Calendar Manager Utility

Directory Structure

The following table provides the directory structure for the Calendar Manager Utility in `<OFSAAI Installed Directory>/database/db_tools/`.

Table 7-14 Calendar Manager Utility Directory Structure

Directory	Description
bin/	Contains executable files, including the shell script set_mantas_date.sh.
lib/	Includes required class files in .jarformat.
mantas_cfg/	Contains configuration files , such as install.cfg and categories.cfg, in which you can configure properties and logging attributes.
logs/	Keeps the calendar_manager.log log file that the utility generates during execution.

Logs

As the utility updates the calendars in the OFSBD system, it generates a log that it enters in the `<OFSAAI Installed Directory>/database/db_tools/logs/calendar_manager.log` file

(the logging process time-stamps all entries). The log file contains relevant information such as status of the various Calendar Manager processes, results, and error records.

You can modify the current logging configuration for the Alert Purge Utility in the `<OFSAAI Installed Directory>/database/db_tools/log4j2.xml` files. For more information about logging in these configuration files, refer to Managing Common Resources for Batch Processing Utilities and Appendix A - Logging.

Calendar Information

The Calendar Manager Utility obtains all holidays and weekly off-days for loading into the OFSBD calendars by retrieving information from the KDD_CAL_HOLIDAY and KDD_CAL_WKLY_OFF tables. These tables contain calendar information that an Oracle client has provided regarding observed holidays and non-business days.

7.5.1 Using the Calendar Manager Utility

The Calendar Manager Utility runs as part of automated processing that a job scheduling tool such as Maestro or Unicenter AutoSys controls. The utility runs through a shell script, using values in parameters that the install.cfg file contains. The utility then populates the KDD_CAL database table with relevant OFSBD business calendar information.

The following sections describe this process, including tasks that you can perform when configuring the utility or running it manually.

Configuring the Calendar Manager Utility

The <OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg file contains common configuration information that Calendar Manager and other utilities require for processing (refer to the install.cfg sample). The following sample section from the install.cfg file provides configuration information specific to this utility, including default numerical values in the utility's two required parameters.

- calendar.lookBack: Determines how many days to iterate backward from the provided date during a calendar update.
- calendar.lookForward: Determines how many days to iterate forward from the provided date during a calendar update.

> **Note:**
>
> The maximum value that you can specify for either of these parameters is 999 days.

```
################ CALENDAR MANAGER CONFIGURATION ###################
The look back and look forward days of the provided date.
# These values are required to update the KDD_CAL table. The
# maximum look back or forward is 999 days.
```

The lookback period should be at least 90 days and as long as any alerts are likely to be open. The lookforward period does not must be more than 10 days. This is used when calculating projected settlement dates during data management.

> **Note:**
>
> **WARNING:**When you have configured the system to calculate alert and case age in Business Days, the calendar date of the current system date and the calendar date of the alert or case creation must be included in the calendar. As such, if you are running with a business date that is substantially behind the current system date, you should set the lookForward parameter for the calendar manager sufficiently high to ensure that the system date is included on the calendar. Additionally, if you have alerts that are open for a very long period, you should set the lookBack parameter sufficiently high to include the dates of your oldest open alerts. If the business calendar does not cover either of these dates, the processing reverts to calculating age in Calendar days.

The utility connects to the database employing the user that the utils.database.username property specifies in the install.cfg file.

7.5.1.1 Executing the Calendar Manager Utility

You can manage the Calendar Manager Utility as part of automated processing. You can run the utility either inside a batch process (that is, after calling the start_mantas_batch.sh script) or outside a batch.

1. Verify that the Behavior Detection database is operational: `tnsping <database instance name>`
2. Verify that the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` configuration file contains the correct source database connection information.
3. Go to the directory where the shell script resides: `cd <OFSAAI Installed Directory>/database/db_tools/bin`
4. Start the calendar manager shell script: `set_mantas_date.sh YYYYMMDD`

   where YYYYMMDD is the date on which you want to base the calendar , such as 20161130 for November 30, 2016. The utility then verifies that the entered date is valid and appears in the correct format.

   If you do not enter a date or enter it incorrectly, the utility terminates and logs a message that describes the error. The error message displays on the console only if you have output to the console enabled in the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/ categories.cfg` configuration file.

7.5.1.2 Updating the KDD_CAL Table

The Calendar Manager Utility retrieves information that it needs for updating OFSBD business calendars from the KDD_CAL_HOLIDAY and KDD_CAL_WKLY_OFF database tables. It then populates the KDD_CAL table accordingly. That is, for each calendar name found in the KDD_CAL_WKLY_OFF and KDD_CAL_HOLIDAY tables, the utility creates entries in KDD_CAL.

The following table provides the contents of the KDD_CAL table.

Table 7-15 KDD_CAL Table Contents

Column Name	Description
CLNDR_NM	Specific calendar name.
CLNDR_DT	Date in the range between the lookback and lookforward periods.
CLNDR_DAY_AGE	Number of calendar days ahead or behind the provided date. The provided date has age 0, the day before is 1, the day after is –1. For example, if a specified date is 20061129, the CLNDR_DAY_AGEof 20061128 = 1, and 20061130 = –1.
BUS_DAY_FL	Flagt hat indicates whether the specified date is a valid business day (set the flag to Y). Set this flag to N if the DAY_OF_WK column contains an entry that appears as a valid non-business day in the KDD_CAL_WKLY_OFF table, or a valid holiday in KDD_CAL_HOLIDAY.
BUS_DAY_AGE	Number of business days ahead or behind the provided date. If BUS_DAY_FL is N, BUS_DAY_AGE receives the value of the previous day's BUS_DAY_AGE.
DAY_OF_WK	Value that represents the day of the week: Sunday=1, Monday=2, Tuesday=3, ... Saturday=7.
WK_BNDRY_CD	Week's start day (SD) and end day (ED). • If this is the last business day for this calendar name for the week (that is, next business day has a lower DAY_OF_WK value), set to ED<x>, where <x> is a numeric counter with the start/end of the week that the provided date is in = 0. • If it is the first business day for this calendar name for this week (that is, previous business day has a higher DAY_OF_WK value), set to SD<x> Weeks before the provided date increment the counter, and weeks after the provided date decrement the counter. Therefore, "ED0" is always on the provided date or in the future, and "SD0" is always on the provided date or in the past.
MNTH_BNDRY_CD	Month's start day (SD) and end day (ED). • If this is the last business day for this calendar name for the month (that is, next business day in a different month), set to ED<y>, where *y* is a numeric counter with the start/end of the month that the provided date is in = 0. • If it is the first business day for this calendar for this month (that is, previous business day in a different month), set to SD<y>. Months before the provided date increment the counter, and months after the provided date decrement the counter. Therefore, "ED0" is always on the provided date or in the future, and "SD0" is always on the provided date or in the past.

Table 7-15 (Cont.) KDD_CAL Table Contents

Column Name	Description
BUS_DAY_TYPE_ CD	Indicates the type of business day: • N =Normal • C =Closed • S =Shortened
SESSN_OPN_TM	Indicates the opening time of the trading session for a shortened day. The format is HHMM.
SESSN_CLS_TM	Indicates the closing time of the trading session for a shortened day. The format is HHMM.
SESSN_TM_OFFST_TX	Indicates the timezone offset for SESSN_OPN_TMand SESSN_CLS_TM. The format is HH:MM.
QRTR_BNDRY_CD	Quarter's start day (SD) and end day (ED). • If this is the last business day for this calendar name for the quarter (that is, next business day in a different quarter), set ED to <y>, where y is a numeric counter with the start/end of the quarter that the provided date is in = 0. • If it is the first business day for this calendar name for this quarter (that is, previous business day is in a different quarter), set SD to <y>. Quarters before the provided date increment the counter, and quarters after the provided date decrement the counter. Therefore, "ED0" is always on the provided date or in the future, and "SD0" is always on the provided date or in the past.

If a batch is running, the system uses the date provided in the call to start the `set_mantas_date.sh` script. This script updates the `KDD_PRCSNG_BATCH_CONTROL.DATA_DUMP_DT` field.

7.6 Managing Data Retention Manager

Behavior Detection relies on Oracle partitioning for maintaining data for a desired retention period, providing performance benefits, and purging older data from the database.

The data retention period for business and market data is configurable. Range partitioning of the tables is by date.

The Data Retention Manager enables you to manage Oracle database partitions and indexes on a daily, weekly, and/or monthly basis (refer to Figure 24). This utility allows special processing for trade-related database tables to maintain open order, execution, and trade data prior to dropping old partitions. As administrator, you can customize these tables.

The utility accommodates daily, weekly, and monthly partitioning schemes. It also processes specially configured Mixed Date partitioned tables. The Mixed Date tables include partitions for Current Day, Previous Day, Last Day of Week for weeks between Current Day and Last Day of Previous Month, and Last Business Day of Previous Two Months.

The Data Retention Manager can:

- Perform any necessary database maintenance activities, such as rebuilding global indexes.
- Add and drop partitions, or both, to or from the date-partitioned tables.

Data Retention Manager provides a set of SQL procedures and process tables in the Behavior Detection database. A shell script and a configuration file that contain the various inputs set the environment that the utility uses.

This section covers the following topics:

Directory Structure

The following table provides the directory structure for the Data Retention Manager.

Table 7-16 Data Retention Manager Directory Structure

Directory	Contents
bin/	Executable files, including the run_drm_utility.shshell script.
lib/	Required class files in .jarformat.
mantas_cfg/	Configuration files , such as install.cfg and categories.cfg, in which you can configure properties and logging attributes.
logs/	File <OFSAAI Installed Directory>/database/db_tools/ logs/DRM_Utility.log that the utility generates during execution.

Logs

Oracle-stored procedures implement Data Retention Manager and conducts some logging on the database server. A configuration parameter in the install.cfg file controls the path to which you store the logs on the database server.

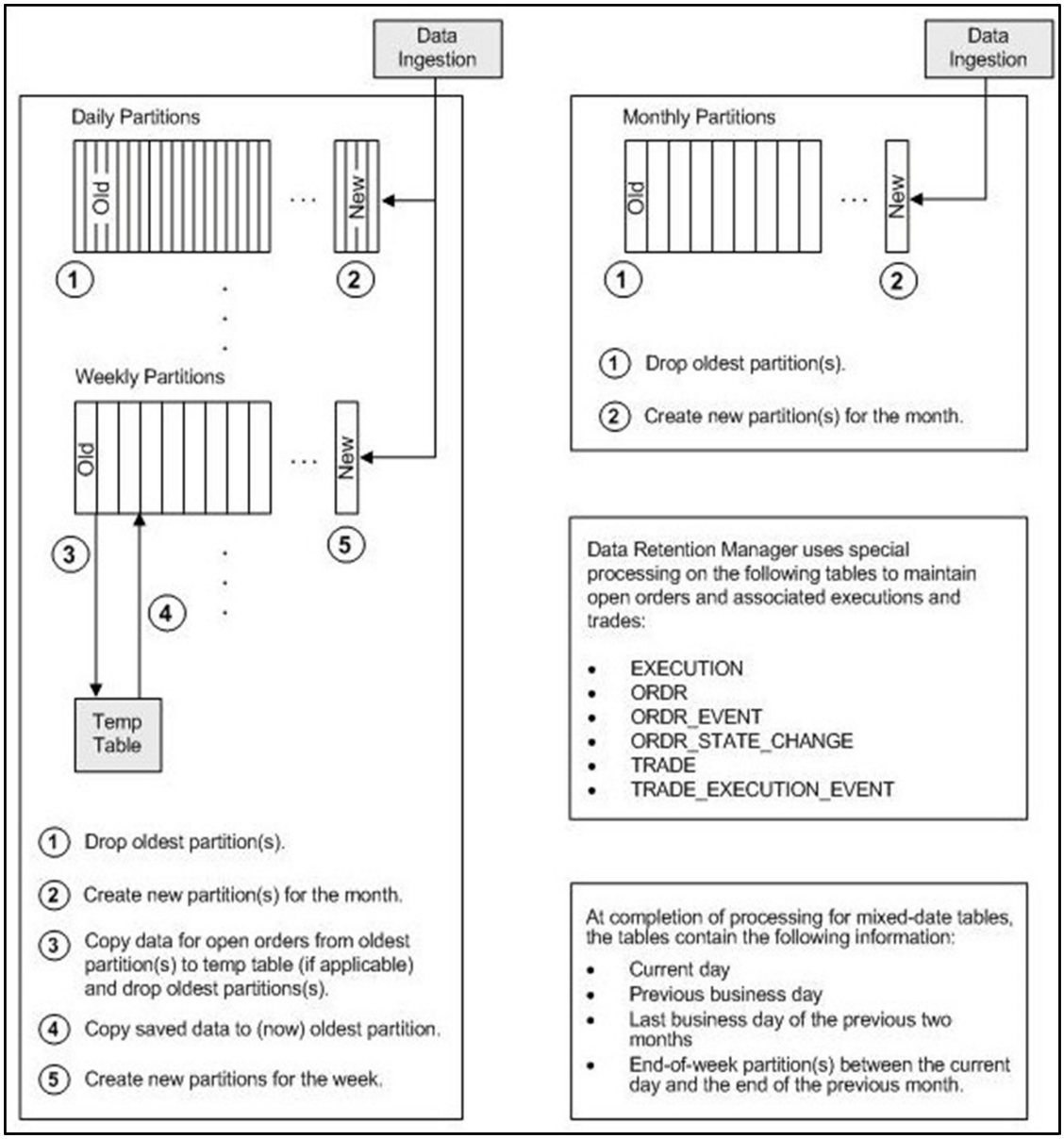
As the Data Retention Manager performs partitioning and indexing activities, it generates a log that it enters in the `<OFSAAI Installed Directory>/database/db_tools/logs/DRM_Utility.log` file (the logging process time-stamps all entries). The log file contains relevant information such as status of the various processes, results, and error records.

You can modify the current logging configuration for the Alert Purge Utility in the `<OFSAAI Installed Directory>/database/db_tools/log4j2.xml` files. For more information about logging in these configuration files, refer to Managing Common Resources for Batch Processing Utilitiesand Appendix A - Logging.

Processing Flow

The following figure illustrates the Data Retention Manager's process flow for daily, weekly, and monthly partitioning. Based on a table's retention period, the utility drops the oldest partition and then adds a new partition.

Figure 7-4 Database Partitioning Process



7.6.1 Using the Data Retention Manager

The Data Retention Manager typically runs as part of automated processing that a job scheduling tool such as Maestro or Unicenter AutoSys controls. However, you can run Data Retention Manager manually on a daily, weekly, or monthly basis to manage database tables.

The following sections describe how to configure and execute the utility and maintain database partitions and indexes.

Configuring the Data Retention Manager

To configure the Data Retention Manager, follow these steps:

1. Navigate to the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` file. This file contains common configuration information that Data Retention Manager and other utilities require for processing.
2. Use the install.cfg sample file to do a configuration.

> **Note:**
>
> The configuration parameters in the install.cfg are only used if command line parameters are not provided. It is strongly recommended that you provide command line parameters instead of using the install.cfg parameters.

The Data Retention Manager automatically performs system checks for any activity that may result in an error , such as insufficient space in the tablespace. If it discovers any such activity, it logs a Warning message that identifies the potential problem. If Data Retention Manager fails to run successfully, you can configure the utility so that the ingestion process for the following day still proceeds.

The following sample section from the install.cfg file provides other configuration information specific to this utility, including required and optional parameters.

```
######### DATA RETENTION MANAGER CONFIGURATION ##################
# Set the Data Retention Manager input variables here.
##
drm_operation=P drm_partition_type=A drm_owner=${schema.mantas.owner}
drm_object_name=A
```

This example shows default values that the system uses only when calling the utility with no command line parameters. The following table describes these parameters.

Table 7-17 Data Retention Manager Processing Parameters

Parameter	Description
drm_operation	Operation type: • P-Partition • AM-Add Monthly Partition • DM -Drop Monthly Partition • RI - Rebuild Indexes • RV - Recompile Views • T-Truncate Current Partition
drm_partition_type	Partition type: • D-Daily • W-Weekly • M- Monthly • X- Mixed-Date • A- All Partitions (Daily, Weekly, Monthly)
drm_owner	Owner of the object (Atomic schema owner).
drm_object_name	Object name. If performing an operation on all objects, the object name is A.
drm_weekly_proc_f l	Flag that determines whether partitioning occurs weekly (Y and N).

> **Note:**
>
> The system processes Daily partitioned tables (`drm_partition_type=D`) and Mixed-date partitioned tables (`drm_partition_type=X`) simultaneously. Therefore, you need only specify **D** or **X** to process these tables.

```
 P20050711 (Current Day)
P20050708 (Previous Day and End of week #1)
P20050701 (End of previous week #2)
P20050630 (End of previous Month #1)
P20050624 (End of previous week #3)
P20050617 (End of previous week #4)
P20050531 (End of previous Month #2)
```

7.6.1.1 Executing the Data Retention Manager

The Data Retention Manager should be executed nightly for Daily partitioned and Mixed-date partitioned tables, after the calendar has been set for the next business day. For weekly and monthly partitioned tables, the Data Retention Manager should be executed prior to the end of the current processing period.

Before you execute the Data Retention Manager, ensure that users are not working on the system. To avoid conflicts, Oracle recommends that you use this utility as part of the end-of-day activities.

> **Note:**
>
> Oracle recommends running the Data Retention Manager on Thursday or Friday for weekly partitioned tables and on or about the 23rd of each month for monthly partitioned tables. Be sure to set the system date with the Calendar Manager Utility prior to running the Data Retention Manager.

7.6.1.1.1 Running the Data Retention Manager

To run the Data Retention Manager manually, follow these steps:

1. Verify that the Behavior Detection database is operational: `tnsping <database instance name>`
2. Verify that the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` configuration file contains the correct source database connection information.
3. Access the directory where the shell script resides: `cd <OFSAAI Installed Directory>/database/db_tools/bin`
4. Start the batch shell script with the parameters in Data Retention Manager Processing Parameters: `run_drm_utility.sh <drm_operation> <drm_partition_type> <drm_owner> <drm_object_name> <drm_weekly_proc_fl>`

   The following are examples of running the script:

- To run the utility for all daily tables in the ATOMIC schema, execute the script: `run_drm_utility.sh P D BUSINESS A N`
- To run the utility to drop a monthly partition of the BUSINESS table ACCT_SMRY_MNTH, execute the script as follows (using the same parameters as in the previous example): `run_drm_utility.sh DM M BUSINESS ACCT_SMRY_MNTH N`

7.6.1.1.2 Creating Partitions

To create partition names, use the formats in the following table

Table 7-18 Partition Name Formats

Partition Type	Format and Description
Monthly	PYYYYMM - where YYYY is the four-digit year and MM is the two-digit month for the data in the partition. For example: Data for November 2006 resides in partition P200611. **Note:** The Data Retention Manager uses information in the KDD_CAL table to determine end-of-week and end-of-month boundary dates.
Weekly or Daily	PYYYYMMDD -where YYYY is the four-digit year, MM is the two-digit month, and DD is either the date of the data (daily) or the date of the following Friday (weekly) for the data in the partition. For example: Data for November 30, 2006 resides in partition P20061130. Data for the week of November 19 - November 23, 2006 resides in partition P20061123. **Note:** The Data Retention Manager uses information in the KDD_CAL table to determine end-of-week and end-of-month boundary dates.

> **Note:**
>
> Data Retention Manager assesses the current status of partitions on the specified table to determine the requested partition. If the system previously fulfilled the request, it logs a warning message.

The Data Retention Manager does not support multiple partition types on a single table. If an Oracle client wants to alter the partitioning scheme on a table, that client must rebuild the table using the new partitioning scheme prior to utilizing the Data Retention Manager. Then you can update the values in the Data Retention Manager tables to reflect the new partitioning scheme.

7.6.1.1.3 Maintaining Partitions

Partition maintenance procedures remove old data from the database so that the database does not continue to grow until space is insufficient. Daily, weekly, or monthly maintenance is necessary for tables that have daily, weekly, and monthly partitions, respectively.

Partition maintenance performs the following tasks:

1. Copies information related to open orders from the oldest partitions to temp tables (*EXECUTION, ORDR, ORDR_EVENT, ORDR_STATE_CHANGE TRADE and TRADE_EXECUTION_EVENT*)
2. Drops the oldest partitions for all partition types.
3. Inserts the saved data into what is now the oldest partition (applicable to tables with open orders).
4. Creates new partitions.
5. Recompiles the views that scenarios use.

Managing Daily Partitioning Alternative

The Data Retention Manager also enables you to build five daily partitions on a weekly basis.

To build partitions, follow these steps:

1. 1. Execute the `run_drm_utility.sh` shell script
2. Set the *drm_weekly_proc_flg* parameter to **Y**. For more information, refer to Data Retention Manager Processing Parameters.

This procedure eliminates the must perform frequent index maintenance; Oracle recommends doing this for large market tables. This approach builds the daily partitions for the next week. When creating the five daily partitions on a weekly basis, the Data Retention Manager should be executed prior to the end of the current week, to create partitions for the next week.

> **Note:**
>
> You must set the *WEEKLY_ADD_FL* parameter in the KDD_DR_MAINT_OPRTN table to **Y** so that the procedure works correctly. For more information about this parameter, refer to BUSINESS.KDD_DR_MAINT_OPRTN Table.

Partition Structures

The structures of business data partitions and market data partitions differ in the following ways:

- Business data partitions are pre-defined so that weekdays (Monday through Friday) are business days, and Saturday and Sunday are *weekly off-days*. Business data tables use all partitioning types.
- You can use the Calendar Manager Utility to configure a business calendar as desired. For more information about this utility, refer to Managing Calendar Manager Utility.
- Market data partitions hold a single day of data. The partitions use the PYYYYMMDD convention, where *YYYYMMDD* is the date of the partition.

Recommended Partition Maintenance

You should run partition maintenance as appropriate for your solution set. Oracle recommends that you run partition maintenance for AML on a daily basis (after setting the business date through the Calendar Manager Utility, and prior to the daily execution of batch processing), and Trading Compliance at least once a week.

Oracle recommends that you use the P (Partition) option when running the Data Retention Manager, as it drops older partitions and adds appropriate partitions in a single run of the utility. When performing monthly maintenance, you can add or drop a partition independently, as the following procedures describe.

Managing Alternative Monthly Partition

As part of an alternative method of monthly partition maintenance, you can either add or drop a monthly database partition. Refer to Data Retention Manager Processing Parameters when following these steps.

- **Adding a Monthly Database Partition:**To add a monthly partition, run the utility's shell script:

  ```
  run_drm_utility.sh AM M BUSINESS <object> N
  ```

  where **AM** is the *drm_operation* parameter that implies adding a monthly partition.

- **Dropping a Monthly Database Partition:** To drop a monthly partition, run the utility's shell script:

  ```
  run_drm_utility.sh DM M BUSINESS <object> N
  ```

  where, **DM** is the *drm_operation* parameter that implies dropping a partition.

7.6.1.1.4 Maintaining Indexes

As part of processing, the Data Retention Manager automatically rebuilds the database index and index partitions that become unusable. You do not need to maintain the indexes separately.

The utility enables you to rebuild global indexes by executing the following command:

```
run_drm_utility.sh RI M BUSINESS <object> N
```

where **RI** is the *drm_operation* parameter that implies rebuilding indexes.

7.6.1.2 Utility Work Tables

The Data Retention Manager uses the KDD_DR_MAINT_OPRTN and KDD_DR_JOB work tables during database partitioning.

KDD_DR_MAINT_OPRTN Table

The KDD_DR_MAINT_OPRTN table contains the processing information that manages Data Retention Manager activities. The following table provides these details.

Table 7-19 BUSINESS.KDD_DR_MAINT_OPRTN Table Contents

Column Name	Description
PROC_ID	Identifies the sequence ID for the operation to perform.
ACTN_TYPE_CD	Indicates the activity that the utility is to perform on the table: • A:Analyze • RI: Rebuild Indexes • P:Partition • RV: Recompile Views

Table 7-19 (Cont.) BUSINESS.KDD_DR_MAINT_OPRTN Table Contents

Column Name	Description
OWNER	Identifies an owner or user of the utility.
TABLE_NM	Identifies a database table.
PARTN_TYPE_CD	Indicates the partition type: • D:Daily • W:Weekly • M:Monthly • X: Mixed Date
TOTAL_PARTN_CT	Specifies the total number of partitions to be created, including the current partition. For example, for a daily partitioning scheme of four previous days and the current day, the value of this field is five (5).
BUFFER_PARTN_CT	Specifies the number of buffer partitions the utility is to maintain, excluding the current partition. For example, a two-day buffer has a value of two (2).
CNSTR_ACTN_FL	Determines whether to enable or disable constraints on the table during processing.
WEEKLY_ADD_FL	Indicates whether daily partitions are added for a week at a time. If set to Y, creates Daily Partitions for the next week. For example, if run on a Thursday, the DRM creates the five (5) partitions for the next week beginning with Monday.
NEXT_PARTN_DATE	Indicates starting date of the next partition that may get created, based on the current partitioned date.

Note:

For weekly partitioned tables, do not set the value to Y.

KDD_DR_JOB Table

The KDD_DR_JOB table stores the start and end date and time and the status of each process that the Data Retention Manager calls. The following table provides these details.

Table 7-20 BUSINESS.KDD_DR_JOB Table Contents

Column Name	Description
JOB_ID	Unique sequence ID.
START_DT	Start date of the process.
END_DT	End date of the process.
STATUS_CD	Status of the process: • RUN:Running • FIN: Finished successfully • ERR: An error occurred • WRN: Finished with a warning

KDD_DR_RUN Table

The KDD_DR_RUN table stores the start and end date and time and status of individual process runs that are associated with a table. The following table provides these details.

Table 7-21 BUSINESS.KDD_DR_RUN Table Contents

Column Name	Description
JOB_ID	Unique sequence ID.
PROC_ID	Process ID.
START_DT	Start date of the process.
END_DT	End date of the process.
RSULT_CD	Result of the process: • FIN: Finished successfully • ERR: An error occurred • WRN: Finished with a warning
ERROR_DESC_TX	Description of a resulting error or warning.

The system also uses the KDD_CAL table to obtain information such as the dates of the last-day-of-previous-month and end-of-weeks. Refer to KDD_CAL table contents.

7.7 Database Statistics Management

The system uses a script to manage Oracle database statistics. These statistics determine the appropriate execution path for each database query.

Logs

The *log.category.RUN_STORED_PROCEDURE* property controls logging for the *process.location* entry in the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/categories.cfg` file.

Using Database Statistics Management

The system calls the script as part of nightly processing at the appropriate time and with the appropriate parameters.

```
analyze_mantas.sh <analysis_type> [TABLE_NAME]
```

The <*analysis_type*> parameter can have one of the following values:

- DLY_POST_LOAD: Use this value to update statistics on tables that the system just loaded (for BUSINESS and MARKET related tables).
- ALL: Use this once per week on all schemas.
- DLY_POST_HDC: Use this value to update statistics of the alert-related archived data (in _ARC tables) that the Behavior Detection UI uses to display alerts. It is recommended that you do not modify this table. The Behavior Detection Historical Data Copy procedures uses this table to archive alert-related data.
- DLY_PRE_HDC: Use this value to update statistics of the Oracle-related tables that contain the alert-related information. It is recommended that you do not modify this table. The Behavior Detection Historical Data Copy procedures uses this table to archive alert-related data.

- DLY_POST_LINK: Use this value to update statistics of the Oracle- related tables that contain network analysis information. Run this option at the conclusion of the network analysis batch process.

The *[TABLE_NAME]* parameter optionally enables you to analyze one table at a time. This allows scheduling of the batch at a more granular level, analyzing each table as processing completes instead of waiting for all tables to complete before running the analysis process.

The metadata in the *KDD_ANALYZE_PARAM* table drive these processes. For each table this table provides information about the method of updating the statistics that you should use for each analysis type. Valid methods include:

- EST_STATS: Performs a standard statistics estimate on the table.

> **Note:**
>
> For the *EST_STATS* and *EST_PART_STATS* parameters, the default sample size that the analyze procedure uses is now based on *DBMS_STATS.AUTO_SAMPLE_SIZE*.

- EST_PART_STATS: Estimates statistics on only the newest partition in the table.
- IMP_STATS: Imports statistics that were previously calculated. When running an ALL analysis, the system exports statistics for the tables for later use.

Failure to run the statistics estimates can result in significant database performance degradation. These scripts connect to the database using the user that the *utils.database.username* property specifies, in the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` file.

The *install.cfg* file also contains *schema.mantas.owner*. The system derives schema name from this property.

For the ATOMIC Schema, there is no separate script for managing Oracle database statistics. But for improved query performance, you must manage the Oracle database statistics periodically.

To analyze table wise use, use the following commands: `ANALYZE table <Table name> compute statistics;`

```
ANALYZE table KDD_ACCOUNT compute statistics;
```

You can also perform whole schema analyze periodically.

7.8 Managing Flag Duplicate Alerts Utility

The Flag Duplicate Alerts Utility enables you to run a script daily after the generation of alerts. This script identifies the pairs of alerts that are possible duplicates. It then adds a system comment to each alert and identifies the paired alert in the comment as a *Possible Duplicate*.

External Entity-focused scenarios in Behavior Detection can generate alerts either on external identifiers , such as external account ID, or on names of parties outside the bank. The logic of the scenarios only generates the name-focused alerts when the name has been found with multiple (or no) external identifiers. This check is made across all transactions, not just the transactions involved in a particular alert. As a result, a single run of an External Entity-focused scenario can generate alerts involving the exact same transactions, one alert focused on the external Party ID, and one alert focused on the external Party Name.

Using Flag Duplicate Alerts Utility

The Flag Duplicate Alerts Utility looks at alerts that meet the following criteria:

- Entity focus (EN)
- Status of New (NW)
- Generated in the current running batch on the current date

The utility selects and compares alerts that meet the listed criteria above. It then determines whether generation of the alert is based on the same set of transactions for the same scenario and with different focuses , such as if one alert is an ID and the other is a Name. The utility flags these alerts as possible duplicates and adds a system comment which can be viewed on the Audit tab of the alert (each alert cross-references the other). For example:

```
Possible duplicate of alert xxxxx.
```

Executing Flag Duplicate Alerts Utility

To execute the Flag Duplicate Alerts Utility, run the following script after the Alert Creator, Assigner, and Auto-Close processes (jobs) have completed: `<OFSAAI Installed Directory>/database/db_tools/bin/flag_duplicate_alerts.sh`

The system writes log information for this process to the following location: `<OFSAAI Installed Directory>/database/db_tools/logs/run_stored_procedure.log`

7.9 Managing Notifications

Notifications appear on the UI on the Home page and help alert users to items requiring their attention. Depending on the method of generation, Event Based and Batch Based.

Event Based

These notifications are always associated with an event. Following are the event based notifications:

- Re-assigned alerts notification: Notification is generated to the new owner of the Alert upon reassignment of the alert. If the user who reassigned the alert is also the new owner, no notification is generated. If the new owner is a pool then notification is generated to all users who are members of the organization represented by that pool.
- Alert Data Transfer Unsuccessful: In Asynchronous alert data transfer mode, if the data transfer during promotion of an alert to a case or linking of an alert to a case is Unsuccessful, then a notification is generated to the user who is taking the action, the owner of the alert, and the owner of the case, and then assigned to the user of the case.

Batch Based

These notifications are the result of processing of end_mantas_batch.sh. Following are the batch based notifications:

- Alerts Near Due Date notifications: Notification is generated to the owner of the alerts if the due date of the alert falls within the configurable parameter set in Installation parameter table.

These notifications are generated after the complete execution of Batch (provide the batch name) and can be seen in the Notification Grid in landing page. Each user sees the notifications which are relevant to them.

> **Note:**
>
> You can set the near due date and display of notification parameters from the Manage Parameters screen. (Refer to the *Configuration Guide* for more information).

7.10 Refreshing Temporary Tables

Some behavior detection patterns use the temporary tables as part of the detection process.

Logs

The *log.category.REFRESH_TEMP_TABLE.location* property in the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/categories.cfg` file controls logging for this process. The system writes log information for this process to the following location:

```
<OFSAAI Installed Directory>/database/db_tools/logs/refresh_temp_table.log
```

Using Refreshing Temporary Tables

The BD ATOMIC schema defines these tables; the tables have corresponding views that are used to populate them. Prior to running these patterns, run the *refresh_temp_table.sh* script.

The script has the following calling signature: `refresh_temp_table.sh <table_name> <view_name>`

Where:

- table_name identifies the name of the table to populate.
- view_name identifies the name of the view to run to populate the table.

This procedure deletes all records in the target table prior to running the view to populate it. It then estimates statistics for the newly populated table. This procedure logs into the database with the user that the *utils.miner.user* property identifies in the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` file.

Populating Temporary Tables for Scenarios

Scenarios typically depend on data management to complete processing. However the following scenarios depend on population of Temp Tables to populate data.

- IML/CU) Hidden Relationships
- (ML/AC) Networks of Accounts, Entities, and Customers
- (FR/AC) Networks of Accounts, Entities, and Customers
- (CST/AC) Customers Who Have Experienced a Large Loss Recently
- (CST/HH) Customers Who Have Experienced a Large Loss Recently

The Link Analysis scenario also depends on the network job creation before the sequence matcher part of the scenario runs.

7.10.1 Enhancing Performance Populating Network Temporary Tables

If you are experiencing performance issue with Refresh Temp Utility views when populating temp tables for network scenarios, add the following entries into the KDD_INSTALL_PARAM table.

These parameters are used in the refreshtemptable job to add HINT in the SQL code.

- PARAM_ID: Unique number. For example, max(PARAM_ID) + 1
- PARAM_NM : Name of the View for which the hint should be added.
- PARAM_VALUE_TX : NULL
- PARAM_VALUE_TYPE_TX : NULL
- PARAM_CAT_CD : RefershTempTable (Hardcoded value)
- ATTR_1_CD: Batch_name (Hardcoded value)
- ATTR_1_VALUE_TX: <batch_name>. For example, DLY
- ATTR_2_CD: Select_Hint (Hardcoded value)
- ATTR_2_VALUE_TX: <hint> that would be used in the query Format /*+ <hint> */ For example, /*+ PARALLEL(8) */
- ATTR_3_CD: Insert_Hint (Hardcoded value)
- ATTR_3_VALUE_TX: <hint> hint to be used in the insert statement Format /*+ <hint> */ For example, /*+ APPEND */

7.10.2 IML-HiddenRelationships-dINST

To populate the temporary tables for IML-HiddenRelationships-dINST scenario, follow these steps:

1. Execute the following refresh temporary table processes (these commands can be run in parallel).
   - If you run a scenario with the *Include records for active batch parameter = 'N'* (All records loaded during lookback period will analyzed regardless of the name of the batch process which means it will include records from other batches in a multi-country installation)
     - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_JRNL TMP_HIDREL_NT_JRNL_VW
     - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_WIRE TMP_HIDREL_NT_WIRE_VW
     - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_ACTAXID TMP_HIDREL_NT_ACTAXID_VW
     - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_ACADDR TMP_HIDREL_NT_ACADDR_VW
     - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_ACPHONE TMP_HIDREL_NT_ACPHONE_VW
     - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_ACEMAIL TMP_HIDREL_NT_ACEMAIL_VW

  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_ACPSWRD TMP_HIDREL_NT_ACPSWRD_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_INST TMP_HIDREL_NT_INST_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_WIREACBENE TMP_HIDREL_NT_WIREACBENE_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_WIREACORIG TMP_HIDREL_NT_WIREACORIG_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_CUACTAXID TMP_HIDREL_NT_CUACTAXID_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_CUACADDR TMP_HIDREL_NT_CUACADDR_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_CUACPHONE TMP_HIDREL_NT_CUACPHONE_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_CUACEMAIL TMP_HIDREL_NT_CUACEMAIL_VW

- If you run scenario with parameter *Include records for active batch = 'Y'* Only records loaded during the lookback period with batch name which is currently active will be analyzed which means it will not include records from other batches in a multi-country installation).
  - <OFSBDF Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_JRNL TMP_HIDREL_NT_JRNL_BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_WIRE TMP_HIDREL_NT_WIRE_ BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_ACTAXID TMP_HIDREL_NT_ACTAXID_ BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_ACADDR TMP_HIDREL_NT_ACADDR_ BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_ACPHONE TMP_HIDREL_NT_ACPHONE_ BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_ACEMAIL TMP_HIDREL_NT_ACEMAIL_ BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_ACPSWRD TMP_HIDREL_NT_ACPSWRD_ BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_INST TMP_HIDREL_NT_INST_ BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_WIREACBENE TMP_HIDREL_NT_WIREACBENE_ BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_WIREACORIG TMP_HIDREL_NT_WIREACORIG_ BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_CUACTAXID TMP_HIDREL_NT_CUACTAXID_ BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_CUACADDR TMP_HIDREL_NT_CUACADDR_ BATCH_VW

- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_CUACPHONE TMP_HIDREL_NT_CUACPHONE_BATCH_VW
- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh
- <OFSBDF Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_HIDREL_NT_CUACEMAIL TMP_HIDREL_NT_CUACEMAIL_BATCH_BATCH_VW

2. Execute the link analysis/network generation job. The product job template ID is **114698616**.
   - If you ran a scenario where the *Include records for active batch parameter = 'N'* (All records loaded during lookback period will analyzed regardless name of batch process) then insert the record to KDD_PARAM_BINDING following these steps:
     a. `insert into KDD_PARAM_BINDING values ('filter_by_batch', 'Link Analysis', <param_set_id>, <true or false>)` For example:

     ```
     insert into KDD_PARAM_BINDING values ('filter_by_batch', 'Link Analysis', 114698653, 'false'

     )
     ```

     b. Run the Link Analysis IGN job which has a **'false**' value in KDD_PARAM_BINDING
3. Execute the scenario job with appropriate value in parameter *Include records for active batch* . The product job template ID is **116200024**.

7.10.3 ML-NetworkOfAcEn-fAC

To populate the temporary tables for ML-NetworkOfAcEn-fAC scenario, follow these steps:

1. Execute these refresh temporary table processes (these commands can be run in parallel):
   - If you run a scenario with parameter *Include records for active batch = 'N'* (All records loaded during lookback period will analyzed regardless of the name of the batch process which means it will include records from other batches in a multi-country installation)
     - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_ACCTADDR TMP_NETACENCU_NT_ACCTADDR_VW
     - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_ACCTEMAIL TMP_NETACENCU_NT_ACCTEMAIL_VW
     - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_ACCTPHONE TMP_NETACENCU_NT_ACCTPHONE_VW
     - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_ACCTPSWRD TMP_NETACENCU_NT_ACCTPSWRD_VW
     - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_ACCTTAXID TMP_NETACENCU_NT_ACCTTAXID_VW
     - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_CUACADDR TMP_NETACENCU_NT_CUACADDR_VW

  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_CUACEMAIL TMP_NETACENCU_NT_CUACEMAIL_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_CUACPHONE TMP_NETACENCU_NT_CUACPHONE_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_CUACTAXID TMP_NETACENCU_NT_CUACTAXID_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_JRNL TMP_NETACENCU_NT_JRNL_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_WIREACBENE TMP_NETACENCU_NT_WIREACBENE_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_WIREACORIG TMP_NETACENCU_NT_WIREACORIG_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_WIRETRXN TMP_NETACENCU_NT_WIRETRXN_VW

- If you run a scenario with parameter *Include records for active batch* = *'Y'* Only records loaded during the lookback period with batch name which is currently active will be analyzed[which means it will not include records from other batches in a multi-country installation)
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_ACCTADDR TMP_NETACEN_ACCTADDR_BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_ACCTEMAIL TMP_NETACEN_ACCTEMAIL_BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_ACCTPHONE TMP_NETACEN_ACCTPHONE_BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_ACCTPSWRD TMP_NETACEN_ACCTPSWRD_BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_ACCTTAXID TMP_NETACEN_ACCTTAXID_BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_CUACADDR TMP_NETACEN_CUACADDR_BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_CUACEMAIL TMP_NETACEN_CUACEMAIL_BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_CUACPHONE TMP_NETACEN_CUACPHONE_BATCH_VW
  - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_CUACTAXID TMP_NETACEN_CUACTAXID_BATCH_VW

- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_JRNL TMP_NETACEN_JRNL_BATCH_VW
- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_WIREACBENE TMP_NETACEN_WIREBENE_BATCH_VW
- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_WIREACORIG TMP_NETACEN_WIREORIG_BATCH_VW
- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_NETACENCU_NT_WIRETRXN TMP_NETACEN_WIRETRXN_BATCH_VW

2. Execute the link analysis/network generation job. The product job template ID is **114698120**.
   - If you run a scenario with parameter *Include records for active batch = 'N'*
     a. Insert a record to KDD_PARAM_BINDING using the following instructions: `insert into KDD_PARAM_BINDING values ('filter_by_batch', 'Link Analysis', <param_set_id>, <true or false>)` For Example:

     ```
     insert into KDD_PARAM_BINDING values ('filter_by_batch', 'Link Analysis', 118745109, 'false')
     ```

     b. Run the Link Analysis IGN job which has a '**false**' value in KDD_PARAM_BINDING.
   - If you run a scenario with parameter *Include records for active batch = 'Y'*
     a. Insert a record to KDD_PARAM_BINDING using the following instructions: `insert into KDD_PARAM_BINDING values ('filter_by_batch', 'Link Analysis', <param_set_id>, <true or false>)` For Example:

     ```
     insert into KDD_PARAM_BINDING values ('filter_by_batch', 'Link Analysis', 118745110, 'true')
     ```

     b. Run the Link Analysis IGN job which has a 'true' value in KDD_PARAM_BINDING
3. Execute the scenario job. The product job template ID is **114698631**.

7.10.4 FR-NetworkOfAcEn-fAC

To populate the temporary tables for FR-NetworkOfAcEn-fAC scenario, follow these steps:

1. Execute these refresh temporary table processes (these commands can be run in parallel.):
   - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_FRNTWRK_NT_ACCTADDR TMP_FRNTWRK_NT_ACCTADDR_VW
   - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_FRNTWRK_ACCTEMAIL TMP_FRNTWRK_ACCTEMAIL_VW
   - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_FRNTWRK_ACCTPHONE TMP_FRNTWRK_ACCTPHONE_VW
   - <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_FRNTWRK_ACCTPSWRD TMP_FRNTWRK_ACCTPSWRD_VW

- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_FRNTWRK_ACCTTAXID TMP_FRNTWRK_ACCTTAXID_VW
- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_FRNTWRK_CUACADDR TMP_FRNTWRK_CUACADDR_VW
- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_FRNTWRK_CUACEMAIL TMP_FRNTWRK_CUACEMAIL_VW
- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_FRNTWRK_CUACPHONE TMP_FRNTWRK_CUACPHONE_VW
- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_FRNTWRK_CUACTAXID TMP_FRNTWRK_CUACTAXID_VW
- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_FRNTWRK_JRNL TMP_FRNTWRK_JRNL_VW
- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_FRNTWRK_WIREACBENE TMP_FRNTWRK_WIREACBENE_VW
- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_FRNTWRK_WIREACORIG TMP_FRNTWRK_WIREACORIG_VW
- <OFSAAI Installed Directory>/database/db_tools/bin/refresh_temp_table.sh TMP_FRNTWRK_WIRETRXN TMP_FRNTWRK_WIRETRXN_VW

2. Execute the link analysis/network generation job. The product job template ID is **118745091**.
3. Execute the scenario job. The product job template ID is **117350084**.

7.10.5 CST-UncvrdLongSales-dRBPC

To populate the temporary table UNCVRD_LONG_TRADE_TEMP for CST-UncvrdLongSales-dRBPC scenario, follow these steps:

This should be run after the ingestion is completed, just before the scenario job runs.

1. Execute this command to refresh temporary table process:

   ```
   <OFSAAI Installed Directory>/database/db_tools/ run_p_uncvrdlongsales_ew.sh
   ```

2. Execute the scenario job.

7.11 Managing Truncate Manager

The data management subsystem calls the run_truncate_manager.sh script to truncate tables that require complete replacement of their data.

Logs

The *log.category.TRUNCATE_MANAGER.location* property in the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/categories.cfg` file controls logging for this utility. The system writes log information for this process to the following location: `<OFSAAI Installed Directory>/database/db_tools/logs/truncate_manager.log`

Using the Truncate Manager

For the *run_truncate_manager.sh* script to take the table name as an argument, the table must exist in the BD ATOMIC schema. The script logs into the database using the user that the

truncate.database.username property specifies in the `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` file.

The script has the following calling signature: `run_truncate_manager.sh <table_name>`

> **Note:**
>
> This process is not intended to be called independently; only the Ingestion Manager subsystem should use it.

7.12 Managing ETL Process for Scenario Tuning

For inserting and updating records into the KDD_TA_ML_DATA, KDD_TA_BC_DATA, and KDD_TA_TC_DATA tables, there are two shell scripts that are used to call the database procedures.

run_insert_ta_utility.sh

This script calls the P_TA_ML_INSERT_BREAKS, P_TA_BC_ INSERT_BREAKS, and P_TA_TC_INSERT_BREAKS procedures, which insert data into the KDD_TA_ML_DATA, KDD_TA_BC_DATA, and KDD_TA_TC_DATA tables, respectively, based on the CREAT_TS of the alerts in relation to the LAST_RUN_DT from KDD_TA_LAST_RUN (values for RUN_TYPE_CD are ML_I, BC_I, and TC_I).

run_update_ta_utility.sh

This script calls the P_TA_ML_UPDATE, P_TA_BC_UPDATE, and P_TA_TC_UPDATE procedures, which update QLTY_RTNG_CD in the KDD_TA_ML_DATA, KDD_TA_BC_DATA, and KDD_TA_TC_DATA tables, respectively, for any Review closed since the last run based on LAST_RUN_DT from KDD_TA_LAST_RUN (values for RUN_TYPE_CD are ML_U, BC_U, and TC_U). The CLS_CLASS_CD value from KDD_REVIEW is used as the new QLTY_RTNG_CD.

> **Note:**
>
> - The log for these scripts is written in the *run_stored_procedure.log* file under the `<OFSAAI Installed Directory>/database/db_tools/logs` directory.
> - The *LAST_RUN_DT* column in the *KDD_TA_LAST_RUN* table is only updated for inserts and updates if at least one or more records were inserted or updated. The *LAST_RUN_DT* column is not updated for significant errors that resulted in no records being updated. These scripts are a part of the database tools and reside in the `<OFSAAI Installed Directory>/database/db_tools/bin` directory.
> - You can run this utility anytime, that is, it is not necessary to run this utility during specific processing activities.

7.12.1 Running Scenario Tuning

To run the scenario tuning utility, follow these steps:

1. Go to ATOMIC schema and execute the following query:

```
select distinct (creat_ts)
from kdd_review t
where t.review_type_cd = 'AL'
and SCNRO_DISPL_NM <> 'User Defined'
and PRCSNG_BATCH_NM = 'DLY';
```

2. Set date as per dates returned from above SQL.

   For example, if *CREATE_TS* is **05/21/2013** in the kdd_review table, then a date of **05/17/2013** (Friday of last week) is set from the `$FICHOME/database/ db_tools/bin` folder.

3. Execute the following command:

```
start_mantas_batch.sh DLY
set_mantas_date.sh 20130517 --(Friday of last week)
```

4. Execute DRM utility to create partitions, refer to Data Retention Manager Processing Parameters for parameter values:

```
run_drm_utility.sh <Partition> <Weekly> <schema> <Table name>
<drm_weekly_proc_fl>
```

   There should be different variations for each Oracle product. For example:

```
run_drm_utility.sh P W ATOMIC KDD_TA_ML_DATA N
run_drm_utility.sh P W ATOMIC KDD_TA_BC_DATA N
run_drm_utility.sh P W ATOMIC KDD_TA_TC_DATA N
```

5. Execute the following Insert and Update Scenario Tuning scripts from the *$FICHOME/ database/ db_tools/bin* folder:

```
run_insert_ta_utility.sh
run_update_ta_utility.sh
```

6. Repeat the process if you have more than one date returned from the query in Step 1.

8

Managing Administrative Utilities

OFSBD provides utilities that enable you to set up or modify a selection of database processes.

Several Behavior Detection database utilities that configure and perform system pre-processing and post-processing activities are not tied to the batch process cycle:

- Data Analysis Tool: Assists a Data Miner or Data Analyst in determining how well a customer has populated the Production Data Model.
- Get Dataset Query with Thresholds Utility: Enables the extraction of dataset SQL complete with substituted thresholds for analysis of the SQL outside of the Behavior Detection application.
- Trusted Pairs and Alert Suppression: OFS BD API offers services to allow OFS BD to consume trusted pair and Alert Suppression information present in the AML Case Management System.
- Scenario Migration Utility: Extracts scenarios, datasets, networks, and associated metadata from a database to flat files and loads them into another environment.
- Investigation Management Configuration Migration Utility: Provides a means to load alert and case configuration metadata into OFSBD as well as allows you to move configuration metadata between installations of OFSBD.

Common Resources for Administrative Utilities

Configuration files enable the utilities to share common resources such as database configuration, directing output files, and setting up logging activities.

8.1 Managing Data Analysis Tool

The Data Analysis Tool enables you to determine how well a customer has populated the Production Data Model.

By reviewing the quality of data in each of the tables that the schema identifies, the Data Analysis Tool indicates how well the supplied data can support scenarios. The tool does not make assumptions about data quality. Rather, it provides a repeatable way to run a set of analytical queries across the data. You can then use the results to direct further analysis.

The following are the key features of the Data Analysis Tool:

- Counts all table rows in the schema.
- Identifies unique values and their distribution against the table.
- Determines the number of null occurrences for a specified column.
- Determines the number of padded spaces that occur for a specified column.
- Checks referential integrity between tables.

The tool provides its results in either a text or Hypertext Markup Language (HTML) file. You can then use these results to direct an investigation for data quality.

> **Note:**
>
> To use the Data Analysis Tool effectively, you must have basic knowledge of Structured Query Language (SQL) and Extensible Markup Language (XML).

The following sections provide instructions for using the tool:

- Configuring Data Analysis Tool
- Using the Data Analysis Tool
- Logs
- Troubleshooting the Data Analysis Tool

8.1.1 Configuring Data Analysis Tool

The Data Analysis Tool uses the install.cfg and analysis.xml (or similar) configuration files. You edit either file in a text editor such as vi. To produce well-formed XML files, however, you should edit the XML file in a validating XML editor.

Configuring General Tool Properties

Use the Data Analysis Tool to configure the general tool properties. To access the Data Analysis Tool, navigate to the *install.cfg* file that resides in `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg`. The following table provides the configuration instructions for the properties that the Data Analysis Tool uses in the install.cfg file.

Table 8-1 Configuring General Tool Properties

Property	Description	Example
database.driver Name	Database connection driver that the utility is to use.	database.driverName =oracle.jdbc.driver. OracleDriver
utils.database.urlName	Database connection string that the Data Analysis Tool is to use.	utils.database.urlName=jdbc:oracle:oci: @PROD_DB
schema.business.owner	Database user for the ATOMIC schema.	schema.business. owner=ATOMIC
dat.database. username	User name for the database. The Data Analysis Tool connects to the database as the ATOMIC USER for the appropriate privileges.	dat.database.username= ATOMIC
dat.database. password	Password for the database. This is set by the Password Manager Utility.	
dat.analysis. input	Path and name for the XML input file. By default, this is the analysis.xml file under the `<OFSAAI Installed Directory>/database/ db_tools/mantas_cfgdirectory`. You can override this at the command line.	dat.analysis.input=/opt/ mantas/ database/ db_tools/ mantas_cfg/ analysis.xml

Table 8-1 (Cont.) Configuring General Tool Properties

Property	Description	Example
dat.analysis. output	Path and file name of output file for the analysis report. You can override this at the command line.	dat.analysis.output=/opt/ mantas/ database/ db_tools/ data/ analysis.html
dat.output. format	Output format for the report. Acceptable output formats are HTML or TEXT.	dat.output.format=HTML
dat.output. delimiter	Not currently used. The delimiter for the format TEXT is always a comma (“,”).	
schema.market. owner	Database user for the ATOMIC schema.	schema.market.owner=ATOMIC

8.1.1.1 Analysis Constraints

For both distinct value counts and null counts, you can specify optional constraints. The XML format for two of the files is identical. For a join analysis, the XML format uses a filter element that is similar to a constraint. However, you must specify the table name.

To specify a constraint, use the <*CONSTRAINT*> element. The <*CONSTRAINT*> element requires three attributes:

- Field: Database field name to which the constraint applies
- Value: Value being compared
- Operator: Operator used in the comparison

The following table lists valid code operators.

Table 8-2 XML Code Operators

XML Code Operator	Comparison Operator
GT	>
LT	<
EQ	=
LTE	<=
GTE	>=
NEQ	<>
EMPTY	Blank Character

The following code sample illustrates the use of the <*CONSTRAINT*> element:

```
<CONSTRAINT field="DATA_DUMP_DT" operator="EQ" value="15-NOV-2006"
    />
```

To include a constraint that filters out null columns, use the *EMPTY* operator and set the value to **is not null**. The following example illustrates the use of the *EMPTY* operator:

```
<CONSTRAINT field="DATA_DUMP_DT" operator="EMPTY" value="is not null"
        />
```

You can also use the *EMPTY* operator to perform more complex comparisons than those that other operators support that Table 55 lists. When using the *EMPTY* operator, the generated SQL statement includes the field name, a space, and the text within the value string. As such, representation of more complex operations is possible.

An AND operator joins any existing, multiple <*CONSTRAINT*> elements.

When adding date constraints as in the first example above, you must specify the date in the same format as the database's NLS Date Format. Oracle recommends **DD-MON-YYYY** as the default format

8.1.1.2 Analyzing Distinct Values for Fields of Interest

Identifying the table and one or more column combinations of interest provides a combination of distinct values and number of occurrences in the table.

The following code illustrates the required structure of this analysis within the following elements:

```
<ANALYSIS>
<TABLES>
<analysis for distinct values occurs here>
</TABLES>
</ANALYSIS>
```

- The name attribute of the <TABLE> element identifies the table against which this analysis is applied.
- The <VALUES> element identifies targeted columns.
- The field attribute of the <COLUMN> element sets each database column.

Application of filters to an analysis is possible if the <*CONSTRAINT*> element identifies the filter. The following code illustrates the structure for using a filter:

```
<TABLE name="table name">
<!-- get distinct value for one column -->
<VALUES>
<COLUMN field="column name"/>
<!-- Constraint feature is optional.
May contain one or more constraints. -->
<CONSTRAINT field="column name" operator="operator"
value="filter value" />
</VALUES>
<!-- get distinct value for many columns -->
<VALUES>
<COLUMN field="column name"/>
<COLUMN field="column name"/>
<!-- Constraint feature is optional.
May contain one or more constraints. -->
<CONSTRAINT field="column name"
operator="operator"value="filter value" />
</VALUES>
</TABLE>
```

The following XML code illustrates use of a filter:

```
<ANALYSIS>
<TABLES>
<TABLE name="ACCT">
<VALUES>
<COLUMN field="ACCT_TYPE1_CD"/>
<COLUMN field="ACCT_TYPE2_CD"/>
</VALUES>
</TABLE>
<TABLE name="CUST">
<VALUES>
<COLUMN field="CUST_TYPE_CD"/>
<CONSTRAINT field="DATA_DUMP_DT" operator="EQ"
value="15-NOV-2006" />
</VALUES>
</TABLE>
</TABLES>
<ANALYSIS>
```

This XML code executes the following queries:

```
select ACCT_TYPE1_CD, ACCT_TYPE2_CD, count(1)
from ACCT
group by ACCT_TYPE1_CD, ACCT_TYPE2_CD
select CUST_TYPE_CD, count(1)
from CUST
where DATA_DUMP_DT='15-NOV-2006'
group by CUST_TYPE_CD
```

8.1.1.3 Analyzing Null and Padded Space Count

Null and padded space count analysis provides the number of occurrences for null values and padded spaces for a particular field in a table.

You perform this analysis by identifying the table and one or more columns of interest. The null analysis feature has the following limitations:

- The feature is optional.
- The field identified for the specified table can be analyzed only once within the <*NULLS*> element per table.
- The filtering feature for the null analysis is optional and can have multiple constraints.

The structure to perform this analysis is:

```
<ANALYSIS>
<TABLES>
<!-- analysis for null counts occurs here -->
</TABLES>
</ANALYSIS>
```

Within the <*TABLE*> element, the name attribute identifies the table to be analyzed. The targeted columns are identified within the <*NULLS*> element. The field attribute in the <*NULL*>

element sets each column name. Apply filters to the analysis within the *<CONSTRAINT>* element. The following code illustrates the structure for the a null and padded space count analysis:

```
<TABLE name="table name">
<!-- May contain one or more columns -->
<NULLS><!-- With no constraints -->
<NULL field="column name"/><!-- With constraints -->
<NULL field="column name">
<!-- Constraint feature is optional.
May contain one or more constraints. -->
<CONSTRAINT field="column name" operator="operator"
value="filter value" />
        </NULL>
    </NULLS>
</TABLE>


<TABLE name="ACCT">
    <NULLS>
        <NULL field="ACCT_TYPE1_CD"/>
        <NULL field="RGSTN_TYPE_CD">
            <CONSTRAINT field="DATA_DUMP_DT" operator="EQ"
                value="15-NOV-2006" />
        </NULL>
    </NULLS>
<TABLE name="ACCT">
```

This code executes the following queries:

```
SELECT sum(case when ACCT_TYPE1_CD is null then 1 else 0 end)as NULL_CT0,
sum(case when ACCT_TYPE1_CD <> ltrim(rtrim(ACCT_TYPE1_CD))
then 1 else 0 end) as SPACE_CT0,
sum(case when RGSTN_TYPE_CD is null
and DATA_DUMP_DT='15-NOV-2006' then 1 else 0 end) as NULL_CT1,
sum(case when RGSTN_TYPE_CD <> ltrim(rtrim(RGSTN_TYPE_CD))
and DATA_DUMP_DT='15-NOV-2006' then 1 else 0 end) as SPACE_CT1
FROM ACCT a
```

8.1.1.4 Analyzing Join Counts

A join identifies the relationship between two tables by common fields. Checking for join counts determines the referential integrity between two or more tables.

Determine join counts as follows:

- Simple join between two or more tables.
- Simple join between two or more tables with filter restriction.
- Join count of distinct values for specific column.

The join count analysis is structured within the following elements:

```
<ANALYSIS>
    <JOINS>
```

```
        <!-- analysis for referential integrity here -->
    </JOINS>
</ANALYSIS>
```

Simple Join

A join is set within the <JOIN> element. To retrieve the join count between two or more tables, the joins are identified within the <MULTIJOIN> element. Within this <MULTIJOIN> element, multiple <JOIN> elements can be set. Because a join retrieves the join count between two or more tables, <LEFT> and <RIGHT> elements are used to indicate the tables. The <LEFT> element identifies the first table and its field using the table and column attributes. The table and column attributes for the <RIGHT> element identify the second table and field. The structure for a simple join count analysis is:

```
<MULTIJOIN>
<!-- May contain more than one JOIN element -->
    <JOIN>
        <LEFT table="table name" column="column" />
        <RIGHT table="table name" column="column" />
    </JOIN>
</MULTIJOIN>


<ANALYSIS>
<JOINS>
<MULTIJOIN>
<JOIN>
<LEFT table="ACCT" column="ACCT_INTRL_ID" />
<RIGHT table="CUST_ACCT" column="ACCT_INTRL_ID" />
</JOIN>
</MULTIJOIN>
<MULTIJOIN>
<JOIN>
<LEFT table="ACCT" column="ACCT_INTRL_ID" />
<RIGHT table="CUST_ACCT" column="ACCT_INTRL_ID" />
</JOIN>
<JOIN>
<LEFT table="CUST" column="CUST_INTRL_ID" />
<RIGHT table="CUST_ACCT" column="CUST_INTRL_ID" />
</JOIN>
</MULTIJOIN>
</JOINS>
</ANALYSIS>
```

This XML code executes the following queries:

```
select count(1)
from ACCT a, CUST_ACCT b
where a.ACCT_INTRL_ID=b.ACCT_INTRL_ID

select count(1)
from ACCT a, CUST_ACCT b, CUST c
where a.ACCT_INTRL_ID=b.ACCT_INTRL_ID
and c.CUST_INTRL_ID=b.CUST_INTRL_ID
```

Simple Join with Filter Restriction

Adding a filter to the joins determines the join count between tables with a restriction. A filter uses the table, field, operator, and value attributes to set the restriction. The operator is limited to the XML code operators.

The structure is organized in the same manner as a Simple Join with an added <FILTER> element. The following code illustrates the structure:

```
<MULTIJOIN>
<JOIN>
<LEFT table="table name" column="column" />
<RIGHT table="table name" column="column" />
</JOIN>
<!-- Optional. May contain one or more filters. -->
<FILTER table="table name" column="column" operator=
"operator" value="filter value" />
</MULTIJOIN>
```

The <FILTER> element is optional in the join analysis. Multiple filters can be applied to a join. The AND operator is appended to each filter condition upon creation of the query. The following XML code illustrates the use of a filter with a simple join analysis:

```
<ANALYSIS>
<JOINS>
<MULTIJOIN>
<JOIN>
<LEFT table="ACCT" column="ACCT_INTRL_ID" />
<RIGHT table="CUST_ACCT" column="ACCT_INTRL_ID" />
</JOIN>
<FILTER table="ACCT" column="DATA_DUMP_DT"
operator="GTE" value="01-NOV-2006" />
<FILTER table="ACCT" column="DATA_DUMP_DT"
operator="LTE" value="05-NOV-2006" />
</MULTIJOIN>
</JOINS>
</ANALYSIS>
```

This code executes the following query:

```
select count(1) from ACCT a, CUST_ACCT b
where a.ACCT_INTRL_ID=b.ACCT_INTRL_ID
and a.DATA_DUMP_DT>='01-NOV-2006' and a.DATA_DUMP_DT<='05-NOV-2006'
```

To filter for values that are null or not null, set the operator to EMPTY and the value to IS NULL or IS NOT NULL, respectively.

Join Count by Distinct Column

To determine a join count of the number of distinct values for a specified column within the joined tables, include the <DISTINCT_COUNT> element as content to the <MULTIJOIN> element. The targeted table and its column are set to the table and column attributes,

respectively. The following sample demonstrates integration of the <DISTINCT_COUNT> element in the analysis:

```
<MULTIJOIN>
<JOIN>
<LEFT table="table name" column="column" />
<RIGHT table="table name" column="column" />
</JOIN>
<!-- Optional. Can only have one DISTINCT_COUNT within
the MULTIJOIN element. -->
<DISTINCT_COUNT table="table name" column="column" />
</MULTIJOIN>
```

> **Note:**
>
> The <DISTINCT_COUNT> element is optional in the join analysis

The following XML sample code illustrates use of the <DISTINCT_COUNT> element:

```
<ANALYSIS>
<JOINS>
<MULTIJOIN>
<JOIN>
<LEFT table="ACCT" column="ACCT_INTRL_ID" />
<RIGHT table="CUST_ACCT" column="ACCT_INTRL_ID" />
</JOIN>
<FILTER table="ACCT" column="DATA_DUMP_DT" operator=
"EQ" value="02-NOV-2006" />
<DISTINCT_COUNT table="ACCT" column="ACCT_TYPE_CD" />
</MULTIJOIN>
</JOINS>
</ANALYSIS>
```

This sample code executes the following query:

```
select count(DISTINCT a.ACCT_TYPE_CD)
from ACCT a, CUST_ACCT b
where a.ACCT_INTRL_ID=b.ACCT_INTRL_ID and a.DATA_DUMP_DT='02-NOV-2006'
```

8.1.1.5 Other Queries

The Data Analysis Tool also supports providing SQL queries directly in the analysis XML file.

A query has two components: the query title and the query itself. As queries often contain characters that are “reserved” in XML, you should follow the example below for “escaping” the SQL to ensure that it does not become corrupted.

```
<QUERIES>
<SQLQUERY title="title">
select col1, col2 from some_table
where some_condition
```

```
</SQLQUERY>
</QUERIES>
```

The following XML sample code illustrates use of the <QUERIES> element:

```
<ANALYSIS>
<QUERIES>
<SQLQUERY title="FO Transaction Roles"><![CDATA[ select
FOT.mantas_PRODUCT_TYPE_CD,
FOTPS.PARTY_ROLE_CD, count(1) as RoleCt
from FO_TRXN_STAGE FOT, FO_TRXN_PARTY_STAGE FOTPS
where FOT.TRXN_INTRL_ID = FOTPS.TRXN_INTRL_ID
```

This code runs the query in the <SQLQUERY> element and writes the results to the output file. For SQL queries, the results are always in HTML. Your code can contain any number of <SQLQUERY> elements. The system runs each query in sequence after the other components of analysis are complete.

SQLQUERY Element Rules

Several cautions and notes are specific to the <SQLQUERY> element:

- If your query contains characters that XML standards reserve, such as > or <, you must place your query within a CDATA block.
- Verify that no white space exists between the SQL query opening tag and the CDATA tags , such as `<![CDATA[ ...)` and the closing tag , such as `...]]>`.
- Processing extracts column headers in the output from the SQL query itself. When performing calculations in return columns, it is best to alias the return columns for output
- Line breaks and comments in the SQL are acceptable, but you should use `/* */` style comments in lieu of single-line comments for safety.
- The tool does not perform any schema-name substitution. Therefore, verify that any schema names match the database contents. The database user , such as ATOMIC, has aliases for most tables you may must analyze. Thus, running the tool as ATOMIC should prevent you from needing schema names in queries.

8.1.2 Using the Data Analysis Tool

After editing the configuration files, you can run the Data Analysis Tool as a foreground or background process.

The following table lists the XML input files delivered for use with the Data Analysis Tool.

Table 8-3 Data Analysis Tool XML Input Files

File	Description
analysis_aml.xml	Analysis configuration specific for data required by Anti-Money Laundering scenarios and Ingestion Manager operations to support them.
analysis_aml_ui.xm l	Analysis configuration specific for data displayed in support of Anti-Money Laundering scenarios.

Table 8-3 (Cont.) Data Analysis Tool XML Input Files

File	Description
analysis_iaml.xml	Analysis configuration specific for data required by Institutional Anti-Money Laundering scenarios and Ingestion Manager operations to support them.
analysis_iaml_ui.x ml	Analysis configuration specific for data displayed in support of Institutional Anti-Money Laundering scenarios.

You can also create your own files using the provided files as a template. Place files that you create in the mantas_cfg directory that the DTD can locate. If you place your files in a different directory, you must modify the DTD reference in the XML files to qualify the path to the DTD.

8.1.2.1 Running the Data Analysis Tool

To run the Data Analysis Tool, follow these steps:

1. Navigate to the `<OFSAAI Installed Directory>/database/db_tools/bin` directory.
2. Execute the following command:

```
run_data_analysis_tool.sh [bg] [-i input_file.xml] [-o outputfile]
```

Table 8-4 Command Line Arguments

Argument	Explanation
bg	If provided, runs the tool in the background. You can then disconnect your Unix or Linux session without interrupting the tool's operation. The system directs any output from the screen to the nohup.out file in the directory from which you ran the tool.
-i input_file	Uses an input analysis file other than the one that install.cfg specifies. Omission of this argument causes the Data Analysis Tool to use the default file in install.cfg.
-o output_file	Writes the output to a file other than the one that install.cfg specifies. Omission of this argument causes the Data Analysis Tool to use the default file in install.cfg.

8.1.3 Logs

The Data Analysis Tool writes status and error messages to the configured log file.

The default location for this log file is:

```
<OFSAAI Installed Directory>/database/db_tools/logs/data_analysis_tool.log
```

The system writes any system-type errors that prevent the tool from connecting to or operating this log file. It also writes data errors to the log and includes them in the data analysis report output. The following section describes the report output.

Understanding the Data Analysis Report

The tool generates a data analysis report, which resides in the location you specified in the install.cfg file or with the command line -o argument.

> **Note:**
>
> Oracle recommends that you view the output report using Microsoft Excel because this HTML file has specific HTML formatting for Excel.

The following table describes sections of the output report.

Table 8-5 Data Analysis Report Output

Section	Description
Table Count Summary	Contains the row count of each table in the configured database excluding the KDD, archive, and temp tables.
Field Distribution Summary Table	Groups by table the unique values for the identified fields and number of times each value occurs in the table. This summary table appears only in the report if the analysis for Distinct Values for Fields of Interest and Its Count was configured in the XML file. In addition, quotes enclose any values with padded spaces to identify spaces in the value.
Null Summary Count Table	Groups by table the number of nulls present and values with padded spaces for the identified fields in each table. This summary table only appears in the report if the analysis for Null and Padded Space Count has been configured in the XML file.
Referential Integrity Table Summary	Displays the join analysis, the number of rows returned between the joined tables, and the table count for each table being joined. This summary only appears in the report if the analysis for Join Counts has been configured in the XML file.
Query Results	Displays the results of queries specified in the QUERIES section of the analysis file.
SQL Report	Lists all of the SQL run to produce the other sections of the report.
Error Report	Displays any errors that occurred when any of the queries were performed.

8.1.4 Troubleshooting the Data Analysis Tool

This topic lists common Data Analysis Tool errors and their solutions.

Table 8-6 Troubleshooting Data Analysis Tool Errors

Error Message	Cause	Solution
java.io. FileNotFoundException <path & filename>	The system cannot find the file specified.	Verify the install.cfgfile indicates the correct path.

Table 8-6 (Cont.) Troubleshooting Data Analysis Tool Errors

Error Message	Cause	Solution
java.lang. RuntimeException: Tables <table 1> and <table2>	Tables <table 1> and <table 2> are already joined in this fashion.	In the analysis.xml file, remove duplicate join contents in the <JOIN> element.

8.2 Managing Get Dataset Query with Thresholds Utility

Processing uses the Get Dataset Query with Thresholds Utility to store a dataset query in the Behavior Detection database with the threshold names and not with the threshold values.

When the Behavior Detection engine executes a scenario, it substitutes the correct threshold values in the SQL query before submitting it to the database. Tracking of the query that executes in the database occurs only through the Behavior Detection engine log file when it runs in trace mode.

This section covers the following topics:

- Using the Get Dataset Query With Thresholds Utility
- Executing the Get Dataset Query with Thresholds Utility

8.2.1 Using the Get Dataset Query With Thresholds Utility

Processing extracts the dataset query and uses it as input for tuning and execution plan generation.

> **Note:**
>
> This utility does not recursively substitute thresholds in child datasets. Therefore, if a dataset being extracted has a reference to another dataset, manual extraction of that dataset must also occur.

The following table describes the parameters to provide with the get_dataset_query.sh script:

Table 8-7 Get Dataset Query Variables

Parameter	Description
Dataset ID	Unique identifier of the dataset for retrieval.
Threshold Set ID	Unique identifier of the threshold set for retrieval.

8.2.2 Executing the Get Dataset Query with Thresholds Utility

To execute the Get Dataset Query with Thresholds Utility, follow these steps.

- After the Alert Creator process completes, execute the get_dataset_query.sh script as follows:

```
<OFSAAI Installed Directory>/database/db_tools/bin/get_dataset_query.sh
<Dataset ID> <Threshold Set ID>
```

The dataset query automatically prints to standard output, which you can copy and paste into any other application.

When the dataset query does not find a dataset, output is:

```
Error: Dataset not found.
```

When the dataset query does not find a threshold set, output is:

```
Error: Threshold Set not found.
```

Optional: Redirect the output into a text file as follows:

```
<OFSAAI Installed Directory>/database/db_tools/bin/get_dataset_query.sh
<Dataset ID> <Threshold Set ID> query.sql
```

8.3 Managing Trusted Pairs and Alert Suppression

OFS BD API offers services to allow OFS BD to consume trusted pair and Alert Suppression information present in the AML Case Management System.

Managing Trusted Pairs

Trusted Pair is the concept of reducing the number of false positives events by identifying transactions between parties viewed as having a trusted relationship. The Trusted Pair API will allow full or filtered Trusted Pair data to be loaded to OFS BD based on inputs provided.

This service allows Oracle Financial Services Behavior Detection (OFS BD) to consume trusted pair information present within AML Case Management System. For more information about this service, see *Oracle Financial Services Behavior Detection API Services Guide*. You can also manage trusted pairs through ingestion. For more information, see *Trusted Pair*.

8.4 Managing Scenario Migration Utility

Use the Scenario Migration Utility to migrate scenarios, datasets, networks, and associated metadata from the development environment to the production environment.

To provide a list of scenarios, datasets, or networks, edit the *scnros.cfg*, *dataset.cfg*, or the *network.cfg* files prior to scenario extraction or loading. The Scenario Migration Utility creates and migrates the following metadata files:

- Scenarios: The `<scenario catalog identifier>.<scenario id>.xml` file contains scenario metadata for core Behavior Detection tables. It also may contain scenario metadata for optional tables.
- Datasets: The `<dataset idDS>.xml` file contains dataset metadata for core Behavior Detection tables.
- Networks: The `<network>NW.xml` file contains network metadata for core Behavior Detection tables.

> **Note:**
>
> When the Scenario Migration Utility extracts these files, you can version-control them or store them in the Oracle client's archival system.

To help avoid accidental loading of a scenario into the incorrect environment, the Scenario Migration utility enables you to name your source and target environments. On extract, you can specify the environment name to which you plan to load the scenario. If you attempt to load it to a different environment, the system displays a warning prompt.

Logs

The Scenario Migration Utility produces two log files, *load.log* and *extract.log*. These files reside in the following location:

```
<OFSAAI Installed Directory>/database/db_tools/logs
```

8.4.1 Configuring the Scenario Migration Utility

To configure the Scenario Migration Utility, navigate to `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg`. The *install.cfg* file contains common configuration information that Scenario Migration and other utilities require for processing.

Sample install.cfg File for Scenario Migration

This section provides sample information from the install.cfg file that is specific to this utility.

```
################ SCENARIO MIGRATION CONFIGURATION ######################
#### GENERAL SCENARIO MIGRATION SETTINGS

#Specify the flags for whether scoring rules and wrapper datasets must be
extracted or loaded
score.include=N
wrapper.include=N

#Specify the Use Code for the scenario. Possible values are 'BRK' or 'EXP'
load.scnro.use=BRK

#If custom patterns exist for a product scenario, set to 'Y' when loading a
scenario hotfix.
#This should normally be set to 'N'.
load.ignore.custom.patterns=N

#Specify the full path of depfile and name of fixfile used for extraction and
loading
#Note : fixfile need not be specified in case of loading
sm.depfile=/scratch/ofsaaapp/OFSBD 8.0.2/OFSBD 8.0.2_B06/BDP62_B06/database/
db_tools/mantas_cfg/dep.cfg

sm.release=5.7.1

#### EXTRACT

# Specify the database details for extraction
```

```
extract.database.username=${utils.database.username}
extract.database.password=${utils.database.password}

# Specify the case schema name for both extraction and load .
caseschema.schema.owner=ATOMIC

# Specify the jdbc driver details for connecting to the source database
extract.conn.driver=${database.driverName}
extract.conn.url=jdbc:oracle:thin:@ofss220074.in.oracle.com:1521:Ti1O11L56
#Source System Id
extract.system.id=
# Specify the schema names for Extract
extract.schema.mantas=${schema.mantas.owner}
extract.schema.case=ATOMIC
extract.schema.business=${schema.business.owner}
extract.schema.market=${schema.market.owner}
extract.user.miner=${load.user.miner}
extract.miner.password=${utils.miner.password}

# File Paths for Extract

#Specify the full path in which to place extracted scenarios
extract.dirname=/scratch/ofsaaapp/OFSBD 8.0.2/OFSBD 8.0.2_B06/BDP62_B06/
database/db_tools/data

#Specify the full path of the directory where the backups for the extracted
scripts would be maintained
extract.backup.dir=/scratch/ofsaaapp/OFSBD 8.0.2/OFSBD 8.0.2_B06/BDP62_B06/
database/db_tools/data/temp

#Controls whether jobs and thresholds are constrained to IDs in the product
range (product.id.range.min
# through product.id.range.max). Values are Y and N. If the range is not
restricted, you can use range.check
# to fail the extract if there are values outside the product range.
extract.product.range.only=N
extract.product.range.check=N

#### LOAD

# Specify the jdbc driver details for connecting to the target database
load.conn.driver=${database.driverName}
load.conn.url=${utils.database.urlName}

#Target System ID
load.system.id=Ti1O11L56

# Specify the schema names for Load
load.schema.mantas=${schema.mantas.owner}
load.schema.case=ATOMIC
load.schema.business=${schema.business.owner}
load.schema.market=${schema.market.owner}
load.user.miner=${utils.miner.user}
load.miner.password=${utils.miner.password}

#Directory where scenario migration files reside for loading
```

```
load.dirname=/scratch/ofsaaapp/OFSBD 8.0.2/OFSBD 8.0.2_B06/BDP62_B06/database/
db_tools/data

# Specify whether threshold can be updated
load.threshold.update=Y

# Specify whether or not to verify the target environment on load
verify.target.system=N
```

> **Note:**
>
> In the install.cfg file, entries are in the form Property1=${Property2}. That is, the value for Property1 is the value that processing assigns to Property2. As such, if you change Property2's value, Property1's value also changes.

Configuring the Environment

To configure the environment for scenario migration, modify the parameters that the sample `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` shows. The tables in the following sections describe the parameters specific to the Scenario Migration Utility.

8.4.1.1 Configuring General Scenario Migration

This topic describes general scenario migration parameters.

Table 8-8 General Scenario Migration Parameters

Parameter	Description
score.include	Flag that indicates whether scenario migration includes scenario scoring metadata; value is "Y" or "N" (the default).
wrapper.include	Flag that indicates whether scenario migration includes wrapper metadata; value is "Y" or "N" (the default).
sm.depfile	Location of the scenario migration dependencies file, <OFSAAI Installed Directory>/ database/ db_tools/mantas_cfg/dep.cfg.
sm.release	Version of the Scenario Migration Utility.

> **Note:**
>
> **Caution:** Oracle strongly recommends that you maintain scores and threshold values in a single environment. Maintaining these attributes in multiple environments and migrating the scenarios between the environments can cause the loss of threshold set-specific scoring rules.

8.4.1.2 Configuring Scenario Extraction

This topic describes scenario extraction parameters.

Table 8-9 Scenario Extraction Parameters

Parameter	Description
extract.database.username	User used to connect to the database when extracting scenarios (ATOMIC).
extract.database.password	Password for the above user.
extract.conn.driver	Database connection driver that the utility is to use (oracle.jdbc.driver.OracleDriver).
extract.conn.url	Database connection string that the Scenario Migration Utility is to use.
extract.system.id	System from which the scenario was extracted.
extract.schema.mantas	ATOMIC schema owner in the database into which extraction of the scenarios occurs (ATOMIC).
extract.schema.business	ATOMIC schema owner in the database into which extraction of the scenarios occurs (ATOMIC).
extract.schema.market	ATOMIC schema owner in the database into which extraction of the scenarios occurs (ATOMIC).
extract.user.miner	ATOMIC schema owner in the database into which extraction of the scenarios occurs (ATOMIC).
extract.miner.password	Password for the above user.
extract.dirname	Full path to the target directory where the utility writes extracted metadata (<OFSAAI Installed Directory>/database/ db_tools/data).
extract.backup.dir	Full path to the target directory where the utility writes backups of the extracted metadata (<OFSAAI Installed Directory>/ database/db_tools/ data/temp).
extract.product.range.only	Indicator (Y or N) of whether to extract custom patterns, jobs, thresholds, threshold sets, and scoring rules when extracting a scenario. Set to Y to prevent extraction of these entities.
extract.product.range.check	(For internal use only.) Indicator (Y or N) of whether to fail the extraction of a scenario if any metadata has sequence IDs outside the product range. Set to Y to fail the extraction.

8.4.1.3 Configuring Scenario Load

This topic describes scenario load parameters.

Table 8-10 Scenario Load Parameters

Parameter	Description
load.conn.driver	Database connection driver that the utility is to use (oracle.jdbc.driver.OracleDriver).
load.conn.url	Database connection string that the Scenario Migration Utility is to use.

Table 8-10 (Cont.) Scenario Load Parameters

Parameter	Description
load.ignore.custom.patterns=N	When set to *N*, custom patterns will not be ignored. This mode should be used when migrating scenarios between environments within the client's environment. If a custom pattern is not in the loaded XML file, then it will be deactivated. When set to *Y*, any custom patterns will be ignored by the load process, and should continue to operate.
load.schema.mantas	ATOMIC schema owner in the database in which loading of the scenario occurs (ATOMIC).
load.schema.business	ATOMIC schema owner in the database in which loading of the scenario occurs (ATOMIC).
load.schema.market	ATOMIC schema owner in the database in which loading of the scenario occurs (ATOMIC).
load.user.miner	ATOMIC schema owner in the database in which loading of the scenario occurs (ATOMIC).
load.miner.password	Password for the above user.
load.threshold.update	Threshold values from the incoming scenario. • Selecting *N* retains the threshold values from the target environment. • Selecting *Y* updates thresholds in the target environment to values from the incoming file.
load.system.id	Name that is assigned to the system into which this instance of Scenario Migration loads metadata. The system compares the value for this setting to the target system in the metadata file.
load.dirname	Directory from which the system loads scenario, network, and dataset XML files.
verify.target.system	Check target name upon loading metadata files. • Setting to *N* prevents Scenario Migration from checking the load.system.id against the target system specified whenthe scenario, network or dataset was extracted. • Setting to *Y* enables this check. If the target in theXML file does not match the setting for load.system.id or the target is present in XML file but the load.system.id is blank then the system prompts you for an appropriate action. You can then continue with load or abandon the load, and you can apply the same answer to all other files in the session of Scenario Migration or allow the utility to continue prompting on each XML file that has a mismatch.

8.4.2 Extracting Scenario Metadata

Scenario metadata includes XML files that contain the table data for scenario, dataset, and network logic.

The sm_extract.sh script invokes a Java tool, which creates these files. You start this script as follows:

```
sm_extract.sh <mode> -notarget | -target <name>
```

where:

- mode (mandatory) is the scenario, network, or dataset.
- -notarget, if included, implies that the system does not save the target environment to the generated XML files.
- -target <name> identifies the same target (in <name>) for all extracted XML files.

If you do not specify -notarget or -target <name> on the command line, the system prompts you to supply a target environment on each extracted file.

To extract scenario, dataset, and network metadata, follow these steps:

1. Navigate to the `cd <OFSAAI Installed Directory>/db_tools` directory.
2. Edit the metadata configuration files with identifying information for the scenarios, datasets, or networks for extraction:
   - <scnro_ctlg_id> in the scnros.cfg file and/or
   - <scnro_ctlg_id>.<scnro_id> in the scnros.cfg file

     > **Note:**
     >
     > Providing both <scnro_ctlg_id> and <scnro_id> in the scnros.cfg file allows finer granularity when extracting scenarios. If you provide both a scenario catalog ID and a scenario ID on a line, you must separate them with a period.

   - <data_set_id> in the dataset.cfg file
   - <network_id> in the network.cfg file
3. Execute the sm_extract.sh script in this order:
   a. Enter sm_extract.sh dataset to extract dataset metadata.
   b. Enter sm_extract.sh scenario to extract scenario metadata.
   c. Enter sm_extract.sh network to extract network metadata.

8.4.3 Loading Scenario Metadata

The sm_load.sh script loads translated XML table data files into the target database.

> **Note:**
>
> To avoid corrupting the Behavior Detection process, never load scenarios while the process is running.

To load scenario, dataset, and network metadata, follow these steps:

1. Navigate to the following directory: `cd <OFSAAI Installed Directory>/db_tools`.
2. **Optional:** Edit the metadata configuration files (that is, scnros.cfg, dataset.cfg, and network.cfg) with identifying information for the scenarios, datasets, or networks that you want to load:
   - <scnro_ctlg_id> in the scnros.cfg file and/or
   - <scnro_id> in the scnros.cfg file

     > **Note:**
     >
     > Providing both <scnro_ctlg_id> and <scnro_id> in the scnros.cfg file allows finer granularity when loading scenarios. You must separate values with a period per line.

   - <data_set_id> in the dataset.cfg file
   - <network_id> in the network.cfg file
3. Copy the XML files you plan to load into the directory that the load.dirname specifies in the install.cfg file.
   - (Optional) <Enter one of the user's choices while performing this step.>
   - (Optional) <Enter another of the user's choices while performing this step.>
4. Execute the sm_load.sh script:
   a. Enter sm_load.sh dataset to load dataset metadata.
   b. Enter sm_load.sh scenario to load scenario metadata.
   c. .Enter sm_load.sh network to load network metadata.

8.4.4 Scenario Migration Best Practices

Migrating scenarios from one environment to another requires a unified process in order to prevent conflicts and errors. This section describes the recommended best practices for scenario migration for any existing OFSBD system.

> **Note:**
>
> **Caution:** Not following the recommended best practices while loading scenarios to the targeted system may cause one or more sequence ID conflicts to occur, and your scenario will not be loaded. Once a conflict occurs, the metadata in the target environment must be corrected before the scenario can be successfully loaded.

To execute the recommended best practices, you should have an intermediate level knowledge of the scenario metadata, and be familiar with scenario patterns, thresholds, threshold sets, and so on. Basic SQL are required, as well as access privileges to the ATOMIC schema. You must also be able to update records through SQLPLUS or a similar DB utility.

Process Overview

Scenario metadata is stored in many tables, with each table using a unique sequence ID for each of its records. If scenarios, thresholds, and scoring rules are modified in multiple environments using the same sequence ID range, then conflicts may occur when you migrate scenarios to these environments. To prevent conflict, you must set different sequence ID ranges in each of the environments.

The recommended best practices contain two basic points:

- Make changes in only one environment
- Separate the sequence ID ranges

Best Practices

Prepare to implement the recommended best practices before installing OFSBD. Once the application is installed you should execute these steps to avoid scenario migration problems.

Making changes in Only One Environment

1. Only make changes to scenarios, thresholds, threshold sets, and scoring rules in the source environment.
2. Test and confirm your changes in the source environment.
3. Extract scenarios from the source environment and migrate them to all of your target environments.

Separating Sequence ID Ranges

Conflicting sequence IDs are often the cause errors when you migrate a scenario, so it is important to separate the sequence ID range.

1. Review the ATOMIC.KDD_COUNTER table, which contains all sequence ID ranges and current values.
2. Start your sequence ID ranger at 10,000,000 and separate each environment by 10,000,000. The OFSBD product sequence ID range is >100,000,000.

Sequences to Modify

You should set these sequences before doing any work on scenarios, thresholds, or scoring rules. The following table lists sequences involved and sample values for the Development environment.

Table 8-11 Environment 1 (Development)

TABLE_NM	SEQUENCE_NAME	CURRENT_VALUE	MIN_VALUE	MAX_VALUE
KDD_ATTR	ATTR_ID_SEQUENCE	10000000	10000000	19999999
KDD_AUGMENTATION	AGMNT_INSTN_ID_SEQ	10000000	10000000	19999999
KDD_DATASET	DATASET_ID_SEQUENC E	10000000	10000000	19999999
KDD_JOB	JOB_ID_SEQ	200000000	10000000	19999999
KDD_LINK_ANLYS_NTWRK_ DEFN	NTWRK_DEFN_ID_SEQ	10000000	10000000	19999999
KDD_LINK_ANLYS_TYPE_C D	TYPE_ID_SEQ	10000000	10000000	19999999
KDD_NTWRK	NTWRK_ID_SEQ	10000000	10000000	19999999
KDD_PARAM_SET	PARAM_SET_ID_SEQ	200000000	10000000	19999999
KDD_PTTRN	PTTRN_ID_SEQ	10000000	10000000	19999999
KDD_RULE	RULE_ID_SEQ	10000000	10000000	19999999
KDD_SCNRO	SCNRO_ID_SEQ	10000000	10000000	19999999
KDD_SCORE	SCORE_ID_SEQ	10000000	10000000	19999999
KDD_SCORE_HIST	SCORE_HIST_SEQ_ID_ SEQ	10000000	10000000	19999999
KDD_TSHLD	TSHLD_ID_SEQ	10000000	10000000	19999999
KDD_TSHLD_HIST	HIST_SEQ_ID_SEQ	10000000	10000000	19999999
KDD_TSHLD_SET	TSHLD_SET_ID_SEQ	10000000	10000000	19999999

The following table lists sequences involved and sample values for the Test/UAT environment.

Table 8-12 Environment 2 (Test/UAT)

TABLE_NM	SEQUENCE_NAME	CURRENT_VALUE	MIN_VALUE	MAX_VALUE
KDD_ATTR	ATTR_ID_SEQUENCE	20000000	20000000	29999999
KDD_AUGMENTATION	AGMNT_INSTN_ID_SEQ	20000000	20000000	29999999
KDD_DATASET	DATASET_ID_SEQUENC E	20000000	20000000	29999999
KDD_JOB	JOB_ID_SEQ	20000000	20000000	29999999
KDD_LINK_ANLYS_NTWRK_ DEFN	NTWRK_DEFN_ID_SEQ	20000000	20000000	29999999
KDD_LINK_ANLYS_TYPE_C D	TYPE_ID_SEQ	20000000	20000000	29999999
KDD_NTWRK	NTWRK_ID_SEQ	20000000	20000000	29999999
KDD_PARAM_SET	PARAM_SET_ID_SEQ	20000000	20000000	29999999

Table 8-12 (Cont.) Environment 2 (Test/UAT)

TABLE_NM	SEQUENCE_NAME	CURRENT_VALUE	MIN_VALUE	MAX_VALUE
KDD_PTTRN	PTTRN_ID_SEQ	20000000	20000000	29999999
KDD_RULE	RULE_ID_SEQ	20000000	20000000	29999999
KDD_SCNRO	SCNRO_ID_SEQ	20000000	20000000	29999999
KDD_SCORE	SCORE_ID_SEQ	20000000	20000000	29999999
KDD_SCORE_HIST	SCORE_HIST_SEQ_ID_ SEQ	20000000	20000000	29999999
KDD_TSHLD	TSHLD_ID_SEQ	20000000	20000000	29999999
KDD_TSHLD_HIST	HIST_SEQ_ID_SEQ	20000000	20000000	29999999
KDD_TSHLD_SET	TSHLD_SET_ID_SEQ	20000000	20000000	29999999

The following table lists sequences involved and sample values for the Production environment.

Table 8-13 Environment 3 (PROD)

TABLE_NM	SEQUENCE_NAME	CURRENT_VALUE	MIN_VALUE	MAX_VALUE
KDD_ATTR	ATTR_ID_SEQUENCE	30000000	30000000	39999999
KDD_AUGMENTATION	AGMNT_INSTN_ID_SEQ	30000000	30000000	39999999
KDD_DATASET	DATASET_ID_SEQUENCE	30000000	30000000	39999999
KDD_JOB	JOB_ID_SEQ	30000000	30000000	39999999
KDD_LINK_ANLYS_NTWRK_ DEFN	NTWRK_DEFN_ID_SEQ	30000000	30000000	39999999
KDD_LINK_ANLYS_TYPE_C D	TYPE_ID_SEQ	30000000	30000000	39999999
KDD_NTWRK	NTWRK_ID_SEQ	20000000	20000000	29999999
KDD_PARAM_SET	PARAM_SET_ID_SEQ	30000000	30000000	39999999
KDD_PTTRN	PTTRN_ID_SEQ	30000000	30000000	39999999
KDD_RULE	RULE_ID_SEQ	30000000	30000000	39999999
KDD_SCNRO	SCNRO_ID_SEQ	30000000	30000000	39999999
KDD_SCORE	SCORE_ID_SEQ	30000000	30000000	39999999
KDD_SCORE_HIST	SCORE_HIST_SEQ_ID_S EQ	30000000	30000000	39999999
KDD_TSHLD	TSHLD_ID_SEQ	30000000	30000000	39999999
KDD_TSHLD_HIST	HIST_SEQ_ID_SEQ	30000000	30000000	39999999
KDD_TSHLD_SET	TSHLD_SET_ID_SEQ	30000000	30000000	39999999

In order to update your database tables with recommended values, use SQLPLUS or a similar tool.

```
UPDATE KDD_COUNTER
set min_value = 10000000,
max_value = 19999999,
current_value = 10000000
where sequence_name in
('DATASET_ID_SEQUENCE',
'ATTR_ID_SEQUENCE',
'PARAM_SET_ID_SEQ',
'PTTRN_ID_SEQ',
'RULE_ID_SEQ',
'SCNRO_ID_SEQ',
'JOB_ID_SEQ',
'TSHLD_ID_SEQ',
'NTWRK_DEFN_ID_SEQ',
'TYPE_ID_SEQ',
'TAB_ID_SEQ',
'TSHLD_SET_ID_SEQ',
'HIST_SEQ_ID_SEQ',
'AGMNT_INSTN_ID_SEQ',
'SCORE_ID_SEQ',
'SCORE_HIST_SEQ_ID_SEQ');
Commit;
```

8.5 Investigation Management Configuration Migration Utility

Use the Investigation Management Configuration Migration Utility to migrate Alert Viewer configuration metadata between environments. This utility provides a means to load alert and case configuration metadata into OFSBD as well as allows you to move configuration metadata between installations of OFSBD.

Configuration metadata is considered to be that metadata associated with the alert workflow, such as actions, action categories, and standard comments. The migration process handles ONLY database metadata and is executed using two separate procedures—extraction and loading. The extraction process pulls metadata from an environment into a file that can be can be moved, configuration controlled, and loaded into another environment. The load process loads these extracted files into the target environment.

To avoid accidental loading of Investigation Metadata into the incorrect environment, the Investigation Management Configuration Migration Utility enables you to name your source and target environments. On extract, you can specify the environment name to which you plan to load the Investigation Metadata. If you attempt to load it to a different environment, the system displays a warning prompt.

> **Note:**
>
> Because not all configuration metadata lies within the database it may be necessary to manually copy over XML files associated with configuration. This manual process is not handled by the Investigation Management Configuration Migration Utility. Any customized XML file pertaining to configuration will must be manually migrated.

Logs

The Investigation Management Configuration Migration Utility produces two log files—load.log and extract.log. These files reside at the following location:

```
<OFSAAI Installed Directory>/database/db_tools/logs
```

8.5.1 Configuring the Investment Configuration Metadata Migration Utility

The `<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/install.cfg` file contains common configuration information that Investment Configuration Metadata Migration Utility and other utilities require for processing.

The following example provides sample information from the install.cfg file that is specific to this utility. This utility migrates data for the following tables:

- KDD_ACTIVITY_TYPE_CD
- KDD_ACTVY_TYPE_REVIEW_STATUS
- KDD_SCNRO_CLASS_ACTVY_TYPE
- KDD_ACTVY_TYPE_RSTRN
- KDD_ACTVY_CAT_CD
- KDD_CMMNT
- KDD_SCNRO_CLASS_CMMNT
- KDD_CMMNT_CAT_CD
- KDD_REVIEW_STATUS
- KDD_ACTIVITY_RESULT_STATUS
- KDD_EXTRL_REF_SRC
- KDD_FOCUS_ALERT_ASGMT
- KDD_AUTO_CLOSE_ALERT
- KDD_BUS_DMN
- KDD_JRSDCN
- KDD_COUNTER
- KDD_CAL_HOLIDAY
- KDD_CAL_WKLY_OFF
- KDD_REPORT_TEMPLATE
- KDD_REPORT_TEMPLATE_PARAM
- KDD_REPORT_DEFN
- KDD_REPORT_DEFN_PARAM
- KDD_REPORT_TEMPLATE_JRSDCN
- KDD_AVERTED_LOSS_TYPE
- KDD_REG_REPORT_TYPE

- KDD_REG_REPORT_STATUS

```
#### EXTRACT (These properties are shared by IMCM with the Scenario Migration
      Utility)
# Specify the database
      details for extraction extract.database.username=$
{utils.database.username}
      extract.database.password=${utils.database.password

# Specify the jdbc driver details for connecting to the source database
      extract.conn.driver=${database.driverName
extract.conn.url= jdbc:oracle:oci:@T2O9S8 #Source System Id
extract.system.id= TEST_ENVIORNMENT # File Paths for Extract
#Specify the full path in which to place extracted Correlation Rules
      extract.dirname=/users/oriont/Mantas5.8/database/db_tools/data
#### LOAD (These properties
      are shared by IMCM Utility with the Scenario Migration Utility) #Target
System
      ID
load.system.id=
      PROD_ENVIRONMENT
# Specify whether or not to verify the target environment on load
```

> **Note:**
>
> This script is part of the Database Tools that reside in the `<OFSAAI Installed Directory>/database/db_tools/bin` directory.

8.5.1.1 Configuring the Environment

To configure the environment for Investigation Metadata Migration, modify the parameters that the sample install.cfg file shows.

The tables in the following sections describe the parameters specific to the Investigation Management Configuration Migration Utility.

Configuring General Investigation Metadata Migration

The following table describes the General Investigation Metadata migration parameters.

Table 8-14 General Investigation Metadata Migration Parameters

Parameter	Description
config.filenm.prefix	Prefix used by the utility for naming the extracted file,

Configuring Investigation Metadata Extraction

The following table describes Investigation Metadata extraction parameters.

Table 8-15 Investigation Metadata Extraction Parameters.

Parameter	Description
extract.database.username	User to connect to the database when extracting Investigation Metadata (DB_UTIL_USER)
extract.database.password	Password for the above user.
extract.conn.driver	Database connection driver that the utility is to use (oracle.jdbc.driver.OracleDriver).
extract.conn.url	Database connection string that the Investigation Metadata Migration Utility is to use.
extract.system.id	System from which the Investigation Metadata was extracted.
extract.dirname	Full path to the target directory where the utility writes extracted metadata ($FIC_WEB_HOME/ database/ db_tools/data).

Configuring Alert Viewer Metadata Load

Table 8-16 Investigation Metadata Load Parameters.

Parameter	Description
utils.database.username	User to connect to the database when loading Investigation Metadata (DB_UTIL_USER).
utils.database.password	Password for the above user.
database.driverName	Database connection driver that the utility is to use (oracle.jdbc.driver.OracleDriver).
utils.database.urlName	Database connection string that the Investigation Metadata Migration Utility is to use.
load.system.id	Namethat is assigned to the system into which this instance of Investigation Metadata Migration loads metadata. The system compares the value for this setting to the target system in the metadata file.
verify.target.system	Check target name upon loading metadata files. • Setting to N prevents Investigation Metadata Migration from checking the load.system.id against the target system specified when the Investigation Metadata was extracted. • Setting to Y enables this check. If the target in the XML file does not match the setting for load.system.id or the target is present in XML file but the load.system.id is blank then the system prompts you for an appropriate action. You can then continue with load or abandon the load, and you can apply the same answer to all other files in the session of Investigation Metadata Migration or allow the utility to continue prompting on each XML file that has a mismatch.

8.5.2 Extracting Investigation Metadata

Investigation metadata includes XML files that contain the table data for the Alert/Case Investigation.

The sm_extract.sh script invokes a Java tool, which creates these files. You start the script as follows:

```
sm_extract.sh investconfig
```

To extract Alert/Case Investigation metadata, execute the *sm_extract.sh* file.

8.5.3 Loading Alert Viewer Metadata

The sm_load.sh script loads translated XML table data files into the target database.

To load the Alert Viewer metadata, execute the *sm_load.sh* file as follows:

```
sm_load.sh investconfig
```

8.6 Managing Watch List Service

Watch list web service enables you to query the Behavior Detection Watch List tables to determine if a given name (or a name closely matching the given name) is on a watch list.

Refer to the Services Guide, for more details on how the service can be called and the results that are returned.

8.7 Configure Password Changes

8.7.1 Modify OFSAA Infrastructure Config Schema Password

To change the Config Schema password, perform the following steps:

1. Change the Config schema User Password in the database.
2. Delete the `$FIC_HOME/conf/Reveleus.SEC` file.
3. Shutdown the OFSAAI App service: `cd $FIC_APP_HOME/common/FICServer/bin ./stopofsaai.sh`
4. Start the Infrastructure Server in foreground directly on the server or through X-Windows software using the command: `./startofsaai.sh`
5. At the prompt, enter **System Password**. Enter the "new Config schema" password. The service will start and initialize itself if it can successfully connect to the DB.
   - If you are using Apache Tomcat as the Web server, update the **<Context> -> Resource** tag details in the Server.xml file from the `$CATALINA_HOME/conf` directory. For Tomcat, both Config Schema (FICMASTER resource) and Atomic Schema (<INFODOM_NAME> resource) exist.
   - If you are using WebSphere as a web server, follow these steps:
   a. Log in to the **WebSphere Administration Console**, from the left side menu.
   b. Navigate to **Resources >JDBC >Data Sources**. A list of data sources are populated on the right side.
   c. Select the appropriate **Data Source** and edit the connection details. (In this case, both Config and Atomic data sources must be modified).
   - If you are using WebLogic as a web server, follow these steps:

a. Log in to the **WebLogic Administration Console**, from the left side menu

b. Under **Domain Structure** list box, expand the appropriate Domain and navigate to **Services > JDBC >Data Sources**. A list of data sources are populated on the right side.

c. Select the appropriate **Data Source** and edit the connection details. (In this case, both Config and Atomic data sources must be modified).

d. Post successful startup of the service, if required, the Infrastructure server may be shut down and restarted in the background using *nohup* mode.

8.7.2 Modify OFSAA Infrastructure Atomic Schema Password

To change the Atomic Schema password, follow these steps:

1. Change the Atomic schema User Password in the database.
2. Log in to the application from the browser using the SYSADMN account or any user ID which has a System Administrator role mapped.
3. Navigate to **System Configuration > Database Details** window. Modify the password as explained in the following steps:

   a. From the **Database Master** window, select the connection whose password you want to modify and click the button from the toolbar.

   b. Click the button corresponding to the **Alias Name**. The Alias Details window is displayed.

   c. Modify the password in the **Auth String** field.

   - If you are using Apache Tomcat as the Web server, update the **<Context> -> Resource** tag details in the Server.xml file from the `$CATALINA_HOME/conf` directory. For Tomcat, both Config Schema (FICMASTER resource) and Atomic Schema (<INFODOM_NAME> resource) exist.
   - If you are using WebSphere as Web server:

   a. Log in to the **WebSphere Administration Console**, from the left side menu.

   b. Navigate to **Resources >JDBC >Data Sources**. A list of data sources are populated on the right side.

   c. Select the appropriate **Data Source** and edit the connection details. (In this case, both Config and Atomic data sources must be modified).

   - If you are using WebLogic as Web server:

   a. Log in to the **WebLogic Administration Console**, from the left side menu.

   b. Under **Domain Structure** list box, expand the appropriate Domain and navigate to **Services > JDBC >Data Sources.** A list of data sources are populated on the right side.

   c. Select the appropriate **Data Source** and edit the connection details. (In this case, both Config and Atomic data sources must be modified).

4. Restart the OFSAAI services.

8.8 Updating Oracle Sequences

The OFSBD framework uses Oracle sequences for BD datamap component. To this end, OFSBD provides the ability to maintain the Oracle sequences used in Behavior Detection.

This utility must be compulsorily run by clients who are upgrading from Informatica to OFSBD at least one time at the end of the stage 1 upgrade process. This utility also doubles up as a maintenance utility for these Oracle sequences.

The shell script which must be executed for invoking this utility is `run_update_ora_seq.sh`. This script in turn calls a database procedure *P_UPDATE_ORACLE_SEQUENCE*. The database procedure *P_UPDATE_ORACLE_SEQUENCE* contains the logic to set the correct start value of Oracle sequences. The procedure internally drops and re-creates Oracle sequences by getting the max value +1 of the *seq_id* column from the base table as specified in the *TABLE_NM* column of metadata table *KDD_ORACLE_SEQUENCE*.

Clients upgrading from previous version of OFSBD to 6.2.1 version must run the script `run_update_ora_seq.sh` without any parameters.

For maintenance work, the script can be executed either by not passing any parameter or by passing either the table name or the Oracle sequence name as its optional parameter.

1. Without any parameter: `run_update_ora_seq.sh`
2. Passing table name or Oracle sequence name as parameter:

```
run_update_ora_seq.sh<TABLE_NAME> OR
run_update_ora_seq.sh<ORACLE_SEQUENCE_NM>
```

If the table name OR the sequence name is not specified, then the utility performs the maintenance activity for all sequences mentioned in the *KDD_ORACLE_SEQUENCE* metadata table. If the script is called by passing the table name or the Oracle sequence name as its parameter, then the maintenance activity is done only for that particular table / Oracle sequence.

> **Note:**
>
> Do not modify the KDD_ORACLE_SEQUENCE metadata table unless specifically requested by the Oracle support team.

The log for this script is written in the *run_stored_procedure.log* file under the `<OFSAAI Installed Directory>/database/db_tools/logs` directory.

This script is a part of database tools and resides in the `<OFSAAI Installed Directory> / database/db_tools/bin` directory.

Clients who are upgrading from Informatica to OFSBD must run this utility at the end of the stage 1 upgrade process. Also, this utility can be run anytime there is a maintenance work on the database affecting the Oracle sequences. Additionally, there can be scenarios when the database is recovered due to some fault in the database requiring run of this utility. Failure to comply with this may result in Unique Constraints violation errors when datamaps are executed.

> **Note:**
>
> When executing `run_update_ora_seq_sh`, it may fail and display the following error: *ORA-04006: START WITH cannot be less than MINVALUE*. To fix this error, update `dim_country set N_COUNTRY_SKEY = 0 where N_COUNTRY_SKEY = -999`

9

Posting External Alerts through Batches

Alerts which are created by external systems can be posted into the Behavior Detection system for further investigation through batch mode. Once the data is available in the processing tables, the system will post the external alerts.

The user must be mapped to the AMMANADMNGR (Mantas Administrator User Group) user group to post external alert data into the processing tables and execute the batch which moves the data into the Alert Viewer table.

Batch Execution

Once the external data is loaded into the processing tables, the BD_EXTRL_ALERT_GENERATION batch has to be executed. The following tasks should be configured with valid values for the batch date and batch name in the BD batch before triggering the BD_EXTRL_ALERT_GENERATION batch. The BD batch should be configured with the batch name and the batch date before triggering the batch:

- BD_SET_BATCH_DATE_FOR_IPE
- BD_START_BATCH_FOR_IPE

For more information about how to execute a batch, refer to the Oracle Financial Services Analytical Applications Infrastructure User Guide.

> **Note:**
>
> - Values for the tasks should be enclosed within double quotes.
> - Batch date should be in the YYYYMMDD format.
> - The application is pre-packaged with one BD batch. The BD batch should be triggered once a day. If there is a need to trigger the BD batch more than once a day, then insert a record into the KDD_PRCSNG_BATCH.
> - The processing table updates from the External Sources System and from IPE. The BD_EXTRL_ALERT_GENERATION batch and BD_GENERATE_ALERTS_FROM_IPE batch should not be executed in parallel.

A
Logging

This appendix describes the mechanism that OFSBD uses when logging system messages.

About System Log Messages

The Common Logging component provides a centralized mechanism for logging Behavior Detection messages, in which the system places all log messages in a single message library file. In the event that a log file becomes very large (one gigabyte or more), the system creates a new log file. The naming convention is to add .x to the log file's name , such as `mantas.log`, `mantas.log.1, mantas.log.2`.

> **Note:**
>
> The log file size is a configurable property; section Log File Sizes on page 251 provides instructions. The default value for this property is 10 MB. The maximum file size should not exceed two gigabytes (2000000000 bytes).

A.1 Message Template Repository

The message template repository resides in a flat text file and contains messages in the format *<message id 1> <message text>*.

The following is an example of a message repository's contents:

```
111 Dataset id {0} is invalid
112 Run id {0} running Pattern {1} failed
113 Checkpoint false, deleting match
```

- 111, 112, and 113 represent message IDs.
- Whitespace and message text follow.
- {0}s and {1}s represent placeholders for code variable values.

Each subsystem has its own repository. The naming convention for each message library file is `mantas_<subsystem>_message_lib_<language-code>.dat`

- `<subsystem>` is the name of the subsystem
- `<language-code>` is the two-character Java (ISO 639) language code.

`mantas_algorithms_message_lib_en.dat.`

The *log.message.library* property that the subsystem's base *install.cfg* file contains the full path to a subsystem's message library file.

A.2 Logging Levels

This topic outlines the logging levels that the Common Logging component supports.

Table A-1 Logging Levels

Severity (Log Level)	Usage
Warning	Recoverable errors that may still enable the application to continue running but should be investigated , such as failed user sessions or missing data fields).
Notice (default)	High-level, informational messaging that highlights progress of an application , such as startup and shutdown of a process or session, or user login and logout.
Diagnostic	Fine-grained diagnostic errors—used for viewing processing status, performance statistics, SQL statements, etc.
Trace	Diagnostic errors—use only for debugging purposes as this level enables all logging levels and may impact performance.
Fatal	Irrecoverable program, process, and thread errors that cause the application to terminate.

The configuration file specifies enabling of priorities in a hierarchical fashion. That is, if Diagnostic is active, the system enables the Notice, Warning, and Fatal levels.

A.3 Logging Message Libraries

Some Behavior Detection subsystems produce log output files in default locations. The following sections describe these subsystems.

Verifying the Schema Creator Log Files

The path of the log files are as follows:

- For batch logs: FTPSHARE/logs
- For Application logs: FIC_HOME/logs

Administration Tools

The following file is the message library for the Administration Tools application:

```
$FIC_WEB_HOME/AM/admin_tools/WEB-INF/classes/conf/mantas_cfg/etc/
mantas_admin_tools_message_lib_en.dat
```

All message numbers that this log contains must be within the range of 50,000 - 89,999.

Database

The following file is the message library for the Database:

```
<OFSAAI Installed Directory>/database/db_tools/mantas_cfg/etc/
mantas_database_message_lib_en.dat
```

All message numbers that this file contains must be within the range of 250,000 - 289,999.

Scenario Manager

The following file is the message library for the Scenario Manager:

```
<OFSAAI Installed Directory>/behavior_detection/toolkit/mantas_cfg/etc/
mantas_toolkit_message_lib_en.dat
```

All message numbers that this section contains must be within the range of 130,000 - 169,999.

Services

The following file is the message library for the Services:

```
<OFSAAI Installed Directory>/services/server/webapps/mantas/WEB-INF/classes/
conf/
mantas_cfg/etc/mantas_alert_management_message_lib_en.dat
```

All message numbers that this section contains must be within the range of 210,000 - 249,999.

A.4 Alert Viewer

The following logs contain the message library for the Alert Viewer application.

Web Server Logs

The following file is the message library for the Web server logs: `$FIC_WEB_HOME/logs/UMMService.log`

Application Server Logs

The following file is the message library for the Application Server logs: `$FIC_APP_HOME/common/ficserver/logs/RevAppserver.log`

Database Objects Logs

DB objects logs used in the application are maintained in the table *KDD_LOGS_MSGS*. An entry in this table represents the timestamp, stage, error code and module.

Ingestion Manager

The following file is the message library for the Ingestion Manager: `<OFSAAI Installed Directory>/ingestion_manager/config/message.dat`

A.5 Logging Configuration File

You can configure common logging through the following files depending on the subsystem you want to modify.

The following table lists the subsystems and their log files:

Table A-2 Configuration File

Subsytem	File
Database	<OFSAAI Installed Directory> /database/ db_tools/ log4j2.xml
Scenario Manager	<OFSAAI Installed Directory>/ behavior_detection/ toolkit/mantas_cfg/ install.cfg
Behavior Detection	<OFSAAI Installed Directory>/ behavior_detection/ algorithms/MTS/ mantas_cfg/install.cfg
Alert Viewer/Administration Tools Web Server logs	$FIC_WEB_HOME/conf/RevLog4jConfig.xml <root> The following logger levels are available: • DEBUG • INFO • WARN • SEVERE • FATAL

Alert Viewer/Administration Tools Application Server logs	`$FIC_WEB_HOME/conf/RevLog4jConfig.xml`
	`<root>`
	`<priority value ="debug" />`
	`<appender-ref ref="ConsoleAppender1"/>`
	`</root>`
	
	The following logger levels are available:
	• DEBUG
	• INFO
	• WARN
	• SEVERE
	• FATAL

Services	<OFSAAI Installed Directory> /services/ server/ webapps/mantas/WEB-INF/log4j2.xml
Ingestion Manager	<OFSAAI Installed Directory> /ingestion_manager/ config/log4j2_common.xml

The configuration file specifies enabling of priorities in a hierarchical fashion. For example, if Diagnostic priority is enabled, Notice, Warning, and Fatal are also enabled, but Trace is not.

In the configuration file, you can specify the following:

- Locations of recorded log messages
- Logging to the console, files, UNIX syslog, e-mail addresses, and the Microsoft Windows Event Viewer
- Routing based on severity and/or category
- Message library location
- Maximum log file size

Monitoring Log Files

When using a tool to monitor a log file, use the message ID to search for a particular log message instead of text within the message itself. Under normal circumstances, the message

IDs are not subject to change between OFSBD releases, but the text of the message can change. If a message ID does change, you can refer to the appropriate readme.txtfile for information about updated IDs.

A.5.1 Sample Configuration File

This topic contains a sample logging configuration file.

Make special note of the comments in the following sample as they contain constraints that relate to properties and logging.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">

<Appenders>


               <RollingFile name="@@CATAGORY@@" append="true"
filePattern="@@PATH@@">
      <FileName>@@PATH@@</FileName>
      <PatternLayout>
        <Pattern>[%d{E  dd/M/yyyy hh:mm:ss}] [@@CATAGORY@@] [%5p] - %m%n</
Pattern>
      </PatternLayout>
      <Policies>
        <SizeBasedTriggeringPolicy size="10000kb"/>
      </Policies>
       <DefaultRolloverStrategy max="20"/>
    </RollingFile>

                                              <Console name="stdout"
target="SYSTEM_OUT">
            <PatternLayout>
                <pattern>
                    [%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{1} - %msg%n
                </pattern>>
            </PatternLayout>
        </Console>
               </Appenders>

               <Loggers>
           <Logger name="@@CATAGORY@@" level="info" additivity="false">
                                                  <AppenderRef
ref="@@CATAGORY@@" level="trace"/>
                                                  <AppenderRef ref="stdout"
level="error"/>
           </Logger>



        <Root level="error">
            <AppenderRef ref="stdout"/>
        </Root>
    </Loggers>
<!--        <root>
```

```
                                <priority value="##PRIORITY##"></priority>
                </root> -->
</log4j:configuration>
```

A.5.2 Configurable Logging Properties

This topic identifies the configurable properties for logging in an Oracle client's environment.

Table A-3 Configurable Logging Properties

Property	Sample Value	Description
log.format	`<Pattern>[%d{E dd/M/yyyy hh:mm:ss}] [@@CATAGORY@@] [%5p] -%m%n</Pattern>`	Identifies the log formatting string. Refer to Apache Software's *Short Introduction to log4j* guide (http://logging.apache.org/ for more details about the log message format.
log.message.library	To be specified at installation.	Identifies the full path and filename of the message library.

log.max.size	`<Policies>`
	`<SizeBasedTriggeringPolicy size=""10000kb""/>`
	`</Policies>`
		Determines the maximum size (in kilobytes) of a log file before the system creates a new log file.

log.category.<catgory_name>.location		Contains routing information for message libraries for this category.
log.categories.file.path	To be specified at installation.	Identifies the full path to the categories.cfg file.
log.<category_name>.<severity>.location		Contains routing information for message libraries with the given severity for the given category.
log4j.config.file	To be specified at installation.	Specifies the full path to the external log4j configuration file.
log.default.location		Contains routing information for message libraries for this category for which there is no location previously specified.
log.mantaslog.location		Contains routing information for message libraries for this category for which there is no location previously specified.
log.smtp.hostname		Identifies the hostname of the SMTP server if e-mail address is specified as log output.
log.fatal	true	Indicates that fatal logging is enabled; **false** indicates that fatal logging is not enabled.

Table A-3 (Cont.) Configurable Logging Properties

Property	Sample Value	Description
log.fatal.synchronous	false	Indicates that fatal level logging should happen asynchronously; true indicates fatal level logging should happen synchronously. **Note:** Setting value to true (synchronous) may have performance impact
log.warning	true	Indicates enabling of warning logging; **false** indicates that warning logging is not enabled.
log.warning.synchronous	false	Indicates that warning level logging should happen asynchronously; true indicates warning level logging should happen synchronously. **Note:** Setting value to true (synchronous) may have performance impact
log.notice	true	Indicates enabling of notice logging; *false* indicates that notice logging is not enabled.
log.notice.synchronous	false	Indicates that notice level logging should happen asynchronously; true indicates notice level logging should happen synchronously. **Note:** Setting value to true (synchronous) may have performance impact
log.diagnostic	false	Indicates that diagnostic logging is not enabled; **true** indicates enabling of diagnostic logging.
log.diagnostic.synchronous	false	Indicates that diagnostic level logging should happen asynchronously; true indicates diagnostic level logging should happen synchronously. **Note:** Setting value to true (synchronous) may have performance impact
log.trace	false	Indicates that trace logging is not enabled; **true** indicates enabling of trace logging.
log.trace.synchronous	true	Indicates that trace level logging should happen asynchronously; true indicates trace level logging should happen synchronously. **Note:** Setting value to true (synchronous) may have performance impact
log.syslog.hostname	hostname	Indicates the host name of syslog for messages sent to syslog.

Table A-3 (Cont.) Configurable Logging Properties

Property	Sample Value	Description
log.time.zone	US/Eastern	Indicates the time zone that is used when logging messages.

B
OFSBD Software Updates

A hotfix is a package that includes one or more files that are used to address a defect or a change request in OFSBD.

Typically, hotfixes are small patches designed to address specific issues reported by the clients. Hotfixes can affect the following areas in Behavior Detection:

- User Interface (UI)
- Scenarios (patterns and datasets)
- Post-Processing jobs
- Performance
- Ingestion/BD

Each hotfix includes a readme.txt file, which describes the step-by-step process to install the hotfix. Hotfixes are delivered to clients by E-mail or Secure FTP.

B.1 Hotfix Effect on Customization

When a hotfix is installed it can affect your customizations on the User Interface and Scenarios.

User Interface

If your UI customizations are correctly isolated to the custom directory, then the impact should be minimal. It is possible, however, that the hotfix changes information in the base product that you have customized. In that case, you cannot see the effect of the hotfix. To minimize this, be sure to avoid copying more than necessary to the custom directory. For example, you should not copy the entire *BF_Business.xml* file to override a few fields, you should create a new file in the custom directory that only contains the fields you are overriding. The hotfixes delivered will include installation and deployment instructions in the fix documentation.

Scenarios

If you have customized scenarios (changed dataset logic or changed scenario logic), then applying a hotfix to that scenario will remove those customizations. If you customized datasets by creating a dataset override file, then your custom dataset continues to be used after applying the hotfix. It is possible that your custom dataset prevents the scenario fix from being evident (if the dataset you customized was one of the items changed by the hotfix). It is also possible that the hotfix changes the fields it expects from the dataset you customized, causing the scenario to fail. For scenarios you have customized, you should always test the scenario hotfix without your customizations in place, then re-apply them to the scenario, if necessary.

C

User Administration

C.1 Managing User Groups and User Roles

User Roles are pre-defined in OFSFCCM solutions. Sample values for User groups are included in the installer but can be modified by clients to meet their specific needs.

The corresponding mappings between User Roles and sample User Groups are pre-defined but can also be modified by clients to either adjust the role to sample user group mapping or to map roles to newly defined user groups.

Mapping Users with User Groups

The SYSADMN user maps a user to a user group in the Behavior Detection (BD) application, which provides the user access to various functions and privileges as per the role. The following tables describe the predefined User Roles and corresponding User Groups.

The following table describes the BD user roles and corresponding user groups.

Table C-1 BD Roles and User Groups

Role	Group Name	User Group Code
Alert Management Administrator	Mantas Administrator User Group	AMMANADMNGR
Alert Viewer	Alert Viewer User Group	ALERTVIEWERGRP
Data Miner	AM Data Miner User Group	AMDATAMNRGRP

This table describes the FATCA Management User Roles and corresponding User Groups.

Table C-2 FATCA Management Roles and User Groups

Role	Group Name	User Group Code
FATCA Analyst	FATCA Analyst User Group	FTCAANALYSTUG
FATCA Auditor	FATCA Auditor User Group	FTCAAUDITORUG
FATCA Administrator	FATCA Admin User Group	FTCAADMINUG
FATCA Supervisor	FATCA Supervisor User Group	FTCASUPERVISRUG

For more information on creating a new user group and mapping it to an existing role, or mapping usesr with user groups, see the Identity Management section of the Oracle Financial Services Analytical Applications Infrastructure User Guide

Note:

When creating a new User Group, you must set precedence as **5001** or greater. Different solutions have different pre-defined/pre-occupied precedence of User Groups. Therefore, if a BD Admin/System Admin is creating a new User Group, do not use the following precedence while providing precedence value:

Table C-3 Solution with Pre-defined Precedence Range

Solution	Precedence Range Already Occupied
Oracle Financial Services Enterprise Case Management	901 to 1000
Oracle Financial Services Know Your Customer	2001 to 3000
Oracle Financial Services Enterprise Regulatory Reporting	3001 to 4000

For more information about the pre-defined user groups for each solution, see the appropriate Administration Guide.

C.2 Managing User Groups

The following sections describe how to manage User Groups.

Defining User Group Maintenance Details

For more information on defining user group maintenance details, see *Identity Management* in the *Oracle Financial Services Analytical Applications Infrastructure User Guide* .

Adding New User Group Details

For more information on adding new user group details, see *Identity Management* in the *Oracle Financial Services Analytical Applications Infrastructure User Guide* .

Mapping Users to User Groups

When mapping users to user groups, consider the following:

- One user can also be used against multiple roles. If multiple roles are allocated to a single user, then the availability of actions depends on the Four Eyes approval option. If Four Eyes approval is off, then the user can take all actions available by the allocated roles, with no duplicates. If Four Eyes approval is on, then action linked to a role that does not require Four Eyes approval takes precedence if there is a conflict.
- Users will have read-only access to Alert if they have been mapped to the ALERTVIEWERGRP user group. Other user groups such as Supervisor, Analyst, Auditor, Executive groups will not be given access to Alert Viewer.

For more information on mapping users to user group, see *Identity Management* in the *Oracle Financial Services Analytical Applications Infrastructure User Guide* .

Mapping a User to a Single User Group

If a user has only one role then that user can be mapped to a single User Group associated with that User Role. For more information on mapping a user to a single user group, see

Identity Management in the *Oracle Financial Services Analytical Applications Infrastructure User Guide* .

Mapping a User to Multiple User Groups

If a user has more than one role within FCCM (that is, within both Alert Viewer and Enterprise Case Management), then the user must be mapped to the different User Groups associated with the corresponding role. When the user logs into FCCM, the user access permissions are the union of access and permissions across all roles.

Mapping a Function to a Role

Functions must be mapped to appropriate Alert and Case User Roles through Function-Role Map function, which is available in the Security Management System, by logging in as the System Administrator in the OFSAAI toolkit. All Alert Viewer user roles should be mapped to the function *AMACCESS* in order to access an alert. Users of roles that are not mapped to this function cannot access the details of the Alerts.

C.2.1 Mapping User Group(s) to Domain(s)

(Required) <Enter a short description here.>

Actions to Role mappings are done through Database tables. Sample action to role mappings are included in the application.

Actions are primarily associated with a User Role, not an individual user. However, the ability to Reassign To All when taking a Reassign action is associated at the individual user level. Reassign To All means that a user is allowed to assign to users and organizations that may not be within their normal viewing privileges.

1. Map all Alert Viewer User Groups to the Alert Viewer Information Domain (Infodom).
2. Map all Know Your Customer User Groups to the Alert Viewer Information Domain (Infodom), Case Management Information Domain (Infodom), and Know Your Customer Information Domain (Infodom).
3. Map all FATCA User Groups to the Alert Viewer Information Domain (Infodom) and Case Management Information Domain (Infodom).

> **Note:**
>
> For more information on mapping user group or groups to domain or domains, see *Identity Management* in the *Oracle Financial Services Analytical Applications Infrastructure User Guide* . For more information on configuring FATCA, see the *FATCA Administration and Configuration Guide*.

C.2.2 Mapping a User to an Organization

If a user is mapped to an organization indicating that it is the line organization for the user and if there exists any child organization for that line organization, then those organizations are implicitly mapped to the user as a business organization.

If the same organization is already mapped as the business organization, then the child of the organizations should not be mapped to the user implicitly by the system.

If an organization is implicitly mapped to the user based on line organization association, the user can still be unmapped from that organization if there is a need to limit them from seeing

the organization. The organization still shows (I) in the Organization list to show that the organization is a child of the line organization. But the fact that it is not selected will prevent the user from being mapped to it.

The following rules apply:

- Users can have only one organization as the line organization.
- A child organization can have only one parent organization

To map organizations, follow these steps:

1. Select a user from the **Select User** drop-down list.
2. Select the line organization or organizations you want to map the user to from the **Line Organization** drop-down list.

> **Note:**
>
> If the user is associated with both line and business organizations, then the business organizations associated to the Line Organization must be implicitly mapped and display the organizations as well.

   The system visually distinguishes the Implicit (I), which is the system determination based on line organization and Explicit (E), which was manually added by the user mapping, of business organizations. The system displays either I or E in the brackets to indicate that the grid displays two different column, one for Implicit and the other one for Explicit mapping.
3. Click **Save**.

D
Managing Data

This appendix provides information about the datamaps used by FSDF CSA Ingestion and Flat File Ingestion.

FSDF CSA Ingestion

This section refers to FSDF Common Staging Area (CSA) ingestion and covers the following topics:

- CSA Datamaps
- Group Dependencies

Flat File Ingestion

This section refers to Behavior Detection (BD) Ingestion Flat Files and covers the following topics:

- BDF.xml File Parameters
- Behavior Detection Flat File Interface

D.1 CSA Datamaps

This topic lists the files which can be run using FSDF Staging.

Files have been grouped in such a way that files in the same group can be executed in parallel to load data. However, you must execute Group 1 through Group 6 in sequence.

> **Note:**
>
> You must run the Country and Customer data files before you run the other files in their respective Groups.

Table D-1 CSA Datamaps Grouped

Group	Logical Table Name
1	Country

Table D-1 (Cont.) CSA Datamaps Grouped

Group	Logical Table Name
2	Account Phone Watch List Account Email Address Insurance Product Insurance Policy Insurance Transaction Insurance Policy Balance Front Office Transaction Account Customer Role Organization Insurance Policy Feature Insurance Policy To Customer Market Index Daily Loan Issuer Loan Daily Activity Market Index Online Account Service Team Insurance Seller Service Team Member Insurance Seller To License Customer Credit Rating Customer Identification Document
3	Account To Peer Group Account Group Peer Group Security Firm Daily Market Index Member Security Security Customer
4	Account Watch List Entry Loan Product Employee Front Office Transaction Party Organization Relationship Restriction List

Table D-1 (Cont.) CSA Datamaps Grouped

Group	Logical Table Name
5	Managed Account
	Account To Customer
	Account Group Member
	Account To Correspondent Account Balance
	Account Address
	Customer To Markets Served
	Customer To Products Offered
	Customer To Customer Relationship
	Anticipatory Profile
	Customer Phone
	Customer Email Address
	Customer Country
	Customer Address
	Online Account To Account
	Controlling Customer
	Employee To Account
	Account Position
	Security Trading Restriction
	Employee Trading Restriction
	Employee Phone
	Employee Email Address
	Employee Address
	Security Group Member
	Security Investment Rating
	Structured Deal
	Account Profit And Loss
	Account Investment Objective
	Account Position Pair
	Mutual Fund Breakpoint
	Market News Event
6	Borrower
	Account Restriction
	Back Office Transaction
	Investment Advisor
	Settlement Instruction
	Loan Origination Document Print Log

Group Dependencies

Processing data in Group1 requires no prerequisite information (dependencies) for Pre-processing. Groups 2-5, however, rely on successful pre-processing of the previous group to satisfy any dependencies. For example, the Ingestion Manager does not run Group 4 until processing of data in Group 3 completes successfully.

Processing bases the dependencies that determine grouping on the referential relationships within the data. If the Oracle client chooses not to perform referential integrity checking,

grouping is not required (except in some instances). In this case, a need still exists to process some reference data files prior to processing trading data.

D.2 BDF.xml File Parameters

This topic provides the parameters which must be configured in the BDF.xml file.

The following table describes the parameters which must be configured in the BDF.xml file under the <OFSAAI Installed Directory>/bdf/config folder for processing DIS files.

Table D-2 Parameters Related to Processing DIS Files

Property Name	Description	Default
DIS.Source	Indicates the source of DIS records. Valid values are: • l FILE for a DIS file • l FSDW for CSA table loading • FILE-EXT for loading DIS file using an external table	FILE
DIS.ArchiveFlag	Indicates whether a DIS file should be archived after it has been processed.	true
DIS.BufferSize	Indicates the size of a byte buffer (in kilobytes) used to read in a line from a DIS file. This should be set to the maximum possible record size (in kilobytes) of a record in a DIS file.	100
DIS.InputFileCharset	Indicates the character set of a DIS file.	UTF8
DIS.Default.Check.Requirement	Indicates whether the mandatory and conditional checks on a DIS record should be done	true
DIS.Default.Reject.Requirement	Indicates whether a mandatory or conditional check failure for a record should result in the record being rejected. If this is set to FALSE and a missing value is attempted to be inserted into a NOT NULL column, then the record will be rejected anyway.	true
DIS.Default.Check.Domain	Indicates whether the domain value checks on a DIS record should be done.	true
DIS.Default.Reject.Domain	Indicates whether a domain value check failure for a record should result in the record being rejected.	true
DIS.Default.Check.Length	Indicates whether the maximum length checks on a DIS record should be done.	true

Table D-2 (Cont.) Parameters Related to Processing DIS Files

Property Name	Description	Default
DIS.Default.Reject.Length	Indicates whether a maximum length check failure for a record should result in the record being rejected. If this is set to FALSE, then the value will be truncated based on the maximum length of the field.	true
DIS.Default.Check.Threshold	Indicates whether the threshold checks (GREATER_THAN_ZERO, etc) on a DIS record should be done.	true
DIS.Default.Reject.Threshold	Indicates whether a threshold check failure for a record should result in the record being rejected.	true
DIS.Default.Check.Lookup	Indicates whether the reference data lookups on a DIS record should be done.	true
DIS.Default.Reject.Lookup	Indicates whether a reference data lookup failure for a record should result in the record being rejected.	true
MITrxnProducttypes	Indicates the parameter which is used to pass a list of product codes for trailing digit purpose (AUG_INSTR_NB derivation).	• l CHECK • CHECK- ACH
CustProfileLookBack	Indicates the parameter which is used to look back at the days in Customer Summary Daily for Customer Summary Month recalculation. In order to look back at a specific time period in Customer Summary Daily, you must have partitions available in Customer Summary Month.	31
CustAcctHolderType	Indicates the parameter which is used to identify customer account types to be included in customer summary.	CI

BD Ingest DIS Data Files by Group

Ingestion Manager processes data files in groups (in a specified order) from Oracle client data in the /inbox directory. The following list of files can be run using CSA in FSDF. Files have been grouped in such a way that files in the same group can be executed in parallel to load data. However, you must execute Group 1 through Group 6 in sequence. The following table lists the data files by group.

Table D-3 BD Ingest DIS Data Files By Group

Group	Data Files
1	Account Phone Watch List Account Email Address Insurance Product Insurance Policy Insurance Transaction Insurance Policy Balance Front Office Transaction Account Customer Role Organization Insurance Policy Feature Insurance Policy To Customer Market Index Daily Loan Issuer Loan Daily Activity Market Index Online Account Service Team Insurance Seller Service Team Member Insurance Seller To License Country
2	Account To Peer Group Account Group Peer Group Security Firm Daily Market Index Member Security Security
3	Account Customer Watch List Entry Loan Product Employee Front Office Transaction Party Organization Relationship Restriction List

Table D-3 (Cont.) BD Ingest DIS Data Files By Group

Group	Data Files
4	Managed Account Account To Customer Account Group Member Account To Correspondent Account Balance Account Address Customer To Markets Served Customer To Products Offered Customer To Customer Relationship Anticipatory Profile Customer Phone Customer Email Address Customer Country Customer Address Online Account To Account Controlling Customer Employee To Account Account Position Security Trading Restriction Employee Trading Restriction Employee Phone Employee Email Address Employee Address Security Group Member Security Investment Rating Structured Deal Account Profit And Loss Account Investment Objective Account Position Pair Mutual Fund Breakpoint Market News Event
5	Borrower Account Restriction Back Office Transaction Investment Advisor Settlement Instruction Loan Origination Document Print Log
6	OpenOrder Order TradeExecutionEvent

D.3 Behavior Detection Flat File Interface

This topic describes the Ingestion Flat File details for products within the BD Application Pack.

Files have been grouped in such a way that files in the same group can be executed in parallel to load data. However, you must execute Group 1 through Group 5 in sequence. For more information, The Staging Representation column indicates whether this file requires a Staging source.

Group 1 Interface Ingestion Flat Files

The following table describes the Group 1 Ingestion Flat File details.

Table D-4 Group 1 Interface Ingestion Flat Files

Interface File Name	AML	Fraud	KYC	FATCA	CTR	Current Ingestion	Staging Representation
Account Phone	Yes	Yes	Yes	Yes	NA	BD Datamaps	Yes
Account Email Address	Yes	Yes	Yes	Yes	NA	BD Datamaps	Yes
Insurance Policy	Yes	Yes	Yes	NA	NA	BD Datamaps	Yes
Insurance Policy Balance	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Account Customer Role	Yes	Yes	NA	Yes	Yes	BD Datamaps	Yes
Insurance Policy Feature	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Insurance Policy to Customer	Yes	Yes	Yes	NA	NA	BD Datamaps	Yes
Loan	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Loan Daily Activity	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Online Account	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Insurance Seller	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Insurance Seller to License	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Country	Yes	Yes	NA	Yes	NA	BD Datamaps	Yes
Watch List	Yes	Yes	Yes	NA	NA	BD Datamaps	Yes
Insurance Product	Yes	Yes	Yes	NA	NA	BD Datamaps	Yes
Insurance Transaction	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Front Office Transaction	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Market Index Daily	NA	NA	NA	NA	NA	BD Datamaps	Yes
Issuer	NA	NA	NA	NA	NA	BD Datamaps	Yes
Organization	NA	NA	NA	NA	Yes	BD Datamaps	Yes
Market Index	NA	NA	NA	NA	NA	BD Datamaps	Yes
Service Team Member	NA	NA	NA	NA	NA	BD Datamaps	Yes
Service Team	NA	NA	NA	NA	NA	BD Datamaps	Yes
CTR Transaction	Yes	Yes	NA	NA	Yes	runDP/ runDL	No
Account Realized Profit and Loss	NA	NA	NA	NA	NA	runDP/ runDL	No
Letter of Intent	NA	NA	NA	NA	NA	runDP/ runDL	No
Collateral Value-Currency	NA	NA	NA	NA	NA	runDP/ runDL	No
Collateral Value-Product	NA	NA	NA	NA	NA	runDP/ runDL	No

Table D-4 (Cont.) Group 1 Interface Ingestion Flat Files

Interface File Name	AML	Fraud	KYC	FATCA	CTR	Current Ingestion	Staging Representation
Commission Product	NA	NA	NA	NA	NA	runDP/ runDL	No
Compliant Registration	NA	NA	NA	NA	NA	runDP/ runDL	No
Complaint Type Rating	NA	NA	NA	NA	NA	runDP/ runDL	No
Employee to Insurance Policy	NA	NA	NA	NA	NA	runDP/ runDL	No
Investment Guideline	NA	NA	NA	NA	NA	runDP/ runDL	No
Investment Guideline to Account	NA	NA	NA	NA	NA	runDP/ runDL	No
System Logon Type	NA	NA	NA	NA	NA	runDP/ runDL	No
Registered Representative Complaint	NA	NA	NA	NA	NA	runDP/ runDL	No
Energy And Commodity Instrument	NA	NA	NA	NA	NA	runDP/ runDL	No

Group 2 Interface Ingestion Flat Files

The following table describes the Group 2 Ingestion Flat File details.

Table D-5 Group 2 Interface Ingestion Flat Files

Interface File Name	AML	Fraud	KYC	FATCA	CTR	Current Ingestion	Staging Representation
Account to Peer Group	Yes	Yes	Yes	NA	NA	BD Datamaps	Yes
Account Group	Yes	Yes	NA	NA	NA	BD	Yes
Peer Group	Yes	Yes	Yes	NA	NA	BD Datamaps	Yes
Security Firm Daily	NA	NA	NA	NA	NA	BD Datamaps	Yes
Security	NA	NA	NA	NA	NA	BD Datamaps	Yes
Market Index Member Security	NA	NA	NA	NA	NA	BD Datamaps	Yes
Security Market State Change	NA	NA	NA	NA	NA	BD Datamaps	Yes
Matched Entity	Yes	Yes	NA	NA	NA	runDP/ runDL	No
Trusted Pair	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Firm Account Position Pair	NA	NA	NA	NA	NA	runDP/ runDL	No
Natural Gas Flow	NA	NA	NA	NA	NA	runDP/ runDL	No

Group 3 Interface Ingestion Flat Files

The following table describes the Group 3 Ingestion Flat File details.

Table D-6 Group 3 Interface Ingestion Flat Files

Interface File Name	AML	Fraud	KYC	FATCA	CTR	Current Ingestion	Staging Representation
Account	Yes	Yes	Yes	Yes	Yes	BD Datamaps	Yes
Customer	Yes	Yes	Yes	Yes	Yes	BD Datamaps	Yes
Watch List Entry	Yes	Yes	Yes	NA	NA	BD Datamaps	Yes
Loan Product	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Employee	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Front Office Transaction Party	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Organization Relationship	NA	NA	NA	NA	Yes	BD Datamaps	Yes
Restriction List	NA	NA	NA	NA	NA	BD Datamaps	Yes
Account Supplemental Attribute	NA	NA	Yes	NA	NA	runDP/ runDL	No
Customer Supplemental Attribute	NA	NA	Yes	NA	NA	runDP/ runDL	No
Market Trading Session	NA	NA	NA	NA	NA	runDP/ runDL	No
Account GroupAddress	Yes	Yes	NA	NA	NA	runDP/ runDL	No
Account Group Investment Objective	NA	NA	NA	NA	NA	runDP/ runDL	No
Account Group IOS Member	NA	NA	NA	NA	NA	runDP/ runDL	No
Account Group Member Experience	NA	NA	NA	NA	NA	runDP/ runDL	No
Loan Origination Action	NA	NA	NA	NA	NA	runDP/ runDL	No
Mail Handling Instruction Activity	NA	NA	NA	NA	NA	runDP/ runDL	No
Banker To Officer	NA	NA	NA	NA	NA	runDP/ runDL	No
Reference Table Detail	NA	NA	NA	NA	NA	runDP/ runDL	No
General Usage List	NA	NA	NA	NA	NA	runDP/ runDL	No
Loan Origination Product	NA	NA	NA	NA	NA	runDP/ runDL	No
Organization To Mortgage Type	NA	NA	NA	NA	NA	runDP/ runDL	No
Securities License	NA	NA	NA	NA	NA	runDP/ runDL	No
Service Vendor	NA	NA	NA	NA	NA	runDP/ runDL	No
Energy and Commodity Trade	NA	NA	NA	NA	NA	runDP/ runDL	No

Group 4 Interface Ingestion Flat Files

The following table describes the Group 4 Ingestion Flat File details.

Table D-7 Group 4 Interface Ingestion Flat Files

Interface File Name	AML	Fraud	KYC	FATCA	CTR	Current Ingestion	Staging Representation
Market News Event	NA	NA	NA	NA	NA	BD Datamaps	No
Managed Account	Yes	Yes	Yes	NA	NA	BD Datamaps	Yes
Account To Customer	Yes	Yes	Yes	Yes	Yes	BD Datamaps	Yes
Branch CTR Transaction	NA	NA	NA	NA	Yes	BD Datamaps	Yes
Branch CTR Conductor	NA	NA	NA	NA	Yes	BD Datamaps	Yes
Branch CTR Summary	NA	NA	NA	NA	Yes	BD Datamaps	Yes
Account Group Member	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Account To Correspondent	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Account Balance	Yes	Yes	Yes	Yes	NA	BD Datamaps	Yes
Account Address	Yes	Yes	Yes	Yes	NA	BD Datamaps	Yes
Customer Identification Document	Yes	Yes	Yes	Yes	NA	BD Datamaps	Yes
Customer To Markets Served	Yes	Yes	Yes	NA	NA	BD Datamaps	Yes
Customer To Products Offered	Yes	Yes	Yes	NA	NA	BD Datamaps	Yes
Customer To Customer Relationship	Yes	Yes	Yes	Yes	NA	BD Datamaps	Yes
Anticipatory Profile	Yes	Yes	Yes	NA	NA	BD Datamaps	Yes
Customer Phone	Yes	Yes	Yes	Yes	Yes	BD Datamaps	Yes
Customer Email Address	Yes	Yes	Yes	Yes	Yes	BD Datamaps	Yes
Customer Country	Yes	Yes	Yes	NA	NA	BD Datamaps	Yes
Customer Address	Yes	Yes	Yes	Yes	Yes	BD Datamaps	Yes
Online Account to Account	Yes	Yes	Yes	NA	NA	BD Datamaps	Yes
Controlling Customer	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Employee To Account	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Account Position	NA	NA	NA	NA	NA	BD Datamaps	Yes
Security Trading Restriction	NA	NA	NA	NA	NA	BD Datamaps	Yes
Employee Trading Restriction	NA	NA	NA	NA	NA	BD Datamaps	Yes
Employee Phone	NA	NA	NA	NA	NA	BD Datamaps	Yes
Employee Email Address	NA	NA	NA	NA	NA	BD Datamaps	Yes
Employee Address	NA	NA	NA	NA	NA	BD Datamaps	Yes
Outside Business Activity	NA	NA	NA	NA	NA	BD Datamaps	Yes
Private Security Transaction	NA	NA	NA	NA	NA	BD Datamaps	Yes

Table D-7 (Cont.) Group 4 Interface Ingestion Flat Files

Interface File Name	AML	Fraud	KYC	FATCA	CTR	Current Ingestion	Staging Representation
Security Group Member	NA	NA	NA	NA	NA	BD Datamaps	Yes
Security Investment Rating	NA	NA	NA	NA	NA	BD Datamaps	Yes
Structured Deal	NA	NA	NA	NA	NA	BD Datamaps	Yes
Account Profit and Loss	NA	NA	NA	NA	NA	BD Datamaps	Yes
Account Position Pair	NA	NA	NA	NA	NA	BD Datamaps	Yes
Account Investment Objective	NA	NA	NA	NA	NA	BD Datamaps	Yes
Mutual Fund Breakpoint	NA	NA	NA	NA	NA	BD Datamaps	Yes
Account Feature	NA	NA	NA	NA	NA	runDP/ runDL	No
Access Events	NA	Yes	NA	NA	NA	runDP/ runDL	No
Customer Balance	NA	Yes	NA	NA	NA	runDP/ runDL	No
Front Office Transaction Remittance Document	Yes	Yes	NA	NA	NA	runDP/ runDL	No
Related Front Office Transaction Information	Yes	Yes	NA	NA	NA	runDP/ runDL	No
Account To Organization	NA	NA	NA	NA	NA	runDP/ runDL	No
Firm Account Position	NA	NA	NA	NA	NA	runDP/ runDL	No
External Investment Account Position	NA	NA	NA	NA	NA	runDP/ runDL	No
Employee To Organization	NA	NA	NA	NA	NA	runDP/ runDL	No
Security Select List Entry	NA	NA	NA	NA	NA	runDP/ runDL	No
Account Fees	NA	NA	NA	NA	NA	runDP/ runDL	No
Account Profile Stage	NA	NA	NA	NA	NA	runDP/ runDL	No
Account Qualification Agreement	NA	NA	NA	NA	NA	runDP/ runDL	No
Account Representative Position	NA	NA	NA	NA	NA	runDP/ runDL	No
Account Asset Allocation	NA	NA	NA	NA	NA	runDP/ runDL	No
Account Scheduled Event	NA	NA	NA	NA	NA	runDP/ runDL	No
Account Identifier Change History	NA	NA	NA	NA	NA	runDP/ runDL	No
Account Position Profile And Loss	NA	NA	NA	NA	NA	runDP/ runDL	No
Uncovered Option Account Position	NA	NA	NA	NA	NA	runDP/ runDL	No
Account Collateral	NA	NA	NA	NA	NA	runDP/ runDL	No

Table D-7 (Cont.) Group 4 Interface Ingestion Flat Files

Interface File Name	AML	Fraud	KYC	FATCA	CTR	Current Ingestion	Staging Representation
Mail Handling Instruction	NA	NA	NA	NA	NA	runDP/ runDL	No
Mutual Fund Family Letter of Intent	NA	NA	NA	NA	NA	runDP/ runDL	No
Employee Disciplinary Action	NA	NA	NA	NA	NA	runDP/ runDL	No
Employee Exam History	NA	NA	NA	NA	NA	runDP/ runDL	No
Employee Firm Transfer History	NA	NA	NA	NA	NA	runDP/ runDL	No
Employee Securities License State Registration	NA	NA	NA	NA	NA	runDP/ runDL	No
Employee Supervision List	NA	NA	NA	NA	NA	runDP/ runDL	No
Employee To Manager History	NA	NA	NA	NA	NA	runDP/ runDL	No
Employee To Securities License	NA	NA	NA	NA	NA	runDP/ runDL	No
Employment History	NA	NA	NA	NA	NA	runDP/ runDL	No
System Logon	NA	NA	NA	NA	NA	runDP/ runDL	No
Plan of Solicitation	NA	NA	NA	NA	NA	runDP/ runDL	No
Mutual Fund Family Configuration	NA	NA	NA	NA	NA	runDP/ runDL	No
Energy And Commodity Market Daily	NA	NA	NA	NA	NA	runDP/ runDL	No
Energy And Commodity Firm Daily	NA	NA	NA	NA	NA	runDP/ runDL	No
Energy And Commodity Reported Market Sale	NA	NA	NA	NA	NA	runDP/ runDL	No
Energy And Commodity Market Trading Session	NA	NA	NA	NA	NA	runDP/ runDL	No
Energy And Commodity Market Center	NA	NA	NA	NA	NA	runDP/ runDL	No
Energy And Commodity Location	NA	NA	NA	NA	NA	runDP/ runDL	No
Energy Flow Mode	NA	NA	NA	NA	NA	runDP/ runDL	No
Energy and Commodity Instrument Position	NA	NA	NA	NA	NA	runDP/ runDL	No

Group 5 Interface Ingestion Flat Files

The following table describes the Group 5 Ingestion Flat File details.

Table D-8 Group 5 Interface Ingestion Flat Files

Interface File Name	AML	Fraud	KYC	FATCA	CTR	Current Ingestion	Staging Representation
Borrower	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Back Office Transaction	Yes	Yes	NA	NA	NA	BD Datamaps	Yes
Account Restriction	NA	NA	Yes	NA	NA	BD Datamaps	Yes
Investment Advisor	NA	NA	NA	NA	NA	BD Datamaps	Yes
Investment Guideline Override	NA	NA	NA	NA	NA	BD Datamaps	Yes
Settlement Instruction	NA	NA	NA	NA	NA	BD Datamaps	Yes
Loan Origination Document Print Log	NA	NA	NA	NA	NA	BD Datamaps	Yes
Change Log	Yes	Yes	Yes	Yes	NA	runDP/ runDL	No
Options Violation	NA	NA	NA	NA	NA	runDP/ runDL	No
Loan Origination Condition	NA	NA	NA	NA	NA	runDP/ runDL	No
Loan Origination Fee Detail	NA	NA	NA	NA	NA	runDP/ runDL	No
Loan Origination Note	NA	NA	NA	NA	NA	runDP/ runDL	No
Loan Origination To Service	NA	NA	NA	NA	NA	runDP/ runDL	No
Investment Guideline Override	NA	NA	NA	NA	NA	runDP/ runDL	No
Loan Origination Condition Type	NA	NA	NA	NA	NA	runDP/ runDL	No
System Logon To System Logon Type	NA	NA	NA	NA	NA	runDP/ runDL	No
System Logon To Organization	NA	NA	NA	NA	NA	runDP/ runDL	No
Registered Representative Account Commission	NA	NA	NA	NA	NA	runDP/ runDL	No
Registered Representative Account Commission Prior Year	NA	NA	NA	NA	NA	runDP/ runDL	No
Registered Representative Commission Monthly Profile	NA	NA	NA	NA	NA	runDP/ runDL	No
Registered Representative Commission Product	NA	NA	NA	NA	NA	runDP/ runDL	No
Currency Transaction	NA	NA	NA	NA	Yes	BD Datamaps	Yes

Group 6 Interface Ingestion Flat Files

The following table describes the Group 6 Ingestion Flat File details.

Table D-9 Group 6 Interface Ingestion Flat Files

Interface File Name	AML	Fraud	KYC	FATCA	CTR	Current Ingestion	Staging Representation
Inside Quote	NA	NA	NA	NA	NA	BD Datamaps	Yes
Market Center Quote	NA	NA	NA	NA	NA	BD Datamaps	Yes
ReportedMarketSale	NA	NA	NA	NA	NA	BD Datamaps	Yes
InsideQuote_Derived	NA	NA	NA	NA	NA	BD Datamaps	Yes
MarketCenterQuote_De rived	NA	NA	NA	NA	NA	BD Datamaps	Yes
ReportedMarketSale_Derived	NA	NA	NA	NA	NA	BD Datamaps	Yes

Note:

The AccountAverageNetWorth file is an exceptional case, and is only intended to be run once before any other files have been loaded. The average net worth amount in the account profile table is built up over time as transactions are ingested. This file allows this value to be set as a starting point before any transactions have been ingested. After transactions are ingested, this file should no longer be used.

Note:

The following derived datamaps must be run after running the corresponding BD scripts.

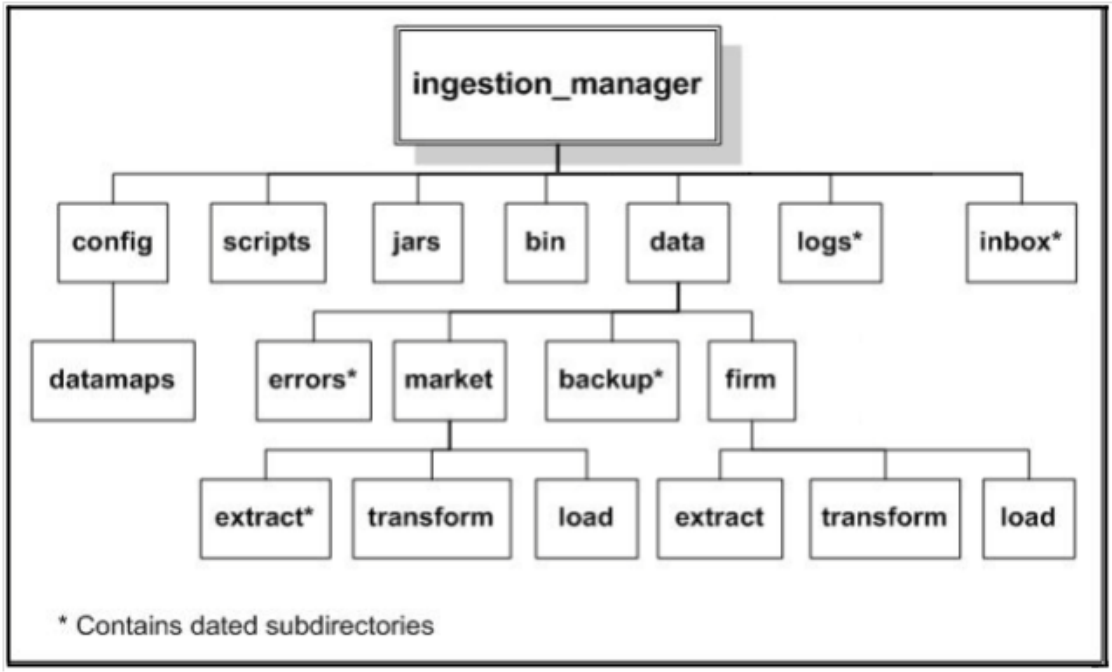
- CurrencyTransaction_ExemptFlagUpd
- SecurityInvestmentRating_PrevInvestmentUpd

D.4 Pre-processing & Loading Directory Structure

Data for Pre-processing and Loading are organized in subdirectories below the ingestion_manager root level.

The following figure illustrates the subdirectories that the `ingestion_manager` directory contains.

Figure D-1 Pre-processing & Loading Directory Structure



The following table lists these important subdirectories that compose the `<OFSAAI Installed Directory>/ingestion_manager` directory structure.

Table D-10 Data Management Directory Structure Description

Directory Name	Description
data/backup	Contains backup files for the various Data Management components.
data/errors	Contains error files for various Data Management components.
data/firm	Contains Oracle client data files that Data Management components write.
inbox	Contains data files that the Oracle client provides.
jars	Contains the Java Archive (JAR) files to run Java Data Management components implemented in Java.
config	Contains files used to configure the Data Management components (see for more information).
logs	Contains log files that Data Management components write.
scripts	Contains all the shell scripts for running Data Management components.
/inbox/<yyyymmdd>	Backup of input files (for restart purposes, if necessary).
/data/<firm or market>/ load	• for loading into the database as <data type>_<yyyymmdd>_<batch_name>_<N>.XDP. • Load controlfiles.

Table D-10 (Cont.) Data Management Directory Structure Description

Directory Name	Description
/logs/<yyyymmdd>	Pre-processing and load status, and error messages.
/data/errors/<yyyymmdd>	Records that failed validation. The file names are the same as those of the input files.
/data/firm/transform	TC trading data files that the FDT processes.

These directories are further explained in the following sections:

- jars Subdirectory
- config Subdirectory
- scripts Subdirectory
- data Subdirectory
  - extract Subdirectory
  - transform Subdirectory
  - load Subdirectory
- inbox Subdirectory
- logs Subdirectory

D.4.1 jars Subdirectory

The jars subdirectory within the ingestion_manager directory contains Java programs that Ingestion Manager uses.

A run script in the scripts subdirectory launches each program. See scripts Subdirectory for more information.

D.4.2 scripts Subdirectory

The scripts subdirectory within the ingestion_manager directory contains the UNIX Bourne Shell scripts to run runtime components.

Executing a run script runs a new instance of a component. If an application component terminates successfully, a script returns a zero return code. If the component fails to terminate successfully, the script returns a non-zero status (normally 1).

The following table defines the run scripts for starting each component and any special instructions.

Table D-11 Run Scripts by Component

Script Names	Description or Special Instructions
runFDT.sh	Launches the FDT. This script stops after it processes all qualifying files that it finds in the /data/firm/transform directory at the time the process starts. The system processes an input file if the processing data and batch name are correct. You can stop the FDT immediately by using the UNIX kill command to stop the process ID for the Java process that is a child of the runFDT.sh process.
runDL.sh <data type>	Launches an instance of the data loader (runDL.sh). For example: runDL.sh Customer To run a specific data loader, specify a valid component that the run script recognizes. If the script does not recognize the component, it exits with an error and identifies the valid list of parameters. For valid component names, see Figure 40 .
runRebuildIndexes.sh	Launches a process to rebuild the indexes of the given component. Processing requires this script only during use of a live market feed. A valid <component> value is one of InsideQuote, ReportedMarketSale, or MarketCenterQuote.
process_firm_summary.sh	Calls a database procedure to build summary statistics about the Oracle client (firm) data.
process_market_summary.sh	Calls a database procedure to build summary statistics about the Market data.
market_analyze.sh	Calls a database procedure to create internal database statistics for Market tables.
firm_analyze.sh	Calls a database procedure to create internal database statistics for Oracle client (firm) tables.
runIMC.sh	Launches the Ingestion Manager Cleaner (IMC) utility. The utility terminates after it finishes removing old data subdirectories and their contents.
env.sh	Contains common configuration settings required to run Data Management subsystem components. The run*.sh scripts use this script.
truncate_table.sh <schema.tablename>	Truncates a specified table in the database. Processing runs this script prior to loading reference data when an Oracle client wants to perform a full refresh of the data.

runUtility.sh <datatype>	Launches a Java based utility to derive the contents of a given database table. You must run runDL.sh <data type> after this script has executed successfully.
	For example:
	
	`runUtility.sh AccountDailySecurityProfile`
	`runDL.sh AccountDailySecurityProfile`

Table D-11 (Cont.) Run Scripts by Component

Script Names	Description or Special Instructions
runDP.sh <data type>	Launches an instance of the data Pre-processor(runDP.sh). For example: runDP.sh Customer To run a specific Data Pre-processor, specify a valid input component that the run script recognizes. If the script does not recognize the input component, it exits with an error and identifies the valid list of parameters. For valid component names, see Figure40

The run scripts in the following table configure the executing environment for the Java component, and then execute it. All run scripts invoke the env.sh script to define environment variables that the components require. The run scripts also start the Java program with appropriate command line parameters, which the following table describes.

Table D-12 Environment Variable Descriptions

Parameter	Description
classpath	Directs the Java Runtime Environment (JRE) to the location of Java programs and supporting Java classes.
Djava.security.policy	Sets the location of the policy file that provides directory and network access rights to the component.
server	Instructs Java JRE to optimize for server-based processing.
Xms<NNNN>*	Indicates the minimum number of megabytes (as NNNN) to reserve for Java memory allocation.
Xmx<NNNN>*	Indicates the maximum number of megabytes (as NNNN) to reserve for Java memory allocation. Note: Setting Xmx too small may result in component failure.

Note:

Default values that are appropriate to the operating system in use , such as Linux or Solaris, are automatically set in the env.sh file:

- For 64-bit operating systems, the maximum value should not be greater than 3500 MB.
- For 32-bit operating systems, the maximum value should not be greater than 1800 MB.

Minimum values vary by component; the env.sh file specifies these values.

D.4.3 config Subdirectory

The config subdirectory within the data_ingest directory contains the application configuration files.

The following sections describe the configuration files:

- DataIngest.properties: Property file that contains settings that are configured at installation. These settings are of the most interest to an Oracle client regarding modification
- DataIngest.xml: XML configuration file that contains settings that normally remain as is.
- DataIngestCustom.xml: XML configuration file that contains overridden settings from DataIngest.xml.

The DataIngest.properties and DataIngest.xml files contain settings for IP addresses, port numbers, file paths, file extensions, and other runtime settings including an application's performance tuning parameters. Property files within the config subdirectory contain database user IDs and encrypted passwords.

The config/datamaps subdirectory also contains XML data maps for parsing input data and mapping processed data to fields in files and in databases. The XML data maps are preset and do not require any modifications.

D.4.3.1 Data Ingest Properties Configuration File

The following table describes the parameters for the DataIngest.properties configuration file.

Table D-13 Data Ingest Properties

Property Name	Description	Example
DB.Connection.Instance	Database instance to connect to on the database servers. Typically, the instance name matches the database name portion of the DB.Connection.URL.	D1O9L2
DB.Connection.User	Database user name that Java ingestion components uses when connecting to the database. The database user must have been assigned the appropriate privileges that Data Management requires for interacting with the database.	ATOMIC
DB.Connection.Password	Password that Java Ingestion components use when connecting with the database. This is set by the Password Manager Utility.	
DB.Type	The type of database being used.	Oracle
MANTAS.DBSchema	Schema name for the ATOMIC database schema. Data Management accesses the ATOMIC schema when allocating sequence IDs to ingested records.	ATOMIC

Table D-13 (Cont.) Data Ingest Properties

Property Name	Description	Example
MARKET.DBSchema	Schema name for the ATOMIC database schema. Data Management stores market data related records in the ATOMIC schema.	ATOMIC
DB.Connection.URL	Database URL for JDBC connections made by Java ingestion components. The content and format of this value is specific to the database vendor and the vendor database driver. Oracle recommends that you use Thin Driver.	jdbc:oracle:thin:@ofss220074.oracle.com:1521:Ti1O11L56
BUSINESS.DBSchema	Schema name for the ATOMIC database schema. Data Management stores market data related records in the ATOMIC schema.	ATOMIC

D.4.3.2 Data Ingest XML Configuration File

The following table describes the parameters for the DataIngest.xml configuration file.

> **Note:**
>
> Default values for properties in this file are suitable for most deployments. Use caution when changing any default values.

Table D-14 Data Ingest Properties

Type	Property Name	Description	Example
ProcessingBatch: Specifies batch settings that override settings in the database. Overrides are primarily useful during testing.	ProcessingBatch.Name	Sets the current batch name. Ingestion components process only input files that contain this value in the batch name portion of the file name. This property should be blank during normal operation.	

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
ProcessingBatch: Specifies batch settings that override settings in the database. Overrides are primarily useful during testing.	ProcessingBatch.Date	Sets the current processing date. Ingestion components process only input files that contain this value in the processing date portion of the file name. This property should be blank during normal operation. The date format is YYYYMMDD.	
ProcessingBatch: Specifies batch settings that override settings in the database. Overrides are primarily useful during testing.	ProcessingBatch.Last	Identifies the flag that indicates processing of the last batch of the day to Data Management. This property should be blank during normal operation.	
Miscellaneous	DefaultSourceSystem.value	Indicates the default value to use for source system when manufacturing reference data records.	MTS
Miscellaneous	BufferSize.value	Specifies the buffer size in kilobytes for I/O byte buffers that the MDS and FDT processes create to read input files. Use care when changing this parameter due to impact on performance and memory requirements.	1024
Miscellaneous	DirectBufferSize.value	Specifies the buffer size in kilobytes for Java NIO direct byte buffers that the MDS, MDT, and FDT processes create to read input files. Use care when changing this parameter due to impact on performance and memory requirements	1024
Miscellaneous	DefaultCurrency.value	Indicates the value to use as the issuing currency when manufacturing security records from order or trade execution records.	USD
Miscellaneous	UseDirectBuffers.value	Specifies whether to make use of Java NIO's direct buffer mechanism.	TRUE

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
Miscellaneous	Separator.value	Specifies the delimiter that separates fields in data file records.	~
Log: Specifies properties used to configure the common logging module.	Log.UseDefaultLog	Specifies whether the system uses the default log file for a component. The default log file has the name of the component and resides in a date subdirectory of the logs directory (in YYYYMMDD format).	TRUE
Log: Specifies properties used to configure the common logging module.	Log.UseDateLog	Specifies whether to put default log file for a component in a date subdirectory. Otherwise, it is placed directly under the logs directory.	TRUE
Log: Specifies properties used to configure the common logging module.	Log.InitDir	Specifies the location of the properties file for configuring the common logging module (install.cfg).	../config
DB: Specifies properties related to database access.	DB.Connection.Driver	Indicates the JDBC driver class name.	oracle.jdbc.driver.OracleDriver
DB: Specifies properties related to database access.	DB.Connection.InitialConnect ions	Specifies the number of connections initially to allocate in the connection pool.	1
DB: Specifies properties related to database access.	DB.Connection.MaximumConnect ions	Indicates the maximum number of connections in the connection pool. You should correlate this parameter to the number of configured threads for the component.	10
DB: Specifies properties related to database access.	DB.Connection.Timeout	Identifies the number of seconds to wait before timing out on a database connection attempt.	10
DB: Specifies properties related to database access.	DB.Connection.NumRetries	Specifies the maximum number of times to attempt to connect to a database before failing.	5
BUSINESS: Specifies properties related to data loaded into the ATOMIC schema.	BUSINESS.ExtractDir	Identifies the parent directory for intermediate files that Pre-processors produce that are applicable to the ATOMIC schema in the database.	../data/firm/extract

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
BUSINESS: Specifies properties related to data loaded into the ATOMIC schema.	BUSINESS.TransformDir	Specifies the working directory for the FDT component which transforms BUSINESS trade-related data.	../data/firm/transform
BUSINESS: Specifies properties related to data loaded into the ATOMIC schema.	BUSINESS.LoadDir	Indicates the parent directory for directories that store ATOMIC schema bound data files prior to loading with the Java data loader component. Control files for native loaders also reside below this directory.	../data/firm/load
MANTAS: Specifies properties related to data loaded into the ATOMIC schema.	MANTAS.ExtractDir	Specifies the parent directory for intermediate files that Pre-processors produce that are applicable to the ATOMIC schema in the database.	../data/mantas/extract
MANTAS: Specifies properties related to data loaded into the ATOMIC schema.	MANTAS.TransformDir	Specifies the working directory for intermediate files that utilities produce that are applicable to the ATOMIC schema in the database.	../data/mantas/ transform
MANTAS: Specifies properties related to data loaded into the ATOMIC schema.	MANTAS.LoadDir	Specifies the parent directory for directories that store ATOMIC schema bound data files prior to loading with the Java data loader component. Control files for native loaders also reside below this directory.	../data/mantas/load
Directory: Specifies properties used to define directory locations.	Directory.Log	Specifies the parent directory for log file directories and log files that Java ingestion components create.	../logs
Directory: Specifies properties used to define directory locations.	Directory.Inbox	Specifies the input directory where Java ingestion components find files that the Oracle client submits. Processing creates subdirectories in the / inboxdirectory for each day of data, to contain a copy of the input data file.	../inbox

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
Directory: Specifies properties used to define directory locations.	Directory.Error	Specifies the parent directory for error directories that contain error data files that Java ingestion components create. Each error data file contains records that were not processed due to error.	../data/errors
Directory: Specifies properties used to define directory locations.	Directory.Archive	Specifiesthe parent directory for directories that contain backup copies of intermediate files that Java ingestion components create.	../data/backup
Directory: Specifies properties used to define directory locations.	Directory.Config	Specifies the directory containing configuration files for Java ingestion server.	../config
Directory: Specifies properties used to define directory locations.	Directory.FuzzyMatcher	Specifies the directory containing files related to fuzzy matcher.	../fuzzy_match
Directory: Specifies properties used to define directory locations.	Directory.DataMap	Specifies the directory that contains XML data map files.	../config/datamaps
FileExtension: Specifies properties used to define extensions for various types of files.	FileExtension.Log	Specifies the file name extension for log files.	.log
FileExtension: Specifies properties used to define extensions for various types of files.	FileExtension.Checkpoint	Specifies the file name extension for checkpoint files. Many of the Java ingestion components create checkpoint files as an aid to recovery when restarted after exiting prematurely.	.cp
FileExtension: Specifies properties used to define extensions for various types of files.	FileExtension.Temporary	Specifies the file name extension for temporary files that Java ingestion components create.	.tmp
FileExtension: Specifies properties used to define extensions for various types of files.	FileExtension.Error	Specifies the file name extension for error files that Java ingestion components create.	.err
FileExtension: Specifies properties used to define extensions for various types of files.	FileExtension.Data	Specifies the file name extension for input data files that the Oracle client submits. The default value of *.dat*is in accordance with the DIS.	.dat

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
Security: Specifies properties used to produce security reference data.	Security.AdditionalColumns	Specifies additional columns of data that ingestion components must populate when manufacturing security records.	SCRTY_SHRT_NM, SCRTY_ISIN_ID, PROD_CTGRY_CD, PROD_TYPE_CD, PROD_SUB_TYPE_CD
Symbol: Specifies properties used for looking up security reference data by security short name.	Symbol.DbTableName	Specifies the name of the database table to use when looking up security records by security short name.	SCRTY
Symbol: Specifies properties used for looking up security reference data by security short name.	Symbol.KeyColumn	Specifies the column name to use when looking up security records by security short name.	SCRTY_SHRT_NM
Symbol: Specifies properties used for looking up security reference data by security short name.	Symbol.ValueColumn	Specifies the column to use for retrieving the Behavior Detection assigned identifier for a security.	SCRTY_INTRL_ID
Symbol: Specifies properties used for looking up security reference data by security short name.	Symbol.Category	Specifies the category of data for the security table. The category is a key for mapping to the database schema in which the security table resides.	BUSINESS
SecurityISIN: Specifies properties used for looking up security ISINs.	SecurityISIN.DbTableName	Specifies the name of the table to use when looking up a security using the Behavior Detection assigned security identifier.	SCRTY
SecurityISIN: Specifies properties used for looking up security ISINs.	SecurityISIN.KeyColumn	Specifies the column name to use when looking up security records by Behavior Detection assigned security identifier.	SCRTY_INTRL_ID
SecurityISIN: Specifies properties used for looking up security ISINs.	SecurityISIN.ValueColumn	Specifies the column to retrieve when looking up a security using the Behavior Detection assigned security identifier.	SCRTY_ISIN_ID

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
SecurityISIN: Specifies properties used for looking up security ISINs.	SecurityISIN.Category	Specifies the category of data in which the security table resides. The category is a key for mapping to the database schema in which the security table resides.	BUSINESS
FDT: Specifies properties used to configure the FDT component.	FDT.NumberOfThreads. Value	Specifies the number of worker threads that the FDT uses when processing data.	4
FDT: Specifies properties used to configure the FDT component.	FDT.LowerDisplayLimit.V alue	Specifies the quantity below which orders are exempt from display.	100
FDT: Specifies properties used to configure the FDT component.	FDT.UpperDisplayLimit. Value	Specifies the quantity above which orders are exempt from display.	10000
FDT: Specifies properties used to configure the FDT component.	FDT.OrderPriceLimit.Val ue	Specifies the dollar value above which orders are exempt from display.	200000
FDT: Specifies properties used to configure the FDT component.	FDT.SequenceBatchSize .OrderEvent	Specifies the batch size when retrieving sequence IDs for OrderEvent records (during end-of-day processing).	1000
FDT: Specifies properties used to configure the FDT component.	FDT.SequenceBatchSize .Order	Specifies the batch size when retrieving sequence IDs for Order records.	10000
FDT: Specifies properties used to configure the FDT component.	FDT.SequenceBatchSize .Trade	Specifies the batch size when retrieving sequence IDs for Trade records.	10000
FDT: Specifies properties used to configure the FDT component.	FDT.SequenceBatchSize .Execution	Specifies the batch size when retrieving sequence IDs for Execution records.	10000
FDT: Specifies properties used to configure the FDT component.	FDT.SequenceBatchSize .DerivedTrad e	Specifies the batch size when retrieving sequence IDs for DerivedTrade records.	10000
FDT: Specifies properties used to configure the FDT component.	FDT.MarketDataSource. Value	Specifies the source of market data. Valid values are File for file based access or RMI for access using an RMI server (not recommended for performance reasons).	File

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
FDT: Specifies properties used to configure the FDT component.	FDT.CalculateDisplayability.Value	Specifies whether to calculate displayability states.	FALSE
FDT: Specifies properties used to configure the FDT component.	FDT.ExplainableCancelCodes.Value	Specifies a comma-separated list of explainable cancellation codes.	
FDT: Specifies properties used to configure the FDT component.	FDT.BufferSize.value	Allows an override to the BufferSize.value property for FDT.	
FDT: Specifies properties used to configure the FDT component.	FDT.LookForFutureEventTimes. value		
FDT: Specifies properties used to configure the FDT component.	FDT.UsePrevailingSale.value	Specifies whether to use the prevailing reported market sales price as an execution's expected print price when no comparable market sales occur during the order's marketable periods.	FALSE
FDT: Specifies properties used to configure the FDT component. Data Management uses these parameters when calculating the expected print price for executions. A reported market sale is comparable to an execution when its size is in the same tier.	FDT.ExecutionSizeThresholds. FirstTierMax	Specifies the maximum size for the first tier.	1000
FDT: Specifies properties used to configure the FDT component. Data Management uses these parameters when calculating the expected print price for executions. A reported market sale is comparable to an execution when its size is in the same tier.	FDT.ExecutionSizeThresholds. SecondTierMax	Specifies the maximum size for the second tier.	5000

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
FDT: Specifies properties used to configure the FDT component.Data Management uses these parameters when calculating the expected print price for executions. A reported market sale is comparable to an execution when its size is in the same tier.	FDT.ExecutionSizeThresholds. ThirdTierMax	Specifies the maximum size for the third tier. Any size bigger than this value is considered part of the fourth tier.	10000
FDT: Specifies properties used to configure the FDT component. Data Management uses these parameters when calculating the marketable time with reasonable size attributes for an order. Processing divides orders into small, medium, and large based on their remaining unit quantities.	FDT.OrderSizeMarketability. MaxSmallSize	Specifies the maximum size for an order to be considered small.	1000
FDT: Specifies properties used to configure the FDT component. Data Management uses these parameters when calculating the marketable time with reasonable size attributes for an order. Processing divides orders into small, medium, and large based on their remaining unit quantities.	FDT.OrderSizeMarketability. MaxMediumSize	Specifies the maximum size for an order to be considered medium.	5000

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
FDT: Specifies properties used to configure the FDT component. Data Management uses these parameters when calculating the marketable time with reasonable size attributes for an order. Processing divides orders into small, medium, and large based on their remaining unit quantities.	FDT.OrderSizeMarketability.SmallMinPercentAtBest	Specifies the minimum percent of a small order's remaining unit quantity that must be available at the best price for execution to be considered reasonable. The minimum percentage value must be represented in its decimal equivalent (for example 1.0 = 100%).	1.0
FDT: Specifies properties used to configure the FDT component. Data Management uses these parameters when calculating the marketable time with reasonable size attributes for an order. Processing divides orders into small, medium, and large based on their remaining unit quantities.	FDT.OrderSizeMarketability.MediumMinPercentAtBest	Specifies the minimum percent of a medium order's remaining unit quantity that must be available at the best price for execution to be considered reasonable. The minimum percentage value must be represented in its decimal equivalent (for example 1.0 = 100%).	1.0
FDT: Specifies properties used to configure the FDT component. Data Management uses these parameters when calculating the marketable time with reasonable size attributes for an order. Processing divides orders into small, medium, and large based on their remaining unit quantities.	FDT.OrderSizeMarketability.LargeMinPercentAtBest	Specifies the minimum percent of a large order's remaining unit quantity that must be available at the best price for execution to be considered reasonable. The minimum percentage value must be represented in its decimal equivalent (for example 1.0 = 100%).	1.0
FDT: Specifies properties used to configure the FDT component.	FDT.TradePurposeFilter.value	Specifies a comma-separated list of trade purpose codes. Processing does not consider trades with one of these purpose codes in firm reference price derivations.	IFADM, OFEA, CONB, CLNT, BTBX

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
FDT: Specifies properties used to configure the FDT component.	FDT.RunBatchesSeparately.val ue	Specifies whether the FDT treats batches as distinct from one another. TRUE: Three defined batches originate from different geographical areas in which the data in each batch does not overlap (that is, an execution in batch A does not occur against an order in batch B). FALSE: Processing does not separate data in each batch into a distinct time interval (that is, an event in batch A occurred at 10am and an event in batch B occurred at 9am, and batch B arrived after batch A).	TRUE
FDT: Specifies properties used to configure the FDT component.	FDT.RegNMSExceptionCodes	Identifies the Order Handling Codes that should be considered as Reg NMS executions.	ISO, BAP, BRD, BOP, SOE, SHE
FDT: Specifies properties used to configure the FDT component.	FDT.TreatLostEventsAsErrors. value	Identifies whether lost events found by the FDT (see Rejection During the Transformation Stage, for a discussion of lost events) should be treated as errors (TRUE) or as lost events to be read in on the next run of FDT (false).	TRUE
FDT: Specifies properties used to configure the FDT component.	FDT.OpenOrderFileExpected.value	Identifies whether an OpenOrder file will be provided by the client during an end of day batch (TRUE) or whether it will not be provided (FALSE).	TRUE
FDT: Specifies properties used to configure the FDT component.	FDT.NonExecutionTradePurpose Codes.value	Specifies a comma-separated list of trade purpose codes. For Trade Execution records that refer to an Order and have one of these codes, the FDT will create a Trade record rather than an Execution record.	CLNT, BTBX

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
FDT: Specifies properties used to configure the FDT component.	FDT.EnableMIFID.value	Identifies whether MiFid related data will be provided (TRUE) or not (FALSE).	FALSE
FDT: Specifies properties used to configure the FDT component.	FDT.IgnoreFutureMarket Refs.value	Identifies whether the FDT will use Reported Market Sales records that occur later in time than a given trade when calculating the market reference price for that trade (FALSE) or not (TRUE).	FALSE
FDT: Specifies properties used to configure the FDT component.	FDT.MaxFutureMarketR efCompTi me.value	Specifies the number of seconds from the time a trade occurs during which any reported sales records cannot have the same price and quantity as the given trade to be considered as a market reference price. -1 means that this condition will not apply, 0 means the condition applies to reported sales with the same time, 5 means the condition applies to reported sales within 5 seconds of the trade, and so on. This parameter is only used if FDT.IgnoreFutureMarket Refs.value= FALSE.	-1

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
FDT: Specifies properties used to configure the FDT component. The next four parameters are used to generate records in the TRADE_TRXN_CORRECTION table, which record when a correction to a field of an execution, trade, or order occurs. The fields to be checked for corrections should be specified in a comma separated list of business field names. Business field names can be found in the corresponding XML data map file in the datamaps directory	FDT.DeriveCorrectionFields.T rade	Specifies which fields of a trade are monitored for corrections.	UnitQuantity, PriceIssuing
FDT: Specifies properties used to configure the FDT component. The next four parameters are used to generate records in the TRADE_TRXN_CORRECTION table, which record when a correction to a field of an execution, trade, or order occurs. The fields to be checked for corrections should be specified in a comma separated list of business field names. Business field names can be found in the corresponding XML data map file in the datamaps directory	FDT.DeriveCorrectionFields.E xecution	Specifies which fields of an execution are monitored for corrections.	UnitQuantity, PriceIssuing

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
FDT: Specifies properties used to configure the FDT component. The next four parameters are used to generate records in the TRADE_TRXN_CORRECTION table, which record when a correction to a field of an execution, trade, or order occurs. The fields to be checked for corrections should be specified in a comma separated list of business field names. Business field names can be found in the corresponding XML data map file in the datamaps directory	FDT.DeriveCorrectionFields.DerivedTrade	Specifies which fields of a derived trade are monitored for corrections.	YieldPercentage, YieldMethodCode
FDT: Specifies properties used to configure the FDT component. The next four parameters are used to generate records in the TRADE_TRXN_CORRECTION table, which record when a correction to a field of an execution, trade, or order occurs. The fields to be checked for corrections should be specified in a comma separated list of business field names. Business field names can be found in the corresponding XML data map file in the datamaps directory	FDT.DeriveCorrectionFields.Order	Specifies which fields of an order are monitored for corrections.	LimitPriceIssuing, UnitQuantity
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.Default.ArchiveFlag	Specifies whether to archive data files. The system copies input files to the backup directory (TRUE) or deletes input files (FALSE).	TRUE

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.Default.ErrorLimit	Specifies the percentage of invalid records to allow before exiting with an error. For example, a value of 10 allows 10 percent of records to be invalid before exiting with an error. A value of 0 allows no invalid records. A value of 100 allows all invalid records.	100
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.Default.TargetDir	Specifies the directory in which to place the resulting output file. If this is blank (the default), output files reside in the corresponding load directory (a subdirectory of market/load or firm/ loaddepending on the schema of the data being processed).	
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.Default.SequenceB atchSiz e	Specifies the batch size when retrieving sequence IDs.	100000
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.Default.AdditionalO utput	Specifies a directory to contain the output file in addition to the target directory.	
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.Default.DoFileLook ups	Specifies whether to do reference data lookups for fields that arrive as part of an input file (TRUE) or not do them (FALSE).	FALSE
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.Default.DiscardLook upFai lures	Specifies whether to discard records that fail a reference data lookup (TRUE) or just log a message (FALSE).	FALSE
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.Default.ValidatorCla ss	Specifies the Java class used to validate records of a given data type. Use of subclasses occurs when the general functionality of AbstractValidator is not enough for a given data type.	AbstractValidator

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.Default.AdapterClass	Specifies the Java class used to process records of a given data type. Use of subclasses occurs when the general functionality of BaseFileAdapter is not enough for a given data type.	BaseFileAdapter
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.Default.NumberOfThreads	Specifies the number of worker threads to be used when Pre-processing a file	2
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.Default.BufferSize	Allows an override to the BufferSize.valueproperty for the XDP.	100
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.Default.InputFileCharset	Specifies the character set of the DIS input files provided by the client. Currently, the only supported character sets are those that are compatible with ASCII.	UTF8
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.Default.SupplementalType	Specifies an additional file type that a given XDP will create when it processes a file of the given type.	TrustedPairMember
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.Account.DeriveAccountToP eerGroup	When processing Account records, specifies whether to derive an AccountToPeerGroup record if the AccountPeerGroupIdentifier field is populated.	

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.EmployeeTradingRestricti on.DescendOrgTree	If an Employee Trading Restriction record contains an Organization Identifier, then it specifies: • Whether to create Employee Trading Restriction records for all employees in the organization and all the related child organizations defined in the Organization Relationship file (TRUE) • l or • Whether to create records onlyfor employees in the specified organization (False).	FALSE
XDP: Specifies properties used to configure the Pre-processor (XDP) component.	XDP.<Data Type>.<Property>	Overrides the given property for the given Pre-processor instance.	
XDL: Specifies properties used to configure the Data Loader (XDL) component.	XDL.Default.FullRefresh	Is valid for data types that have a load operation of *Overwrite* as defined in the DIS. This parameter specifies replacement of the entire table (TRUE) or provision of deltas (FALSE).	TRUE
XDL: Specifies properties used to configure the Data Loader (XDL) component.	XDL.Default.DataFileExt s	Specifies the possible file extensions for an input file.	.XDP, .FDT, .MDT, .XDT
XDL: Specifies properties used to configure the Data Loader (XDL) component.	XDL.Default.CommitSize	Specifies the number of records to update or insert before committing (not used when Direct=TRUE).	500
XDL: Specifies properties used to configure the Data Loader (XDL) component.	XDL.Default.ErrorLimit	Specifies the number of rejected records to allow before exiting with an error. If left blank (the default), processing sets no limit.	

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
XDL: Specifies properties used to configure the Data Loader (XDL) component.	XDL.Default.DbErrorCodes	Specifies a comma-separated list of database vendor-specific error codes that indicate data level errors , such as data type and referential integrity. This results in rejection of records with a warning instead of a fatal failure.	1, 1400, 1401, 1407, 1438, 1722, 1840,1841 2291, 2359, 1839,1847,12899
These properties apply only to the Oracle adapter.	XDL.Default.MaxBindSize	Specifies the maximum number of bytes (integer) to use in the bind array for loading data into the database.	4194304
These properties apply only to the Oracle adapter.	XDL.Default.Direct	Specifies whether to use direct path loading (TRUE) or conventional path loading (FALSE).	FALSE
These properties apply only to the Oracle adapter.	XDL.Default.Parallel	Specifies whether a direct path load will be donein parallel (TRUE). This will be the case when multiple loaders for the same data type are run in parallel, such as with multiple ingestion instances.	FALSE
These properties apply only to the Oracle adapter.	XDL.Default.Unrecoverable	Specifies whether a direct path load does not use redo logs (TRUE) or uses redo logs (FALSE).	FALSE
These properties apply only to the Oracle adapter.	XDL.Default.Partitioned	Specifies whether a direct path load uses the current date partition (TRUE) or any partition (FALSE).	FALSE
These properties apply only to the Oracle adapter.	XDL.Default.SkipIndexes	Specifies whether a direct path load skips index maintenance (TRUE) or maintains indexes (FALSE). If set to TRUE, rebuilding of indexes must occur after running the Data Loader.	FALSE
These properties apply only to the Oracle adapter.	XDL.Default.SkipIndexErrorCo de	Specifies a database vendor specific error code that occurs in the log file when skipping indexes.	26025

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
These properties apply only to the Oracle adapter.	XDL.Default.IndexParalle lLev el	Specifies the parallel level of an index rebuild (that is, number ofconcurrent threads for rebuilding an index).	4
These properties apply only to the Oracle adapter.	XDL.Default.DoAnalyze	Specifies whether to run a stored procedure to analyze a database table after loading data into it.	FALSE
These properties apply only to the Oracle adapter.	XDL.Default.DoImportSt atisti cs	Specifies whether to run a stored procedure to import statistics for a database table after loading data into it.	FALSE
These properties apply only to the Oracle adapter.	XDL.Default.ImportStatis tics Type	Specifies the type of statistic import to perform if DoImportStatistics has a value of True.	DLY_POST_LOAD
These properties apply only to the Oracle adapter.	XDL.Default. ImportStatisticsLogDir	Saves the directory to which the stored procedure writes the log file if DoImportStatistics has a value of True. This log directory must reside on the server that hosts the database.	/tmp
These properties apply only to the Oracle adapter.	XDL.Default.TableDoesN otExis tErrorCode	Specifies the database error code that indicates a database table does not exist.	942
These properties apply only to the Oracle adapter.	XDL.Default.UseUpdate Loader	Specifies whether JDBC updates should be used instead of a delete/insert when updating a database record. This is only valid for data types that have a load operation of Update.	FALSE
These properties apply only to the Oracle adapter.	XDL.<Data Type>.<Property>	Overrides the specified property for a given Data Loader instance.	
IMC: Specifies properties for configuring the Directory Cleanup (IMC) component.	Directory[1..N].Name	Identifies the directory to clean up. The system removes date subdirectories (in *YYYYMMDD* format) from this directory.	../data/backup
IMC: Specifies properties for configuring the Directory Cleanup (IMC) component.	Directory[1..N].DaysToK eep	Specifies the number of days to keep for this directory. The system does not delete date subdirectories with the latest dates.	3

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
DBUtility: Specifies properties used to configure various utility processes. Valid utility names are SecurityFirmDaily, AccountChangeLogSummary, CustomerChangeLogSummary, AccountToCustomerChangeLogSummary.	<UtilityName>.NumberofThread s	Specifiesthe number of worker threads that the give component uses when processing data.	4
DBUtility: Specifies properties used to configure various utility processes. Valid utility names are SecurityFirmDaily, AccountChangeLogSummary, CustomerChangeLogSummary, AccountToCustomerChangeLogSummary.	<UtilityName>.SequenceBatchs ize	Specifies the batch size when retrieving sequence IDs for records generated by given component.	10000
Watch List Service: Specifies properties used to configure the Scan Watch List Web Service.	Timeout.value	Specifies how many seconds a call to the Watch List Service made through the scanWatchList.shscript will wait for the service request to finish. This value should be set to the longest wait time expected based on the volume of data and system configuration. Setting it very high will not affect performance since the call will return as soon as it is complete.	600
Watch List Service: Specifies properties used to configure the Scan Watch List Web Service.	Log.UseDateLog	Overrides the default Log.UseDateLog property.	FALSE
Watch List Service: Specifies properties used to configure the Scan Watch List Web Service.	WatchListScannerClass.value	Identifies the Java class used to scan a watch list for a given name.	MantasWatchListScanner

Table D-14 (Cont.) Data Ingest Properties

Type	Property Name	Description	Example
Watch List Service: Specifies properties used to configure the Scan Watch List Web Service.	NameMatcherClass.value	Identifies the Java class used to match a name against a list of names.	FuzzyNameMatcher
Watch List Service: Specifies properties used to configure the Scan Watch List Web Service.	FuzzyMatcher.SecondToPoll	Identifies the number of seconds to wait between querying the WATCH_LIST table for new names that are added by the Watch List Management Utility.	
Watch List Service: Specifies properties used to configure the Scan Watch List Web Service.	FuzzyMatcher.MaximumAddedNames	Identifies the maximum number of names that can be added to the Watch List Service after it is initialized. If additional names must be added, the service must be re- initialized.	

D.4.3.3 Data Ingest Custom XML Configuration File

Oracle clients can modify the DataIngest.xml file to override default settings that the system provides.

However, this file is subject to change in future OFSBD releases. Therefore, upon installation of a newer OFSBD version the client must reapply any modifications in the current DataIngest.xml file to the newer DataIngest.xml file.

To simplify this process, the DataIngestCustom.xml file is available for use. This file holds all site-specific changes to the DataIngest.xml file. The client can override any settings in DataIngest.xml by placing the modifications in DataIngestCustom.xml. After installing a newer OFSBD version, the client can copy the older DataIngestCustom.xml file to DataIngestCustom.xml in the new installation.

D.4.4 data Subdirectory

The data subdirectory within the ingestion_manager directory contains additional subdirectories for organizing Market data files and Oracle client data files.>

The system creates these files during the Pre-processing, transformation and data-loading stages of the ingestion process. The Market data and Oracle client data files appear in subdirectories that are indicative of the processing stages (or workflow steps) that the Data Management subsystem components perform. The following sections describe the contents of each subdirectory and the components that read or write to each subdirectory.

> **Note:**
>
> Processing date stamps should appear as YYYYMMDD for Data Management directories and subdirectories. The system provides this processing date to the set_mantas_date.sh shell script when starting the first batch for the day.

D.4.4.1 data/errors Subdirectory

The errors subdirectory within the data subdirectory stores error files that Data Management subsystem components create or move upon detection of errors during file processing.

The system places error files in subdirectories within the errors subdirectory. These error file subdirectories are name-based on the processing date for the files that they contain. The date has the format YYYYMMDD, where YYYY is the four-digit year, MM is the two-digit month, and DD is the two-digit day. The files in the errors subdirectory have the same name as the file in which the error was detected. However, the component that identified the errors appends its extension to the end of the file.

The following table identifies the error file signatures that each component can output to the errors subdirectory.

Table D-15 Error File Signatures Output by Component

Component	Error File
Pre-processor	`<data type>_*.XDP.err`
Data Loader	`<data type>_*.XDL.err`
FDT	`Order_*.FDT.err` `TradeExecution_*.FDT.err`

MDS	`InsideQuote_*.MDS.err`
	`MarketCenterQuote_*.MDS.err`
	`ReportedMarketSale_*.MDS.err`

The IMC utility, *runIMC.sh*, cleans up the errors subdirectory. The IMC's configuration file defines the number of days that error files age before their removal.

D.4.4.2 data/backup Subdirectory

The backup subdirectory stores files that Data Management subsystem components processed and require no further processing.

That is, they are considered to be in a final form after successful processing.

- Transformers back up files that they receive and create.
- Loaders back up files that they finished loading. Each file in the backup directory appears in a subdirectory with the date as its name. The name is in the format YYYYMMDD, where YYYY is the four-digit year, MM is the two-digit month, and DD is the two-digit day.

The IMC component, *runIMC.sh*, cleans up the backup subdirectory. The IMC's configuration file defines the number of days that backup files age before removal. The following table references the files that the system writes to the backup subdirectory, by component.

Table D-16 Backed Up Files by Component

Component	Data Files
FDT	*.XDP
Data Loader	*.XDP, *.FDT

D.4.4.3 data/firm Subdirectory

The *firm* subdirectory within the *data* subdirectory contains the extract, transform and load subdirectories that correspond directly to the workflow steps that Firm data moves through during Data Management.

The following sections describe each subdirectory.

extract Subdirectory

The *extract* subdirectory within the *firm* subdirectory contains checkpoint data and working files for each Pre-processor during Pre-processing.

Each Pre-processor also maintains checkpoint files that enable it to recover after a failure and without the loss of data integrity; an FDT removes the files after it successfully Pre-processes its data. When finished, each Pre-processor moves its final Pre-processed files to either the *transform* subdirectory for processing by FDT, or to the *load* subdirectory for loading into the database.

The.XDP file type identifies files that the Pre-processor creates.

transform Subdirectory

The *transform* subdirectory within the *firm* subdirectory contains the FDT's checkpoint data and working files during transformation. When finished, the FDT moves its final transformed Firm data files to the *load* subdirectories for loading into the database. The system writes the transformed data to files and then moves the files to the *load* subdirectory. The.FDT file type identifies the files that the FDT creates.

The FDT also maintains several checkpoint files that allow it to recover after a failure, without the loss of data integrity.

load Subdirectory

The *load* subdirectory within the *firm* subdirectory contains additional subdirectories that contain Pre-processed and transformed Firm data that the system queues for loading into the database. Each loader component monitors its respective subdirectory (that is, data queue) looking for data to load into the database—a subdirectory exists for each kind of Oracle client data that processing loads into the database. After loading data files into the database, each loader moves the processed files to the backup subdirectory.

D.4.5 inbox Subdirectory

The *inbox* subdirectory within the *ingestion_manager* directory is an electronic mailbox or queue in which the Oracle client writes its data files for subsequent processing by Data Management subsystem Data Pre-processor components.

Each Market or Firm Data Pre-processor retrieves the file it is assigned to process from the *inbox* subdirectory and then moves the file to the appropriate *extract* subdirectory for Pre-processing. The DIS describes the naming convention and content of each data file that an Oracle client provides.

D.4.6 logs Subdirectory

The *logs* subdirectory contains a log file for each component running on a host computer.

Each log file in the *logs* subdirectory appears in a subdirectory with the date as its name, in the format **YYYYMMDD**, where `YYYY` is the four-digit year, `MM` is the two-digit month, and `DD` is the two-digit day. The subdirectory's date is based on the processing date for data to which the log files pertain.

The IMC utility, *runIMC.sh*, cleans up the *logs* subdirectory. The IMC utility's configuration file defines the number of days that log files age before their removal. The following table identifies log files for each component, based on the file name's prefix.

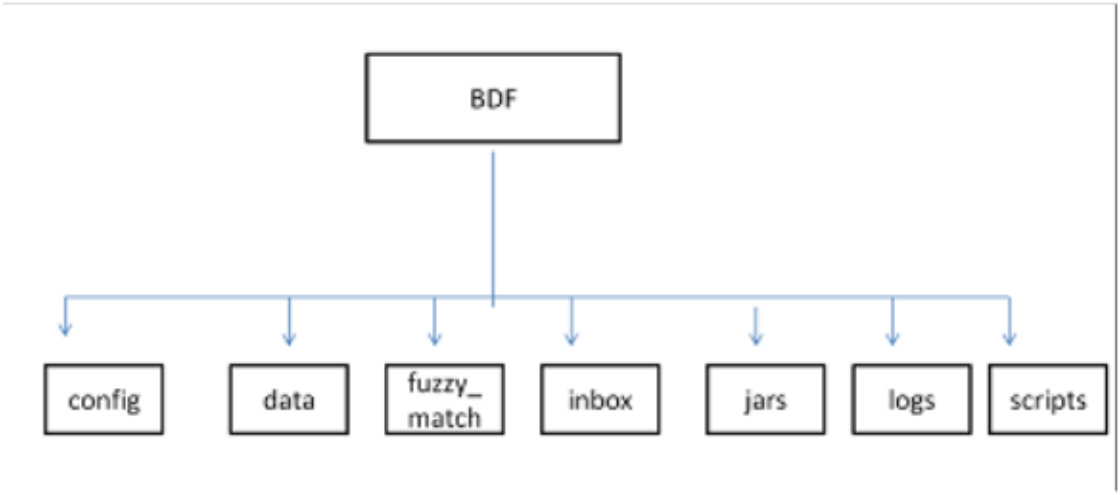
Table D-17 Log Files Output by Component

Prefix	Component
XDP	Pre-processor
XDL	Data loader
FDT	File Data Transformer
IMC	IMC

D.5 BD Directory Structure

The BD Datamap component is organized as subdirectories below the *<OFSAAI Installed Directory>/bdf* file.

Figure D-2 BD Subsystem Directory Structure



The following table provides details about each subdirectory.

Table D-18 Directory Structure Description

Directory Name	Description
logs	Log files containing status and error messages produced by BD components
config	Files used to configure BD components
config/datamaps	XML files containing data map definitions for individual BD components
jars	Java Archive (JAR) files used to run BD components
data/errors	Files containing error records produced by BD components
data/temp	Temporary files produced by BD components
inbox	Data files provided by the Oracle client in DIS format
fuzzy_match	C++ library files used for the purpose of fuzzy matching names
scripts	Shell scripts for running BD components, setting the environment, and changing passwords

The following sections describe the BD directory structure.

- scripts
- logs
- parameters
- config

D.5.1 scripts Folder

The scripts folder contains the env.sh and execute.sh files.

The *env.sh* file sets ups the shell environment of BD components, while the *execute.sh* file executes BD components.

Running these files in the BD subsystem improves performance time.

```
<OFSAAI Installed Directory>/bdf/scripts/execute.sh <component>
<OFSAAI Installed Directory>/bdf/scripts/execute.sh CorrespondentBankProfile
```

Component in this document means a batch process which is part of the BD Datamap subsystem. For the most part, these components will refer to XML data maps. For example, the *AccountProfile_Balance* component refers to the *AccountProfile_Balance.xml* data map.

D.5.2 logs Folder

The *log* file has information about the warnings, errors, and status of the component.

Additional information can be obtained from a component by turning on diagnostic logging. This can be done by setting the *Log.DIAGNOSTIC.Enabled* parameter to **true**. In a production environment, this should be left as **false** and only changed to true when debugging errors or performance issues.

Log files for each component are written to a log file named for the component inside a subdirectory of the logs directory named for the current processing date in *YYYYMMDD* format.

```
 <OFSAAI Installed Directory>/bdf/logs/<processing date>/<component>.log
<OFSAAI Installed Directory>/bdf/logs/20130313/CorrespondentBankProfile.log
```

When an external table is used as the DIS file loading mechanism, there are additional log files containing log output from the external table utility. The log files are named the same as the external table being loaded. The name of the external table is the name of the table being loaded with a prefix of *DIS_*.

```
<OFSAAI Installed Directory>/bdf/logs/20130313/DIS_ACCT.log
```

D.5.3 parameters Folder

Parameters in BD Datamaps are specified as elements in an XML file.

The XSD containing a description of these elements can be found in the `<OFSAAI Installed Directory>/bdf/config/ParameterSet.xsd` directory.

The Parameter element defines a parameter and its value, and contains the following attributes:

- name - The name of the parameter.
- type - The data type of the parameter. Valid values are STRING, REAL, INTEGER, BOOLEAN, FILE, and CLASS.
- value - The value of the parameter, which must map the type of the parameter.
- list - A boolean value specifying that the value is a single value (false - the default) or a comma separated list of values (true).

```
<Parameter name="MinimumGeographyRisk" type="INTEGER" value="0"/>
<Parameter name="InternalAccountCodeList" type="STRING" value="IA,GL"
list="true"/>
```

If the value of the parameter is a string containing characters which are not allowed in an XML attribute, then a CDATA element can be used as the element's text.

```
<Parameter name="PassThruExpressionSeparators" type="STRING">
<![CDATA[~: \t/#-]]>
</Parameter>
```

Parameters in the main BDF.xml file should not be modified. Instead, any customizations to parameter values should be placed in the `<OFSAAI Installed Directory>/bdf/config/custom/BDF.xml` file. Parameters can be overridden at the component level by placing them in the `custom/<component>.xml` file. Also, parameters can be overridden on the command line by passing the parameter name and value as parameters to the *execute.sh* script after the component name:

```
<OFSAAI Installed Directory>/bdf/scripts/execute.sh <component> [parameter
name=value]*
<OFSAAI Installed Directory>/bdf/scripts/execute.sh CorrespondentBankProfile
NumberOfThreads=4
```

When a given parameter is read by a component, the order of precedence for where the parameter value is taken from is as follows:

```
command line
<OFSAAI Installed Directory>/bdf/config/custom/<component>.xml
<OFSAAI Installed Directory>/bdf/config/<component>.xml
<OFSAAI Installed Directory>/bdf/config/custom/BDF.xml
<OFSAAI Installed Directory>/bdf/config/BDF.xml
```

D.5.4 config Folder

The config subdirectory contains configuration files.

- *<OFSAAI Installed Directory>/bdf/config/BDF.xml* contains all default product configuration parameters. It should not be modified.
- *<OFSAAI Installed Directory>/bdf/config/install/BDF.xml* contains all configuration parameters set at installation time (refer to the *Installation Guide* for more information).
- *<OFSAAI Installed Directory>/bdf/config/custom/BDF.xml* contains any product configuration parameters that have been overridden for this installation. It is initially empty. Any changes to default product configuration parameters should be put here.

Individual BD components can have their own configuration file which overrides default product parameters. These files would be named using the following format: `<OFSAAI Installed Directory>/bdf/config/<component>.xml`

`<OFSAAI Installed Directory>/bdf/config/CorrespondentBankProfile.xml`

Component configuration files in this directory are part of the product and should not be modified. If any parameters must be overridden at the individual component level, the component configuration file should be created in *<OFSAAI Installed Directory>/bdf/config/custom.*

- The *datamaps* subdirectory contains XML files holding the data map definitions for BD components.
- The *derivations* subdirectory contains SQL derivations for individual fields.
- The *queries* subdirectory contains SQL queries for individual data maps.

D.5.4.1 BDF.xml Configuration Parameters

The following table describes the BD properties configurations mentioned in the *<OFSAAI Installed Directory>/bdf/config/BDF.xml* file.

Table D-19 BDF.xml File Configuration Parameters

Type	Parameter Name	Description	Example
MISCELLANEOUS	NumberOfThreads	The number of worker threads used by some BD components	4
MISCELLANEOUS	SequenceBatchSize	The batch size when retrieving sequence IDs for new records	100000
MISCELLANEOUS	SourceSystem	he default value for source system when one is not provided	MTS

Table D-19 (Cont.) BDF.xml File Configuration Parameters

Type	Parameter Name	Description	Example
MISCELLANEOUS	Currency	The default value for issuing currency when one is not provided	USD
MISCELLANEOUS	Separator	The delimiter that separates fields in data file records.	~
DB: Parameters related to database access.	DB.Connection.Driver	The JDBC driver class name.	oracle.jdbc.OracleDriver
DB: Parameters related to database access.	DB.Timeout	The number of seconds to wait before timing out on a database connection attempt.	10
DB: Parameters related to database access.	DB.NumRetries	The maximum number of times to attempt to connect to a database before failing.	5
DB: Parameters related to database access.	DB.MaxNumberOfDeadlocks	The maximum number of times a deadlock is encountered during a JDBC insert or update operation, before an error is generated.	10
Directory: Parameters used to define directory locations.	Directory.Inbox	The input directory where the Oracle client will write DIS files. Date subdirectories will be created in this directory where these files will be archived	../inbox
Directory: Parameters used to define directory locations.	Directory.InternalData	The directory where files generated by BD components will reside. This includes log files, error files, and any temporary processing files.	..
Log: Parameters used to configure the common logging module	Log.Format	Identifies the log formatting string.	%d [%t] %p - %m%n
Log: Parameters used to configure the common logging module	Log.UseDefaultLog	Specifies whether the system uses the default log file for a component. The default log file has the name of the component and resides in a date subdirectory of the logs directory (in YYYYMMDD format).	true
Log: Parameters used to configure the common logging module	Log.SysLogHostName	The host name of syslog for messages sent to syslog.	hostname

Table D-19 (Cont.) BDF.xml File Configuration Parameters

Type	Parameter Name	Description	Example
Log: Parameters used to configure the common logging module	Log.SMTPHostName	The host name of the SMTP server for messages that processing sends to an e-mail address.	hostname
Log: Parameters used to configure the common logging module	Log.MaxSize	The maximum size (in MB) of a log file before the system creates a new log file.	2000MB
Log: Parameters used to configure the common logging module	Log.MaxIndex	If a log file exceeds Log.MaxSize, this will be the maximum number of additional log files that are created (Component.log.1, Component.log.2, etc).	10
Log: Parameters used to configure the common logging module	Log.TRACE.Enabled	Indicates that trace logging is not enabled; true indicates enabling of trace logging.	false
Log: Parameters used to configure the common logging module	Log.TRACE.Location	Specifies additional locations to send TRACE log messages to, other than the default BD log file (logs/ YYYYMMDD/ Component.log). If the value is not provided, considers the default BD log location.	false
Log: Parameters used to configure the common logging module	Log.TRACE.Synchronous	Specify whether logging for a particular level should be performed synchronously or asynchronously.	false
Log: Parameters used to configure the common logging module	Log.DIAGNOSTIC.Enabled	DIAGNOSTIC logging is used to log database statements and will slow down performance. Make it true if needed.	false
Log: Parameters used to configure the common logging module	Log.DIAGNOSTIC.Location	Additional locations to send DIAGNOSTIC log messages to, other than the default BD log file (logs/YYYYMMDD/ Component.log). If the value is not provided, considers the default BD log location.	
Log: Parameters used to configure the common logging module	Log.DIAGNOSTIC.Synchronous	Specify whether logging for a particular level should be performed synchronously or asynchronously.	false

Table D-19 (Cont.) BDF.xml File Configuration Parameters

Type	Parameter Name	Description	Example
Log: Parameters used to configure the common logging module	Log.NOTICE.Enabled	Indicates enabling of notice logging; false indicates that notice logging is not enabled.	true
Log: Parameters used to configure the common logging module	Log.NOTICE.Location	Specifies additional locations to send NOTICE log messages to, other than the default BD log file (logs/YYYYMMDD/Component.log). If the value is not provided, considers the default BD log location.	
Log: Parameters used to configure the common logging module	Log.NOTICE.Synchronous	Specify whether logging for a particular level should be performed synchronously or asynchronously.	false
Log: Parameters used to configure the common logging module	Log.WARN.Enabled	Indicates enabling of warning logging; false indicates that warning logging is not enabled.	true
Log: Parameters used to configure the common logging module	Log.WARN.Location	Specifies additional locations to send WARN log messages to, other than the default BD log file (logs/YYYYMMDD/Component.log).	
Log: Parameters used to configure the common logging module	Log.WARN.Synchronous	Specify whether logging for a particular level should be performed synchronously or asynchronously.	false
Log: Parameters used to configure the common logging module	Log.FATAL.Enabled	Indicates enabling of Fatal logging; false indicates that fatal logging is not enabled.	true
Log: Parameters used to configure the common logging module	Log.FATAL.Location	Specifies additional locations to send FATAL log messages to, other than the default BD log file (logs/YYYYMMDD/Component.log).	
Log: Parameters used to configure the common logging module	Log.FATAL.Synchronous	Specify whether logging for a particular level should be performed synchronously or asynchronously.	false

Table D-19 (Cont.) BDF.xml File Configuration Parameters

Type	Parameter Name	Description	Example
Load: Parameters used to configure common Loading data	Load.FullRefresh	For DIS files defined as Overwrite, whether to fully replace FSDM tables with the contents of the DIS file (true) or to treat the DIS file as a delta (false)	True
Load: Parameters used to configure common Loading data	Load.BatchSize	The batch size when loading data.	5000
Load: Parameters used to configure common Loading data	Load.Direct	Specifies whether to use direct path loading (TRUE) or conventional path loading (FALSE).	false
Load: Parameters used to configure common Loading data	Load.Unrecoverable	Specifies whether a direct path load does not use redo logs (TRUE) or uses redo logs (FALSE).	false
Load: Parameters used to configure common Loading data	Load.Partitioned	Specifies whether a direct path load uses the current date partition (TRUE) or any partition (FALSE).	false
Load: Parameters used to configure common Loading data	Load.SkipIndexes	Specifies whether a direct path load skips index maintenance (TRUE) or maintains indexes (FALSE). If set to TRUE, rebuilding of indexes must occur after running the DataMap XML.	false
Load: Parameters used to configure common Loading data	Load.DoAnalyze	Specifies whether to run a stored procedure to analyze a database table after loading data into it.	true
Load: Parameters used to configure common Loading data	Load.AnalyzeType	Specifies the type of analyze statistics has to perform if DoAnalyze has a value of True.	DLY_POST_L OAD
Load: Parameters used to configure common Loading data	Load.LogRecordInterval	Specifies how often to log a message saying how many records a particular thread has inserted/updated,	1000

Table D-19 (Cont.) BDF.xml File Configuration Parameters

Type	Parameter Name	Description	Example
Load: Parameters used to configure common Loading data	Load.MaxErrorRate	Specifies the percentage of invalid records to allow before exiting with an error. For example, a value of 10 allows 10 percent of records to be invalid before exiting with an error. A value of 0 allows no invalid records. A value of 100 allows all invalid records.	100
Load: Parameters used to configure common Loading data	Load.RecordQueueSize	Specifies the number of records the query reader thread will write to a database writer thread queue before waiting for the reader thread to catch up. Higher values will require more memory usage.	100
Load: Parameters used to configure common Loading data	Load.SkipIndexesErrorCode	Specifies a database error code that occurs in the log file when skipping index maintenance.	26025
Load: Parameters used to configure common Loading data	Load.IndexParallelLevel	Specifies the parallel level of an index rebuild (that is, number of concurrent threads for rebuilding an index).	1
Load: Parameters used to configure common Loading data	Load.DataErrorCodes	Specifies a comma-separated list of database error codes that indicate data level errors , such as data type and referential integrity. This results in rejection of records with a warning instead of a fatal failure.	1,1400,1401,1407,1438,1722,1840,1841,2291,2359,1839,1847,12899
Load: Parameters used to configure common Loading data	Load.ParallelLevel	Specifies the level of parallelization to apply when loading data from a set of source tables to a target table.	8
Load: Parameters used to configure common Loading data	Load.WriteErrorFiles	Whether to check a DIS file for errors before loading as an external table (true) or not (false)	True

Table D-19 (Cont.) BDF.xml File Configuration Parameters

Type	Parameter Name	Description	Example
DIS: Parameters related to processing DIS files	DIS.Source	The mechanism used to load DIS data. FILE: DIS files will be provided and will be loaded using SQL*Loader processes running on the application server. FILE-EXT: DIS files will be provided and will be loaded using external tables with the DIS files accessed directly by the database. FSDW: DIS data will be obtained from database tables in the FSDW.	FILE
DIS: Parameters related to processing DIS files	DIS.ArchiveFlag	Whether DIS files will be archived to a date subdirectory (true) or not (false).	True
DIS: Parameters related to processing DIS files	DIS.BufferSize	The size in KB of the byte buffer used to read in DIS file records.	100
DIS: Parameters related to processing DIS files	DIS.InputFileCharset	The character set of the DIS files. Note that output data is always written in UTF8, this parameter just allows the DIS files to be in a different character set.	
DIS: Parameters related to processing DIS files	DIS.Default.Check.Requi remen t	Whether to check for mandatory fields on DIS records (true) or not (false).	True
DIS: Parameters related to processing DIS files	DIS.Default.Reject.Requi remen t	Whether to reject DIS records for failing a mandatory field check (true) or to log a warning and attempt to load the record (false).	True
DIS: Parameters related to processing DIS files	DIS.Default.Check.Doma in	Whether to check that a DIS field has a valid domain value (true) or not (false).	True
DIS: Parameters related to processing DIS files	DIS.Default.Reject.Doma in	Whether to reject DIS records that fail a domain check (true) or not (false).	True
DIS: Parameters related to processing DIS files	DIS.Default.Check.Lengt h	Whether a DIS field should be checked for a valid length (true) or not (false).	True

Table D-19 (Cont.) BDF.xml File Configuration Parameters

Type	Parameter Name	Description	Example
DIS: Parameters related to processing DIS files	DIS.Default.Reject.Length	Whether to reject DIS records that fail a length check (true) or not (false)	True
DIS: Parameters related to processing DIS files	DIS.Default.Check.Threshold	Whether a DIS field should be checked that it is within an acceptable threshold (i.e. greater than 0) (true) or not (false).	True
DIS: Parameters related to processing DIS files	DIS.Default.Reject.Threshold	Whether to reject DIS records that fail a threshold check (true) or not (false).	True
DIS: Parameters related to processing DIS files	DIS.Default.Check.Lookup	Not currently supported.	True
DIS: Parameters related to processing DIS files	DIS.Default.Reject.Lookup -	Not currently supported	True
Parameters used by queries defined in the data maps	MinimumGeographyRisk	Defines what is considered High Risk For the Account Profile attributes related to High Risk Geography , such as Incoming High Risk Wire Count. Processing compares this parameter using a strict greater-than operation.	0
Parameters used by queries defined in the data maps	AccountInactivityInMonths	Specifies the number of months that processing aggregated to determine whether an account is inactive. If the sum of trades and transactions over this number of months is <= 3, the account is considered inactive. This setting can impact the Escalation in Inactive Accounts scenario. The default value is six months.	6
Parameters used by queries defined in the data maps	TransactionsReversalLookbac kDays	This parameter controls how many days of transactions to look across. Verify whether the new data contains reversals of prior transactions.	7

Table D-19 (Cont.) BDF.xml File Configuration Parameters

Type	Parameter Name	Description	Example
Parameters used by queries defined in the data maps	LowPriceSecurityThreshold	Defines Low Priced in the base currency for the Account Profile attributes named Low-Priced Equity Range # Opening Trade Count. Processing compares the value of this parameter to the Trade table's Last Execution Price-Base.	5000
Parameters used by queries defined in the data maps	CommissionEquityPercentUp perLimit	Defines the upper limit for Commission Versus Average Daily Equity Percentage in Account Profile Calculation.	5
Parameters used by queries defined in the data maps	TurnOverRateUpperLimit	Defines the upper limit for Total Turnover Rate in Account Profile Calculation.	5
Parameters used by queries defined in the data maps	BankCodeListWithIA	Defines the List of Financial Institution Identifier Types, these are type of unique identifiers which are used to represent the financial institutions. This parameter also contains IA (Internal Account Identifier) to be used in datamaps and is mainly used in Correspondent Bank related datamap derivations. Below are the list of examples • BIC: SWIFT Bank Identifier Code(BIC) • CHU: CHIPS Participant UserIdentifier • CO: CorporateIdentifier • CHP: CHIPS ParticipantIdentifier • FED: Federal Reserve Routing (ABA)Number • CU: CustomerIdentifier • GL: General LedgerAccount • l IA: Internal AccountIdentifier	BIC,FED,CHP, CHU, DTC,CDL,EPN, KID, CBI,CSN,OTF, BLZ,I BAN,ABLZ,B SB,CP AP, SDIC, HEBIC, BCHH, NSC, IFSC, IDIC, PNCC, RCBIC, UKDSC, Swiss BC, Swiss SIC,IA

Table D-19 (Cont.) BDF.xml File Configuration Parameters

Type	Parameter Name	Description	Example
Parameters used by queries defined in the data maps	IdRiskWinLevel	Defines the Risk level to calculate Effective Risks for internal parties (Account/ Customer). For example: Account 1234 has an Effective Risk of 5, IdRiskWinLevel can be set by the client. If the party identifier effective risk is greater than the set IdRiskWinLevel, then the party identity risk wins compared to fuzzy matcher (Party Name Risk). If not, fuzzy matcher wins.	1
Parameters used by queries defined in the data maps	InternalAccountCodeList	Codes to define types of Internal Entities with client, for example: • IA: Internal AccountIdentifier • GL: General LedgerAccount	IA, GL
Parameters used by queries defined in the data maps	ExternalEntityCodeList	Codes to define types of External Entities with client, for example: • XA: External AccountIdentifier • CO: CorporateIdentifier • DL: DriverLicense • IBAN: International Bank AccountNumber	XA,CC,CO,DL, GM, GP,LE,MC,ND, NR, PP,SS,TX,AR, OT,IB AN
Parameters used by queries defined in the data maps	TrustedPairReviewReasonText 1	Defines the reason text1 for recommendation of canceling the Trusted Pair, due to increase in Risk of parties involved in trusted pair.	Risk of <Party1> increased from <A> to <b>
Parameters used by queries defined in the data maps	TrustedPairReviewReasonText 2	Defines the reason text2 for recommendation of canceling the Trusted Pair, due to increase in Risk of parties involved in trusted pair.	Risk of <Party2> increased from <C> to <D>
Parameters used by queries defined in the data maps	CorporateActionLookBackDay s	This parameter determines the how many days trades to look back from the Corporate Effective Date.	7

Table D-19 (Cont.) BDF.xml File Configuration Parameters

Type	Parameter Name	Description	Example
Parameters used by queries defined in the data maps	BankCodeList	Defines the List of Financial Institution Identifier Types, these are type of unique identifiers which are used to represent the financial institutions excluding Internal Account (IA). This parameter does not contain IA (Internal Account Identifier) to be used in datamaps and is typically used to derive financial institutions. Below are the list of examples • l BIC: SWIFT Bank Identifier Code(BIC) • l CHU: CHIPS Participant UserIdentifier • l CO: CorporateIdentifier • l CHP: CHIPS ParticipantIdentifier • l FED: Federal Reserve Routing (ABA)Number • l CU: CustomerIdentifier • l GL: General LedgerAccount	BIC,FED,CHP, CHU, DTC,CDL,EPN, KID, CBI,CSN,OTF, BLZ,IBAN,ABLZ,B SB,CP AP, SDIC, HEBIC, BCHH, NSC, IFSC, IDIC, PNCC, RCBIC, UKDSC,Swiss BC, Swiss SIC
Parameters used by queries defined in the data maps	DealNearTermMaturityD ays	Defines the maximum number of days between the End Date and Trade Date. This helps to calculate Structured Deals Initiated w/ Near-Term Exp. In Customer Profile/ Institutional Account Profile.	7

Table D-19 (Cont.) BDF.xml File Configuration Parameters

Type	Parameter Name	Description	Example
Parameters used by queries defined in the data maps	ProfitLossUpperLimit	Helps determine how much a security must move by the end of the day to be considered a win or loss. If the security moves by less than a specified percentage, processing does not count it either way. If it moves by this percentage or more, it counts as a win or a loss, depending on whether the movement was beneficial to the account that made the trade.	5
Parameters used by queries defined in the data maps	HouseholdTurnOverRate Uppe rLimit	Defines the upper limit for Total Turnover Rate in Household Profile Calculation.	10000
Parameters used by queries defined in the data maps	HouseholdCommissionE quity PercentUpperLimit	Defines the upper limit for Commission Versus Average Daily Equity Percentage in Account Profile Calculation.	10000
Parameters used by queries defined in the data maps	OptionTradeAmountRan ge1 OptionTradeAmountRan ge2 OptionTradeAmountRan ge3 OptionTradeAmountRan ge4 OptionTradeAmountRan ge5 OptionTradeAmountRan ge6	Define the lower bound of each range for the Account Profile attributes named Options Range # Opening Trade Count. Processing compares each parameter to the Trade table's Last Principal Amount- Base. Each range is from the lower bound entered here to the lower bound of the next range.	
Parameters used by queries defined in the data maps	EquityTradeAmountRang e1 EquityTradeAmountRang e2 EquityTradeAmountRang e3 EquityTradeAmountRang e4 EquityTradeAmountRang e5 EquityTradeAmountRang e6	Define the lower bound of each range for the Account Profile attributes named Equity Range # Opening Trade Count. Processing compares each parameter to the Trade table's Last Principal Amount- Base. Each range is from the lower bound entered here to the lower bound of the next range.	

Table D-19 (Cont.) BDF.xml File Configuration Parameters

Type	Parameter Name	Description	Example
Parameters used by queries defined in the data maps	LowPricedEquityTradeAmountRange1 LowPricedEquityTradeAmountRange2 LowPricedEquityTradeAmountRange3 LowPricedEquityTradeAmountRange4 LowPricedEquityTradeAmountRange5 LowPricedEquityTradeAmountRange6	Define the lower bound of each range for the Account Profile attributes named Low-Priced Equity Range # Opening Trade Count. Processing compares each parameter to the Trade table's Last Principal Amount-Base. Each range is from the lower bound entered here to the lower bound of the next range.	
Parameters used by queries defined in the data maps	MutualFundTradeAmountRange1 MutualFundTradeAmountRange2 MutualFundTradeAmountRange3 MutualFundTradeAmountRange4 MutualFundTradeAmountRange5 MutualFundTradeAmountRange6	Define the lower bound of each range for the Account Profile attributes named Mutual Fund Range # Opening Trade Count. Processing compares each parameter to the Trade table's Last Principal Amount-Base. Each range is from the lower bound entered here to the lower bound of the next range.	
Parameters used by queries defined in the data maps	UnrelatedWhenOffsetAccount IsNull	This parameter is used to assign unrelated party code as "J" in the BackOfficeTransaction table, If OFFST_ACCT_INTRL_ID is null and UnrelatedWhenOffsetAccountIsNull is "Y", If OFFST_ACCT_INTRL_ID is null and UnrelatedWhenOffsetAccountIsNull is "N", then unrelated party code is NULL.	Y

D.5.4.2 BD Datamap Configuration File

Oracle clients can modify the BDF.xml file under the *bdf/config/custom* folder to override default settings that the system provides.

You can also reapply any modifications in the current BDF.xml file to the newer BDF.xml file. Override any settings in BDF.xml by placing the modifications in BDF.xml under the *bdf/config/custom* folder. During installation, the following parameters are configured by the installer:

- AccountTrustFromCustomer
- DefaultJurisdiction
- UseTaxidForUnrelatedPartyCode
- BaseCountry
- ProcessForeignFlag
- ProcessBankToBank
- ProcessTransactionXRefFlag
- TrustedPairRiskReviewFlag

These parameters are stored in the *<OFSAAI Installed Directory>/bdf/config/install/BDF.xml* file.

The *DefaultJurisdiction* and *BaseCountry* parameters are defined in the InstallConfig.xml file during Silent Installation. Refer to the *Installation Guide* for more information.

The Installer sets the default value for other parameters as follows:

- `<Parameter name="AccountTrustFromCustomer" type="STRING" value="Y"/>`
- `<Parameter name="DefaultJurisdiction" type="STRING" value="AMEA"/>`
- `<Parameter name="UseTaxidForUnrelatedPartyCode" type="STRING" value="Y"/ >`
- `<Parameter name="BaseCountry" type="STRING" value="US"/>`
- `<Parameter name="ProcessForeignFlag" type="STRING" value="N"/>`
- `<Parameter name="ProcessBankToBank" type="STRING" value="N"/>`
- `<Parameter name="ProcessTransactionXRefFlag" type="STRING" value="Y"/>`
- `<Parameter name="TrustedPairRiskReviewFlag" type="STRING" value="N"/>`

To change the default value of these parameters, before running ingestion, go to *<OFSAAI Installed Directory>/bdf/config/install/BDF.xml* and change the value to '**Y**' or '**N**' as needed.

The following table describes the parameters defined in BDF.xml:

Table D-20 BD Datamap Configuration Parameters

Property Name	Description	Example
DB.Connection.Instance	Database instance to connect to on the database servers. Typically, the instance name matches the database name portion of the DB.Connection.URL.	D1O9L2
DB.Schema.MANTAS	Schema name for the Oracle ATOMIC database schema. BD accesses the ATOMIC schema when allocating sequence IDs to ingested records.	ATOMIC
DB.Schema.MARKET	Schema name for the ATOMIC database schema. Data Management stores market data related records in the ATOMIC schema.	ATOMIC

Table D-20 (Cont.) BD Datamap Configuration Parameters

Property Name	Description	Example
DB.Schema.BUSINESS	Schema name for the ATOMIC database schema. Data Management stores business data related records in the ATOMIC schema.	ATOMIC
DB.Schema.CONFIG	Name of the configuration schema owner.	REVELEUS
DB.Schema.CASE	Name of the ATOMIC schema owner.	ATOMIC
DB.Alg.Connection.User	Database user for running Behavior Detection post-processing jobs.	ATOMIC
DB.Alg.Connection.Password	Password for the DB.Alg.Connection.User.	
DB.Connection.URL	Database URL for JDBC connections made by BD components. The content and format of this value is specific to the database vendor and the vendor database driver.	jdbc:oracle:thin:@solitaire.mantas.com:1521:D1O9 L2

There are also configuration files for individual components that are delivered as part of the product as `<OFSAAI Installed Directory>/bdf/config/<component>.xml`

These can also be created in `<OFSAAI Installed Directory>/bdf/config/custom/<component>.xml`

D.6 Alternate Process Flow for MiFID Clients

Derivations done by the FDT process for the MiFID scenarios, which use the Order Size Category, require the use of the Four-week Average Daily Share Quantity (4-wk ADTV) to define an order as small, medium, or large based on how it compares to a percentage of the 4-wk ADTV.

The 4-wk ADTV is derived on a daily basis by the process_market_summary.sh script in the end-of day batch once the Daily Market Profile is collected for each security from the relevant market data source.

For firms using the MiFID scenarios and running a single end-of-day batch, the process_market_summary.sh script must be executed prior to running the runFDT.sh script such that the 4-wk ADTV for the Current Business Day incorporates the published Current Day Traded Volume.

The following figure depicts dependency between the *process_market_summary.sh* script and the *runFDT.sh* script. For intra-day batch ingestion or intra-day execution of the MiFID scenarios, the process flow does not change from this structure. Since the current day's 4-wk ADTV is not available until the end of the day, the previous day's 4-wk ADTV is used to determine order size. For additional information on configuring the percentage values used to define a MiFID-eligible order as Small, Medium, or Large, see section *Market Supplementary Guidance*, in the *Data Interface Specification*.

Figure D-3 Dependency between process_market_summary.sh and runFDT.sh

E

Processing Derived Tables and Fields

This appendix tells how to process the derived fields and tables.>

This appendix covers the following topics:

E.1 Customizing Scripts

For OFSAAI to execute the shell scripts, the customized scripts have to be placed in the ficdb layer.

The customized scripts should be placed under `<Installed Path>ficdb/bin`. When the customized scripts are called from OFSAAI, it appends the Batch Flag and Wait Flag parameters. This must be internally handled in the customized script to eliminate these additional parameters.

> **Note:**
>
> The Batch Flag and Wait Flag are the default parameters expected by the AAI Batch. For more information on these parameters refer the Oracle Financial Services Analytical Applications Infrastructure User Guide.

The following paths should be set inside the scripts:

- MANTAS_HOME: The path where the solution is installed. For Example: `/scratch/ofsaaapp/FCCM804`
- INGESTION_HOME: The path under installed area pointing to the ingestion_manager subsystem. For Example: `/scratch/ofsaaapp/FCCM804/ingestion_manager`
- DB_TOOLS_HOME: The path under installed area pointing to database subsystem. For Example: `/scratch/ofsaaapp/FCCM804/database/db_tools`
- BDF_HOME: The path under the installed area pointing to the BD subsystem. For Example: `/scratch/ofsaaapp/FCCM804/bdf`

> **Note:**
>
> BDF_HOME should be exported only if Ingestion has to be run through the BD subsystem.

After exporting the respective paths inside the script, the product script must be called from the customized script. For more information about how to create an OFSAA Batch and add a task for executing the custom script, refer to the Oracle Financial Services Analytical Applications Infrastructure User Guide.

Sample customized script for execute.sh is given below:

```
#!/bin/sh
if [[ $# == 0 || $# > 3 ]]; then
##echo "Usage: run_GD_dpdl.sh YYYYMMDD"
exit -1;
fi
export MANTAS_HOME=/scratch/ofsaadb/BD_801_BUILD2/BD_801C2WL
export BDF_HOME=$MANTAS_HOME/bdf
export DB_TOOLS_HOME=$MANTAS_HOME/database/db_tools
##export DIS_FILES=$HOME/GD_Scripts/disfile.cfg
export FILE_NAME=$1
$BDF_HOME/scripts/execute.sh $FILE_NAME
err=$?
if [ $err -ne 0 ]
then
echo " BDF Execution failed"
exit 1
fi
```

The above script is used to trigger BD Ingestion using execute.sh. This script expects only the file name (such as Account)as a parameter. Since AAI batch appends two additional default parameters (Batch Flag and Wait Flag) during batch execution, these should be handled inside the script and only the file name should be passed as a parameter. Internally this customized script calls the product script, execute.sh. Similarly, other scripts can also be customized.

E.2 Derivations

These utilities populate a single table in the data model.

These utilities populate a single table in the data model. They should be executed after all the files have been loaded. A utility should not be executed until its predecessors have executed successfully.

Execute the following commands:

```
<OFSAAI Installed Directory>/ingestion_manager/scripts/runUtility.sh <Utility
Name>
<OFSAAI Installed Directory>/ingestion_manager/scripts/runDL.sh <Utility Name>
```

AccountDailySecurityProfile

The AccountDailySecurityProfile Utility is used to populate the Account Daily Security Profile table. This Utility reads the Trade table, and processes the trade records to populate the ACCT_SCRTY_SMRY_DAILY table.

Execute the following commands:

```
runUtility.sh <Utility Name>
runDL.sh <Utility Name>
```

While executing these commands, replace `<Utility Name>` with *AccountDailySecurityProfile*

```
 runUtility.sh AccountDailySecurityProfile
runDL.sh AccountDailySecurityProfile
```

E.3 Guidelines for Duplicate Record Handling

The Ingestion Manager considers records as duplicates if the primary business key for multiple records are the same.

The Ingestion Manager manages these records by performing either an insert or update of the database with the contents of the first duplicate record. The system inserts the record if a record is not currently in the database with the same business key. The record updates the existing database record if one exists with the same business key. The Ingestion Manager handles additional input records with the same business key by performing database updates. Therefore, the final version of the record reflects the values that the last duplicate record contains.

E.4 Data Rejection During Ingestion

The Ingestion Manager can reject records at the Pre-processing, Transformation, or Loading stages.

The following sections provide an overview of the most frequent types of conditions that cause transactions to be rejected:

- Rejection During Pre-processing Stage: Describes how rejections occur during the Pre-processing stage and offers guidance on ways to resolve rejections.
- Rejection During Transformation Stage: Describes how rejections occur during the Transformation stage and offers guidance on ways to resolve rejections.
- Rejection During Loading Stage: Describes how rejections occur during the Loading stage and offers guidance on ways to resolve rejections.

Rejection During the Pre-processing Stage

The first stage of ingestion is Pre-processing. At this stage, Data Management examines Oracle client reference and trading data for data quality and format to ensure the records conform to the requirements in the DIS. Common reasons for rejection of data during Pre-processing include problems with data type, missing data, referential integrity, and domain values.

During normal operation, the number of rejections at the Pre-processor stage should be minimal. If the volume of rejections at this stage is high, a decision threshold can halt

processing and allow manual inspection of the data. The rejections are likely the result of a problem in the data extraction process. It is possible to correct the rejections and then reingest the data.

- **Data Type**: Every field in a record that processing submits to the Ingestion Manager must meet the data type and length requirements that the DIS specifies. Otherwise, the process rejects the entire record. For example, fields with a Date Type must appear in the format YYYYMMDD. Thus, the date April 30, 2005 has a format of 20050430 and, therefore, is unacceptable. In addition, a field cannot contain more characters or digits than specified. Thus, if an Order Identifier in an Order record contains more than the maximum allowed length of 40 characters, rejection of the entire record occurs.
- **Missing Data:** The DIS defines fields that are mandatory, conditional, and optional. If a record contains a field marked mandatory, and that field has a null value, processing rejects the record. For example, all Trade Execution records must contain a Trade Execution Event Number. If a field is marked conditional, it must be provided in some cases. Thus, an Order record for a limit order must contain a Limit Price, but an Order record for a market order need not contain a Limit Price.
- **Referential Integrity:** In some cases, you can configure Ingestion Manager to reject records that refer to a missing reference data record. For example, Ingestion Manager can reject an order that refers to a deal that does not appear in the Deal file. The default behavior is not to reject records for these reasons.
- **Domain Values:** Some fields are restricted to contain only one of the domain values that the DIS defines. The Ingestion Manager rejects records that contain some other value. For example, Ingestion Manager rejects any Order record that contains an Account Type other than CR, CI, FP, FB, ER, IA, EE or any Special Handling Code other than that in the DIS.

Rejection During the Transformation Stage

The second stage of ingestion is Transformation. At this stage, the Ingestion Manager derives the order and trade life cycles, and other attributes, that are necessary for trade-related surveillance. The Ingestion Manager rejects order records during Transformation for the following reasons:

- New and Cancel or Replace order events if the order identifier and placement date combination already exists; order identifiers must be unique during a given day.
- New order events for child orders if the referenced parent order is itself a child order; only one level of a parent-child relationship is allowed.

The Ingestion Manager rejects trade execution records for New and Cancel or Replace trade execution events if the trade execution identifier and trade execution date combination already exists. Trade execution identifiers must be unique during a given day.

Other problems can occur that do not cause rejection of records but cause handling of the records to be different, such as Lost Events or Out of Sequence Events.

Lost Events

If the system receives an order event other than a New or Cancel or Replace in a set of files before receiving the corresponding New or Cancel or Replace, it writes the order event to a lost file. The system examines events in the lost file during processing of subsequent sets of files to determine whether the system received the corresponding New or Cancel or Replace event. If so, processing of this event is normal. If an event resides in the lost file when execution of open order processing occurs (that is, execution of runDP.sh OPEN_ORDER), processing rejects the event. The same applies to trade execution events. In addition, if a New trade execution event references an order but the system did not receive the order, the New event also resides in the lost file subject to the same rules. If rejection of a New or Cancel or Replace order or trade execution occurs during the Pre-processor stage, all subsequent events

are considered lost events. Submission of missing New or Cancel or Replace event can occur in a subsequent set of files, and processing of the lost events continue normally.

Out-of-Sequence Events

An out-of-sequence event is an order or trade execution event (other than New or Cancel or Replace) that the system processes in a set of files after processing the set of files that contains the corresponding New or Cancel or Replace event. Such an event that has a timestamp prior to the timestamp of the last event against that order or trade is considered an out-of-sequence event.

File Set 1 contains the following events:

- NW order event, timestamp 09:30:00.
- MF order event, timestamp 09:45:00.

File Set 2 contains NW trade execution event (references the above order), timestamp 09:40:00.

This trade execution event is considered out of sequence.

It is important to note that this also includes market data. If, in a given batch, market data up to 10:00:00 is used to derive attributes for a given order, any event in a subsequent file against that order with a timestamp prior to 10:00:00 is considered out of sequence.

An out-of-sequence event has no effect on the order or trade that it references. Processing sets the out-of-sequence flag for the event to Y(Yes) and the system writes the event to the database. An Out of Sequence event has no effect on the order or trade that it refers if processing sets the Out-of-sequence flag set for the event to Y

For end-of-day processing, this may not be an issue. For Intra-day processing, subsequent files should contain data in an ever-increasing time sequence. That is, the first set of files should contain data from 09:00:00 to 11:00:00, the second set of files should contain data from 11:00:00 to 12:00:00, and so on. This only affects events in a single order or trade's life cycle.

Batch 1 contains the following events:

- NW order event for order X, timestamp 09:30:00.
- MF order event for order X, timestamp 09:45:00.

This order event is not considered out of sequence; processing continues normally.

Rejection During the Loading Stage

The last stage of ingestion is Loading. At this stage, the Ingestion Manager loads orders, executions, and trades into the database. The Ingestion Manager rejects records during Loading if configuration of the database is incorrect , such as setup of partitions, are incorrect for the data being ingested).

E.5 Alternatives to Standard Data Management Practices

This topic describes the various alternatives to standard data management practices.

Data Management Archiving

During ingestion processing, the system moves processed files into an archive directory. Firms can use these files to recover from processing malfunctions, and they can copy these files to off-line media for backup purposes.

The Pre-processor moves files in the /inbox directory. All other components move their input files to date-labeled subdirectories within the /backup directory.

Periodically, an Oracle client can run the runIMC.sh script to perform the Ingestion Manager cleanup activities. This script deletes old files from the archive area based on a configurable retention date. Periodic running of the cleanup script ensures that archive space is available to archive more recent data.

E.5.1 Fuzzy Name Matcher Utility

During BD Datamap processing, the Fuzzy Name Matcher utility is used to match names of individuals and corporations (candidates) against a list of names (targets).

The utility calculates a score that indicates how strongly the candidate name matches the target name. All matches are case-insensitive.

The Fuzzy Name Matcher engine supports matching on ASCII, extended ASCII, AND the first 128 encoded characters of the UTF-8 character set (which is equivalent to ASCII, as the same encoding). Any UTF-8 characters beyond this (such as Chinese, Arabic, and so on) will be ignored (will not cause the engine to crash, but such names will not match). Any encoding other than ASCII, extended ASCII, and UTF-8 will cause unpredictable behavior and likely cause the engine to crash (as they are not supported).

Using the Fuzzy Name Matcher Utility

The utility typically runs as part of automated processing that a job scheduling tool such as Maestro or Unicenter AutoSys manages. You can also execute the utility through a UNIX shell script, which the next section describes.

Executing the Fuzzy Name Matcher Utility

To execute the Fuzzy Name Matcher Utility manually, type the following at the UNIX command line:

```
fuzzy_match.sh -t <target_name_list> -c <candidate_name_list> -r <result_file>
```

E.5.1.1 Configuring the Fuzzy Name Matcher Utility

The Fuzzy Name Matcher utility can be used through Ingestion Manager as a standalone Fuzzy Name Matcher, or through BD Datamaps.

To use the Fuzzy Matcher Utility through the Ingestion Manager as a standalone Fuzzy Name Matcher, refer to Executing the Fuzzy Name Matcher Utility. Configure the Fuzzy Name Matcher by modifying `<ingestion_manager>/fuzzy_match/mantas_cfg/install.cfg`.

To use the Fuzzy Matcher Utility through the BD Datamaps (NameMatchStaging.xml,RegOToBorrower.xml) file in folder (<OFSAAI Installed Directory>/bdf/config/datamaps). For more information, refer to *Chapter 4, "Managing Data.*". Configure the Fuzzy Name Matcher by modifying `<ingestion_manager>/fuzzy_match/mantas_cfg/install.cfg`.

The following section provides a sample configuration appearing in `<OFSAAI Installed Directory>/bdf/fuzzy_match/mantas_cfg/install.cfg`.

Sample BDF.xml Configuration Parameters

```
#############################################################    ##
#
#    Fuzzy Name Matcher System Properties file (install.cfg) #
#############################################################    ##

#         #    Log configuration items
#
# Specify which priorities are enabled in a hierarchical fashion, i.e., if
# DIAGNOSTIC priority is enabled, NOTICE, WARN, and FATAL are also enabled, #
but TRACE is not.
# Uncomment the desired log level to turn on appropriate level(s).
# Note, DIAGNOSTIC logging is used to log database statements and will slow
# down performance. Only turn on if you need to see the SQL statements being
# executed.
# TRACE logging is used for debugging during development. Also only turn on #
TRACE if needed.
#log.fatal=true #log.warning=true log.notice=true #log.diagnostic=true
#log.trace=true

# Specify where a message should get logged -- the choices are mantaslog, #
syslog, console, or a filename (with its absolute path).
# Note that if this property is not specified, logging will go to the
console. log.default.location=mantaslog

# Specify the location (directory path) of the mantaslog, if the mantaslog #
was chosen as the log output location anywhere above.

# Logging will go to the console if mantaslog was selected and this property
is # not given a value.
log.mantaslog.location=mp

#         #    Fuzzy Name Matcher configuration items
#     fuzzy_name.match_multi=true fuzzy_name.file.delimiter=~
fuzzy_name.default.prefix=P fuzzy_name.max.threads=1
fuzzy_name.max.names.per.thread=1000 fuzzy_name.max.names.per.process=250000
fuzzy_name.min.intersection.first.letter.count=2
fuzzy_name.temp_file.directory=/scratch/ofsaaapp/BD805/BD805/bdf/data/temp

fuzzy_name.B.stopword_file=/scratch/ofsaaapp/BD805/BD805/bdf/fuzzy_match/
share/ stopwords_b.dat
fuzzy_name.B.match_threshold=80 fuzzy_name.B.initial_match_score=75.0
fuzzy_name.B.initial_match_p1=2 fuzzy_name.B.initial_match_p2=1
fuzzy_name.B.extra_token_match_score=100.0
fuzzy_name.B.extra_token_min_match=2 fuzzy_name.B.extra_token_pct_decrease=50
fuzzy_name.B.first_first_match_score=1

fuzzy_name.P.stopword_file=/scratch/ofsaaapp/BD805/BD805/bdf/fuzzy_match/
share/ stopwords_p.dat
fuzzy_name.P.match_threshold=70 fuzzy_name.P.initial_match_score=75.0
fuzzy_name.P.initial_match_p1=2 fuzzy_name.P.initial_match_p2=1
fuzzy_name.P.extra_token_match_score=50.0
```

The following table describes the utility's configuration parameters as they appear in the BDF.xml file. Note that all scores have percentage values.

Table E-1 Fuzzy Name Matcher Parameters

Parameter	Description
fuzzy_name.stopword_file	Identifies the file that stores the stop word list. The stop word file is either corporate or personal. The <prefix> token identifies corporate as *B* and personal as *P*. Certain words such as *Corp, Inc*, *Mr*, *Mrs*, or *the*, do not add value when comparing names.
fuzzy_name.match_threshold	Indicates the score above which two names are considered to match each other. The utility uses this parameter only when the match_multi property has a value of *true*. The allowable range is from 0 to100.
fuzzy_name.initial_match_score	Specifies the score given for matching to an initial. The allowable range is 0 to 100; the recommended default is 75.
fuzzy_name.initial_match_p1	Specifies the number of token picks that must be made before awarding initial_match_score. The value is an integer >= 0. The default value is 2.
fuzzy_name.initial_match_p2	Specifies the number of token picks that must be made before awarding initial_match_score if only initials remain in one name. The value is an integer >= 0. The default value is 1.
fuzzy_name.extra_token_match_score	Indicates the score given to extra tokens. The allowable range is 0 to 100; the recommended default is 50.
fuzzy_name.extra_token_min_match	Specifies the minimum number of matches that occur before awarding extra_token_match_score. The range is any integer >= 0. The recommended setting for corporations is 1; for personal names is 2.
fuzzy_name.extra_token_pct_decrease	Determines the value of the extra_token_match_score parameter in regard to extra tokens. If multiple extra tokens are present, reduction of extra_token_match_score occurs for each additional extra token. The utility multiplies it by this number. For example, if extra_token_match_score = 50, and extra_pct_decrease is 50 (percent), the first extra token gets 50 percent, the second extra token gets 25 percent, the third token gets 12.5 percent, the fourth 6.25 percent, the fifth 3.125 percent, etc. The allowable range is 0 to 100. The recommended percentage for corporations is 100 (percent); for personal names, 50 (percent).
fuzzy_name.first_first_match_score	Allows the final score to be more heavily influenced by how well the first token of name #1 matches the first token of name #2. The allowable value is any real number >= 0. The recommended value for corporate names is 1.0; for personal names, 0.0.

Table E-1 (Cont.) Fuzzy Name Matcher Parameters

Parameter	Description
fuzzy_name.match_multi	Determines how to handle multiple matches above the match_threshold value. If set to "*true,"* the utility returns multiple matches. If set to "*false*," it returns only the match with the highest score.
fuzzy_name.file.delimiter	Specifies the delimiter character used to separate each columns in the result file and target name list file.
fuzzy_name.min.intersection.firs t.letter.count	Specifies the number of words per name whose first letters match. For example, if parameter value = 1 only the first letter of the first or last name would have to match to qualify. If the value = 2, the first letter of both the first and last name would have to match to qualify. **Warning:** By default, the value is set to 2. Oracle recommends using the default value. You must not change the value to 1 or your system performance may slow down.
fuzzy_name.default.prefix	For entries that are not specified as business or personal name, default to this configuration set.
fuzzy_name.max.names.per.process	This property variable determines whether or not the fuzzy matcheralgorithm will be run as a single process or as multiple sequential processes. If the total number of names between both the candidate name list and the target name list is less than the value of this property, then a single process will be run. If the number of names exceeds this property's value, then multiple processes will be run, based on how far the value is exceeded. For example, if the candidate name list contains 50 names, the target name list contains 50 names, and the fuzzy_name.max.names.per.process property is set to 200, then one process will be run (because the total number of names, 100, does not exceed 200). If the candidate list contains 400 names, the target name list contains 200 names, and the fuzzy_name.max.names.per.process property is set to 300, then four processes will be run (each with 100 candidate names and 200 target names so that the max number of names per process never exceeds 300). The ability to break apart one large fuzzy matcher process into multiple processes through this property can help to overcome per-process memory limitations imposed by certain Behavior Detection architectures.
fuzzy_name.max.threads	This parameter controls the number of threads to use when Fuzzy Name Matcher is being run. Oracle recommends that this value is not set to a number higher than the number of processing cores on the system.

Table E-1 (Cont.) Fuzzy Name Matcher Parameters

Parameter	Description
fuzzy_name.max.names.per.thread	This parameter keeps the processing threads balanced so that they perform work throughout the course of the fuzzy matcher job. That is, instead of splitting the number of names to process evenly across the threads, the value of this parameter can be set to a smaller batch-size of names so that threads that finish ahead of others can keep working.

E.5.2 Refresh Temporary Tables Commands

Prior to running post-processing, you must execute database scripts after ingestion and prior to running AML scenarios.

These scripts refresh the required temporary tables for selected AML scenario detection.

E.5.3 Using Control Data

After installing the OFSBD software, you can use control data provided to test end-to-end processing of data (that is, running data management, executing scenarios, and viewing generated alerts in the Alert Viewer UI). Thus, you can verify that installation of the software is correct and works as designed.

To prepare the system for testing, follow these steps:

1. Complete the prerequisites for using control data.
2. Prepare for ingestion of the control data.
3. Install the control data.
4. Run Behavior Detection on control data to generate alerts.

E.5.3.1 Prerequisites for Using Control Data

Before you use control data to test your Behavior Detection installation, the following prerequisites must be fulfilled.

1. The maximum lookback that control data considers is of 13 months, which is for change in behavior scenarios. Hence, while creating control data ensure that it is spread over 25 different dates in 13 months.
2. The current day according to control data is 20151210.
3. Unless specified, set the current date as 20151210, to generate alerts on control data, before running Behavior Detection Platform.

> **Note:**
>
> For more information about control data on your site, contact your OFSBD Administrator.

E.5.3.2 Control Data Management

Control data uses a specific set of dates to ensure that all the OFSBD lock-stock scenarios are tested using this data.

The maximum lookback that control data considers is of 13 months, which is for change in behavior scenarios. The control data is spread over 25 different dates in 13 months. The dates (YYYYMMDD format) being used by control data are:

On all these dates, ingest the data and run the complete Behavior Detection batch for the respective date. Except for Behavior Detection and Post-Processing tasks, perform all other activities for the Control Data Management dates. Activities required during any Behavior Detection Framework business day are - `START BATCH > DRM > DATA INGESTION > BEHAVIOR DETECTION > POST PROCESSING > END BATCH`.

Prior to running Behavior Detection on the control data, you must complete the following procedures.

1. Copy all control data from the golden data directory in the database subsystem (`/database/ golden_data directory`) to the Ingestion Manager /inbox directory bdf /inbox.
2. Run ingestion for all the control Data Management dates. Refer to Ingestion Timeline - Intra-Day Ingestion Processing, for more information about the ingestion process.

> **Note:**
>
> You must adjust the partitions of the database tables as per the new dates, if you intend to process Control Data after the database upgrade to OFSBD.

E.5.3.3 Loading Control Data Thresholds

To generate breaks on the control data, specific threshold sets and jobs are created.

These threshold sets must be installed to the Behavior Detection system for use of control data and generation of test alerts.

1. Navigate to the directory `<OFSAAI Installed Directory>/database/golden_data/threshold_sets`. This directory consists of test threshold sets of all the scenarios that are available with the OFSAAI system.
2. Execute `shell script load_tshld_set.sh`. This shell script installs the control data threshold sets for all the scenarios that are installed at your site. It also creates new jobs and template group IDs corresponding to all the scenarios installed. These template group IDs are same as the scenario IDs of installed scenarios.
3. Once the control data thresholds are installed, the system is ready for a test run, that is, generating test alerts.

E.5.3.4 Running Behavior Detection on Control Data

In order to generate alerts on the ingested control data, execute the new scenario jobs. These jobs consists of same template group ID as the scenario ID.

Refer to *Chapter 5, "Behavior Detection Jobs."* to get information regarding about running Behavior Detection Jobs.

Important Notes

1. Run loaded scenarios with the system date as 20151210 with the following exceptions:
   - For Portfolio Pumping scenario, the system date must be 20151204
   - For Active Trading scenario, the system date must be 20151130
2. Check for system errors in the appropriate logs (refer to “APPENDIX A - Logging.”, for more information).
3. Run post-processing procedures.
4. Close the batch to enable display of alerts in the Behavior Detection UI.
5. Log in to the Behavior Detection UI with the correct user credentials.
6. Verify that you can view alerts in the UI.

The display of alerts signifies that installation of the system is correct and works as designed.

> **Note:**
>
> The alerts that you can view depend on your user privileges.

F

BD Datamap Details

This appendix lists the BD datamaps used in OFSAAI and a brief explanation of the each datamap.

This appendix contains the following topics:

> **Note:**
>
> Oracle recommends all datamaps are run in the order described in the following tables.

F.1 AML Brokerage - Pre-Watch List Datamaps

Pre-Watch List Datamaps are used to facilitate the application to populate various business areas, such as Financial Institutions, Account To Client Bank, Settlement Instructions, Front Office and Back Office Transaction.

These datamaps populate the relevant data which is used by watch list datamaps in calculating risks.

Table F-1 AML Brokerage - Pre-Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
50010	Customer_TotAcctUpd	NA
10010	EmployeeControlledAccount (*Optional*)	NA
10015	FrontOfficeTransactionParty_SecondaryNames	NA
10020	FinancialInstitution_ThomsonDataInstitutionInsert (*Optional*)	NA
10030	AccountToClientBank_ThomsonDataInstitutionInsert (*Optional*)	10020
10040	FinancialInstitution_AIIMSPopulation	NA
10050	AccountToClientBank_AIIMSInstitutionInsert	10040
10060	AccountToClientBank_InstitutionInsert	10050
10070	AccountToClientBank_InstitutionUpd	10060
10080	FinancialInstitution_FOTPSPopulation	10020 10030 10040 10050 10060 10070
10090	AccountToClientBank_FOTPSInstitutionInsert	10020 10030 10040 10050 10060 10070 10080
10095	AccountCustomerRole	10095
10096	AccountToCustomer	NA
10100	AccountManagementStage	NA
10111	LoanDailyActivity_RepCurrencyUpd	NA
10110	LoanProfile_LoanProfileStage	10111
10114	BackOfficeTransaction_UnrelatedPartyCodeUpd	NA
10116	BackOfficeTransaction_CollateralUpd	10114
10120	BackOfficeTransaction_OriginalTransactionReversalUpd	NA
10130	BackOfficeTransaction_CancelledTransactionReversalCredi tUpd	NA
10140	BackOfficeTransaction_CancelledTransactionReversalDebit Upd	NA
10150	FrontOfficeTransactionParty_InstnSeqID	10020 10030 10040 10050 10060 10070 10090
10160	FrontOfficeTransactionParty_HoldingInstnSeqID	10150
10170	FinancialInstitution_AnticipatoryProfile	10020 10030 10040 10050 10060 10070
10180	AccountToClientBank_AnticipatoryProfile	10020 10030 10040 10050 10060 10070 10170

Table F-1 (Cont.) AML Brokerage - Pre-Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10190	AnticipatoryProfile_AccountToClientBank	10020 10030 10040 10050 10060 10070 10170 10180
50020	DailyAggregateStage	NA
50030	OffsettingAccountPairStage	NA
50040	TradeDailyTotalCountStage	NA
10200	CustomerAccountStage_FrontOfficeTransactionParty	NA
10210	FrontOfficeTransaction_UnrelatedPartyUpd	10120 10130 10140 10200
10220	FinancialInstitution_SettlementInstruction	10020 10030 10040 10050 10060 10070
10230	AccountToClientBank_SettlementInstruction	10020 10030 10040 10050 10060 10070 10220
10240	SettlementInstruction_AccountToClientBank	10020 10030 10040 10050 10060 10070 10230
10014	FrontOfficeTransaction_PassThroughFlag	NA

Note:

- **FrontOfficeTransaction_PassThroughFlag** - This data map should only be run if the *Pass Through Indicator* field is not being provided in the Front Office Transaction DIS file,and the client requires support to derive this datamap.
- **FrontOfficeTransactionParty_SecondaryNames** - This data map should only be run if Secondary Originator and Secondary Beneficiary party records are not being provided in the Front Office Transaction Party DIS file, and the client requires support to derive them from the Bank- to-Bank Instructions and Originator-to-Beneficiary Instructions fields.

F.2 AML Brokerage - Watch List Datamaps

(Watch List Datamaps facilitate the application of customer-supplied measures of risk to corresponding entities, transactions, and instructions.

These datamaps assist other datamaps which are used to calculate Effective Risk and Activity Risk for various entities, such as Account, Customer, Transaction Tables, and so on.

Table F-2 AML Brokerage - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10245	WLMProcessingLock	NA
10250	WatchListEntry_WatchListEntryCurrDayInsert	10020 10030 10040 10050 10060 10070 10245
10260	WatchListAudit_StatusUpd	10020 10030 10040 10050 10060 10070

Table F-2 (Cont.) AML Brokerage - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10270	WatchList_WatchListSourceAuditInsert	10020 10030 10040 10050 10060 10070
10280	WatchList_WatchListSourceAuditUpd	10020 10030 10040 10050 10060 10070
10290	WatchList_WatchListSourceUpd	10020 10030 10040 10050 10060 10070
10300	WatchListEntry_WatchListAuditUpd	10020 10030 10040 10050 10060 10070 10260
10310	WatchListEntryAudit_WatchListEntryUpdate	10020 10030 10040 10050 10060 10070 10300
10320	Customer_KYCRiskUpd	NA
10330	DerivedAddress_SettlementInstructionInsert	NA
10340	DerivedAddress_SettlementInstructionUpd	NA
10350	SettlementInstruction_PhysicalDlvryAddrUpd	NA
10360	DerivedAddress_FrontOfficeTransactioPartyStageInsert	NA
10370	DerivedAddress_FrontOfficeTransactioPartyStageUpd	NA
10380	FrontOfficeTransactionParty_DerivedAddress	10360 10370
10390	DerivedEntity_FrontOfficeTransactionPartyInsert	10080 10090
10400	DerivedEntity_FrontOfficeTransactionPartyUpd	10080 10090
10410	DerivedEntity_SettlementInstructionInsert	10220 10230 10240
10420	DerivedEntity_SettlementInstructionUpd	10220 10230 10240
10430	CorrespondentBank_FrontOfficeTransactionPartyStageInsert	10080 10090
10440	CorrespondentBank_FrontOfficeTransactionPartyStageUpd	10080 10090
10441	ExternalPartyStage	NA
10442	ExternalParty	10441
10443	ExternalPartyAddress	10441
10444	DerivedAddress_ExternalOrganizationStageInsert	10360 10442
10445	DerivedAddress_ExternalOrganizationStageUpd	10444
10446	ExternalParty_DerivedAddress	10442 10444 10445
10447	ExternalPartyAddress_DerivedAddress	10441 10443 10444 10445 10446
10448	DerivedEntity_ExtrlOrgInsert	10390 10442
10449	DerivedEntity_ExtrlOrgUpd	10448

Table F-2 (Cont.) AML Brokerage - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10450	WatchListStagingTable_WatchList	10250 10260 10270 10280 10290 10300 10310
10501	LinkStaging_ExtrlOrg	10490 10442
10460	WatchListStagingTable_WatchList InstnIDUpd	10250 10260 10270 10280 10290 10300 10310
10470	PreviousWatchList_WatchList	10250 10260 10270 10280 10290 10300 10310
10480	DerivedAddress_WatchListNewC ountries	10250 10260 10270 10280 10290 10300 10310
10485	WLMProcessingUnlock	10480
10489	ExternalParty_ExternalEntitySeq Upd	10390 10442
10490	LinkStaging_FrontOfficeTransacti onParty	10360 10370 10380 10390 10400 10485
10500	LinkStaging_InstructionDerivedE ntDerivedAdd	10330 10340 10350 10410 10420
10510	NameMatchStaging	10450 10460 10470 10480 10390 10400
10520	WatchListStagingTable_NameMat chStageInsert	10510
10530	DerivedEntityLink_LinkStage	10490 10500
10540	DerivedEntitytoDerivedAddress_L inkStage	10490 10500
10550	DerivedEntitytoInternalAccount_L inkStage	10490 10500
10560	DerivedAddresstoInternalAccount _LinkStage	10490 10500
10570	WatchListStagingTable2_WatchLi stStage2AcctExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10580	WatchListStagingTable2_WatchLi stStage2CBExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10590	WatchListStagingTable2_WatchLi stStage2CustExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10600	WatchListStagingTable2_WatchLi stStage2DAExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10610	WatchListStagingTable2_WatchLi stStage2EEExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10620	WatchListStagingTable2_WatchLi stStage	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10630	WatchListStagingTable2_AcctList MembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440

Table F-2 (Cont.) AML Brokerage - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10640	WatchListStagingTable2_CBListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10650	WatchListStagingTable2_CustListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10660	WatchListStagingTable2_EEListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10670	WatchListStagingTable2_EEListMembershipStatusUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10680	WatchListStagingTable2_DAListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10690	WatchListStagingTable2_DAListMembershipStatusUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10700	WatchListStagingTable2_WatchListStage2SeqIdUpd	10570 10580 10590 10600 10610 10620 10630 10640 10650 10660 10670 10680 10690
10710	WatchListStagingTable2_WatchListStage2IntrlIdUpd	10570 10580 10590 10600 10610 10620 10630 10640 10650 10660 10670 10680 10690
10720	Customer_WatchListStage2ListRisk	10320 10700 10710
10730	CorrespondentBank_WatchListStage2EffectiveRisk	10320 10700 10710
10740	Customer_WatchListStage2EffectiveRisk	10320 10700 10710
10750	DerivedAddress_WatchListStage2EffectiveRisk	10320 10700 10710
10760 10700 10710	DerivedEntity_WatchListStage2EffectiveRisk	10320 10700 10710
10770	WatchListStagingTable2_WatchListStage2SeqId	10320 10700 10710
10780	AccountListMembership_WatchListStage2Insert	10700 10710
10790	AccountListMembership_WatchListStage2Upd	10700 10710
10800	CorrespondentBankListMembership_WatchListStage2Insert	10700 10710
10810	CorrespondentBankListMembership_WatchListStage2Upd	10700 10710
10820	CustomerListMembership_WatchListStage2Insert	10700 10710
10830	CustomerListMembership_WatchListStage2Upd	10700 10710

Table F-2 (Cont.) AML Brokerage - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10840	DerivedAddressListMembership_WatchListStage2Insert	10700 10710
10850	DerivedAddressListMembership_WatchListStage2Upd	10700 10710
10860	DerivedEntityListMembership_WatchListStage2Insert	10700 10710
10870	DerivedEntityListMembership_WatchListStage2Upd	10700 10710
10875	Account_EffectiveRiskFactorTxtUpd	10700 10710
10880	Account_OverallEffectiveRiskUpd	10720 10730 10740 10750 10760 10770 10780 10790 10800 10810 10820 10830 10840 10850 10860 10870
10881	Account_AccountCustRiskUpd	10880
10890	Account_EffRiskUpdAfterWLRiskRemoval	10720 10730 10740 10750 10760 10770 10880
10900	Account_WatchListStage2EffectiveRisk	10720 10730 10740 10750 10760 10770 10880
10910	WatchListStagingTable2_WatchListStage2IntrlId	10320 10700 10710
10920	BackOfficeTransaction_EffectiveAcctivityRiskUpd	10890 10900
10930	SettlementInstruction_EntityActivityRiskUpd	10890 10900
10940	FrontOfficeTransactionPartyRiskStage_EntityActivityRiskInse rt	10890 10900

Note:

If you are running any of these combinations you must run datamap 10320 and 10880.

- OFSBD AML and KYC
- OFSBD Fraud and KYC
- OFSBD AML, Fraud, and KYC

F.3 AML Brokerage - Post-Watch List Datamaps

Post-Watch List Datamaps are used to populate or rather ingest data into various transaction tables using Front Office and Back Office Transaction files, these are executed only after the Watch List Datamaps are run.

These datamaps are used to populate data into the Cash, Wire, and Monetary Instruments tables. These are also used to update Trusted Pair and Jurisdiction information into various other entities. The table below describes the Post-Watch List datamaps for AML Brokerage.

Oracle clients can configure the Risk Zones and customize the Review Reason Text for the following datamaps:

- TrustedPair_StatusRRCInsert (Datamap Number 11080)
- TrustedPair_StatusRRCUpd (Datamap Number11090)
- TrustedPairMember_AcctExtEntEffecRiskUpd (Datamap Number11070)

Table F-3 AML Brokerage - Post Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10960	AccountGroup_JurisdictionUpd	NA
10970	TransactionPartyCrossReference_BackOfficeTransaction	10360 10370 10380 10940
10980	CashTransaction_FrontOfficeTransaction	10360 10370 10380 10940
10990	MonetaryInstrumentTransaction_FrontOfficeTransaction	10360 10370 10380 10940
11000	TransactionPartyCrossReference_FrontOfficeTransaction	10360 10370 10380 10940 11060 11070 11080 11090
11010	WireTransaction_FrontOfficeTransaction	10360 10370 10380 10940
11020	WireTransactionInstitutionLeg_FrontOfficeTransaction	10360 10370 10380 10940
11030	CashTransaction_FrontOfficeTransactionRevAdj	10970 10980 10990 11000 11010 11020
11040	MonetaryInstrumentTransaction_FrontOfficeTransactionRevAdj	10970 10980 10990 11000 11010 11020
11050	WireTransaction_FrontOfficeTransactionRevAdj	10970 10980 10990 11000 11010 11020
11060	TrustedPair_StatusEXPUpd	10970 10980 10990 11000 11010 11020
11070	TrustedPairMember_AcctExtEntEffecRiskUpd	10970 10980 10990 11000 11010 11020
11080	TrustedPair_StatusRRCInsert	11160
11090	TrustedPair_StatusRRCUpd	11170
11100	ApprovalActionsAudit_TrustedPair	10970 10980 10990 11000 11010 11020
11110	TrustedPairMember_StatusRRCInsert	10970 10980 10990 11000 11010 11020
11120	BackOfficeTransaction_TrustedFlagsUpd	11060 11070 11080 11090 11100 11110
11140	MonetaryInstrumentTransaction_TrustedFlagsUpd	11060 11070 11080 11090 11100 11110
11150	WireTransaction_TrustedFlagsUpd	11060 11070 11080 11090 11100 11110

F.4 AML Brokerage - Summary Datamaps

Summary Datamaps are used to calculate aggregations across various entities using the Trade, Transaction, Positions and Balances Tables.

These datamaps populate various profile tables for different entities like Account Profile, Household Profile, Correspondent Bank Profile.The aggregation is done daily, weekly or monthly depending on the business areas.

Table F-4 AML Brokerage - Summary Datamaps

Datamap Number	Datamap Name	Predecessors
50045	ExternalEntityDailyProfile	10390 10980 10990 11010
50046	ExternalEntityProfile	50045
50050	CustomerDailyProfile_BOT	NA
50060	CustomerDailyProfile_FOTPS	NA
50070	InstitutionalAccountDailyProfile_DEAL	NA
50080	CustomerDailyProfile_DEAL	NA
50090	InstitutionalAccountDailyProfile_INST	NA
50100	CustomerDailyProfile_INST	NA
50110	InstitutionalAccountDailyProfile_CorpAction	NA
50120	CustomerDailyProfile_CorpAction	NA
50130	InstitutionalAccountDailyProfile_Trade	NA
50140	CustomerDailyProfile_Trade	NA
11160	AccountDailyProfile-Trade	NA
11170	AccountDailyProfile-Transaction	NA
11180	AccountProfile_Trade	10940 11160 11170
11190	AccountProfile_Transaction	10940 11160 11170
11200	AccountProfile_Stage	NA
11210	AccountProfile_Position	11180 11190 11200
11220	AccountProfile_Balance	11180 11190 11200 11210
50150	InstitutionalAccountProfile	50070 50090 50110 50130
50160	CustomerProfile	50050 50060 50080 50100 50120 50140
11230	ChangeLog_AcctProfileInactivity	11180 11190 11200 11210 11220
11240	AccountPeerGroupMonthlyTransactionProfile	11180 11190 11200 11210 11220
11300	AccountChangeLogSummary	The datamap should be executed once the change log processing is done.
11310	AccountToCustomerChangeLogSummary	The datamap should be executed once the change log processing is done.
11320	CustomerChangeLogSummary	The datamap should be executed once the change log processing is done.

> **Note:**
>
> The AccountChangeLogSummary, AccountToCustomerChangeLogSummary, and CustomerChangeLogSummary datamaps must be run with `execute.sh` from 8.0.2 onwards.

F.5 AML Brokerage - Balances and Positions Datamaps

Balances and Positions Datamaps derive attributes that are useful in assessment of the financial status of an account, customer, or Household.

These datamaps are used to populate business areas, such as account balance, account position, portfolio manager positions, and so on.

Table F-5 AML Brokerage - Balances and Positions Datamaps

Datamap Number	Datamap Name	Predecessors
50170	CustomerBalance_ActiveOTCTradeCtUpd	NA

F.6 AML Banking - Pre-Watch List Datamaps

Pre-Watch List Datamaps are used to facilitate the application to populate various business areas like Financial Institutions, Account To Client Bank, Settlement Instructions, Front Office and Back Office Transaction.

These datamaps populate the relevant data which are used by watch list datamaps in calculating risks.

Optional Datamaps are used to perform processing to support other datamaps in multiple functional areas. These datamaps may or may not be completely relevant to a particular solution set. Execute the datamap if a scenario in your implementation requires this information.

Table F-6 AML Banking - Pre-Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10010	EmployeeControlledAccount (*Optional*)	NA
10015	FrontOfficeTransactionParty_SecondaryNames	NA
10020	FinancialInstitution_ThomsonDataInstitutionInsert (*Optional*)	NA
10030	AccountToClientBank_ThomsonDataInstitutionInsert (*Optional*)	10020
10040	FinancialInstitution_AIIMSPopulation	NA
10050	AccountToClientBank_AIIMSInstitutionInsert	10040

Table F-6 (Cont.) AML Banking - Pre-Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10060	AccountToClientBank_InstitutionInsert	10050
10070	AccountToClientBank_InstitutionUpd	10060
10080	FinancialInstitution_FOTPSPopulation	10020 10030 10040 10050 10060 10070
10090	AccountToClientBank_FOTPSInstitutionInsert	10020 10030 10040 10050 10060 10070 10080
10095	AccountCustomerRole	10095
10096	AccountToCustomer	NA
10100	AccountManagementStage	NA
10110	LoanProfile_LoanProfileStage	NA
10114	BackOfficeTransaction_UnrelatedPartyCodeUpd	NA
10116	BackOfficeTransaction_CollateralUpd	10114
10120	BackOfficeTransaction_OriginalTransactionReversalUpd	NA
10150	FrontOfficeTransactionParty_InstnSeqID	10020 10030 10040 10050 10060 10070 10090
10160	FrontOfficeTransactionParty_HoldingInstnSeqID	10020 10030 10040 10050 10060 10070 10150
10200	CustomerAccountStage_FrontOfficeTransactionParty	NA
10210	FrontOfficeTransaction_UnrelatedPartyUpd	10120 10130 10140 10200
10014	FrontOfficeTransaction_PassThroughFlag	NA

Note:

- **FrontOfficeTransaction_PassThroughFlag** - This data map should only be run if the P*ass Through Indicator*field is not being provided in the Front Office Transaction DIS file,and the client requires support to derive this datamap.
- **FrontOfficeTransactionParty_SecondaryNames** - This data map should only be run if Secondary Originator and Secondary Beneficiary party records are not being provided in the Front Office Transaction Party DIS file, and the client requires support to derive them from the Bank- to-Bank Instructions and Originator-to-Beneficiary Instructions fields.

F.7 AML Banking - Watch List Datamaps

Watch List Datamaps facilitate the application of customer-supplied measures of risk to corresponding entities, transactions, and instructions.

These datamaps finally assist other datamaps which are used to calculate Effective Risk and Activity Risk for various entities, such as Account, Customer, Transaction, and so on.

Table F-7 AML Banking - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10245	WLMProcessingLock	NA
10250	WatchListEntry_WatchListEntryCurrDayInsert	10020 10030 10040 10050 10060 10070 10245
10260	WatchListAudit_StatusUpd	10020 10030 10040 10050 10060 10070
10270	WatchList_WatchListSourceAuditInsert	10020 10030 10040 10050 10060 10070 10260
10280	WatchList_WatchListSourceAuditUpd	10020 10030 10040 10050 10060 10070
10290	WatchList_WatchListSourceUpd	10020 10030 10040 10050 10060 10070
10300	WatchListEntry_WatchListAuditUpd	10020 10030 10040 10050 10060 10070 10260
10310	WatchListEntryAudit_WatchListEntryUpdate	10020 10030 10040 10050 10060 10070 10300
10320	Customer_KYCRiskUpd	NA
10360	DerivedAddress_FrontOfficeTransactioPartyStageInsert	NA
10370	DerivedAddress_FrontOfficeTransactioPartyStageUpd	NA
10380	FrontOfficeTransactionParty_DerivedAddress	10360 10370
10390	DerivedEntity_FrontOfficeTransactionPartyInsert	10080 10090
10400	DerivedEntity_FrontOfficeTransactionPartyUpd	10080 10090
10410	DerivedEntity_SettlementInstructionInsert	10220 10230 10240
10420	DerivedEntity_SettlementInstructionUpd	10220 10230 10240
10430	CorrespondentBank_FrontOfficeTransactionPartyStageInsert	10080 10090
10440	CorrespondentBank_FrontOfficeTransactionPartyStageUpd	10080 10090
10450	WatchListStagingTable_WatchList	10250 10260 10270 10280 10290 10300 10310
10460	WatchListStagingTable_WatchListInstnIDUpd	10250 10260 10270 10280 10290 10300 10310
10470	PreviousWatchList_WatchList	10250 10260 10270 10280 10290 10300 10310
10480	DerivedAddress_WatchListNewCountries	10250 10260 10270 10280 10290 10300 10310
10485	WLMProcessingUnlock	10480
10490	LinkStaging_FrontOfficeTransactionParty	10360 10370 10380 10390 10400 10485
10500	LinkStaging_InstructionDerivedEntDerivedAdd	10330 10340 10350 10410 10420

Table F-7 (Cont.) AML Banking - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10510	NameMatchStaging	10450 10460 10470 10480 10390 10400
10520	WatchListStagingTable_NameMatchStageInsert	10510
10530	DerivedEntityLink_LinkStage	10490 10500
10540	DerivedEntitytoDerivedAddress_LinkStage	10490 10500
10550	DerivedEntitytoInternalAccount_LinkStage	10490 10500
10560	DerivedAddresstoInternalAccount_LinkStage	10490 10500
10570	WatchListStagingTable2_WatchListStage2AcctExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10580	WatchListStagingTable2_WatchListStage2CBExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10590	WatchListStagingTable2_WatchListStage2CustExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10600	WatchListStagingTable2_WatchListStage2DAExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10610	WatchListStagingTable2_WatchListStage2EEExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10620	WatchListStagingTable2_WatchListStage	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10630	WatchListStagingTable2_AcctListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10640	WatchListStagingTable2_CBListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10650	WatchListStagingTable2_CustListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10660	WatchListStagingTable2_EEListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10670	WatchListStagingTable2_EEListMembershipStatusUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10680	WatchListStagingTable2_DAListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10690	WatchListStagingTable2_DAListMembershipStatusUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440

Table F-7 (Cont.) AML Banking - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10700	WatchListStagingTable2_WatchListStage2SeqIdUpd	10570 10580 10590 10600 10610 10620 10630 10640 10650 10660 10670 10680 10690
10710	WatchListStagingTable2_WatchListStage2IntrlIdUpd	10570 10580 10590 10600 10610 10620 10630 10640 10650 10660 10670 10680 10690
10720	Customer_WatchListStage2ListRisk	10320 10700 10710
10730	CorrespondentBank_WatchListStage2EffectiveRisk	10320 10700 10710
10740	Customer_WatchListStage2EffectiveRisk	10320 10700 10710
10750	DerivedAddress_WatchListStage2EffectiveRisk	10320 10700 10710
10760	DerivedEntity_WatchListStage2EffectiveRisk	10320 10700 10710
10770	WatchListStagingTable2_WatchListStage2SeqId	10320 10700 10710
10780	AccountListMembership_WatchListStage2Insert	10700 10710
10790	AccountListMembership_WatchListStage2Upd	10700 10710
10800	CorrespondentBankListMembership_WatchListStage2Insert	10700 10710
10810	CorrespondentBankListMembership_WatchListStage2Upd	10700 10710
10820	CustomerListMembership_WatchListStage2Insert	10700 10710
10830	CustomerListMembership_WatchListStage2Upd	10700 10710
10840	DerivedAddressListMembership_WatchListStage2Insert	10700 10710
10850	DerivedAddressListMembership_WatchListStage2Upd	10700 10710
10860	DerivedEntityListMembership_WatchListStage2Insert	10700 10710
10870	DerivedEntityListMembership_WatchListStage2Upd	10700 10710
10875	Account_EffectiveRiskFactorTxtUpd	10700 10701
10880	Account_OverallEffectiveRiskUpd	10720 10730 10740 10750 10760 10770 10780 10790 10800 10810 10820 10830 10840 10850 10860 10870
10881	Account_AccountCustRiskUpd	10880
10890	Account_EffRiskUpdAfterWLRiskRemoval	10720 10730 10740 10750 10760 10770 10880
10900	Account_WatchListStage2EffectiveRisk	10720 10730 10740 10750 10760 10770 10880

Table F-7 (Cont.) AML Banking - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10910	WatchListStagingTable2_WatchListStage2IntrlId	10320 10700 10710
10920	BackOfficeTransaction_EffectiveAcctivityRiskUpd	10890 10900
10940	FrontOfficeTransactionPartyRiskStage_EntityActivityRiskInsert	10890 10900

F.8 AML Banking - Post-Watch List Datamaps

Post-Watch List Datamaps are used to ingest data into various transaction tables using Front Office and Back Office Transaction files, these are executed only after the Watch List Datamaps are run.

These datamaps are used to populate data into the Cash, Wire, and Monetary Instruments tables, and to update Trusted Pair and Jurisdiction information into various other entities.

Note:

The following datamaps can be run in parallel:

- 10970
- 10980
- 10990
- 11000
- 11010
- 11020

Table F-8 AML Banking - Post-Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
20010	CorrespondentBank_JurisdictionUpd	10430 10440
20020	CorrespondentBank_AcctJurisdictionReUpd	10430 10440
20030	FinancialInstitution_InstNameUpd	10430 10440
10960	AccountGroup_JurisdictionUpd	NA
10970	TransactionPartyCrossReference_BackOfficeTransaction	10360 10370 10380 10940
10980	CashTransaction_FrontOfficeTransaction	10360 10370 10380 10940
10990	MonetaryInstrumentTransaction_FrontOfficeTransaction	10360 10370 10380 10940
11000	TransactionPartyCrossReference_FrontOfficeTransaction	10360 10370 10380 10940

Table F-8 (Cont.) AML Banking - Post-Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
11010	WireTransaction_FrontOfficeTransaction	10360 10370 10380 10940
11020	WireTransactionInstitutionLeg_FrontOfficeTransaction	10360 10370 10380 10940
11030	CashTransaction_FrontOfficeTransactionRevAdj	10970 10980 10990 11000 11010 11020
11040	MonetaryInstrumentTransaction_FrontOfficeTransactionRevAdj	10970 10980 10990 11000 11010 11020
11050	WireTransaction_FrontOfficeTransactionRevAdj	10970 10980 10990 11000 11010 11020
11060	TrustedPair_StatusEXPUpd	10970 10980 10990 11000 11010 11020
11070	TrustedPairMember_AcctExtEntEffecRiskUpd	10970 10980 10990 11000 11010 11020
11080	TrustedPair_StatusRRCInsert	10970 10980 10990 11000 11010 11020
11090	TrustedPair_StatusRRCUpd	10970 10980 10990 11000 11010 11020
11100	ApprovalActionsAudit_TrustedPair	10970 10980 10990 11000 11010 11020 11060 11080 11090
11110	TrustedPairMember_StatusRRCInsert	10970 10980 10990 11000 11010 11020
11120	BackOfficeTransaction_TrustedFlagsUpd	11060 11070 11080 11090 11100 11110
11140	MonetaryInstrumentTransaction_TrustedFlagsUpd	11060 11070 11080 11090 11100 11110
11150	WireTransaction_TrustedFlagsUpd	11060 11070 11080 11090 11100 11110

F.9 AML Banking - Summary Datamaps

Summary Datamaps are used to calculate aggregations across various entities using the Trade, Transaction, Positions and Balances tables.

These datamaps populate various profile tables for different entities such as Account Profile, Household Profile, and Correspondent Bank Profile. The aggregation is done either daily, weekly or monthly depending on the business areas.

Optional Datamaps are used to perform processing to support other datamaps in multiple functional areas. These datamaps may or may not be completely relevant to a particular solution set. Execute the datamap if a scenario in your implementation requires this information.

Table F-9 AML Banking - Summary Datamaps

Datamap Number	Datamap Name	Predecessors
11160	AccountDailyProfile-Trade	NA
11170	AccountDailyProfile-Transaction	NA

Table F-9 (Cont.) AML Banking - Summary Datamaps

Datamap Number	Datamap Name	Predecessors
11180	AccountProfile_Trade	11160
11190	AccountProfile_Transaction	11170
11200	AccountProfile_Stage *Optional:* Run the datamap if there is any record in Account Profile Stage.	NA
11210	AccountProfile_Position	11180 11190
11220	AccountProfile_Balance	10940 11160 11170 11180 11190 11210
20040	CorrespondentBankProfile	11180 11190 11200 11210 11220
20050	AccountATMDailyProfile	10940
11230	ChangeLog_AcctProfileInactivity	11180 11190 11200 11210 11220
11240	AccountPeerGroupMonthlyTransactionProfile	11180 11190 11200 11210 11220
20060	CorrespondentBankPeerGroupTransactionProfile	20040
20070	AccountChannelWeeklyProfile	10940
11300	AccountChangeLogSummary	The datamap should be executed once the change log processing is done.
11310	AccountToCustomerChangeLogSummary	The datamap should be executed once the change log processing is done.
11320	CustomerChangeLogSummary	The datamap should be executed once the change log processing is done.

Note:

The AccountChangeLogSummary, AccountToCustomerChangeLogSummary, and CustomerChangeLogSummary datamaps must be run with `execute.sh` from 8.0.2 onwards.

F.10 Fraud Detection - Pre-Watch List Datamaps

Pre-Watch List Datamaps are used to facilitate the application to populate various business areas such as, Financial Institutions, Account To Client Bank, Settlement Instructions, Front Office and Back Office Transaction.

These datamaps populate the relevant data which would be used in watch list datamaps in calculating risks.

Optional Datamaps are used to perform processing to support other datamaps in multiple functional areas. These datamaps may or may not be completely relevant to a particular solution set. Execute the datamap if a scenario in your implementation requires this information.

Table F-10 Fraud Detection - Pre-Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10015	FrontOfficeTransactionParty_SecondaryNames	NA
10020	FinancialInstitution_ThomsonDataInstitutionInsert (*Optional*)	NA
10030	AccountToClientBank_ThomsonDataInstitutionInsert (*Optional*)	10020
10040	FinancialInstitution_AIIMSPopulation	NA
10050	AccountToClientBank_AIIMSInstitutionInsert	10040
10060	AccountToClientBank_InstitutionInsert	10050
10070	AccountToClientBank_InstitutionUpd	10060
10080	FinancialInstitution_FOTPSPopulation	10020 10030 10040 10050 10060 10070
10090	AccountToClientBank_FOTPSInstitutionInsert	10020 10030 10040 10050 10060 10070 10080
10095	AccountCustomerRole	10095
10096	AccountToCustomer	NA
10100	AccountManagementStage	NA
10114	BackOfficeTransaction_UnrelatedPartyCodeUpd	NA
10116	BackOfficeTransaction_CollateralUpd	10114
10120	BackOfficeTransaction_OriginalTransactionReversalUpd	NA
10130	BackOfficeTransaction_CancelledTransactionReversalCreditUpd	NA
10010	EmployeeControlledAccount (*Optional)*	NA
10140	BackOfficeTransaction_CancelledTransactionReversalDebitUpd	NA
10150	FrontOfficeTransactionParty_InstnSeqID	10020 10030 10040 10050 10060 10070
10160	FrontOfficeTransactionParty_HoldingInstnSeqID	10150
10170	FinancialInstitution_AnticipatoryProfile	10020 10030 10040 10050 10060 10070
10180	AccountToClientBank_AnticipatoryProfile	10020 10030 10040 10050 10060 10070 10170
10190	AnticipatoryProfile_AccountToClientBank	10170 10180
10200	CustomerAccountStage_FrontOfficeTransactionParty	NA

Table F-10 (Cont.) Fraud Detection - Pre-Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10210	FrontOfficeTransaction_UnrelatedPartyUpd	10120 10130 10140 10200
10220	FinancialInstitution_SettlementInstruction	10020 10030 10040 10050 10060 10070
10230	AccountToClientBank_SettlementInstruction	10020 10030 10040 10050 10060 10070 10220
10240	SettlementInstruction_AccountToClientBank	10020 10030 10040 10050 10060 10070 10230
10014	FrontOfficeTransaction_PassThroughFlag	NA

Note:

- **FrontOfficeTransaction_PassThroughFlag** - This data map should only be run if the *Pass Through Indicator* field is not being provided in the Front Office Transaction DIS file, and the client requires support to derive this datamap.
- **FrontOfficeTransactionParty_SecondaryNames** - This data map should only be run if *Secondary Originator* and *Secondary Beneficiary* party records are not being provided in the Front Office Transaction Party DIS file, and the client requires support to derive them from the Bank to- Bank Instructions and Originator-to-Beneficiary Instructions fields.

F.11 Fraud Detection - Watch List Datamaps

Watch List Datamaps facilitate the application of customer-supplied measures of risk to corresponding entities, transactions, and instructions.

These datamaps finally assist other datamaps which are used to calculate Effective Risk and Activity Risk for various entities, such as Account, Customer, Transaction, and so on.

Table F-11 Fraud Detection - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10245	WLMProcessingLock	NA
10250	WatchListEntry_WatchListEntryCurrDayInsert	10020 10030 10040 10050 10060 10070 10245
10260	WatchListAudit_StatusUpd	10020 10030 10040 10050 10060 10070
10270	WatchList_WatchListSourceAuditInsert	10020 10030 10040 10050 10060 10070 10260
10280	WatchList_WatchListSourceAuditUpd	10020 10030 10040 10050 10060 10070
10290	WatchList_WatchListSourceUpd	10020 10030 10040 10050 10060 10070

Table F-11 (Cont.) Fraud Detection - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10300	WatchListEntry_WatchListAuditUpd	10020 10030 10040 10050 10060 10070 10260
10310	WatchListEntryAudit_WatchListEntryUpdate	10020 10030 10040 10050 10060 10070 10300
10320	Customer_KYCRiskUpd	NA
10330	DerivedAddress_SettlementInstructionInsert	NA
10340	DerivedAddress_SettlementInstructionUpd	NA
10350	SettlementInstruction_PhysicalDlvryAddrUpd	NA
10360	DerivedAddress_FrontOfficeTransactioPartyStageInsert	NA
10370	DerivedAddress_FrontOfficeTransactioPartyStageUpd	NA
10380	FrontOfficeTransactionParty_DerivedAddress	NA
10390	DerivedEntity_FrontOfficeTransactionPartyInsert	10080 10090
10400	DerivedEntity_FrontOfficeTransactionPartyUpd	10080 10090
10410	DerivedEntity_SettlementInstructionInsert	10220 10230 10240
10420	DerivedEntity_SettlementInstructionUpd	10220 10230 10240
10430	CorrespondentBank_FrontOfficeTransactionPartyStageInsert	10080 10090
10440	CorrespondentBank_FrontOfficeTransactionPartyStageUpd	10080 10090
10450	WatchListStagingTable_WatchList	10250 10260 10270 10280 10290 10300 10310
10460	WatchListStagingTable_WatchListInstnIDUpd	10250 10260 10270 10280 10290 10300 10310
10470	PreviousWatchList_WatchList	10250 10260 10270 10280 10290 10300 10310
10480	DerivedAddress_WatchListNewCountries	10250 10260 10270 10280 10290 10300 10310
10485	WLMProcessingUnlock	10480
10490	LinkStaging_FrontOfficeTransactionParty	10360 10370 10380 10390 10400 10485
10500	LinkStaging_InstructionDerivedEntDerivedAdd	10330 10340 10350 10410 10420
10510	NameMatchStaging	10450 10460 10470 10480 10390 10400
10520	WatchListStagingTable_NameMatchStageInsert	10510
10530	DerivedEntityLink_LinkStage	10490 10500

Table F-11 (Cont.) Fraud Detection - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10540	DerivedEntitytoDerivedAddress_LinkStage	10490 10500
10550	DerivedEntitytoInternalAccount_LinkStage	10490 10500
10560	DerivedAddresstoInternalAccount_LinkStage	10490 10500
10570	WatchListStagingTable2_WatchListStage2AcctExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10580	WatchListStagingTable2_WatchListStage2CBExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10590	WatchListStagingTable2_WatchListStage2CustExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10600	WatchListStagingTable2_WatchListStage2DAExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10610	WatchListStagingTable2_WatchListStage2EEExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10620	WatchListStagingTable2_WatchListStage	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10630	WatchListStagingTable2_AcctListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10640	WatchListStagingTable2_CBListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10650	WatchListStagingTable2_CustListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10660	WatchListStagingTable2_EEListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10670	WatchListStagingTable2_EEListMembershipStatusUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10680	WatchListStagingTable2_DAListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10690	WatchListStagingTable2_DAListMembershipStatusUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10700	WatchListStagingTable2_WatchListStage2SeqIdUpd	10570 10580 10590 10600 10610 10620 10630 10640 10650 10660 10670 10680 10690
10710	WatchListStagingTable2_WatchListStage2IntrlIdUpd	10570 10580 10590 10600 10610 10620 10630 10640 10650 10660 10670 10680 10690

Table F-11 (Cont.) Fraud Detection - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10720	Customer_WatchListStage2ListRisk	10320 10700 10710
10730	CorrespondentBank_WatchListStage2EffectiveRisk	10320 10700 10710
10740	Customer_WatchListStage2EffectiveRisk	10320 10700 10710
10750	DerivedAddress_WatchListStage2EffectiveRisk	10320 10700 10710
10760	DerivedEntity_WatchListStage2EffectiveRisk	10320 10700 10710
10770	WatchListStagingTable2_WatchListStage2SeqId	10320 10700 10710
10780	AccountListMembership_WatchListStage2Insert	10700 10710
10790	AccountListMembership_WatchListStage2Upd	10700 10710
10800	CorrespondentBankListMembership_WatchListStage2Insert	10700 10710
10810	CorrespondentBankListMembership_WatchListStage2Upd	10700 10710
10820	CustomerListMembership_WatchListStage2Insert	10700 10710
10830	CustomerListMembership_WatchListStage2Upd	10700 10710
10840	DerivedAddressListMembership_WatchListStage2Insert	10700 10710
10850	DerivedAddressListMembership_WatchListStage2Upd	10700 10710
10860	DerivedEntityListMembership_WatchListStage2Insert	10700 10710
10870	DerivedEntityListMembership_WatchListStage2Upd	10700 10710
10875	Account_EffectiveRiskFactorTxtUpd	10700 10710
10880	Account_OverallEffectiveRiskUpd	10720 10730 10740 10750 10760 10770 10780 10790 10800 10810 10820 10830 10840 10850 10860 10870
10881	Account_AccountCustRiskUpd	10880
10890	Account_EffRiskUpdAfterWLRiskRemoval	10720 10730 10740 10750 10760 10770 10880
10900	Account_WatchListStage2EffectiveRisk	10720 10730 10740 10750 10760 10770 10880
10910	WatchListStagingTable2_WatchListStage2IntrlId	10320 10700 10710
10920	BackOfficeTransaction_EffectiveAcctivityRiskUpd	10890 10900
10930	SettlementInstruction_EntityActivityRiskUpd	10890 10900

Table F-11 (Cont.) Fraud Detection - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10940	FrontOfficeTransactionPartyRiskStage_EntityActivityRiskInsert	10890 10900

F.12 Fraud Detection - Post-Watch List Datamaps

Fraud Detection - Post-Watch List Datamaps Post-Watch List Datamaps are used to populate or rather ingest data into various transaction tables using Front Office and Back Office Transaction files, these are executed only after the Watch List Datamaps are run.

These datamaps are used to populate data into Cash, Wire, Monetary Instruments tables, and to update Trusted Pair and Jurisdiction information into various other entities.

Note:

The following datamaps can be run in parallel:

- 10970
- 10980
- 10990
- 11000
- 11010
- 11020

Optional Datamaps are used to perform processing to support other datamaps in multiple functional areas. These datamaps may or may not be completely relevant to a particular solution set. Execute the datamap if a scenario in your implementation requires this information.

Table F-12 Fraud Detection - Post-Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10960	AccountGroup_JurisdictionUpd	NA
10970	TransactionPartyCrossReference_BackOfficeTransaction	10360 10370 10380 10940
10980	CashTransaction_FrontOfficeTransaction	10360 10370 10380 10940
10990	MonetaryInstrumentTransaction_FrontOfficeTransaction	10360 10370 10380 10940
11000	TransactionPartyCrossReference_FrontOfficeTransaction	10360 10370 10380 10940 11060 11070 11080 11090
11010	WireTransaction_FrontOfficeTransaction	10360 10370 10380 10940
11020	WireTransactionInstitutionLeg_FrontOfficeTransaction	10360 10370 10380 10940

Table F-12 (Cont.) Fraud Detection - Post-Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
11030	CashTransaction_FrontOfficeTransactionRevAdj	10970 10980 10990 11000 11010 11020
11040	MonetaryInstrumentTransaction_FrontOfficeTransactionRevAdj	10970 10980 10990 11000 11010 11020
11050	WireTransaction_FrontOfficeTransactionRevAdj	10970 10980 10990 11000 11010 11020
11060	TrustedPair_StatusEXPUpd	10970 10980 10990 11000 11010 11020
11070	TrustedPairMember_AcctExtEntEffecRiskUpd	10970 10980 10990 11000 11010 11020
11080	TrustedPair_StatusRRCInsert	10970 10980 10990 11000 11010 11020
11090	TrustedPair_StatusRRCUpd	10970 10980 10990 11000 11010 11020
11100	ApprovalActionsAudit_TrustedPair	10970 10980 10990 11000 11010 11020
11110	TrustedPairMember_StatusRRCInsert	10970 10980 10990 11000 11010 11020
11120	BackOfficeTransaction_TrustedFlagsUpd	11060 11070 11080 11090 11100 11110
11140	MonetaryInstrumentTransaction_TrustedFlagsUpd	11060 11070 11080 11090 11100 11110
11150	WireTransaction_TrustedFlagsUpd	11060 11070 11080 11090 11100 11110

F.13 Fraud Detection - Summary Datamaps Detection

Summary Datamaps are used to calculate aggregations across various entities using Trade, Transaction, Positions and Balances tables.

These datamaps populate various profile tables for different entities, such as Account Profile, Household Profile, and Correspondent Bank Profile. The aggregation is done either daily, weekly or monthly depending on the business areas.

Optional Datamaps are used to perform processing to support other datamaps in multiple functional areas. These datamaps may or may not be completely relevant to a particular solution set. Execute the datamap if a scenario in your implementation requires this information.

Table F-13 Fraud Detection - Summary Datamaps

Datamap Number	Datamap Name	Predecessors
11160	AccountDailyProfile-Trade	NA
11170	AccountDailyProfile-Transaction	10940
11180	AccountProfile_Trade	11160
11190	AccountProfile_Transaction	11170

Table F-13 (Cont.) Fraud Detection - Summary Datamaps

Datamap Number	Datamap Name	Predecessors
11200	AccountProfile_Stage *Optional:* Run the datamap if there is any record in Account Profile Stage	11180 11190
11210	AccountProfile_Position	11180 11190
11220	AccountProfile_Balance	11180 11190 11210
11230	ChangeLog_AcctProfileInactivity	11180 11190 11200 11210 11220
11240	AccountPeerGroupMonthlyTransactionProfile	11180 11190 11200 11210 11220
11300	AccountChangeLogSummary	The datamap should be executed once the change log processing is done.
11310	AccountToCustomerChangeLogSummary	The datamap should be executed once the change log processing is done.
11320	CustomerChangeLogSummary	The datamap should be executed once the change log processing is done.

Note:

The AccountChangeLogSummary, AccountToCustomerChangeLogSummary, and CustomerChangeLogSummary datamaps must be run with `execute.sh` from 8.0.2 onwards

F.14 Insurance - Pre-Watch List Datamaps

Pre-Watch List Datamaps are used to facilitate the application to populate various business areas such as, Financial Institutions, Account To Client Bank, Settlement Instructions, Front Office and Back Office Transaction.

These datamaps populate the relevant data which would again be used in watch list datamaps in calculating risks.

Optional Datamaps are used to perform processing to support other datamaps in multiple functional areas. These datamaps may or may not be completely relevant to a particular solution set. Execute the datamap if a scenario in your implementation requires this information.

Table F-14 Insurance - Pre-Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10010	EmployeeControlledAccount (*Optional*)	NA
10020	FinancialInstitution_ThomsonDataInstitutionInsert (*Optional*)	NA

Table F-14 (Cont.) Insurance - Pre-Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10030	AccountToClientBank_ThomsonDataInstitutionInsert (*Optional*)	10020
10040	FinancialInstitution_AIIMSPopulation	NA
10050	AccountToClientBank_AIIMSInstitutionInsert	10040
10060	AccountToClientBank_InstitutionInsert	10050
10070	AccountToClientBank_InstitutionUpd	10060
10080	FinancialInstitution_FOTPSPopulation	10020 10030 10040 10050 10060 10070
10090	AccountToClientBank_FOTPSInstitutionInsert	10020 10030 10040 10050 10060 10070 10080
10095	AccountCustomerRole	10095
10096	AccountToCustomer	NA
10100	AccountManagementStage	NA
10114	BackOfficeTransaction_UnrelatedPartyCodeUpd	NA
10116	BackOfficeTransaction_CollateralUpd	10114
10150	FrontOfficeTransactionParty_InstnSeqID	10020 10030 10040 10050 10060 10070
10160	FrontOfficeTransactionParty_HoldingInstnSeqID	10150
10170	FinancialInstitution_AnticipatoryProfile	10020 10030 10040 10050 10060 10070
10180	AccountToClientBank_AnticipatoryProfile	10020 10030 10040 10050 10060 10070 10170
10190	AnticipatoryProfile_AccountToClientBank	10020 10030 10040 10050 10060 10070 10180
10220	FinancialInstitution_SettlementInstruction	10020 10030 10040 10050 10060 10070
10230	AccountToClientBank_SettlementInstruction	10020 10030 10040 10050 10060 10070 10220
10240	SettlementInstruction_AccountToClientBank	10020 10030 10040 10050 10060 10070 10230
40010	FinancialInstitution_InsuranceTransaction	10020 10030 10040 10050 10060 10070
40020	AccountToClientBank_InsuranceTransaction	10020 10030 10040 10050 10060 10070 40010
40030	InsuranceTransaction_AccountToClientBank	10020 10030 10040 10050 10060 10070 40020

F.15 Insurance - Watch List Datamaps

Watch List Datamaps facilitate the application of customer-supplied measures of risk to corresponding entities, transactions, and instructions.

These datamaps assist other datamaps which are used to calculate Effective Risk and Activity Risk for various entities, such as, Account, Customer, Transaction tables, and so on.

Table F-15 Insurance - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10245	WLMProcessingLock	NA
10250	WatchListEntry_WatchListEntryCurrDayInsert	10020 10030 10040 10050 10060 10070 10245
10260	WatchListAudit_StatusUpd	10020 10030 10040 10050 10060 10070
10270	WatchList_WatchListSourceAuditInsert	10020 10030 10040 10050 10060 10070
10280	WatchList_WatchListSourceAuditUpd	10020 10030 10040 10050 10060 10070
10290	WatchList_WatchListSourceUpd	10020 10030 10040 10050 10060 10070
10300	WatchListEntry_WatchListAuditUpd	10020 10030 10040 10050 10060 10070
10310	WatchListEntryAudit_WatchListEntryUpdate	10020 10030 10040 10050 10060 10070
10320	Customer_KYCRiskUpd	NA
10360	DerivedAddress_FrontOfficeTransactioPartyStageInsert	NA
10370	DerivedAddress_FrontOfficeTransactioPartyStageUpd	NA
10380	FrontOfficeTransactionParty_DerivedAddress	NA
40040	DerivedAddress_InsuranceTransactionInsert	NA
40050	DerivedAddress_InsuranceTransactionUpd	NA
40060	InsuranceTransaction_InstitutionAddrUpd	NA
40070	DerivedEntity_InsuranceTransactionInsert	40010 40020 40030
40080	DerivedEntity_InsuranceTransactionUpd	40010 40020 40030
10390	DerivedEntity_FrontOfficeTransactionPartyInsert	10080 10090
10400	DerivedEntity_FrontOfficeTransactionPartyUpd	10080 10090
10410	DerivedEntity_SettlementInstructionInsert	10220 10230 10240
10420	DerivedEntity_SettlementInstructionUpd	10220 10230 10240

Table F-15 (Cont.) Insurance - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10430	CorrespondentBank_FrontOfficeTransactionPartyStageInsert	10080 10090
10440	CorrespondentBank_FrontOfficeTransactionPartyStageUpd	10080 10090
10450	WatchListStagingTable_WatchList	10250 10260 10270 10280 10290 10300 10310
10460	WatchListStagingTable_WatchListInstnIDUpd	10250 10260 10270 10280 10290 10300 10310
10470	PreviousWatchList_WatchList	10250 10260 10270 10280 10290 10300 10310
10480	DerivedAddress_WatchListNewCountries	10250 10260 10270 10280 10290 10300 10310
10485	WLMProcessingUnlock	10480
10490	LinkStaging_FrontOfficeTransactionParty	10360 10370 10380 10390 10400
40090	LinkStaging_InsTrxnDerivedEntDerivedAdd	40040 40050 40060 40070 40080
10500	LinkStaging_InstructionDerivedEntDerivedAdd	10330 10340 10350 10410 10420
10510	NameMatchStaging	10450 10460 10470 10480 10390 10400
10520	WatchListStagingTable_NameMatchStageInsert	10510
10530	DerivedEntityLink_LinkStage	40090 10490 10500
10540	DerivedEntitytoDerivedAddress_LinkStage	40090 10490 10500
10550	DerivedEntitytoInternalAccount_LinkStage	40090 10490 10500
10560	DerivedAddresstoInternalAccount_LinkStage	40090 10490 10500
10570	WatchListStagingTable2_WatchListStage2AcctExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10580	WatchListStagingTable2_WatchListStage2CBExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10590	WatchListStagingTable2_WatchListStage2CustExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10600	WatchListStagingTable2_WatchListStage2DAExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10610	WatchListStagingTable2_WatchListStage2EEExistence	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10620	WatchListStagingTable2_WatchListStage	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440

Table F-15 (Cont.) Insurance - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10630	WatchListStagingTable2_AcctListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10640	WatchListStagingTable2_CBListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10650	WatchListStagingTable2_CustListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10660	WatchListStagingTable2_EEListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10670	WatchListStagingTable2_EEListMembershipStatusUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10680	WatchListStagingTable2_DAListMembershipUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10690	WatchListStagingTable2_DAListMembershipStatusUpd	10450 10460 10470 10480 10390 10400 10510 10520 10410 10420 10430 10440
10700	WatchListStagingTable2_WatchListStage2SeqIdUpd	10570 10580 10590 10600 10610 10620 10630 10640 10650 10660 10670 10680 10690
10710	WatchListStagingTable2_WatchListStage2IntrlIdUpd	10570 10580 10590 10600 10610 10620 10630 10640 10650 10660 10670 10680 10690
10720	Customer_WatchListStage2ListRisk	10320 10700 10710
10730	CorrespondentBank_WatchListStage2EffectiveRisk	10320 10700 10710
10740	Customer_WatchListStage2EffectiveRisk	10320 10700 10710
10750	DerivedAddress_WatchListStage2EffectiveRisk	10320 10700 10710
10760	DerivedEntity_WatchListStage2EffectiveRisk	10320 10700 10710
10770	WatchListStagingTable2_WatchListStage2SeqId	10320 10700 10710
10780	AccountListMembership_WatchListStage2Insert	10700 10710
10790	AccountListMembership_WatchListStage2Upd	10700 10710
10800	CorrespondentBankListMembership_WatchListStage2Insert	10700 10710
10810	CorrespondentBankListMembership_WatchListStage2Upd	10700 10710
10820	CustomerListMembership_WatchListStage2Insert	10700 10710

Table F-15 (Cont.) Insurance - Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
10830	CustomerListMembership_WatchListStage2Upd	10700 10710
10840	DerivedAddressListMembership_WatchListStage2Insert	10700 10710
10850	DerivedAddressListMembership_WatchListStage2Upd	10700 10710
10860	DerivedEntityListMembership_WatchListStage2Insert	10700 10710
10870	DerivedEntityListMembership_WatchListStage2Upd	10700 10710
10875	Account_EffectiveRiskFactorTxtUpd	10700 10710
10880	Account_OverallEffectiveRiskUpd	10720 10730 10740 10750 10760 10770 10780 10790 10800 10810 10820 10830 10840 10850 10860 10870
10881	Account_AccountCustRiskUpd	10880
10890	Account_EffRiskUpdAfterWLRiskRemoval	10720 10730 10740 10750 10760 10770 10880
10900	Account_WatchListStage2EffectiveRisk	10720 10730 10740 10750 10760 10770 10880
10910	WatchListStagingTable2_WatchListStage2IntrlId	10320 10700 10710
10940	FrontOfficeTransactionPartyRiskStage_EntityActivityRiskInsert	10890 10900
40100	InsuranceTransaction_EntityActivityRiskUpd	10890 10900

F.16 Insurance - Post-Watch List Datamaps

Post-Watch List Datamaps are used to populate or ingest data into various transaction tables using Front Office and Back Office Transaction files, these are executed only after the Watch List Datamaps are run.

These datamaps are used to populate data into Cash, Wire, Monetary Instruments tables, and to update Trusted Pair and Jurisdiction information into various other entities.

Table F-16 Insurance - Post-Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
11060	TrustedPair_StatusEXPUpd	10970 10980 10990 11000 11010 11020
11070	TrustedPairMember_AcctExtEntEffecRiskUpd	10970 10980 10990 11000 11010 11020
11080	TrustedPair_StatusRRCInsert	10970 10980 10990 11000 11010 11020
11090	TrustedPair_StatusRRCUpd	10970 10980 10990 11000 11010 11020 11060 11070 11080 11090

Table F-16 (Cont.) Insurance - Post-Watch List Datamaps

Datamap Number	Datamap Name	Predecessors
11100	ApprovalActionsAudit_TrustedPair	10970 10980 10990 11000 11010 11020
11110	TrustedPairMember_StatusRRCInsert	10970 10980 10990 11000 11010 11020
11120	BackOfficeTransaction_TrustedFlagsUpd	11060 11070 11080 11090 11100 11110
11130	InsuranceTransaction_TrustedFlagsUpd	11060 11070 11080 11090 11100 11110
11140	MonetaryInstrumentTransaction_TrustedFlagsUpd	11060 11070 11080 11090 11100 11110
11150	WireTransaction_TrustedFlagsUpd	11060 11070 11080 11090 11100 11110

F.17 Insurance - Summary Datamaps

Summary Datamaps are used to calculate aggregations across various entities using Trade, Transaction, Positions and Balances tables.

These datamaps populate various profile tables for different entities such as Account Profile, Household Profile, Correspondent Bank Profile, the aggregation is done either daily, weekly or monthly depending on the business areas. The following table describes the Summary datamaps for Insurance.

> **Note:**
>
> To execute the datamap *WatchListStagingTable_WatchListInstnIDUpd* against 1.5 million records, the tempspace should be set to 400GB or above.

Table F-17 Insurance - Summary Datamaps

Datamap Number	Datamap Name	Predecessors
40110	InsurancePolicyDailyProfile_InsTrxnInsPolicyBal	NA
40120	InsurancePolicyProfile_InsurancePolicyDailyProfile	40110
11300	AccountChangeLogSummary	The datamap should be executed once the change log processing is done.
11310	AccountToCustomerChangeLogSummary	The datamap should be executed once the change log processing is done.
11320	CustomerChangeLogSummary	The datamap should be executed once the change log processing is done.

> **Note:**
>
> The AccountChangeLogSummary, AccountToCustomerChangeLogSummary, and CustomerChangeLogSummary datamaps must be run with `execute.sh` from 8.0.2 onwards.

F.18 Processing BD Datamaps

The following table provides a list of datamaps and description for each datamap.

These datamaps are listed in order.

Table F-18 BD Datamaps

Datamap Number	Datamap Name	Description
10010	EmployeeControlledAccount	This datamap creates entry for Employee personal accounts and Employee Related account using same tax ID
60010	PortfolioManagerPosition	The datamap is used to populate the portfolio manager positions. It reads tables (Account and Account Position), populated while executing Pre-processors and creates records to populate the PORTFOLIO_MGR_POSN table.
60020	AccountGroupProductAllocation	The datamap captures the actual proportionate distribution of holdings for an account group aggregated by reporting classifications.
60030	AccountProductAllocation	The datamap captures the actual proportionate distribution of holdings for an account aggregated by product classifications.
60040	UncoveredOptionExposureDaily	This datamap derives the value from the uncvrd_optns_smry_dly table and insert/updates the records in UNCVRD_OPTNS_EXPOSURE_DLY table.
60050	InvestmentAdvisorProfile	This datamap updates the Investment Manager Summary Month table from the daily activity
60060	RegisteredRepresentativeProfile	This datamap updates the Registered Representative Summary Month table with daily activity
60070	RegOToBorrower	This datamap use the fuzzy match logic to match the Regulation O list against the Borrower.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
60080	InterestedPartyToEmployee	This datamap use fuzzy matcher to match Interested Parties in Account Scheduled Event table against Employee name.
50010	Customer_TotAcctUpd	This datamap calculates the total number of accounts for an institutional customer.
10015	FrontOfficeTransactionParty_SecondaryN ames	This datamap kicks off the Pass Thru process. It generates second orgininator and beneficiary records for Front Office Transaction. It also sets the pass thru flag based on the a set of expressions.
10020	FinancialInstitution_ThomsonDataInstituti onInsert	This datamap builds the many-to-one relationship in INSTN_MASTER that is the relationships between bics and feds with INSTN_SEQ_ID. The INSTN_MASTER table gets populated from BANK_REFERENCE_STAGE table.
10030	AccountToClientBank_ThomsonDataInstit utionInsert	This datamap builds the many-to-one relationship in ACCT_ID_INSTN_ID_MAP that is the relationships between bics and feds with INSTN_SEQ_ID. The ACCT_ID_INSTN_ID_MAP table gets populated from BANK_REFERENCE_STAGE table.
10040	FinancialInstitution_AIIMSPopulation	This datamap inserts new records in Financial Institution table from the ACCT_INSTN_MAP_STAGE table, the datamap creates unique identifiers for banks based on the third party vendors.
10050	AccountToClientBank_AIIMSInstitutionIns ert	This datamap creates unique identifiers for banks based BIC records on the third party vendors. 1) Retrieve Institution information from ACCT_INSTN_MAP_STAGE in comparison of INSTN_MASTER and loads it into ACCT_ID_INSTN_ID_MAP.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10060	AccountToClientBank_InstitutionInsert	This datamap creates unique identifiers for banks based on the third party vendors. 1) Retrieve Institution information from ACCT_INSTN_MAP_STAGE and load it into ACCT_ID_INSTN_ID_MAP.
10070	AccountToClientBank_InstitutionUpd	This datamap updates unique identifiers for banks based on the third party vendors. 1) Retrieve Institution information from ACCT_INSTN_MAP_STAGE and update it into ACCT_ID_INSTN_ID_MAP.
10080	FinancialInstitution_FOTPSPopulation	This datamap inserts new records in Financial Institution table for the institutions found in front office transaction party table for both party ID type code as IA and BIC, INSTN_SEQ_ID are OFSAAI generated.
10090	AccountToClientBank_FOTPSInstitutionIns ert	This datamap marks all institutions with an OFSAAI generated INTSN_SEQ_ID in FOTPS. 1) Prior to this datamap execution the predecessor datamaps finds the new institutions from the transaction data and loads them in the INSTITUTION_MASTER. 2) This data map finds the new institutions from the transaction data for IA and BIC party ID type and loads them in the ACCT_ID_INSTN_ID_MAP table using OFSAAI generated INTSN_SEQ_ID from INSTITUTION_MASTER.
10100	AccountManagementStage	This datamap identifies the relationship between accounts and the employees who have a management role on that account. Management roles include positions such as Financial Advisor, Banker, and Registered Representative.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10110	LoanProfile_LoanProfileStage	This datamap is used to populate Loan Summary from LOAN_SMRY_MNTH_STAGE table. 1) Select set of information/ columns from LOAN_SMRY_MNTH_STAGE table, if the record is new insert the details in LOAN_SMRY_MNTH else update the existing record.
10112	ServiceTeam_SprvsncdUpd	This datamap updates service team table with the Employee Maximum Supervision Code.
10113	InvestmentAdvisor_MangdAcctUpd	This datamap updates ManagedAccountNetworth and ActiveSubAccountCount column in InvestmentAdvisor table.
10114	Security_CIRRatingUpd	This datamap derives the column CIRRating and updates back to Security table.
10116	BackOfficeTransaction_CollateralUpd	This datamap updates Collateral Percentage , Collateral Value for that transaction.
10120	BackOfficeTransaction_OriginalTransactio nReversalUpd	This datamap handles reverserals for Back Office Transactions. 1) Select the set of information from today's BackOfficeTransaction to update records with columns CXL_PAIR_TRXN_INTRL_ID in BackOfficeTransaction table. 2) Updates the "cancellation pair" column in the original back office transaction table as per the "Internal ID" of the reversing or adjusting record.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10130	BackOfficeTransaction_CancelledTransacti onReversalCreditUpd	This datamap updates Cancelled Transaction details for CREDIT record of Back Office Transactions. 1) Finds original-reversal back-office transaction pairs, links them via their respective transaction identifiers. 2) For original transactions: update Canceled Pairing Transaction Identifier by reversal transaction ID; 3) For reversal transactions: update the transaction's Debit Credit Code, Unit Quantity, Transaction Amount, Canceled Pairing Transaction Identifier by original transaction's field values, and Mantas Transaction Adjustment Code by 'REV'.
10140	BackOfficeTransaction_CancelledTransacti onReversalDebitUpd	This datamap updates Cancelled Transaction details for DEBIT record of Back Office Transactions. 1) Finds original-reversal back-office transaction pairs, links them via their respective transaction identifiers. 2) For original transactions: update Canceled Pairing Transaction Identifier by reversal transaction ID; 3) For reversal transactions: update the transaction's Debit Credit Code, Unit Quantity, Transaction Amount, Canceled Pairing Transaction Identifier by original transaction's field values, and Mantas Transaction Adjustment Code by 'REV'.
10150	FrontOfficeTransactionParty_InstnSeqID	This datamap marks all the records of FO_TRXN_PARTY_STAGE table with institutions by OFSAAI generated INTSN_SEQ_ID.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10160	FrontOfficeTransactionParty_HoldingInstn SeqID	This datamap marks all the records of FO_TRXN_PARTY_STAGE table with institutions by OFSAAI generated INTSN_SEQ_ID. 1) To update HOLDG_INSTN_SEQ_ID and HOLDG_ADDR_CNTRY_CD based on DATA_DUMP_DT and country code (BASE_COUNTRY).
10170	FinancialInstitution_AnticipatoryProfile	This datamap inserts new records in Financial Institution table for the institutions found in Anticipatory Profile table, INSTN_SEQ_ID are OFSAAI generated. This datamap should be executed before AccountToClientBank_AnticipatoryProfile datamap as generated INSTN_SEQ_ID will be used to populate Anticipatory Profile table.
10180	AccountToClientBank_AnticipatoryProfile	This datamap marks all institutions with an OFSAAI generated INTSN_SEQ_ID in FOTPS. 1) Prior to this datamap execution the predecessor datamaps finds the new NTCPTRY_PRFL from the transaction data and loads them in the INSTITUTION_MASTER. 2) This data map finds the new institutions from the NTCPTRY_PRFL data and loads them in the ACCT_ID_INSTN_ID_MAP table using OFSAAI generated INTSN_SEQ_ID from INSTITUTION_MASTER.
10190	AnticipatoryProfile_AccountToClientBank	This datamap marks all institutions with an OFSAAI generated INTSN_SEQ_ID in the Anticipatory Profile tables. It should be executed after FinancialInstitution_AnticipatoryProfile and AccountToClientBank_AnticipatoryProfile datamaps are executed.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
50020	DailyAggregateStage	This datamap populates DAILY_AGG_STAGE table with aggregated TRADE Data. DAILY_AGG_STAGE table in turn is used to populate OFFSETING_ACCT_PAIRS and TRADE_DAILY_TOT_CT_STAGE tables.
50030	OffsettingAccountPairStage	This datamap is used to populate OFFSETING_ACCT_PAIRS table by self-joining the table DAILY_AGG_STAGE to generate offsetting trade account pairs.The accounts have the lower ACCT_INTRL_ID while the offsetting accounts have the higher ACCT_INTRL_ID.
50040	TradeDailyTotalCountStage	This datamap aggregates the total trades done by that account for the current processing day.
10200	CustomerAccountStage_FrontOfficeTrans actionParty	This datamap populates the Customer Account Stage table with the Cust-Acct pairs which appears inFOTPS with Party type as IA.
10210	FrontOfficeTransaction_UnrelatedPartyUp d	This datamap updates the FOT table for records where UNRLTD_PARTY_FL is 'Y' with a value as 'N', by determining the pairs of parties (internal) in the role of Orig & Benef having either common Tax ID/Common Customer/Common HH.
10220	FinancialInstitution_SettlementInstruction	This datamap inserts new records in Financial Institution records for the institutions found in INSTRUCTION that have not been previously identified, INSTN_SEQ_ID are OFSAAI generated. This datamap should be executed before AccountToClientBank_Settlement Instruction datamap.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10230	AccountToClientBank_Settlement Instructi on	This datamap marks all institutions with an OFSAAI generated INTSN_SEQ_ID in FOTPS. 1) Prior to this datamap execution the predecessor datamaps finds the new INSTRUCTION from the transaction data and loads them in the INSTITUTION_MASTER. 2) This data map finds the new institutions from the INSTRUCTION data and loads them in the ACCT_ID_INSTN_ID_MAP table using OFSAAI generated INTSN_SEQ_ID from INSTITUTION_MASTER. .
10240	SettlementInstruction_AccountTo ClientBa nk	This datamap updates Destination Institution and Physical Delivery Institution in INSTRUCTION table using the values from ACCT_ID_INSTN_ID_MAP table.
40010	FinancialInstitution_InsuranceTra nsaction	This datamap inserts new records in Financial Institution table for the institutions found in Insurance Transactions, INSTN_SEQ_ID are OFSAAI generated. This datamap should be executed before AccountToClientBank_Insurance Transaction datamap as generated INSTN_SEQ_ID will be used to populate Anticipatory Profile table.
40020	AccountToClientBank_Insurance Transacti on	This datamap marks all institutions with an OFSAAI generated INTSN_SEQ_ID in FOTPS. 1) Prior to this datamap execution the predecessor datamaps finds the new institutions from the transaction data and loads them in the INSTITUTION_MASTER. 2) This data map finds the new institutions from the INSURANCE_TRXN data and loads them in the ACCT_ID_INSTN_ID_MAP table using OFSAAI generated INTSN_SEQ_ID from INSTITUTION_MASTER.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
40030	InsuranceTransaction_AccountToClientBa nk	This datamap marks all institutions with an OFSAAI generated Institution Identifier in Insurance Transaction records. 1) Prior to this datamap execution Financial Institution and Account To Client Bank records are inserted. 2) Henceforth this datamap uses the Account To Client Bank table and updates Institution Identifier in Insurance table.
10245	WLMProcessingLock	This datamap applies lock to restrict UI accessibility for Watch list Management.
10250	WatchListEntry_WatchListEntryCurrDayIns ert	This datamap checks for records in watch list from source files for the current day, if there is no records, create the current day watch list records from the previous day.
10260	WatchListAudit_StatusUpd	This datamap take care of watchlist table for the modifications of the WL based on the new user interface WL utility.
10270	WatchList_WatchListSourceAuditInsert	This datamap takes into account the modifications of the watchlist based on the new user interface WL utility. 1) Get all the records that are active from audit table. Order by created time. 2) Take the latest change for each LIST_SRC_CD Watch List and insert records in WATCH_LIST_SOURCE table.
10280	WatchList_WatchListSourceAuditUpd	This datamap takes into account the modifications of the watchlist based on the new user interface WL utility. 1) Get all the records that are active from audit table. Order by created time. 2) Take the latest change for each LIST_SRC_CD Watch List and update records in WATCH_LIST_SOURCE table.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10290	WatchList_WatchListSourceUpd	This datamap takes into account the modifications of the watchlist based on the new user interface WL utility. 1) Get all the records that are active from audit table. Order by created time. 2) Take the latest change for each LIST_SRC_CD Watch List and update records in WATCH_LIST_SOURCE table.
10300	WatchListEntry_WatchListAuditUpd	This datamap takes care of watch list entry table for the modifications of the WL based on the new user interface WL utility.
10310	WatchListEntryAudit_WatchListEntryUpda te	Thisdatamap take care of watchlist entry audit table for the modifications of the WL based on the new user interface WL utility.
10320	Customer_KYCRiskUpd	This datamap calculates risk, If the risk was List driven, then this can ignore that record. If it was BUS/GEO driven and there is KYC risk. Apply KYC Risk in Customer table.
60090	CorrespondentBankToPeerGroup	This datamap populates the CLIENT_BANK_PEER_GRP table by associating peer group identifiers in the ACCT_PEER_GRP table with institution identifiers in the ACCT_ID_INSTN_ID_MAP table.
10330	DerivedAddress_SettlementInstructionIns ert	This datamap inserts new addresses in the Derived Address table. It derives the addresses from the INSTRUCTION table.
10340	DerivedAddress_SettlementInstructionUp d	This datamap derives the addresses from the INSTRUCTION table. It updates addresses in the Derived Address table, if already existing.
10350	SettlementInstruction_PhysicalDlvryAddr Upd	This datamap updates Mantas Physical Delivery Address Identifier in INSTRUCTION table.
10360	DerivedAddress_FrontOfficeTransactioPar tyStageInsert	This datamap selects the distinct set of addresses from today's front-office transactions and if non-existent, inserts new address records into Derived Address.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10370	DerivedAddress_FrontOfficeTransactioPar tyStageUpd	This datamap selects the distinct set of addresses from today'sfront-office transactions and if existent, updates new address records into Derived Address.
10380	FrontOfficeTransactionParty_DerivedAddr ess	This datamap maintains the addresses in the DerivedAddress table. It derives the addresses from the FrontOfficeTransactionParty table.
40040	DerivedAddress_InsuranceTransactionInse rt	This datamap derives the addresses from the INSURANCE table, and inserts the addresses in to the Derived Address table.
40050	DerivedAddress_InsuranceTransactionUpd	This datamap derives the addresses from the INSURANCE table. If the address already exists in Derived Address table, it will update the addresses in to the Derived Address table.
40060	InsuranceTransaction_InstitutionAddrUpd	This datamap updates Mantas Institution Address Identifier in the Insurance Transaction table. 1) A new record is created in Derived Address table prior to this datamap execution. 2) Update the same Derived Address Sequence ID in INSURANCE_TRXN for CP_ADDR_MSTR_SEQ_ID column.
40070	DerivedEntity_InsuranceTransactionInsert	This datamap maintains the External Entity table. It derives the entities from the INSURANCE table on current processing date.
40080	DerivedEntity_InsuranceTransactionUpd	This datamap maintains the External Entity table. It derives the entities from the INSURANCE table on current processing date.
10390	DerivedEntity_FrontOfficeTransactionPart yInsert	This datamap maintains the External Entity table. It derives the entities from the Front Office and Front Office Party transaction table.
10400	DerivedEntity_FrontOfficeTransactionPart yUpd	This datamap maintains the External Entity table. It derives the entities from the Front Office and Front Office Party transaction table.
10410	DerivedEntity_SettlementInstructionInsert	This datamap maintains the External Entity table. It derives the entities from the Instruction table on current processing date.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10420	DerivedEntity_SettlementInstructionUpd	This datamap maintains the External Entity table. It derives the entities from the INSTRUCTION table. 1) Select the distinct set of names, accounts, institutions from today's Instructions and updates matching records in the External Entity table.
10430	CorrespondentBank_FrontOfficeTransacti onPartyStageInsert	This datamap populates the client bank table for current day transactions where there is an institution involved.
10440	CorrespondentBank_FrontOfficeTransacti onPartyStageUpd	This datamap maintains the Correspondent Bank table. Itderives the records from the FOTPS table. If there is an existing correspond bank record available, this datamap updates the LAST_ACTVY_DT for that record.
10450	WatchListStagingTable_WatchList	This datamap determines changes in the Watch List table Each entry is classified as Add, No Change, or Retire based on the comparison of the current-day watch list data to the previous-day watch list data.
10460	WatchListStagingTable_WatchList InstnIDU pd	This datamap only processes watch list entries that are External Accounts, Financial Institutions, and Internal Accounts. 1) It updates the Watch List Stage table with the corresponding Institution Sequence ID of the institution or account.
10470	PreviousWatchList_WatchList	This datamap save off current day's watch list records into PREV_WATCH_LIST
10480	DerivedAddress_WatchListNewCountries	This datamap inserts new countries from WL in the derived addresses table.
10485	WLMProcessingUnlock	This datamap releases the lock for Watch list Management.
10490	LinkStaging_FrontOfficeTransactionParty	This datamap loads the Link Stage with any entity associations from FOTPS, depending on the combination of Link Type Code defined.
40090	LinkStaging_InsTrxnDerivedEntDerivedAd d	This datamap loads the Link Stage with any entity associations from INSURANCE.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10500	LinkStaging_InstructionDerivedEntDerive dAdd	This datamap loads the Link Stage with any entity associations from instruction. Define the entity association based on existence of entity and address associations in data.
10510	NameMatchStaging	This datamap use fuzzy match to matchCandidate Name against the List Name and inserts records in Name Match Stage table.
10520	WatchListStagingTable_NameMatchStageI nsert	This datamap is a wrapper for the fuzzy matching mappings and scripts. 1) For each processing day, this datamap joins fuzzy names to their matched watch list records to create additional watch list records for subsequent application to transactional tables.
10530	DerivedEntityLink_LinkStage	This datamap selects the external entity links from today's Link Stage table and insert records in External Entity Link table in associations to various link tables.
10540	DerivedEntitytoDerivedAddress_LinkStage	This datamap writes link-stage associations to various link tables in External Entity Address Table.
10550	DerivedEntitytoInternalAccount_LinkStage	This datamap writes link-stage associations to various link tables in External Entity Account Table.
10560	DerivedAddresstoInternalAccount_LinkSta ge	This datamap writes link-stage associations to various link tables in Derived Account Address Table.
10570	WatchListStagingTable2_WatchListStage2 AcctExistence	This datamap validates each watch list entry and inserts into the processing table WATCH_LIST_STAGE2. 1) Processes all watch list entries that have a possible match with ACCT entity. 2) For IA (ACCT table) watch list entries, the error status is assigned if the entity does not exist in the entity table because these entity records are expected to exist.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10580	WatchListStagingTable2_WatchListStage2 CBExistence	This datamap validates each watch list entry and inserts into the processing table WATCH_LIST_STAGE2. 1) Processes all watch list entries that have a possible match with CLIENT_BANK entity. 2) Evaluates the existence of the CLIENT_BANK entity and assigns a 'Warning"" status to the record if the entity does not exist in the entity table because these entity records are expected to exist.
10590	WatchListStagingTable2_WatchListStage2 CustExistence	This datamap validates each watch list entry and inserts into the processing table WATCH_LIST_STAGE2. 1) Processes all watch list entries that have a possible match with CUST entity. 2) For CU (CUST table) watch list entries, the error status is assigned if the entity does not exist in the entity table because these entity records are expected to exist.
10600	WatchListStagingTable2_WatchListStage2 DAExistence	This datamap validates each watch list entry and inserts into the processing table WATCH_LIST_STAGE2. 1) Processes all watch list entries that have a possible match with DERIVED_ADDRESS entity. 2) Evaluates the existence of the DERIVED_ADDRESS record and assigns status to the record accordingly.
10610	WatchListStagingTable2_WatchListStage2 EEExistence	This datamap validates each watch list entry and inserts into the processing table WATCH_LIST_STAGE2. 1) Processes all watch list entries that have a possible match with EXTERNAL_ENTITY entity. 2) Evaluates the existence of the EXTERNAL_ENTITY record and assigns a 'Warning""""" status to the record if the entity does not exist in the entity table because these entity records are expected to exist.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10620	WatchListStagingTable2_WatchListStage	Thisdatamap validates each watch list entry and inserts into the processing table WATCH_LIST_STAGE2. 1) Check for watch list stage CUST_INTRL_ID flag if it is 'Y' means that this name is fuzzy matched. 2) Insert the watch list entry into the second processing table that is Watch list stage 2 table for both the fuzzy matched as well as exact name records.
10630	WatchListStagingTable2_AcctListMembers hipUpd	The datamap checks for entry membership in the corresponding entity list membership table.
10640	WatchListStagingTable2_CBListMembersh ipUpd	This datamap validates each watch list entry and inserts into the processing table WATCH_LIST_STAGE2. 1) Processes all watch list entries that have a possible match with CB_LIST_MEMBERSHIP entity. 2) Evaluates the existence of the CB_LIST_MEMBERSHIP record and assigns a 'Warning"" status to the record if the entity does not exist in the entity table because these entity records are expected to exist.
10650	WatchListStagingTable2_CustListMember shipUpd	This datamap validates each watch list entry and inserts into the processing table WATCH_LIST_STAGE2. 1) Processes all watch list entries that have a possible match with CUST_LIST_MEMBERSHIP entity. 2) Evaluates the existence of the CUST_LIST_MEMBERSHIP record and assigns a 'Warning"" status to the record if the entity does not exist in the entity table because these entity records are expected to exist.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10660	WatchListStagingTable2_EEListMembershi pUpd	This datamap validates each watch list entry and inserts into the processing table WATCH_LIST_STAGE2. 1) Processes all watch list entries that have a possible match with EXTERNAL_NTITY_LIST_MEMBERSHIP entity. 2) Evaluates the existence of the EXTERNAL_NTITY_LIST_MEMBERSHIP record and assigns a 'Warning"" status to the record if the entity does not exist in the entity table because these entity records are expected to exist.
10670	WatchListStagingTable2_EEListMembershi pStatusUpd	This datamap validates each watch list entry and inserts into the processing table WATCH_LIST_STAGE2. 1) It validates the list membership status of External Entities whose Last Activity Date is earlier than the current date. 2) Update the status of the watch list entry based the existence or non-existence of a corresponding list membership record.
10680	WatchListStagingTable2_DAListMembersh ipUpd	This datamap validates each watch list entry and inserts into the processing table WATCH_LIST_STAGE2. 1) Processes all watch list entries that have a possible match with DERIVED_ADDR_LIST_MEMBERSHIP entity. 2) Evaluates the existence of the DERIVED_ADDR_LIST_MEMBERSHIP record and assigns a 'Warning"" status to the record if the entity does not exist in the entity table because these entity records are expected to exist.
10690	WatchListStagingTable2_DAListMembersh ipStatusUpd	This datamap validates each watch list entry and inserts into the processing table WATCH_LIST_STAGE2. 1) It validates the list membership status of DERIVED_ADDRESS whose Last Activity Date is earlier than the current date. 2) Update the status of the watch list entry based the existence or non-existence of a corresponding list membership record.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10700	WatchListStagingTable2_WatchListStage2 SeqIdUpd	This datamap updates the list risk of each valid watch list entity based on the entity Sequence ID. The datamap sets various flags and derives the highest List Risk value for each entity on the watch list.
10710	WatchListStagingTable2_WatchListStage2I ntrIIdUpd	This datamap updates the list risk of each valid watch list entity based on the entity Internal ID. The datamap sets various flags and derives the highest List Risk value for each entity on the watch list.
10720	Customer_WatchListStage2ListRisk	This datamap calculates the customer's effective risk and set the risk factor if the risk is not found for the current day in watch list stage table. After calculating the risk updates the CUST table. Use nulls for the List Risk and the List Source Code.
10730	CorrespondentBank_WatchListStage2Effe ctiveRisk	This datamap calculates the Client Bank Effective Risk and applies the Effective Risk and the List Risk to the CLIENT_BANKrecord.
10740	Customer_WatchListStage2EffectiveRisk	This datamap calculates the Effective Risk of Customer and applies the Effective Risk and the List Risk to the CUST record.
10750	DerivedAddress_WatchListStage2Effective Risk	This datamap calculates the Effective Risk of all derived address entities and applies the Effective Risk and the List Risk to the DERIVED_ADDRESS record.
10760	DerivedEntity_WatchListStage2EffectiveRi sk	This datamap calculates the Effective Risk of all external entities and applies the Effective Risk and the List Risk to the EXTERNAL_ENTITY record.
10770	WatchListStagingTable2_WatchListStage2 SeqId	This datamap calculates the Effective Risk of all entities and applies the Effective Risk and the List Risk to the entity record where sequence ID is not null.
10780	AccountListMembership_WatchListStage2 Insert	This datamap inserts List Membership records for entities into ACCT_LIST_MEMBERSHIP table that are new to a list.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10790	AccountListMembership_WatchListStage2 Upd	This datamap updates the existing retired ACCT_LIST_MEMBERSHIP records by setting List Removal Date to the current processing date.
10800	CorrespondentBankListMembership_Watc hListStage2Insert	This datamap inserts List Membership records for entities that are new to a list into CB_LIST_MEMBERSHIP table.
10810	CorrespondentBankListMembership_Watc hListStage2Upd	This datamap updates the existing retired CB_LIST_MEMBERSHIP records by setting List Removal Date to the current processing date.
10820	CustomerListMembership_WatchListStage 2Insert	This datamap inserts List Membership records for entities that are new to a list into CUST_LIST_MEMBERSHIP table.
10830	CustomerListMembership_WatchListStage 2Upd	This datamap updates the existing retired CUST_LIST_MEMBERSHIP records by setting List Removal Date to the current processing date.
10840	DerivedAddressListMembership_WatchLis tStage2Insert	This datamap maintains the Derived Address List membership table based on the current WL processing results.
10850	DerivedAddressListMembership_WatchLis tStage2Upd	This datamap maintains the Derived Address List membership tables based on the current WL processing results by setting List Removal Date to the current processing date.
10860	DerivedEntityListMembership_WatchListS tage2Insert	This datamap inserts List Membership records for entities that are new to a list into EXTERNAL_NTITY_LIST_MEMBERSHIP table.
10870	DerivedEntityListMembership_WatchListS tage2Upd	This datamap maintains the External Entity membership tables based on the current WL processing results by setting List Removal Date to the current processing date.
10880	Account_OverallEffectiveRiskUpd	This datamap updates the risk on the ACCT based on KYC, Primary customer, as well as other external risks.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10881	Account_AccountCustRiskUpd	This data map updates the risk on the ACCT based on KYC, Primary customer, as well as other external risks.
10890	Account_EffRiskUpdAfterWLRisk Removal	This datamap Updates the account Effective Risk to the maximum of the business risk, geographic risk, and customer risk. The account Effective Risk was already set to the higher of the customer-supplied business and geography risk. List risk is ignored here, as this mapping is where we're removing list risk.
10900	Account_WatchListStage2Effectiv eRisk	This datamap calculates all risk related values like Effective Risk of Acct and applies the Effective Risk, List Risk to the ACCT record.
10910	WatchListStagingTable2_WatchLi stStage2I ntrlId	This datamap calculates the Effective Risk of all entities and applies the Effective Risk and the List Risk to the entity record based on NTITY_INTRL_ID.
10920	BackOfficeTransaction_EffectiveA cctivityR iskUpd	This datamap updates the risk related values to all parties involved in Back Office Transaction. 1) Select risk values from BACK_OFFICE_TRXN, ACCT, Offset Account in the sub query. 2) Derive the effective and activity risks from the transaction. 3) Update BACK_OFFICE_TRXN table using BO_TRXN_SEQ_ID in the main query.
10930	SettlementInstruction_EntityAcctivityRisk Upd	This datamap updates Entity Risk and Activity Risk in INSTRUCTION table
10940	FrontOfficeTransactionPartyRiskStage_En tityActivityRiskInsert	This datamap populates the Effective Risk and Activity Risk related values to all the parties in FO_TRXN_PARTY_RISK_STAGE table.
10955	AccountGroup_InvestmentObjectiveUpd	This datamap updates Investment Objective column in Account Group table.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
40100	InsuranceTransaction_EntityActivityRisk Upd	This datamap updates the risk related values to all parties in Insurance Transaction. 1) Select different risk related values from various tables like watchlist, external entity and derived address etc. 2) Updates Entity Risk and Activity Risk in INSURANCE_TRXN table.
20010	CorrespondentBank_Jurisdiction Upd	This datamap updates the JRSDCN_CD and BUS_DMN_LIST_TX for an existing client bank record where either the JRSDCN_CD or the BUS_DMN_LIST_TX is null.
20020	CorrespondentBank_AcctJurisdictionReUp d	This datamap updates the jurisdiction for CLIENT_BANK (Correspondent Bank).
20030	FinancialInstitution_InstNameUpd	This datamap updates INSTN_NM for an existing INSTN_MASTER record.
10960	AccountGroup_JurisdictionUpd	This datamap updates the primary account in a HH with the jurisdiction & business domain present in Account table for it.
10970	TransactionPartyCrossReference_BackOffi ceTransaction	This datamap is used to build the record for Transaction Party Cross Reference table from today's Back Office Transactions. 1) Select the set of information from today's Back Office Transactions and insert records in Transaction Party Cross Reference table. 2) Parameter ProcessTransactionXRefFlag = 'N' or 'Y' accordingly.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
10980	CashTransaction_FrontOfficeTransaction	This datamap is used to build the record for Cash Transaction Table from today's Front Office Transaction and Front Office Transaction Party. 1) Select the set of Cash Transaction categories information from today's Front Office Transaction and Front Office Transaction Party to Insert records In Cash Transaction Table. 2) Some fields are not null-able. The NVL function is used in the SQL to plug the default values in place of a null. Also, various "NB" fields are set to zero whenever they are null in the expression prior to the inserting them into the target table.
10990	MonetaryInstrumentTransaction_FrontOff iceTransaction	This datamap select the set of information from today's Front Office Transaction and Front Office Transaction Party to Insert records In Monetary Instrument Transaction Table.
11000	TransactionPartyCrossReference_FrontOff iceTransaction	This datamap is used to build the record for Transaction Party Cross Reference table from today's Front Office Transaction and Front Office Transaction Party. 1) Select the set of information from today's Front Office Transaction and Front Office Transaction Party to Insert records In Transaction Party Cross Reference Table. 2) Some fields are not null-able. The NVL function is used in the SQL to plug the default values in place of a null. Also, various "NB" fields are set to zero whenever they are null in the expression prior to the inserting them into the target table. 3) Parameter ProcessTransactionXRefFlag = 'N' or 'Y' accordingly.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
11010	WireTransaction_FrontOfficeTransaction	This datamap is used to build the record for Wire Transaction Table from today's Front Office Transaction and Front Office Transaction Party. 1) Select the set of Wire Transaction categories information from today's Front Office Transaction and Front Office Transaction Party to Insert records In Wire Transaction Table. 2) Some fields are not null-able. The NVL function is used in the SQL to plug the default values in place of a null. Also, various "NB" fields are set to zero whenever they are null in the expression prior to the inserting them into the target table. 3) Parameter ProcessBankToBank= 'N' or 'Y' accordingly.
11020	WireTransactionInstitutionLeg_FrontOffic eTransaction	This datamap is used to build the record for Wire Transaction Institution Leg Table from today's Front Office Transaction and Front Office Transaction Party. 1) Select the set of Wire Transaction categories and it should have more than 1 leg information from today's Front Office Transaction and Front Office Transaction Party to Insert records In Wire Transaction Institution Leg Table. 2) Some fields are not null-able. The NVL function is used in the SQL to plug the default values in place of a null. Also, various "NB" fields are set to zero whenever they are null in the expression prior to the inserting them into the target table. 3) Parameter ProcessBankToBank = 'N' or 'Y' accordingly.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
11030	CashTransaction_FrontOfficeTransactionR evAdj	This datamap adjusts the reversals for Cash Transaction table. 1) Select the set of information from today's Front Office Transaction to update records with columns CXL_PAIR_TRXN_INTRL_ID, REBKD_TRXN_INTRL_ID in Cash Transaction table.
11040	MonetaryInstrumentTransaction_FrontOff iceTransactionRevAdj	This datamap adjusts the reversals for front office transaction tables in Monetary Instrument Transaction table
11050	WireTransaction_FrontOfficeTransactionR evAdj	This datamap adjusts the reversals for Wire Transaction table. 1) Select the set of information from today's Front Office Transaction to update records with columns CXL_PAIR_TRXN_INTRL_ID, REBKD_TRXN_INTRL_ID in Wire Transaction table.
50050	CustomerDailyProfile_BOT	This datamap aggregates Back Office Transaction data by Customer and Date and updates into CUST_SMRY_DAILY table.
50060	CustomerDailyProfile_FOTPS	This datamap aggregates Front Office Transaction data by Customer and Date and updates into CUST_SMRY_DAILY table.
50070	InstitutionalAccountDailyProfile_DEAL	This datamap updates INSTL_ACCT_SMRY_DAILY table from Deal, grouping by account and data dump date.
50080	CustomerDailyProfile_DEAL	This datamap updates CUST_SMRY_DAILY table from Structured Deal, grouping by customer and data dump date.
50090	InstitutionalAccountDailyProfile_INST	This datamap updates INSTL_ACCT_SMRY_DAILY table from Instruction, grouping by account and data dump date.
50100	CustomerDailyProfile_INST	This datamap updates CUST_SMRY_DAILY table from Instruction data, grouping by Customer and data dump date.
50110	InstitutionalAccountDailyProfile_CorpActi on	This datamap aggregates institutional trading activity, grouping by Account ID and data dump date.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
50120	CustomerDailyProfile_CorpAction	This datamap aggregates Corporate Action trading activity, grouping by Customer ID.
50130	InstitutionalAccountDailyProfile_Trade	This datamap updates INSTL_ACCT_SMRY_DAILY table from Trade, grouping by account and data dump date.
50140	CustomerDailyProfile_Trade	This datamap updates CUST_SMRY_DAILY table from Trade data, grouping by customer and data dump date.
60100	ManagedAccountDailyProfile_SameDayTr ade	This datamap is used for the daily aggregation of the block allocation day trades data. This populates the managed account daily summary.
60110	ManagedAccountDailyProfile_Trade	This datamap is used for the daily aggregation of the block allocation trades data. This populates the managed account daily summary .
60120	ManagedAccountDailyProfile_BOT	This datamap populates MANGD_ACCT_SMRY_DAILY table using Back Office Transaction.
11160	AccountDailyProfile-Trade	This datamap performs daily aggregation of trades fromtrade table , Profit Loss from Account Realized Profit Loss table.
11170	AccountDailyProfile-Transaction	This datamap populates the table ACCT_TRXN_SMRY_DAILY using both Front office and Back Office transaction for that account on current processing date.
11180	AccountProfile_Trade	This datamap populates the table ACCT_SMRY_MNTH using ACCT_TRADE_SMRY_DAILY table for that account starting from Month Start date till current processing date.
11190	AccountProfile_Transaction	This datamap populates the table ACCT_SMRY_MNTH using ACCT_TRXN_SMRY_DAILY table for that account starting from Month Start date till current processing date.
11200	AccountProfile_Stage	This datamap populates the table ACCT_SMRY_MNTH using ACCT_PRFL_STAGE table for that account starting from Month Start date till current processing date.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
11210	AccountProfile_Position	This datamap populates the table ACCT_SMRY_MNTH using ACCT_POSN table for that account starting from Month Start date till current processing date. Updates values by calculating aggregate values for AGGR_SHRT_PUT_EXPSR_AM, AGGR_SHRT_CALL_EXPSR_AM, SHRT_PUT_EXPSR_RATIO and SHRT_CALL_EXPSR_RATIO for each account internal ID present in ACCT_SMRY_MNTH.
11220	AccountProfile_Balance	This datamap populates the ACCT_SMRY_MNTH table usingACCT_BAL_POSN_SMRY. If there is already record in Account summary Month for Account and Month Start Date, then it will update the record. Else it will do insert, remaining columns defaulted to 0.
60130	HouseholdProfile	This datamap aggregates monthly account summaries into their respective households. All monthly records must be processed each day since account households are subject to change daily.
50150	InstitutionalAccountProfile	This datamap performs Insert or Update of Institutional Account Summary Month Table from its corresponding Daily table. Aggregate daily activity with counts and amounts for the current month. If already record exists for the account in the current month, the datamap will update the record, else insert a new record.
50160	CustomerProfile	This Datamap loads into CUST_SMRY_MNTH from CUST_SMRY_DAILY table. Check for the customer record exists for t he month, if record not available Insert records in CUST_SMRY_MNTH table
60140	ManagedAccountProfile	This datamap updates the Managed AccountSummary Month Table from its corresponding Managed Account Daily Summary table.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
60145	AccountPosition_PercentofPortfolioUpd	This datamap updates Percent of Portfolio column in Account Position table.
20040	CorrespondentBankProfile	This datamap performs daily re-aggregation of the Correspondent Bank Summary Month table out of the account summary month table.
20050	AccountATMDailyProfile	This datamap calculates the total Transaction Amount for Account ATM Daily Profile Select information from Front Office Transaction, Account and Account ATM Daily Profile and insert or update (if record exist) into ACCT_ATM_SMRY_DAILY
11230	ChangeLog_AcctProfileInactivity	This datamap creates Change Log records that indicate a change in an accounts activity level as measured by the sum of deposits, withdrawals, and trades over a configurable time period (months).
11240	AccountPeerGroupMonthlyTransactionPro file	This datamap calculates average values and insert into Account Peer Group Monthly Transaction Profile. Select and calculate average values for withdrawal amount and count from ACCT_SMRY_MNTH table Insert the above values into ACCT_PEER_TRXN_SMRY_MNTH.
20060	CorrespondentBankPeerGroupTransaction Profile	This datamaps populate CorrespondentBankPeerGroupTransactionProfile from Client Bank Summary Month. 1) Select set of information from CLIENT_BANK_SMRY_MNTH, CLIENT_BANK_PEER_GRP 2) Data is populated in the target table after aggregating the required columns.
20070	AccountChannelWeeklyProfile	This datamap populates the table ACCT_CHANL_SMRY_WKLY using FO_TRXN, BACK_OFFICE_TRXN table for that account starting from Weekly Start date till current processing date.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
40110	InsurancePolicyDailyProfile_InsTrxnInsPoli cyBal	This datamap performs inserts or updates of Insurance Policy Summary Daily Table from the Insurance Transaction table on the current processing day.
40120	InsurancePolicyProfile_InsurancePolicyDail yProfile	This datamap performs updates of Insurance Policy Summary Month Table using the values from Insurance Policy Daily Profile table. 1) Records are inserted into Insurance Policy Daily Profile table prior to this datamap execution. 2) This datamap inserts new records or Updates matched records in Insurance Policy Profile table using the values from Insurance Policy Daily Profile table.
50170	CustomerBalance_ActiveOTCTradeCtUpd	This datamap counts the records in the Deal table which has an end date greater than or equal to the current date by customer and update the ACTV_OTC_TRD_CT column in customer balance table.
60150	AccountPositionDerived	This datamap processes account option position pair data and updates the corresponding account position records. Updates are made to attributes relating to uncovered option contracts
60160	AccountBalance_AcctPosnPair	This datamap processes account option position pair data and updates the corresponding account balance records. Updates are made to option market value long attributes.
60170	AccountBalance_Acctposn	This datamap aggregates current-day security positions by product category and account for update of the account balance record. Rejoins for single update to avoid deadlocks.
60180	HouseholdBalance	This datamap aggregates daily records of account balances data and inserts into household balances table based household group id.

Table F-18 (Cont.) BD Datamaps

Datamap Number	Datamap Name	Description
11300	AccountChangeLogSummary	This datamap inserts new records to the ACCT_CHG_LOG_SMRY table. The datamap should be executed once the change log processing is done.
11310	AccountToCustomerChangeLogSummary	This datamap inserts new records to the CUST_ACCT_CHG_LOG_SMRY. The datamap should be executed once the change log processing is done.
11320	CustomerChangeLogSummary	This datamap inserts new records to the CUST_CHG_LOG_SMRY table. The datamap should be executed once the change log processing is done.

Note:

The AccountChangeLogSummary, AccountToCustomerChangeLogSummary, and CustomerChangeLogSummary datamaps must be run with `execute.sh` from 8.0.2 onwards.

F.19 Firm Data Transfer Datamaps

The following table lists the Firm Data Transfer (FDT) Datamaps and the order they must be run in.

Table F-19 FDT Datamaps

Datamap Number	Datamap Name	Predecessors
70010	Scrty_TradeExecutionStageInsert	NA
70020	Scrty_OrderStageInsert	NA
70030	MktCntr_OrderStageInsert	NA
70040	OrderStage_DQupdate	NA
70050	TradeExecutionEventStage_DQupdate	NA
70060	OrderStage_FDTupdate	70040
70070	OrderStage_RmngQtupdate	70040 70060
70080	OrderSummary	70040 70060 70070
70090	OrderSummary_OpenOrdrInsrt	70040 70060 70070 70080
70100	OrderSummary_QtyUpdate	70040 70060 70070 70080 70090
70110	OrderStage_OpenOderUpd	70040 70060 70070 70080 70090 70100

Table F-19 (Cont.) FDT Datamaps

Datamap Number	Datamap Name	Predecessors
70120	OrderSummary_Update	70040 70060 70070 70080 70090 70100 70110
70130	OrderStage_OrdrSeqUpd	70040 70060 70070 70080 70090 70100 70110 70120
70140	OrderEvent_OrderStage	70040 70060 70070 70080 70090 70100 70110 70120 70130
70150	Execution_NewEvents	70010 70050
70160	Execution_CancelAndReplace	70010 70050
70170	Execution_CancelEvents	70010 70050 70150 70160
70180	Execution_CorrectionEvents	70010 70050 70150 70160 70170
70190	Trade_NewEvents	70010 70050
70200	Trade_CancelAndReplace	70010 70050
70210	Trade_CorrectionEvents	70010 70050 70190 70200
70220	Trade_CancelEvents	70010 70050 70190 70200 70210
70230	Trade_DerivedTrade	70010 70050 70190 70200 70210 70220
70240	Trade_OrigSeqIDUpd	70010 70050 70190 70200 70210 70220 70230
70250	Trade_ParentSeqIDUpd	70010 70050 70190 70200 70210 70220 70230 70240
70260	Trade_RplcngSeqIDUpd	70010 70050 70190 70200 70210 70220 70230 70240 70250
70270	TradeExecutionEvent_Trade	70010 70050
70280	TradeExecutionEvent_Execution	70010 70050
70290	TradeExecutionEvent_CancelReplaceTrade	70010 70050
70300	TradeExecutionEvent_FirmRefTrade	70010 70050 70270 70280 70290
70310	TradeExecutionEvent_MktRefTrade	70010 70050 70270 70280 70290
70320	Trade_RefData	70010 70050 70270 70280 70290
70330	Execution_Update	70010 70050 70150 70160 70170 70180

The following table provides a list of datamaps and description for each datamap. These datamaps are listed in order.

> **Note:**
>
> To execute the TRADE_EXECUTION_EVENT_STAGE datamap, the corresponding dat files must be modified at the following location: `<FIC_HOME>/database/golden_data`. The name must be changed from `TradeExecutionStage_yyyymmdd_DLY_01.dat` to `TradeExecutionEventStage_yyyymmdd_DLY_01.dat`.

Table F-20 FDT Datamap Description

Datamap Number	Datamap Name	Description
70010	Scrty_TradeExecutionStageInsert	This datamap populates the SCRTY table using ingested trade records present at TRADE_EXECUTION_EVENT_STAGE for that scurity, if security is not present already in the SCRTY table
70020	Scrty_OrderStageInsert	Thisdatamap populates the SCRTY table using ingested order records present at ORDR_STAGE for that security, if the security is not present already in the SCRTY table.
70030	MktCntr_OrderStageInsert	This datamap populates the MARKET_CENTER table using ingested order records present at ORDR_STAGE for that market centre, if themarket centre is not present already in the MARKET_CENTER table.
70040	OrderStage_DQupdate	This datamap updates the ORDR_STAGE table to mark invalid records.
70050	TradeExecutionEventStage_DQupdate	This datamap updates the TRDE_EXECUTION_EVENT_STAGE table to mark invalid trade events.
70060	OrderStage_FDTupdate	This datamap calculates and update information for each order event indentifying corresponding trade and quote information.
70070	OrderStage_RmngQtupdate	This datamap calculates and updates remaining units for each order event.
70080	OrderSummary	This datamap aggregates order events properties to identify the property for order, and populates the ORDR table.
70090	OrderSummary_OpenOrdrInsrt	This datamap populates the ORDR table based on the records in the OPEN_ORDR_STAGE table if required.
70100	OrderSummary_QtyUpdate	The datamap calculates the various quantity units and updates the ORDR table using those values.
70110	OrderStage_OpenOderUpd	This datamap populates the ORDR_STAGE table with order events not provided by customer but evident from the information provided by customer.

Table F-20 (Cont.) FDT Datamap Description

Datamap Number	Datamap Name	Description
70120	OrderSummary_Update	This datamap updates the ORDR table for various events and trades occurred for order.
70130	OrderStage_OrdrSeqUpd	This datamap updates the ORDR_STAGE table using the corresponding order_seq_id from the ORDR table
70140	OrderEvent_OrderStage	This datamap populates the ORDR_EVENT table with records processed and calculated at the ORDR_STAGE table.
70150	Execution_NewEvents	This datamap populates the EXECUTION table identifying NEW events in the TRADE_EXECUTION_EVENT_STAGE table.
70160	Execution_CancelAndReplace	This datamap populates the EXECUTION table identifying CANCEL AND REPLACE events in the TRADE_EXECUTION_EVENT_STAGE table.
70170	Execution_CancelEvents	This datamap updates the EXECUTION table identifying CANCEL events in the TRADE_EXECUTION_EVENT_STAGE table.
70180	Execution_CorrectionEvents	This datamap updates the EXECUTION table identifying CORRECTION events in the TRADE_EXECUTION_EVENT_STAGE table.
70190	Trade_NewEvents	This datamap populates the TRADE table identifying NEW events in the TRADE_EXECUTION_EVENT_STAGE table.
70200	Trade_CancelAndReplace	This datamap populates the TRADE table identifying CANCEL AND REPLACE events in the TRADE_EXECUTION_EVENT_STAGE table.
70210	Trade_CorrectionEvents	This datamap updates the TRADE table identifying CORRECTION events in the TRADE_EXECUTION_EVENT_STAGE table.
70220	Trade_CancelEvents	This datamap updates the TRADE table identifying CANCEL events in the TRADE_EXECUTION_EVENT_STAGE table.

Table F-20 (Cont.) FDT Datamap Description

Datamap Number	Datamap Name	Description
70230	Trade_DerivedTrade	This datamap populates TRADE tables identifying DERIVED TRADES in the TRADE_EXECUTION_EVENT_STAGE table.
70240	Trade_OrigSeqIDUpd	This datamap updates the original sequence identifier for non replaced trades.
70250	Trade_ParentSeqIDUpd	This datamap updates the parent sequence identifiers for the TRADE table.
70260	Trade_RplcngSeqIDUpd	This datamap updates the replacing sequence identifiers for the TRADE table.
70270	TradeExecutionEvent_Trade	This datamap populates the TRADE_EXECUTION_EVENT table with non order based trade records from the TRADE_EXECUTION_EVENT_STAGE.
70280	TradeExecutionEvent_Execution	This datamap populates the TRADE_EXECUTION_EVENT table with executed order records from the TRADE_EXECUTION_EVENT_STAGE table.
70290	TradeExecutionEvent_CancelReplaceTrade	This datamap populates the TRADE_EXECUTION_EVENT table with CANCEL AND REPLACE event executed order records from the TRADE_EXECUTION_EVENT_STAGE table.
70300	TradeExecutionEvent_FirmRefTrade	This datamap updates firm reference information in the TRADE_EXECUTION_EVENT table using the EXECUTION and TRADE tables.
70310	TradeExecutionEvent_MktRefTrade	This datamap updates market reference information in the TRADE_EXECUTION_EVENT table using the REPORTED SALE and TRADE tables.
70320	Trade_RefData	This datamap updates market and firm reference data in the TRADE table using the TRADE_EXECUTION_EVENT table.

Table F-20 (Cont.) FDT Datamap Description

Datamap Number	Datamap Name	Description
70330	Execution_Update	This datamap updates the EXECUTION table in using various events which occur for the trade in the TRADE_EXECUTION_EVENT_STAGE table.

G
Datamaps Matrix

This appendix provides a single window view of datamaps required for each solution set.

'X' denotes mandatory datamaps for each solution set.

'NA' denotes not applicable datamaps for the same solution set.

Table G-1 BD Datamaps

Datamap Number	Datamap Name	AML	Fraud	Insurance	AML Brokerage
10010	EmployeeControlledAccount	X	X	X	X
60010	PortfolioManagerPosition	NA	NA	NA	NA
60020	AccountGroupProductAllocation	NA	NA	NA	NA
60030	AccountProductAllocation	NA	NA	NA	NA
60040	UncoveredOptionExposureDaily	NA	NA	NA	NA
60050	InvestmentAdvisorProfile	NA	NA	NA	NA
60060	RegisteredRepresentativeProfile	NA	NA	NA	NA
60070	RegOToBorrower	NA	NA	NA	NA
60080	InterestedPartyToEmployee	NA	NA	NA	NA
50010	Customer_TotAcctUpd	NA	NA	NA	X
10015	FrontOfficeTransactionParty_SecondaryNames	X	X	NA	X
10020	FinancialInstitution_ThomsonDataInstitutionInsert	X	X	X	X
10030	AccountToClientBank_ThomsonDataInstitutionInsert	X	X	X	X
10040	FinancialInstitution_AIIMSPopulation	X	X	X	X

Table G-1 (Cont.) BD Datamaps

Datamap Number	Datamap Name	AML	Fraud	Insurance	AML Brokerage
10050	AccountToClientBank_AIIMSInstitutionInsert	X	X	X	X
10060	AccountToClientBank_InstitutionInsert	X	X	X	X
10070	AccountToClientBank_InstitutionUpd	X	X	X	X
10080	FinancialInstitution_FOTPSPopulation	X	X	X	X
10090	AccountToClientBank_FOTPSInstitutionInsert	X	X	X	X
10095	AccountCustomerRole	X	X	X	X
10096	AccountToCustomer	X	X	X	X
Existing 10100	AccountManagementStage	X	X	X	X
10110	LoanProfile_LoanProfileStage	X	NA	NA	X
10112	ServiceTeam_SprvsncdUpd	NA	NA	NA	NA
10113	InvestmentAdvisor_MangdAcctUpd	NA	NA	NA	NA
10114	Security_CIRRatingUpd	X	X	X	X
10116	BackOfficeTransaction_CollateralUpd	X	X	X	X
10120	BackOfficeTransaction_OriginalTransactionReversalUpd	X	X	NA	X
10130	BackOfficeTransaction_CancelledTransactionReversalCreditUpd	X	X	NA	X
10140	BackOfficeTransaction_CancelledTransactionReversalDebitUpd	X	X	NA	X
10150	FrontOfficeTransactionParty_InstnSeqID	X	X	X	X

Table G-1 (Cont.) BD Datamaps

Datamap Number	Datamap Name	AML	Fraud	Insurance	AML Brokerage
10160	FrontOfficeTransactionParty_HoldingInstnSeqID	X	X	X	X
10170	FinancialInstitution_AnticipatoryProfile	NA	X	X	X
10180	AccountToClientBank_AnticipatoryProfile	NA	X	X	X
10190	AnticipatoryProfile_AccountToClientBank	NA	X	X	X
50020	DailyAggregateStage	NA	NA	NA	X
50030	OffsettingAccountPairStage	NA	NA	NA	X
50040	TradeDailyTotalCountStage	NA	NA	NA	X
10200	CustomerAccountStage_FrontOfficeTransactionParty	X	X	NA	X
10210	FrontOfficeTransaction_UnrelatedPartyUpd	X	X	NA	X
10220	FinancialInstitution_SettlementInstruction	NA	X	X	X
10230	AccountToClientBank_SettlementInstruction	NA	X	X	X
10240	SettlementInstruction_AccountToClientBank	NA	X	X	X
40010	FinancialInstitution_InsuranceTransaction	NA	NA	X	NA
40020	AccountToClientBank_InsuranceTransaction	NA	NA	X	NA
40030	InsuranceTransaction_AccountToClientBank	NA	NA	X	NA
10245	WLMProcessingLock	X	X	X	X
10250	WatchListEntry_WatchListEntryCurrDayInsert	X	X	X	X

Table G-1 (Cont.) BD Datamaps

Datamap Number	Datamap Name	AML	Fraud	Insurance	AML Brokerage
10260	WatchListAudit_StatusUpd	X	X	X	X
10270	WatchList_WatchListSourceAuditInsert	X	X	X	X
10280	WatchList_WatchListSourceAuditUpd	X	X	X	X
10290	WatchList_WatchListSourceUpd	X	X	X	X
10300	WatchListEntry_WatchListAuditUpd	X	X	X	X
10310	WatchListEntryAudit_WatchListEntryUpdate	X	X	X	X
10320	Customer_KYCRiskUpd	X	X	X	X
60090	CorrespondentBankToPeerGroup	NA	NA	NA	NA
10330	DerivedAddress_SettlementInstructionInsert	NA	X	NA	X
10340	DerivedAddress_SettlementInstructionUpd	NA	X	NA	X
10350	SettlementInstruction_PhysicalDlvryAddrUpd	NA	X	NA	X
10360	DerivedAddress_FrontOfficeTransactioPartyStageInsert	X	X	X	X
10370	DerivedAddress_FrontOfficeTransactioPartyStageUpd	X	X	X	X
10380	FrontOfficeTransactionParty_DerivedAddress	X	X	X	X
40040	DerivedAddress_InsuranceTransactionInsert	NA	NA	X	NA
40050	DerivedAddress_InsuranceTransactionUpd	NA	NA	X	NA
40060	InsuranceTransaction_InstitutionAddrUpd	NA	NA	X	NA

Table G-1 (Cont.) BD Datamaps

Datamap Number	Datamap Name	AML	Fraud	Insurance	AML Brokerage
40070	DerivedEntity_InsuranceTransactionInsert	NA	NA	X	NA
40080	DerivedEntity_InsuranceTransactionUpd	NA	NA	X	NA
10390	DerivedEntity_FrontOfficeTransactionPartyInsert	X	X	X	X
10400	DerivedEntity_FrontOfficeTransactionPartyUpd	X	X	X	X
10410	DerivedEntity_SettlementInstructionInsert	X	X	X	X
10420	DerivedEntity_SettlementInstructionUpd	X	X	X	X
10430	CorrespondentBank_FrontOfficeTransactionPartyStageIn sert	X	X	X	X
10440	CorrespondentBank_FrontOfficeTransactionPartyStageU pd	X	X	X	X
10450	WatchListStagingTable_WatchList	X	X	X	X
10460	WatchListStagingTable_WatchListInstnIDUpd	X	X	X	X
10470	PreviousWatchList_WatchList	X	X	X	X
10480	DerivedAddress_WatchListNewCountries	X	X	X	X
10485	WLMProcessingUnlock	X	X	X	X
10490	LinkStaging_FrontOfficeTransactionParty	X	X	X	X
40090	LinkStaging_InsTrxnDerivedEntDerivedAdd	NA	NA	X	NA
10500	LinkStaging_InstructionDerivedEntDerivedAdd	X	X	X	X

Table G-1 (Cont.) BD Datamaps

Datamap Number	Datamap Name	AML	Fraud	Insurance	AML Brokerage
10510	NameMatchStaging	X	X	X	X
10520	WatchListStagingTable_NameMatchStageInsert	X	X	X	X
10530	DerivedEntityLink_LinkStage	X	X	X	X
10540	DerivedEntitytoDerivedAddress_LinkStage	X	X	X	X
10550	DerivedEntitytoInternalAccount_LinkStage	X	X	X	X
10560	DerivedAddresstoInternalAccount_LinkStage	X	X	X	X
10570	WatchListStagingTable2_WatchListStage2AcctExistence	X	X	X	X
10580	WatchListStagingTable2_WatchListStage2CBExistence	X	X	X	X
10590	WatchListStagingTable2_WatchListStage2CustExistence	X	X	X	X
10600	WatchListStagingTable2_WatchListStage2DAExistence	X	X	X	X
10610	WatchListStagingTable2_WatchListStage2EEExistence	X	X	X	X
10620	WatchListStagingTable2_WatchListStage	X	X	X	X
10630	WatchListStagingTable2_AcctListMembershipUpd	X	X	X	X
10640	WatchListStagingTable2_CBListMembershipUpd	X	X	X	X
10650	WatchListStagingTable2_CustListMembershipUpd	X	X	X	X

Table G-1 (Cont.) BD Datamaps

Datamap Number	Datamap Name	AML	Fraud	Insurance	AML Brokerage
10660	WatchListStagingTable2_EEListMembershipUpd	X	X	X	X
10670	WatchListStagingTable2_EEListMembershipStatusUpd	X	X	X	X
10680	WatchListStagingTable2_DAListMembershipUpd	X	X	X	X
10690	WatchListStagingTable2_DAListMembershipStatusUpd	X	X	X	X
10700	WatchListStagingTable2_WatchListStage2SeqIdUpd	X	X	X	X
10710	WatchListStagingTable2_WatchListStage2IntrlIdUpd	X	X	X	X
10720	Customer_WatchListStage2ListRisk	X	X	X	X
10730	CorrespondentBank_WatchListStage2EffectiveRisk	X	X	X	X
10740	Customer_WatchListStage2EffectiveRisk	X	X	X	X
10750	DerivedAddress_WatchListStage2EffectiveRisk	X	X	X	X
10760	DerivedEntity_WatchListStage2EffectiveRisk	X	X	X	X
10770	WatchListStagingTable2_WatchListStage2SeqId	X	X	X	X
10780	AccountListMembership_WatchListStage2Insert	X	X	X	X
10790	AccountListMembership_WatchListStage2Upd	X	X	X	X

Table G-1 (Cont.) BD Datamaps

Datamap Number	Datamap Name	AML	Fraud	Insurance	AML Brokerage
10800	CorrespondentBankListMembership_WatchListStage2Ins ert	X	X	X	X
10810	CorrespondentBankListMembership_WatchListStage2Up d	X	X	X	X
10820	CustomerListMembership_WatchListStage2Insert	X	X	X	X
10830	CustomerListMembership_WatchListStage2Upd	X	X	X	X
10840	DerivedAddressListMembership_WatchListStage2Insert	X	X	X	X
10850	DerivedAddressListMembership_WatchListStage2Upd	X	X	X	X
10860	DerivedEntityListMembership_WatchListStage2Insert	X	X	X	X
10870	DerivedEntityListMembership_WatchListStage2Upd	X	X	X	X
10875	Account_EffectiveRiskFactorTxtUpd	X	X	X	X
10880	Account_OverallEffectiveRiskUpd	X	X	X	X
10881	Account_AccountCustRiskUpd	X	X	X	X
10890	Account_EffRiskUpdAfterWLRiskRemoval	X	X	X	X
10900	Account_WatchListStage2EffectiveRisk	X	X	X	X
10910	WatchListStagingTable2_WatchListStage2IntrlId	X	X	X	X

Table G-1 (Cont.) BD Datamaps

Datamap Number	Datamap Name	AML	Fraud	Insurance	AML Brokerage
10920	BackOfficeTransaction_EffectiveAcctivityRiskUpd	X	X	NA	X
10930	SettlementInstruction_EntityActivityRiskUpd	NA	X	NA	X
10940	FrontOfficeTransactionPartyRiskStage_EntityActivityRiskInsert	X	X	X	X
10955	AccountGroup_InvestmentObjectiveUpd	NA	NA	NA	NA
40100	InsuranceTransaction_EntityActivityRiskUpd	NA	NA	X	NA
20010	CorrespondentBank_JurisdictionUpd	X	NA	NA	NA
20020	CorrespondentBank_AcctJurisdictionReUpd	X	NA	NA	NA
20030	FinancialInstitution_InstNameUpd	X	NA	NA	NA
10960	AccountGroup_JurisdictionUpd	X	X	NA	X
10970	TransactionPartyCrossReference_BackOfficeTransaction	X	X	NA	X
10980	CashTransaction_FrontOfficeTransaction	X	X	NA	X
10990	MonetaryInstrumentTransaction_FrontOfficeTransaction	X	X	NA	X
11000	TransactionPartyCrossReference_FrontOfficeTransaction	X	X	NA	X
11010	WireTransaction_FrontOfficeTransaction	X	X	NA	X
11020	WireTransactionInstitutionLeg_FrontOfficeTransaction	X	X	NA	X

Table G-1 (Cont.) BD Datamaps

Datamap Number	Datamap Name	AML	Fraud	Insurance	AML Brokerage
11030	CashTransaction_FrontOfficeTransactionRevAdj	X	X	NA	X
11040	MonetaryInstrumentTransaction_FrontOfficeTransaction RevAdj	X	X	NA	X
11050	WireTransaction_FrontOfficeTransactionRevAdj	X	X	NA	X
11060	TrustedPair_StatusEXPUpd	X	X	X	X
11070	TrustedPairMember_AcctExtEntEffecRiskUpd	X	X	X	X
11080	TrustedPair_StatusRRCInsert	X	X	X	X
11090	TrustedPair_StatusRRCUpd	X	X	X	X
11100	ApprovalActionsAudit_TrustedPair	X	X	X	X
11110	TrustedPairMember_StatusRRCInsert	X	X	X	X
11120	BackOfficeTransaction_TrustedFlagsUpd	X	X	X	X
11130	InsuranceTransaction_TrustedFlagsUpd	NA	NA	X	NA
11140	MonetaryInstrumentTransaction_TrustedFlagsUpd	X	X	X	X
11150	WireTransaction_TrustedFlagsUpd	X	X	X	X
50050	CustomerDailyProfile_BOT	NA	NA	NA	X
50060	CustomerDailyProfile_FOTPS	NA	NA	NA	X
50070	InstitutionalAccountDailyProfile_DEAL	NA	NA	NA	X
50080	CustomerDailyProfile_DEAL	NA	NA	NA	X

Table G-1 (Cont.) BD Datamaps

Datamap Number	Datamap Name	AML	Fraud	Insurance	AML Brokerage
50090	InstitutionalAccountDailyProfile_INST	NA	NA	NA	X
50100	CustomerDailyProfile_INST	NA	NA	NA	X
50110	InstitutionalAccountDailyProfile_CorpAction	NA	NA	NA	X
50120	CustomerDailyProfile_CorpAction	NA	NA	NA	X
50130	InstitutionalAccountDailyProfile_Trade	NA	NA	NA	X
50140	CustomerDailyProfile_Trade	NA	NA	NA	X
60100	ManagedAccountDailyProfile_SameDayTrade	NA	NA	NA	NA
60110	ManagedAccountDailyProfile_Trade	NA	NA	NA	NA
60120	ManagedAccountDailyProfile_BOT	NA	NA	NA	NA
11160	AccountDailyProfile-Trade	X	X	NA	X
11170	AccountDailyProfile-Transaction	X	X	NA	X
11180	AccountProfile_Trade	X	X	NA	X
11190	AccountProfile_Transaction	X	X	NA	X
11200	AccountProfile_Stage	X	X	NA	X
11210	AccountProfile_Position	X	X	NA	X
11220	AccountProfile_Balance	X	X	NA	X
60130	HouseholdProfile	NA	NA	NA	NA
50150	InstitutionalAccountProfile	NA	NA	NA	X
50160	CustomerProfile	NA	NA	NA	X
60140	ManagedAccountProfile	NA	NA	NA	NA

Table G-1 (Cont.) BD Datamaps

Datamap Number	Datamap Name	AML	Fraud	Insurance	AML Brokerage
60145	AccountPosition_PercentofPortfolioUpd	NA	NA	NA	NA
20040	CorrespondentBankProfile	X	NA	NA	NA
20050	AccountATMDailyProfile	X	NA	NA	NA
11230	ChangeLog_ActProfileInactivity	X	X	NA	X
11240	AccountPeerGroupMonthlyTransactionProfile	X	X	NA	X
20060	CorrespondentBankPeerGroupTransactionProfile	X	NA	NA	NA
20070	AccountChannelWeeklyProfile	X	NA	NA	NA
40110	InsurancePolicyDailyProfile_InsTrxnInsPolicyBal	NA	NA	X	NA
40120	InsurancePolicyProfile_InsurancePolicyDailyProfile	NA	NA	X	NA
50170	CustomerBalance_ActiveOTCTradeCtUpd	NA	NA	NA	X
60150	AccountPositionDerived	NA	NA	NA	NA
60160	AccountBalance_AcctPosnPair	NA	NA	NA	NA
60170	AccountBalance_Acctposn	NA	NA	NA	NA
60180	HouseholdBalance	NA	NA	NA	NA
60190	AccountManagementStage	*	*	*	*
11300	AccountChangeLogSummary	X	X	X	X
11310	AccountToCustomerChangeLogSummary	X	X	X	X
11320	CustomerChangeLogSummary	X	X	X	X

Note:

- The AccountChangeLogSummary, AccountToCustomerChangeLogSummary, and CustomerChangeLogSummary datamaps must be run with `execute.sh` from 8.0.2 onwards.
- BackOfficeTransaction must be loaded after the AccountManagementStage utility has been executed.

H

Configuring Administration Tools

If the administration tool is deployed on a separate web application server, then follow these steps:

1. Log in as an Administrator User. The Home page displays.
2. Click **Manage Configuration** from the LHS menu.
3. Select **Manage Common Parameters**.
4. In the **Parameter Category** drop-down, select **Used for Design**.
5. In the **Parameter Name** drop-down, select **Admin Tools**.
6. Set the **Attribute 2 Value** as follows:

   ```
   <PROTOCOL>:// <AdminTools_WEB_SERVER_NAME>:<PORT>
   ```

   - <PROTOCOL> is web page access PROTOCOL (http or https).
   - <AdminTools_WEB_SERVER_NAME> is the FQDN of the web application server hosting Administrative Tools.
   - <PORT> is the web application server port hosting Admin Tools.
7. Set the **Attribute 4 Value** as follows:

   ```
   <PROTOCOL>://<AdminTools_WEB_SERVER_NAME>:<PORT>/
   <CONTEXT_NAME>
   ```

   - <PROTOCOL> is web page access PROTOCOL (http or https).
   - <AdminTools_WEB_SERVER_NAME> is the FQDN of the web application server hosting Administrative Tools.
   - <PORT> is the web application server port host
   - <CONTEXT_NAME> is the context name given during installation
8. Set the **Attribute 5 Value** as follows:

   Infodom Name Associated with each Atomic Schema and ADDON.
9. Set the **Attribute 8 Value** as follows:

   Segment name of the application, such as FCCMSEGMNT

Right to Be Forgotten

This appendix provides instructions on how to configure the Right to be Forgotten feature used in the OFSAA Data Foundation applications.

Right to be Forgotten is the task of dropping PII (Personally Identifiable Information) of a Data Subject for the given Party. The financial institution can drop PII for those Data Subjects who have exercised the Right to be Forgotten functionality. The Data Subjects may have made significant financial transactions, and (or) financial information may be required for regulatory or compliance reporting. Deleting the complete record that consists of PII may lead to issues in data reconciliation. In OFSAA, the PII data is replaced with randomized values, and therefore, the complete Data Subject record is retained. As a result, financial information is retained; however, the associated Party PII is removed permanently.

I.1 Data Redaction

Oracle Financial Services Analytical Application Infrastructure (OFSAAI) is enhanced to enable masking of sensitive data and Personal Identification Information (PII) to adhere to Regulations and Privacy Policies.

Oracle Data Redaction provides selective, on-the-fly redaction of sensitive data in database query results prior to display by applications so that unauthorized users cannot view the sensitive data. The stored data remains unaltered, while displayed data is transformed to a pattern that does not contain any identifiable information.

For more information, see Oracle Financial Services Advanced Analytical Applications Infrastructure Administration and Configuration Guide Release 8.1.x.

> **Note:**
>
> - The Redacted user must not have the Edit option in the UI
> - To display the default value in the UI, configure the AAI_FF_FORM_CONTROLS_B table
> - If you do not have data redact function rights and try to update any value in PII data, the application updates all redacted column values to null which leads to data loss.

Configuring Redaction for FCC Grids

Redaction can be configured for FCC grids that are configured using the FCC UI MODULE CONF table. The V MODULE PROP column must be configured under the columnProperties section. The following is an example for the Related Party grid in the Customer tab (see the highlighted attributes):

```
"key": "customer_name",
"locale_code":"RENDERER.CM_RP_CUST_NAME",
"align": "left",
```

```
"headerAlign": "left",
"width": "0.13",
"dataType": "string",
"draggable":true,
"resizable":true,
"sortable":true,
"readOnly":true,
"visible": true,
"addToColMenu": true,
"isRedactedColumn": true,
"redactedValueToDisplay": "********"
```

Date, Integer, and Float fields will appear as blank, regardless of the value provided for the redactedValueToDisplay attribute.

Redaction on UI Screens

The FCC_NATIVE_REDACTION_CONFIG table in the Atomic schema allows you to set the value of redaction under the V_REDACTED_VALUE_TO_DISPLAY column, based on your requirements.

Date fields will be displayed as blank if redacted, regardless of the value is configured in the V_REDACTED_VALUE_TO_DISPLAY column.

You can also configure redaction for Accounts and External Entities. Refer to the FCC_NATIVE_REDACTION_CONFIG table for more information. Additionally you can also see the Redaction_Account_External Entity_Info spreadsheet in MOS.

I.2 Implementing Right to be Forgotten by OFSAA

To implement Right to be Forgotten, follow these steps:

1. Use the FSI_PARTY_RIGHT_TO_FORGET table to collect the input list of Party IDs for which PII must be removed from the system. The financial institution must source this Party ID list into the FSI_PARTY_RIGHT_TO_FORGET table, and then call the batch *(<<INFODOM>>_RightToForget)* or schedule it.

   > **Note:**
   >
   > - For the sample query, see the Sample Query for the FSI_PARTY_RIGHT_TO_FORGET Table section.
   > - If Redaction is already performed and if you want to implement Right to Forget, you must revert the redaction policy. For more information, see the Disabling Data Redaction section in Oracle Financial Services Advanced Analytical Applications Infrastructure Administration and Configuration Guide Release 8.1.x.

2. Use the AAI table AAI_DRF_FUNCTION_COLUMN_MAP to store the PII attribute list. During the Right to Forget batch execution, the AAI_DRF_FUNCTION_COLUMN_MAP table is referred to as randomize the PII values. See the Data Redaction section in Oracle Financial Services Advanced Analytical Applications Infrastructure Administration and Configuration Guide Release 8.1.x.

3. Use the AAI table AAI_DRF_QUERY_METADATA to store the query metadata, which is used during the <<INFODOM>>_RightToForget batch execution. This is the query metadata table that can lead to the following types of queries:

   - When the table consists of Party Identifier as an attribute, a simple record is required in the metadata query table.

   For example: Select v_party_id from Dim_Party where v_party_id='10'

   - When the table does not consist of Party Identifier as an attribute, an interrelated set of records is required in the metadata query table AAI_DRF_QUERY_METADATA. Compose these set of records in a systematic way such that, for the selected Party Identifier, the table join procedure can be performed and traversed to reach the required PII attribute.

   For example: The DIM_CLAIM table does not consist of N_CLAIM_SKEY (N_CLAIM_SKEY is the required Primary Key for the PII Attribute N_DRIVER_SKEY). Therefore, perform a table join procedure similar to the following query:

```
Dim_driver.n_driver_skey from dim_driver dim_driver, fct_driver_details
fct_driver_details,
        Fct_Claim_Driver_Vehicle_Map Fct_Claim_Driver_Vehicle_Map,
Dim_Claim Dim_Claim where
        dim_driver.n_driver_skey=fct_driver_details.n_driver_skey and
fct_driver_details.n_driver_skey=Fct_Claim_Driver_Vehicle_Map.n_driver_skey
 and
        Fct_Claim_Driver_Vehicle_Map.n_claim_skey=Dim_Claim.n_claim_skey
and
        v_claim_id='GDPR'
```

   In the preceding scenario, DIM_CLAIM.N_CLAIM_SKEY is a Number Datatype.

   > **Note:**
   >
   > For more information about this table, see Table Definition for AAI_DRF_QUERY_METADATA in the *OIDF Application Pack Data Protection Implementation Guide.*
   >
   > - To arrive at the query above, see Steps to Perform the Table Join Procedure.
   > - For a pictorial representation of this query, see the AAI_DRF_QUERY_METADATA Table section.
   > - For more sample queries generated using the query metadata table, see Sample Queries Using the AAI_DRF_QUERY_METADATA Table.

   You must arrive at the SKey or equivalent column in the table, which consists of the required PII attributes. Then the <<INFODOM>>_RightToForget batch uses this key to filter records (For example DIM_DRIVER) and randomize all the PIIs listed in the AAI_DRF_FUNCTION_- COLUMN_MAP for that table.

4. After completing these steps, PII attributes can be queried and the values are randomized.